RTL9310

LAYER 2/3 MANAGED 48*10/100M/1000M + 6*10GE -PORT SWITCH CONTROLLER

Developer Guide

Rev. 1.00
25 July 2018

Realtek Semiconductor Corp.
No. 2, Innovation Road II, Hsinchu Science Park, Hsinchu 300, Taiwan
Tel.: +886-3-578-0211 Fax: +886-3-577-6047
www.realtek.com

USING THIS DOCUMENT
This document is intended for use by the system engineer when integrating with Realtek switch products. Though every effort has been made to assure that this document is current and accurate, more information may have become available subsequent to the production of this guide. In that event, please contact your Realtek representative for additional information that may help in the development process.

Revision	Release Date	Summary
1.0	2018/07/25	Initial

Contents

Table List

Figure List

1 Overview

The RTL9310 chip family is a layer 3 stackable 10-Gigabit Ethernet switch with 130Gbps forwarding bandwidth. The RTL9310 integrates a Dual-Core 1GHz MIPS interAptive CPU subsystem with SPI NOR/NAND flash and DDR3/4 SDRAM support.

In addition to normal forwarding pipeline, it further implements an OpenFlow pipeline which complies with OpenFlow standard for SDN application. The RTL9310 is a cost-effective solution for Software-Defined Network (SDN), Small-Medium Business (SMB) and Carrier Ethernet Access/Edge applications with wire-speed performance for 1GE/2.5GE/10GE platforms.

1.1 Chip Family

The RTL9310 family consists of several members with different port configuration. Each family member with its port configuration is listed as below table.

Table 1-1 Family Member List

Chip Name	1GE Ports	2.5GE Ports	10GE Ports	USXGMII
RTL9311-CG	48	X	6	X
RTL9312-CG	X	X	6	6
RTL9313-CG	X	X	12	X

1.2 Feature List

The RTL9310 major features including CPU subsystem and capacity are shown in Table 1-2.

Table 1-2 Feature List

Feature	Description
CPU sub-system	
CPU	MIPS interAptive Dual Core 1GHz
L1/L2 cache	32KB/256KB
On-chip SRAM	64KB
Flash Interface	SPI NOR up to 64MB SPI NAND up to 512MB
Memory Interface	32-bit DDR3/DDR4-1600 up to 2GB
UART	2*UART
USB Host 2.0	USB drive
I2C/SPI Master	I2C master support 12 peripherals with 2 clocks SPI master support 2 peripherals with 2 chip select *SDA, SCL, CS are shared with GPIO pins
GPIO/External GPIO	32/74

External CPU Interface	PCIe Gen2
Hardware Capability and Interface	
Integrated PHY Chipset	X
Switch Capacity	- 48G+6*10G - 6*USXGMII+6*10G - 12*10G
Packet Buffer	16Mbit
Jumbo Frame	12KB
Stacking	
Maximum Stackable Device	16
Stacking Topology	Line and Ring
Stacking Port	- Max. 16 stacking ports - Uplink stacking port supports 12 queues - Fast link fault detection by CCM (1~1024ms)
Stacking Supported Feature	- 128 Trunk across devices - 4 Mirror across devices
MAC and Port Capability	
MAC Address Table Size	32K (2-left 4-way hash)
Multicast Port Mask Table Size	4K
L2 Entry Notification	Notify L2 table change to CPU. Support Packet mode and NIC mode.
Mirror & Sampling	
Mirroring set	- 4 mirror sets - One mirroring port and multiple mirrored ports - Flow-based mirror by ACL
RSPAN	Source, Intermediate, Destination
Sampling (sFlow)	Ingress port, Egress port
VLAN	
1Q and QinQ VLAN	- 4K VLANs - IVL, SVL, IVL/SVL mixed mode. - Flexible QinQ
Protocol VLAN	Global 16 protocol VLAN configurations
Mac-based VLAN	share with 2K Ingress VLAN Translation table
IP-Subnet-based VLAN	share with 2K Ingress VLAN Translation table
VLAN Translation	2K Ingress and 1K Egress
N:1 VLAN Translation	Via MAC Address Table
VLAN Profile	16 VLAN profiles which define L2 learning enable/disable and unknown L2/IPv4/IPv6 multicast flooding domain.
VLAN Filtering	Per ingress/egress port enable VLAN filtering
Trunk (IEEE 802.3ad LACP)	
Groups	128

Distribution Algorithm for Load Balance	Known unicast hash key selection: L2: SPP/SMAC/DMAC/VID L3: SPP/SMAC/DMAC/VID /SIP/DIP/L4SPORT/L4DPORT/ Protocol ID/Flow Label Non-unicast hash by fixed key below: L2: SMAC, DMAC, SPP L3: SIP, DIP, SPP Note: SPP stands for source physical port.
Local First Load Balance	Local trunk member priorities selected in stacking mode
Stacking Port Trunk	Specific trunk group setting for stacking ports
Ports for Each Group	8
Trunk Fail Over	Stand-alone mode: Hardware auto fail-over for link down ports. Stacking mode: Handled by software.
Traffic Separation	Unknown unicast, broadcast and known multicast traffic separation.
Multicast	
IGMP/MLD Snooping	ASM and SSM
Multicast Lookup Method	MAC-based: (DMAC,VID) or (DMAC,FID) IP-based: (VRF, SIP, GIP, VID/IF)+(VRF, *, GIP, VID/IF)
Multicast Groups	MAC-based: Up to 32K Groups with 4K different Port Mask IP-based: Up to 6K IPv4/2K IPv6 Groups with 4K different Port Mask
QoS	
Amount of Queues	Normal Port: 8 egress queues Internal CPU port: 32 egress queues
Storm Control	Per port specify rate (bps or pps): - Unknown unicast or unicast(unknown + known) - Unknown multicast or multicast(unknown + known) - Broadcast
Control Protocol Storm Control	Per port specify rate (pps): - BPDU - ARP/IPv6 NS - IGMP/MLD - DHCP
Ingress Bandwidth Control	Rate 0~10Gbps, unit: 16Kbps.
Egress Bandwidth Control	- Rate 0~10Gbps, unit: 16Kbps. - CPU port can specify bps or pps mode - Queue-based Assured and Fixed bandwidth control
Scheduling Method	WRR/WFQ/Strict/Strict+WRR/Strict+WFQ
WRED	Simple WRED
Remarking	Per egress port enable remarking: - Inner VLAN - Outer VLAN - DSCP - DEI
ACL	
Number of Entries	4K (share with OpenFlow)

Entry Width	240-bits
Template Number	10 templates(5 pre-defined, 5 user-defined)
ACL Phase	VLAN-based ACL (VACL), Ingress ACL (IACL) and Egress ACL (EACL)
Meter/Marker	- 512 (DLB/srTCM/trTCM) - Hierarchical policing - CAR Supportive
Counter	- 4K (packet-based or byte-based) - Multiple counter execution
Range Check	- 16 (I-VID/O-VID/L4-SPORT/L4-DPORT/Packet length/L3 packet length) - 8 IPv4 or 2 IPv6 or 4 IPv6 suffix
Field Selector	14 entries (per entry 2 byte)
Security	
IP+MAC+PORT+VLAN Binding	1K flexible binding entry
MAC Learning Constraint	System/Port/VLAN
DoS Prevention	L2~L4
MAC White List	Combine with L2 table
MAC Black Hole	Source and Destination MAC
Port Movement	Per port specify dynamic and static port movement action
Layer 3	
IP Routing	- IPv4/IPv6 unicast and multicast routing - 12K network route and 12K host route - Longest prefix match (LPM) based routing - 256 VRF Domain
uRPF	Strict and loose mode
Policy-based Routing	By ACL
ECMP/WCMP	- Distribution by source/destination IP address, DSCP, TCP/UDP port
Tunneling	
IP Tunnel	- IP-in-IP - L3 GRE - 6to4 - 6rd - ISATAP
VXLAN	- L2 Bridging between Traditional VLANs and VXLANs - L3 Routing Into and Out of VXLAN Tunnel - MAC Address Learning with VTEP/VNI as L2 interface - 384 VTEP - 2048 VNI - VXLAN and VXLAN-GPE Encapsulation Support
Carrier Ethernet	

MPLS L3 VPN	- Up to two MPLS labels lookup - Pop/Swap/Push/PHP operations - 2K labels - TTL support Uniform/Short Pipe/Pipe - TC/DP support E-LSP/L-LSP - Trap special label(0-15) - Exception handling
802.3ah OAM Loopback	Per port specify Multiplexer and Parser state
802.3ah OAM Dying Gasp	Hardware-based dying gasp with software-defined payload
802.1ag CFM	- Link fault detection for G.8031/8032 Support - 8 instances which can have different CCM payload - HW transmits CCM packet from 1~1024ms - Trigger interrupt when link fault is detected
Y.1731 CFM	One-Way/Two-Way ETH-DM
IEEE 1588 v1/v2	Support by PHY (Accuracy is 8ns)
SyncE	Support by PHY.
Power Saving	
EEE	V
System Idle Mode	V
AVB (Audio Video Bridging)	
PTP Timestamp	Support by PHY
SDN (Software Defined Network)	
OpenFlow	- OpenFlow v1.5.1 Compliant - OpenFlow hybrid switch - 32K flow entries(4K Full Match+16K L2+12K L3) - Five Flow Tables (3 Full Match+L2 Optimized+L3 Optimized) - 4 Ingress Flow Tables + 1 Egress Flow Table - Extensible Flow Tables - Group table support - Meter table support
Virtual Network Function	
802.1BR(Bridge Port Extension)	- Control Bridge (8K Multicast Replication Number) - 32K ECID Channel for Port Extender Bridge - 256 Name Space Group for Port Extender Bridge - ECID Extension - ETAG-based QoS

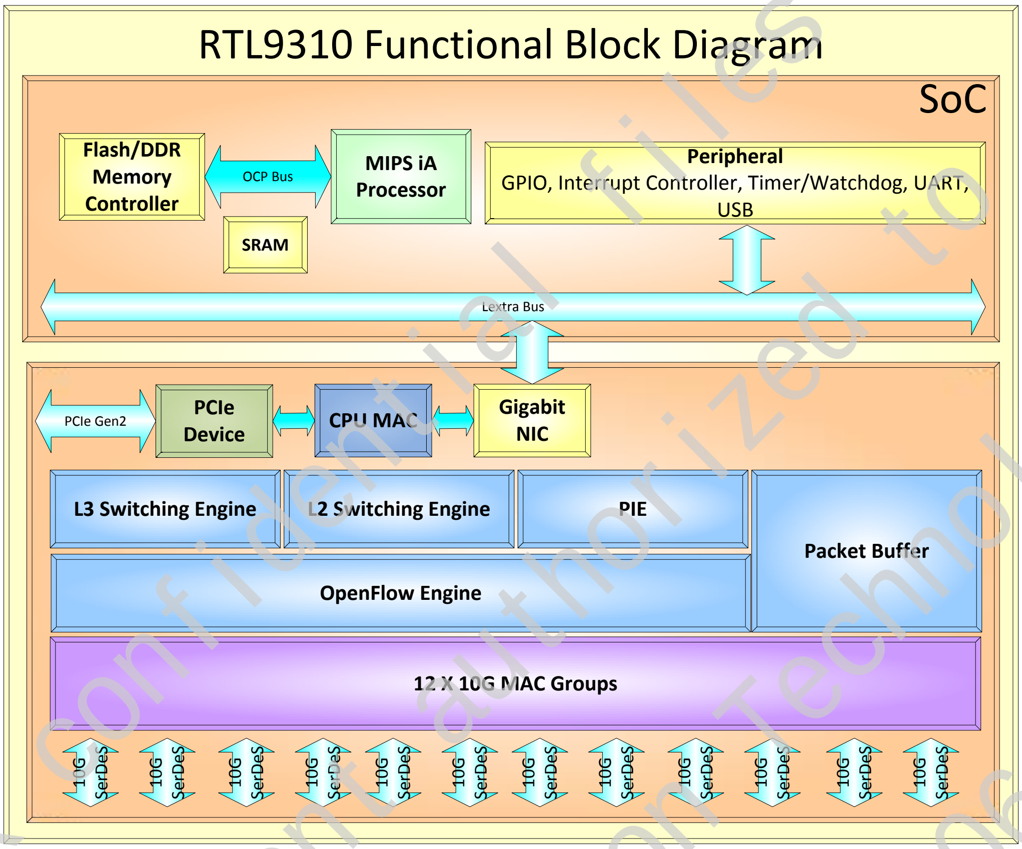
1.3 Switching Architecture

1.3.1 Device Block Diagram

The RTL9310 consists of multiple module blocks including CPU interface, packet buffer, L2/L3 switching engine, packet inspect engine (PIE) and OpenFlow engine. It supports 14 SERDES in total to connect to

Fast/Gigabit/10Gigabit Ethernet PHYs. 10G-R (XFI/SFI) interface is supported for 10GBase-T PHY connection or direct 10G fiber (XFP/SFP+) connection.

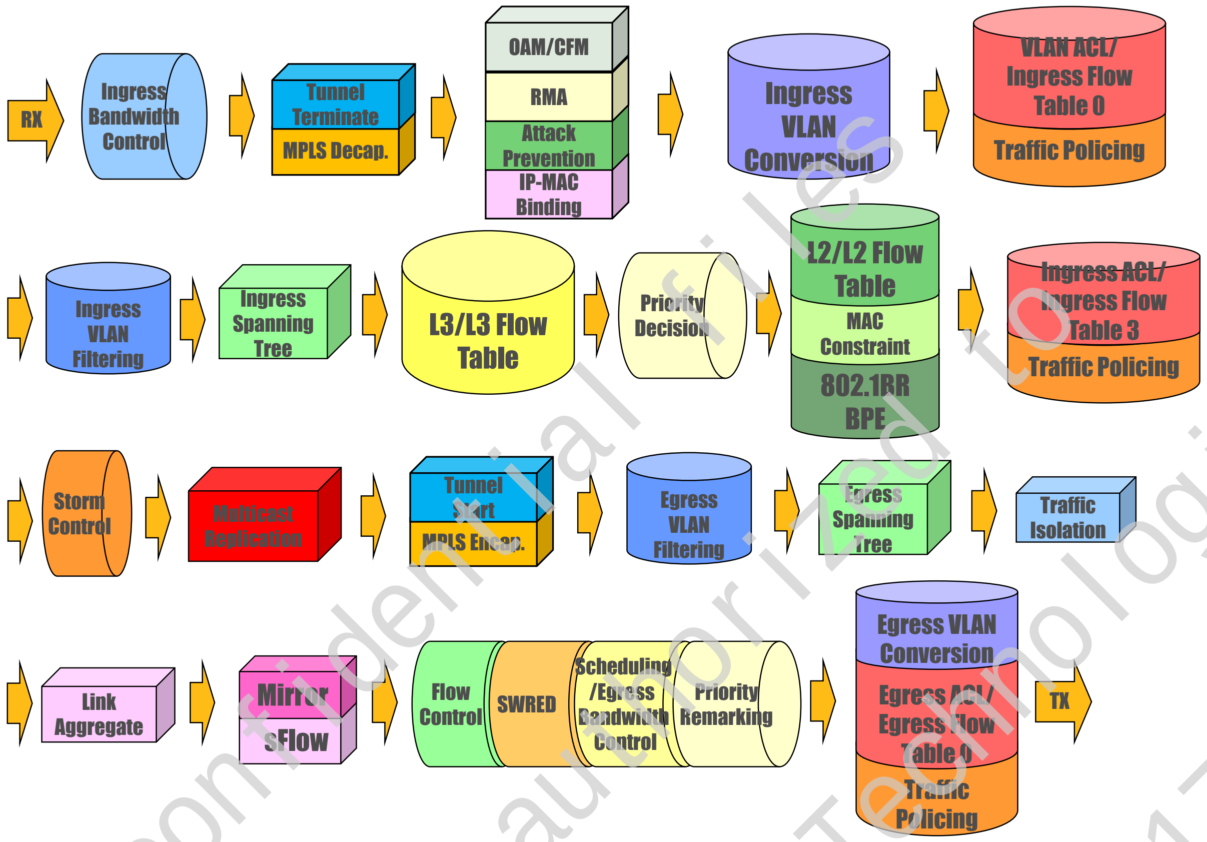
Figure 1-1 Block Diagram



1.3.2 Pipeline WalkThrough

The packet goes through RTL9310 is pipelined as below switching architecture.

Figure 1-2 Switching Pipeline



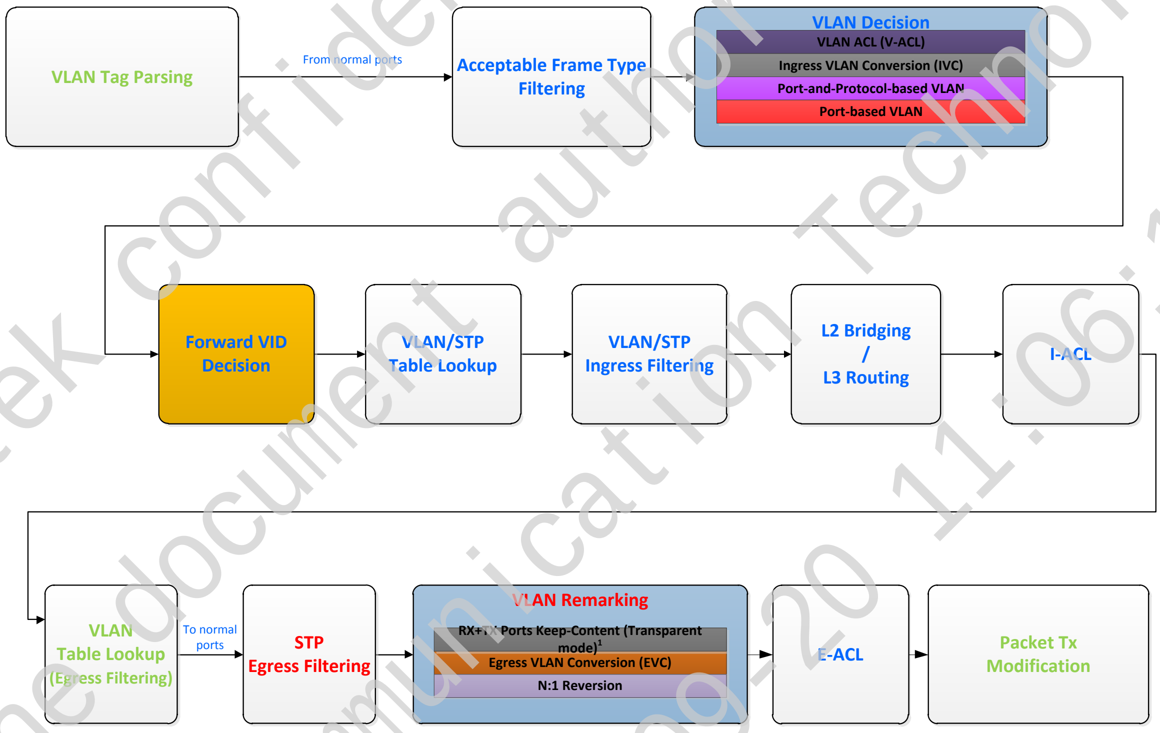
2 VLAN

The system supports IEEE 802.1Q and Q-in-Q VLAN (802.1ad). When it receives a packet, it will base on the settings to parse inner tag and outer tag, and then base on inner or outer tag to do the VLAN learning and forwarding, it can also assign the packet to be un-tag or inner tag or outer tag or double tag when transmitting packets.

In Q-in-Q, depends on the connected network is a provider network or a customer network, each port can be separated into two types: NNI(Network Network Interface) and UNI(User Network Interface), NNI port recognizes inner and outer tag while UNI port usually only recognizes inner tag.

2.1 VLAN Overview

Figure 2-1 VLAN Block



The system supports:

4096 VLANs	Includes VLAN member, untagged member and multiple spanning tree instance etc
8 VLAN profiles	VLAN additional configuration
8 Protocol-and-Port-based VLAN entries	Specifies IEEE 802.1v(Protocol-and-Port-based VLAN) configuration. Global eight protocol values are supported and each port can have different PPB VLAN value for each global protocol value

1024 Ingress VLAN conversion table/MAC-based VLAN table/IP-subnet-based VLAN entries	Adds or removes C-TAG and S-TAG, shared with MAC-based/IP-subnet-based VLAN
512 Egress VLAN conversion entries	Egress VLAN conversion

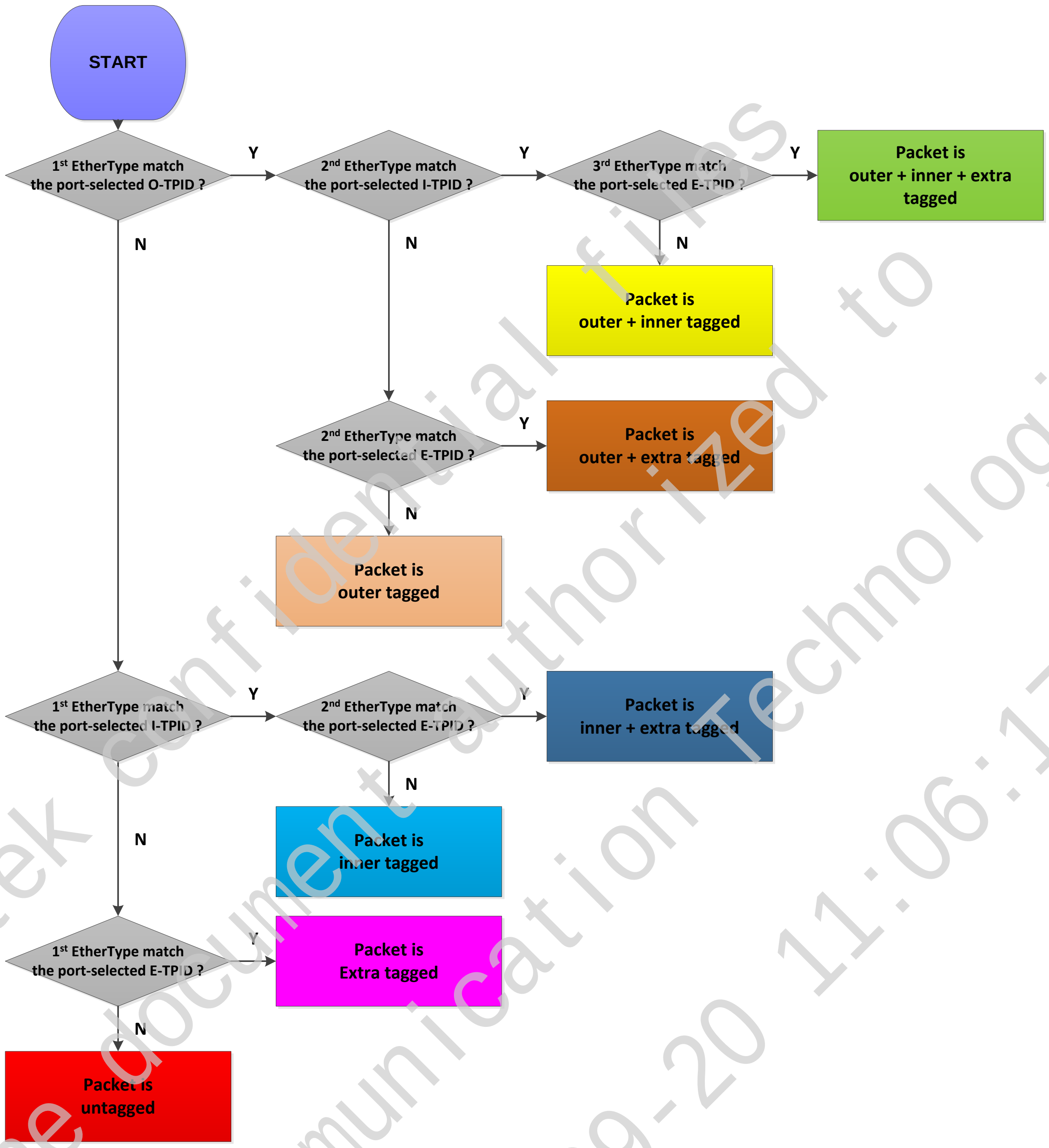
2.1.1 Tag parsing

C-TAG's TPID is defined to be 0x8100 by standard while S-TAG's TPID is 0x88A8, but considering that some special environments may need to define its own TPID (ex: 0x9100, 0x9200) and may need to support multiple uplink port(more than one ISP), so the system provides four inner TPID(I-TPID) values and four outer TPID(O-TPID) values for user to define, and per-port provides a 4-bit I-TPID mask and 4-bit O-TPID mask to indicate which set of I-TPID and O-TPID to compare,

In addition, the system provides one global E-TPID values and ETPID_CMP bit for user to define and select whether to compare the extra TPID or not.

Once a packet matches configured O-TPID/I-TPID/E-TPID, it is considered as an outer/ inner/extra tagged packet. Packets with no tags are called untagged, packets with one tag are called single tagged, and packets with two tags are double tagged.

Figure 2-2 Inner and Outer Tag Parsing


START
1st EtherType match the port-selected O-TPID ?
2nd EtherType match the port-selected I-TPID ?
3rd EtherType match the port-selected E-TPID ?
Packet is outer + inner + extra tagged
Packet is outer + inner tagged
2nd EtherType match the port-selected E-TPID ?
Packet is outer + extra tagged
Packet is outer tagged
1st EtherType match the port-selected I-TPID ?
2nd EtherType match the port-selected E-TPID ?
Packet is inner + extra tagged
Packet is inner tagged
1st EtherType match the port-selected E-TPID ?
Packet is Extra tagged
Packet is untagged
Y
N

2.1.2 Acceptable Frame Type

IEEE 802.1Q standard defines each port should provide an attribute called “Acceptable Frame Type”, this attribute is to tell the port which frame type it can receive, and attribute value could be the following:

Admit All Frames	Accept all the frame type packets
Admit Only VLAN-Tagged frames	Only accept VLAN-Tagged (doesn’t include Priority-Tagged) frames
Admit Only Untagged and Priority-Tagged frames	Only accept Untagged and Priority-Tagged frames

System per-port provides following configurations:

Table 2-1 VLAN_PORT_AFT Register

Bits	Field	Description
3	OTAG_UNTAG_ACCEPT	Permit outer VLAN untagged and priority-tagged frame.
2	OTAG_ACCEPT	Permit outer VLAN tagged frame.
1	ITAG_UNTAG_ACCEPT	Permit inner VLAN untagged and priority-tagged frame.
0	ITAG_ACCEPT	Permit inner VLAN tagged frame. 0b0: drop 0b1: forward

System doesn't provide a setting for "admit all", because which can be achieved through configuring both VLAN_PORT_AFT.ITAG_ACCEPT (VLAN_PORT_AFT.OTAG_ACCEPT) and VLAN_PORT_AFT.ITAG_UNTAG_ACCEPT (VLAN_PORT_AFT.OTAG_UNTAG_ACCEPT) to be 1(Accept) at the same time.

2.1.3 CFI Operation

System provides configurations below to trap a tagged/priority-tagged packet with CFI=1.

Table 2-2 VLAN_PORT_AFT Register

Bits	Field	Description
3:2	OCFI_ACT	Action to take if outer CFI=1. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: trap to master CPU
1:0	ICFI_ACT	Action to take if inner CFI=1. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: trap to master CPU

2.2 VLAN Assignment

In the ingress phase, system will decide the ingress inner VID(I-VID) and ingress outer VID(O-VID) respectively, depending on the Port-based VLAN setting, Protocol-and-Port-based VLAN setting, the ingress VLAN translation setting , the MAC-based VLAN setting, the IP Subnet-based VLAN setting, and the VLAN ACL module setting. After that, according to the per-port setting VLAN_PORT_FWD_CTRL, system will select either ingress I-VID or ingress O-VID as the final forwarding VID which is used to do the VLAN/L2 table lookup and forwarding. But in routing case, something is different because the forwarding VID before routing and after routing may be different.

In addition, the module such as Protocol-and-Port-based VLAN, the ingress VLAN translation, the MAC-based VLAN, the IP Subnet-based VLAN, and the VLAN ACL module can also allow assigning the priority which will be introduced in detail in the priority decision and TX remark chapter related. Here in this VLAN chapter we will only focus on the VID assignment logic explanation, but for convenience we still will list the related priority action while description.

The system should follow the flow below to determine the ingress I-VID and ingress O-VID for a received packet:

Figure 2-3 Ingress I-VID Decision

[**ori_itag_if**]: original itag existed or not
[**ori_ivid**]: original itag vid value
[**ppb_hit**]: protocal port-based vlan hit
[**ppb_ivid**]: ppb assigned ivid
[**ivc_hit**]: ingress vlan conversion hit
[**ivc_ivid**]: ivc/mb-vlan/ipsub_vlan assigned ivid
[**mb_hit**]: mac-based vlan hit
[**ipsub_hit**]: ipsubnet vlan hit
[**vacl_hit**]: vlan acl hit
[**vacl_ivid**]: vacl assigned ivid
[**int_ivid**]: ingress i-vid

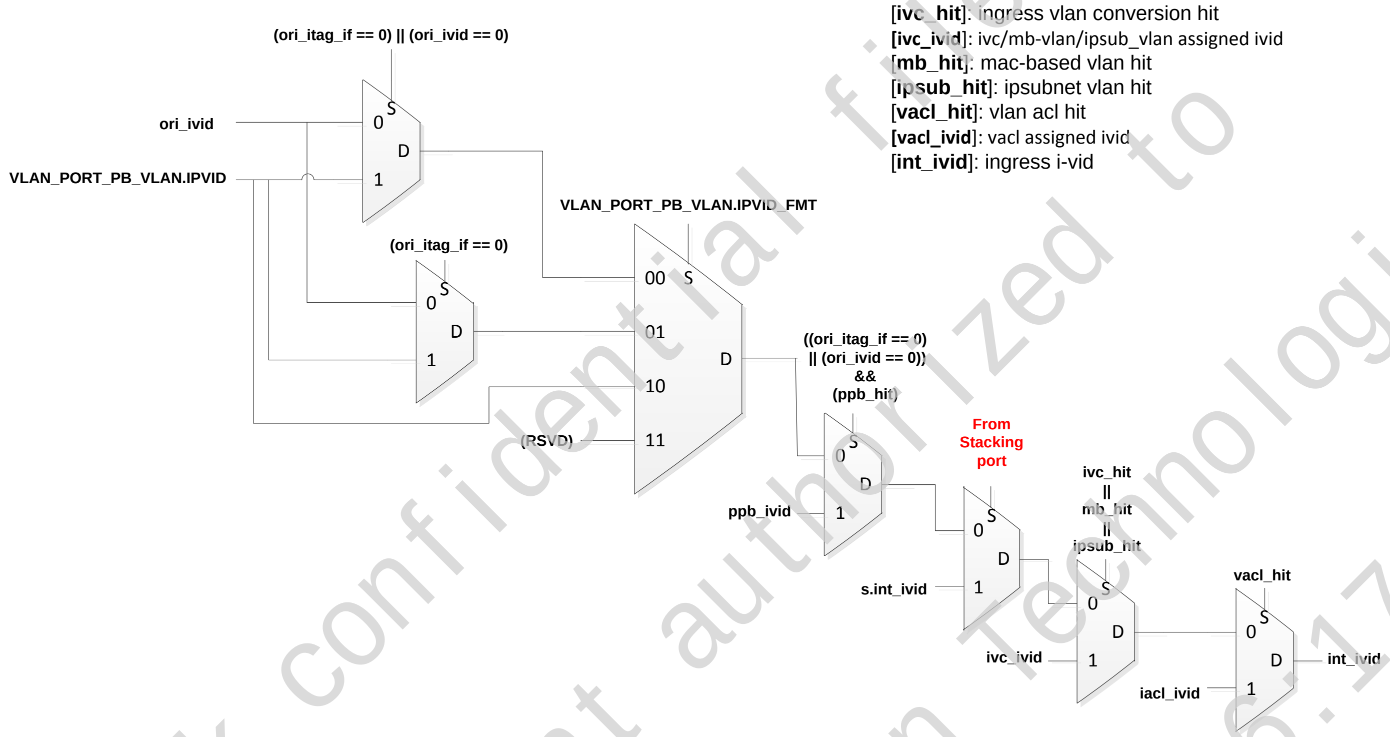
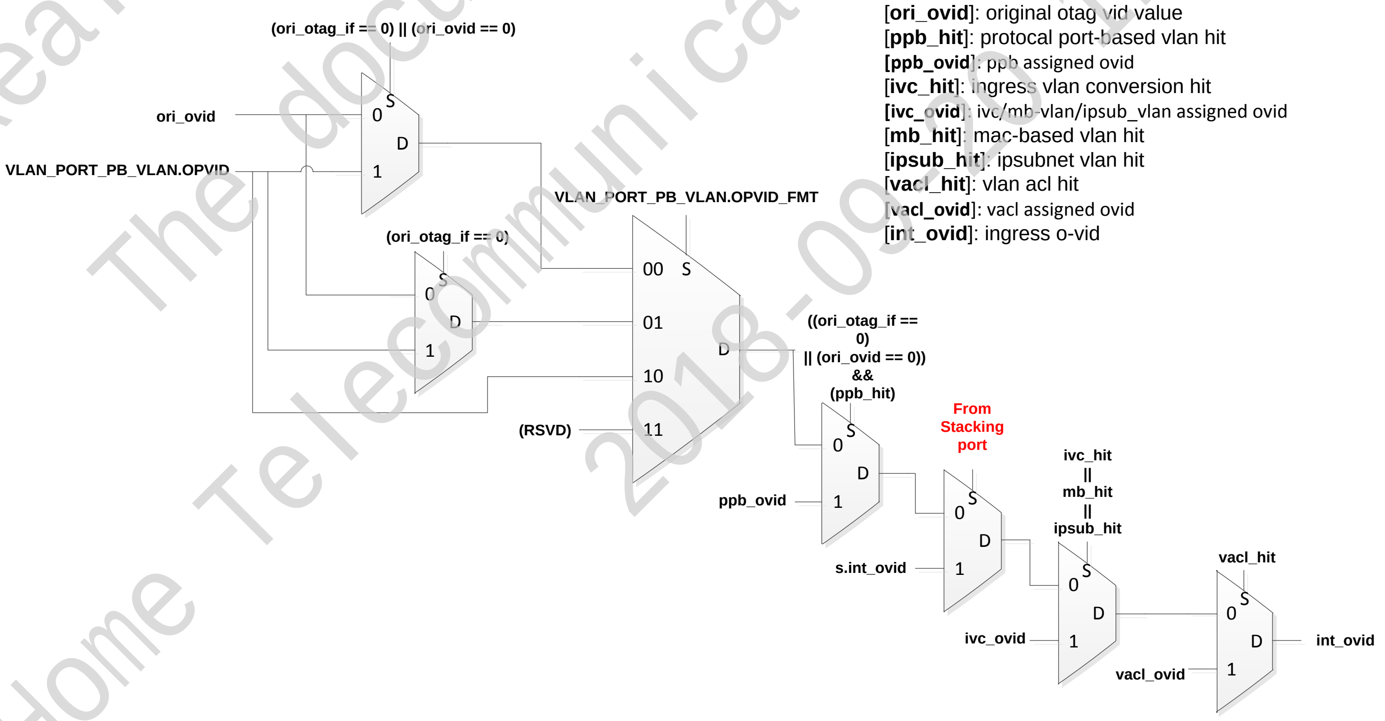


Figure 2-4 Ingress O-VID Decision

[**ori_otag_if**]: original otag existed or not
[**ori_ovid**]: original otag vid value
[**ppb_hit**]: protocal port-based vlan hit
[**ppb_ovid**]: ppb assigned ovid
[**ivc_hit**]: ingress vlan conversion hit
[**ivc_ovid**]: ivc/mb-vlan/ipsub_vlan assigned ovid
[**mb_hit**]: mac-based vlan hit
[**ipsub_hit**]: ipsubnet vlan hit
[**vacl_hit**]: vlan acl hit
[**vacl_ovid**]: vacl assigned ovid
[**int_ovid**]: ingress o-vid



2.2.1 Port-based VLAN

Each port has a Port-based VLAN configuration (OPVID_FMT/IPVID_FMT) used to control whether assigning the (OPVID/IPVID) value ONLY for the untagged packets, OR for both tagged and untagged packets. IPVID is used to give the port's inner VID while OPVID is used to give the port's outer VID.

Table 2-3 VLAN_PORT_PB_VLAN Register

Bits	Field	Description
27:16	OPVID	Port-based Outer VLAN ID
15:14	OPVID_FMT	Apply outer port-based VLAN configuration on 00: outer untagged and outer priority tagged packet 01: outer untagged packet 10: all packet(outer untagged, outer priority-tagged, outer tagged) 11: reserved
13:2	IPVID	Port-based Inner VLAN ID
1:0	IPVID_FMT	Apply inner port-based VLAN configuration on 00: inner untagged and inner priority tagged packet 01: inner untagged packet 10: all packet(inner untagged, inner priority-tagged, inner tagged) 11: reserved

2.2.2 Protocol-and-Port-based VLAN

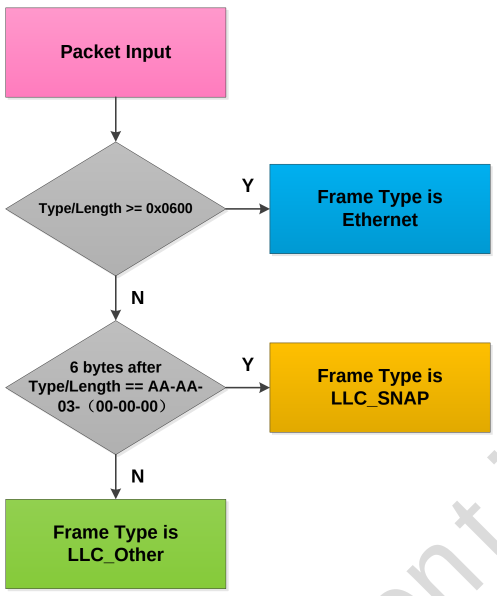
Protocol-and-Port-based VLAN provides the ability to assign a VID and priority based on packet's frame type, ether type and ingress port. Globally system provides 8 entries to compare the FRAME_TYPE & FRAME_TYPE_VALUE with every packet received. The entry format is as follows:

Table 2-4 VLAN_PPB_VLAN_SET Entry Format

Bits	Field	Description
17:16	FRAME_TYPE	Packet frame type format. 00: Unknown 01: Ethernet 10: LLC_Other 11: LLC_SNAP
15:0	FRAME_TYPE_VALUE	Frame type value.

Frame type supports three frame types: Ethernet II, LLC_Other and LLC_SNAP.

Figure 2-5 Frame Type Decision Flow



Meanwhile, each port supports eight VLAN_PORT_PPB_VLAN_SET configurations to map to eight global VLAN_PPB_VLAN_SET table. If the entry of the VLAN_PPB_VLAN_SET table is matched, the corresponding VLAN_PORT_PPB_VLAN_SET action of RX port will be referred to. The per-port VLAN_PORT_PPB_VLAN_SET action format is as follows.

Table 2-5 VLAN_PORT_PPB_VLAN_SET Entry Format

Bits	Field	Description
17	VLAN_TYPE	To indicate the VLAN type: 0b0: Inner VLAN 0b1: Outer VLAN
16	PRI_AS	Indicate to assign Priority.
15:13	PRI	Protocol-and-Port-based Priority.
12	VID_AS	Indicate to assign VLAN ID.
11:0	VID	Protocol-and-Port-based VLAN ID.

2.2.3 Ingress VLAN translation

The function of IVC (Ingress VLAN Conversion) is to convert the VLAN information before VLAN lookup based on the original VLAN information of the packet. The system supports a 2K entries Ingress VLAN conversion table shared with MAC-based and IP Subnet-based VLAN.

The 2K entries are divided into 16 blocks (each block contains 128 entries), system supports per block instead of per entry configuration to indicate the block entries are used for IVC or MAC-based or IP Subnet-based VLAN.

IVC supports search key, including port, original inner tag, original outer tag and VLAN range etc., and below is detailed definitions:

Table 2-6 IVC Search Key

Fields	Description

VALID	Valid bit
PORT_TYPE	Type of the ingress port (0 for individual port, 1 for trunk port) This is used as search key for ingress packets to determine the converted VLAN info.
PORT_ID	For individual port, to indicate the port id For trunk port, to indicate the trunk group
ORI_OTAG_IF	Original outer tag status
ORI_OPRI	Original outer priority (PCP)
ORI_OVID	Original outer VLAN ID
ORI_ITAG_IF	Original inner tag status
ORI_IPRI	Original inner priority (PCP)
ORI_IVID	Original inner VLAN ID
ORI_VID_RNG_CHK	Range check result given by 32 VID range check configurations, each bit represents a range check configuration result and can be marked off through PIE care bits.
BMSK_PORT_TYPE	Care bitmask of PORT_TYPE
BMSK_PORT_ID	Care bitmask of PORT_ID
BMSK_ORI_OTAG_IF	Care bitmask of ORI_OTAF_IF
BMSK_ORI_OPRI	Care bitmask of ORI_OPRI
BMSK_ORI_OVID	Care bitmask of ORI_OVID
BMSK_ORI_ITAG_IF	Care bitmask of ORI_ITAG_IF
BMSK_ORI_IPRI	Care bitmask of ORI_IPRI
BMSK_ORI_IVID	Care bitmask of ORI_IVID
BMSK_ORI_VID_RNG_CHK	Care bitmask of ORI_VID_RNG_CHK

IVC specifies VID actions below:

- Don’t assign (null operation)
- Assign a new I-VID with IVID directly
- Shift I-VID by adding IVID value
- Copy from the original O-VID

Table 2-7 IVC Action List

Fields	Description
OTPID_ ASSIGN	Assign O-TPID 0: Don’t assign 1: Assign new O-TPID with OTPID_IDX indexing value
OTPID_IDX	Index of the O-TPID (active when OTPID_ASSIGN = 1)
OTAG_STS_ASSIGN	Assign the O-TAG status 0: Don’t assign 1: Assign new O-TAG status with OTAG_STS value.

OTAG_STS	New outer tag status (active when OTAG_STS_ASSIGN = 1) 0: Assign outer tag status to be untagged 1: Assign outer tag status to be tagged
OPRI_ASSIGN	Assign outer priority (PCP) value 00: Don't assign(null operation) 01: Assign an outer priority with OPRI value 10: Copy from the original Inner priority 11: Reserved Note: that if the packet is inner-untagged and the action is "copy", then the action will be skipped (null operation).
OPRI	New outer priority (active when OPRI_ASSIGN = 1)
OVID_ASSIGN	Assign a new O-VID 00: Don't assign (null operation) 01: Assign a new O-VID with OVID directly 10: Shift O-VID by adding OVID value 11: Copy from the original I-VID Note: that if the packet is inner-untagged and the action is "copy", then the action will be skipped (null operation).
OVID	Outer VID value for OVID_ASSIGN using.
ITPID_ ASSIGN	Assign I-TPID 0: Don't assign 1: Assign new I-TPID with ITPID_IDX indexing value
ITPID_IDX	Index of the I-TPID (active when ITPID_ASSIGN = 1)
ITAG_STS_ASSIGN	Assign the I-TAG status 0: Don't assign(null operation) 1: Assign new I-TAG status with ITAG_STS value.
ITAG_STS	New inner tag status (active when ITAG_STS_ASSIGN = 1) 0: Assign inner tag status to be untagged 1: Assign inner tag status to be tagged
IPRI_ASSIGN	Assign inner priority (PCP) value 0: Don't assign 1: Assign an inner priority with IPRI value
IPRI	New inner priority (active when IPRI_ASSIGN = 1)
IVID_ASSIGN	Assign a new I-VID 00: Don't assign (null operation) 01: Assign a new I-VID with IVID directly 10: Shift I-VID by adding IVID value 11: Copy from the original O-VID Note: that if the packet is outer-untagged and the action is "copy", then the action will be skipped (null operation).
IVID	Inner VID value for IVID_ASSIGN using.
INTF_ID_ASSIGN	Assign L3 interface ID
INTF_ID	L3 interface ID

Each IVC entry(NOT include MAC/IP-Subnet Based VLAN) have one hit indication bit, if packet match IVC rule, corresponding hit indication bit will be set to 1,if multiple entries hit, lowest index hit indication bit will be set. When the tagged packet doesn't hit any IVC rule, it may be dropped. System provides a register to indicate the lookup miss action.

2.2.3.1 Ingress VID range check

The system supports 4 sets of IVC VID range check tables, each table has 32 dedicated range check entries. The 4 sets of rang check table which is used can be selected based on ingress port. The VID range check entry format is as below:

Table 2-8 VLAN_IGR_VID_RNG_CHK_SET Register

Bits	Field	Description
24	TYPE	Range check data type 0: original I-VID range check 1: original O-VID range check
23:12	UPPER	VID range upper bound
11:0	LOWER	VID range lower bound

2.2.4 MAC-based VLAN

System allows assigning VID based on source mac of packet using MAC-based VLAN table. MAC-based VLAN search key has ingress port, OVID_VALID, IVID_VALID. In addition, system provides an option to control per port enable or disable MAC-based or IP Subnet-based VLAN function.

MAC-based VLAN search key is as follows.

Table 2-9 MAC-based VLAN Search Key

Fields	Description
VALID	Valid bit
PORT_TYPE	Type of the ingress port (0 for individual port, 1 for trunk port) This is used as search key for ingress packets to determine the converted VLAN info.
PORT_ID	For individual port, to indicate the port id For trunk port, to indicate the trunk group
OVID_VALID	0 For outer untagged/priority-tagged packet 1 For outer tagged packet
IVID_VALID	0 For inner untagged/priority-tagged packet 1 For inner tagged packet
MAC Address	Source MAC address
BMSK_PORT_TYPE	Care bitmask of PORT_TYPE
BMSK_PORT_ID	Care bitmask of PORT_ID
BMSK_OVID_VALID	Care bitmask of OVID_VALID
BMSK_IVID_VALID	Care bitmask of IVID_VALID
BMSK_MAC_ADDRESS	Care bitmask of MAC Address

Table 2-10 MAC-based VLAN Action list

Fields	Description
FWD_ACT	Assign the forwarding action 0x0: Drop 0x1: Forward

	0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved
IGNORE_IGR_FLTR	Indicate the packet should ignore Ingress VLAN Filtering 0b0: Don't ignore (follow the decision of Ingress VLAN Filtering) 0b1: Ignore the checking
VLAN_TYPE	To indicate the VID assign for VLAN type (active when VID_ASSIGN = 1) 0b0: Inner VLAN 0b1: Outer VLAN
TPID_ ASSIGN	Assign TPID according to VLAN_TYPE 0: Don't assign 1: Assign new TPID with TPID_IDX indexing value
TPID_IDX	Index of the TPID (active when TPID_ASSIGN = 1)
TAG_STS_ASSIGN	Assign the TAG status 0: Don't assign 1: Assign new TAG status with TAG_STS value.
TAG_STS	New tag status (active when TAG_STS_ASSIGN = 1) 0: Assign tag status to be untagged 1: Assign tag status to be tagged
PRI_ASSIGN	Assign tag priority (PCP) value 0: Don't assign 1: Assign a tag priority with PRI value
PRI	New internal tag priority (active when PRI_ASSIGN = 1)
VID_ASSIGN	Assign a new VLAN-ID 0b0: Don't assign (keep original/PVID) 0b1: Assign a new VLAN-ID with VID directly
VID	VID value for VID_ASSIGN using.

2.2.5 IP Subnet-based VLAN

The IP Subnet-based VLAN is similar to MAC-based VLAN, just the main search key changing from source MAC address to source IP address. System provides an ability to per port enable or disable MAC-based or IP Subnet-based VLAN function too.

Table 2-11 IP Subnet-based VLAN Search Key

Fields	Description
VALID	Valid bit
PORT_TYPE	Type of the ingress port (0 for individual port, 1 for trunk port) This is used as search key for ingress packets to determine the converted VLAN info.
PORT_ID	For individual port, to indicate the port id For trunk port, to indicate the trunk group
OVID_VALID	0 For outer untagged/priority-tagged packet 1 For outer tagged packet

IVID_VALID	0 For inner untagged/priority-tagged packet 1 For inner tagged packet
SIP	Source IP address
BMSK_PORT_TYPE	Care bitmask of PORT_TYPE
BMSK_PORT_ID	Care bitmask of PORT_ID
BMSK_OVID_VALID	Care bitmask of OVID VALID
BMSK_IVID_VALID	Care bitmask of IVID_VALID
BMSK_SIP	Care bitmask of SIP

The action list of IP-subnet-based VLAN is the same as MAC-based VLAN. If want to know more, please refer to the MAC-based VLAN chapter above

Note

Mac/IP-Subnet based VLAN assign internal priority, and priority assigned by IVC will be a TX priority remarking source.

2.3 VLAN Forwarding and Filtering

2.3.1 Forwarding VID Decision

The system per-ingress-port and per-internal-tag-status provides a bit configuration to indicate the packet's learning and forwarding VID selected from ingress IVID or ingress OVID, and the forwarding VID is used for the VLAN table lookup and ingress/egress VLAN filtering etc.

Bear in mind that internal tag status mentioned above is different from original tag status which packets received. It means the tag status after processed by VLAN ACL, IVC and MAC-based/IP-Subnet based VLAN, etc.

Table 2-12 VLAN_PORT_FWD_CTRL Register

Bits	Field	Description
3	DBL_TAG	For double (outer + inner) tagged packets, learning & forwarding & filtering base on inner VID or outer VID
2	OTAG	For single outer packets, learning & forwarding & filtering base on inner VID or outer VID
1	ITAG	For single inner packets, learning & forwarding & filtering base on inner VID or outer VID
0	UNTAG	For untagged packets, learning & forwarding & filtering base on inner VID or outer VID 0: inner VID 1: outer VID

2.3.2 VLAN Table Format

When a packet is determined a learning and forwarding VID, the system utilizes the VID to lookup the VLAN table. The system provides a 4K entries VLAN table, therefore it can fully support 4K VLAN, and in the Q-in-Q VLAN application, the table is shared by C-VLAN and S-VLAN. The fields of VLAN table and their definitions are shown as below:

Table 2-13 VLAN Table Entry

Fiels	Description
MBR	Define the VLAN member port, the field is utilized by VLAN Ingress/Egress filtering
UNTAG	Define the egress packet should carry VLAN tag or not. System bases on this field and the tag status configuration of egress port to determine the tag status for the transmitted packet
MSTI	The system totally provides 64 Multiple Spanning Tree Instances. System will base on the MSTI to forward/drop the packet
L2_HKEY_UCAST	Select L2 lookup hash key for L2 unicast and broadcast traffic 0: VID (IVL mode) 1: FID = VLAN_CTRL.UC_SVL_FID or VLAN.SVL_FID is controlled by VLAN_CTRL.SVL_FID_SRC (SVL mode)
L2_HKEY_MCAST	Select L2 lookup hash key for L2 multicast and IP multicast traffic 0: FID = VID (IVL mode) 1: FID = VLAN_CTRL.MC_SVL_FID or VLAN.SVL_FID is controlled by VLAN_CTRL.SVL_FID_SRC (SVL mode)
L3_INTF_ID	Layer 3 interface ID
VLAN_PROFILE	Index to VLAN profile
GROUP_MASK	User-defined bits for customer's application. Pre-defined bits: Bit[0]: referred by ARP protocol storm (Protocol-Storm - ARP request) Bit[1]: referred by DHCP protocol storm (Protocol-Storm - IPv4) Bit[2]: referred by VLAN_APP_PKT_CTRL.IGMP_ACT. (Application-Packet) Bit[3]: referred by VLAN_APP_PKT_CTRL.MLD_ACT. (Application-Packet) Bit[4]: referred by VLAN_APP_PKT_CTRL.DHCP_ACT. (Application-Packet) Bit[5]: referred by VLAN_APP_PKT_CTRL.DHCP6_ACT. (Application-Packet) Bit[6]: referred by VLAN_APP_PKT_CTRL.ARP_REQ_ACT. (Application-Packet) Bit[7]: referred by VLAN_APP_PKT_CTRL.ARP_REP_ACT. (Application-Packet) Note: For per-VLAN configuration, user can use the bits to indicate the enable status of the ACL rule. (ACL can refer the bits as search key)
SVL_FID	SVL FID value

2.3.3 VLAN Ingress Filter

The field MBR looked up form VLAN Table can be used to do the ingress/egress VLAN filter check, depending on the per-port setting of VLAN_PORT_IGR_FLTR to Forward, Drop or Trap the packet.

Table 2-14 VLAN_PORT_IGR_FLTR Register

Bits	Field	Description
1:0	IGR_FLTR_ACT	Action to take for a packet that violate the VLAN ingress filtering. 0x0: Forward 0x1: Drop 0x2: Trap to local CPU 0x3: Trap to master CPU

2.3.4 VLAN Egress Filter

System provides control setting to enable or disable egress VLAN filter check. The related register is as follows.

Table 2-15 VLAN_PORT_EGR_FLTR Register

Bits	Field	Description
0	EGR_FLTR_EN	Enable VLAN egress filtering. 0x0:disable 0x1:enable

The per egress port provides a configuration for enabling or disabling the Egress Filtering, when the EGR_FLTR_EN setting of packet's outgoing port is disabled, all the traffic (L2 unicast/L2 multicast/IP multicast/broadcast) forwards to that port will ignore the Egress Filtering checking. Otherwise, system will check whether the outgoing port belongs to the forwarding VLAN to deciding to forwarding to that port not.

When Egress Filtering is enabled, system provides an extra configuration VLAN_CTRL.LEAKY to support MVR (Multicast VLAN Registration), known L2/IP4/IP6 Multicast traffic can bypass the VLAN Egress Filtering checking (other traffic is still be filtered) if VLAN_CTRL.LEAKY is enabled, the packets can be forwarded no matter whether the destination ports are in the same VLAN with ingress port.

When known multicast packet is leaked (LEAKY_EN = 1), Spanning tree egress leaking can work. When VLAN_CTRL.STP_LEAK_EN == 0, packet will follow STP Egress filter process. When VLAN_CTRL.STP_LEAK_EN == 1, for block/learn/forward port STP state, packet will not be dropped, but for disable STP status, packet will be still filtered.

Table 2-16 VLAN_CTRL Register

Bits	Field	Description
5	STP_LEAK_EN	Enable spanning tree egress leaking when known multicast packet is leakyed (LEAKY_EN = 1). System provides the configuration in case ingress port and egress ports are in different MSTI. 0: Disable 1: Enable

4	LEAK_EN	Enable leaking known L2/IP/IP6 multicast packet if the dest. port doesn't belong to the given VLAN of the packet 0: Disable 1: Enable

Note

While routing case, packet will be assigned a new forwarding vid retrieved from Next-hop entry after routing, the forwarding VID before routing is called ingress FVID, and the other one is called egress FVID. The ingress vlan filter and ingress port spanning-tree state checking will refer to ingress FVID, but the egress VLAN filter and egress port spanning-tree state checking will refer to egress FVID. If packet is not decided to do routing(bridging), forwarding VID used by ingress and egress vlan filter to look up vlan table are no difference..

2.3.5 VLAN profile

The system supports eight global VLAN profiles which includes many L2 and L3 control settings. Each VLAN can select to bind one of the 8 VLAN profiles in the VLAN entry to decide the L2/L3 forwarding action.

Table 2-17 VLAN_PROFILE_SET Register

Field	Description
L2_LRN_EN	L2 table learning behavior when receives a packet with new SMAC. 0b00: ASIC Auto Learn 0b01: learn as a suspend entry 0b10: not Learn 0b11: reserved
L2_NEW_SA_ACT	Packet forwarding behavior when receives a packet with new SMAC. 0x0: Forward 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved
IP4_MC_BDG_MODE	Bridge Mode for IPv4 multicast packets 0:MAC-based 1:IP-based
IP6_MC_BDG_MODE	Bridge Mode for IPv6 multicast packets
L2_MC_BDG_LU_MIS_ACT	L2 Multicast Packet look up miss action 0x0: Forward (Flooding to L2_UNKN_MC_FLD_PMSK) 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved

IP4_MC_L2BDG_LU_MIS_ACT	IPv4 Multicast Packet lookup miss action 0x0: Forward (Flooding to IP4_UNKN_MC_FLD_PMSK) 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved
IP6_MC_L2BDG_LU_MIS_ACT	IPv6 Multicast Packet lookup miss action 0x0: Forward (Flooding to IP6_UNKN_MC_FLD_PMSK) 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved
L2_UNKN_MC_FLD_PMSK	Unknown L2 multicast flooding portmask
IP4_UNKN_MC_FLD_PMSK	Unknown IP4 multicast flooding portmask
IP6_UNKN_MC_FLD_PMSK	Unknown IP6 multicast flooding portmask

2.3.6 VLAN application trap

In some cases, we may want a group discrete or continuous of VLANs to do the same ACL action, but limited ACL entry number is allowed. In other cases, we may want a group of VLANs to trap some common-used protocol packet, but we don't want to use ACL to cover. The VLAN application trap mechanism provided here is a good choice.

Per VLAN entry has 16 bits GROUP_MASK field which can be used as the VACL/IACL search key for VLAN-based application. The related VLAN application trap control register is as follows. For example, if ARP_REP_ACT is set to 0x1(trap to local CPU), packet received from all the VLAN with the GROUP_MASK.bit7==0x1 will be trapped to local CPU. The GROUP_MASK.bit7 can be used as the ACL key, packet from all the VLAN with the GROUP_MASK.bit7==0x1 will do the same ACL action if matched.

Table 2-18 VLAN_APP_PKT_CTRL Register

Bits	Field	Description
17:15	ARP_REP_ACT	Select the action for ARP reply packet while VLAN_GROUP_MASK[7] of the fowarding VLAN is set. 0x0: Null operation 0x1: Trap to local CPU 0x2: Copy to local CPU 0x3: Trap to master CPU 0x4: Copy to master CPU 0x5 0x7: Reserved

14:12	ARP_REQ_ACT	Select the action for ARP request packet while VLAN_GROUP_MASK[6] of the fowarding VLAN is set. 0x0: Null operation 0x1: Trap to local CPU 0x2: Copy to local CPU 0x3: Trap to master CPU 0x4: Copy to master CPU 0x5 0x7: Reserved
11:9	DHCP6_ACT	Select the action for DHCPv6 Snooping packet while VLAN_GROUP_MASK[5] of the fowarding VLAN is set. 0x0: Null operation 0x1: Trap to local CPU 0x2: Copy to local CPU 0x3: Trap to master CPU 0x4: Copy to master
8:6	DHCP_ACT	Select the action for DHCP Snooping packet while VLAN_GROUP_MASK[4] of the fowarding VLAN is set. 0x0: Null operation 0x1: Trap to local CPU 0x2: Copy to local CPU 0x3: Trap to master CPU 0x4: Copy to master CPU
5:3	MLD_ACT	Select the action for MLD Snooping packet while VLAN_GROUP_MASK[3] of the fowarding VLAN is set. 0x0: Null operation 0x1: Trap to local CPU 0x2: Copy to local CPU 0x3: Trap to master CPU 0x4: Copy to master CPU 0x5 0x7: Reserved
2:0	IGMP_ACT	Select the action for IGMP Snooping packet while VLAN_GROUP_MASK[2] of the fowarding VLAN is set. 0x0: Null operation 0x1: Trap to local CPU 0x2: Copy to local CPU 0x3: Trap to master CPU 0x4: Copy to master CPU 0x5 0x7: Reserved

The detailed packet format definition is as follows.

Table 2-19 Application Trap

Fields in VLAN Table	Packet Content Description
Bit[2] IGMP Snooping	IPv4 packet && IP protocol = 2 (IGMP)
Bit[3] MLD Snooping	IPv6 packet && IP protocol = 0x3A (ICMPv6) &&ICMPv6 type = 0x82/0x83/0x84/0x8F (one of them)
Bit[4] DHCP	UDP/IP packet && UDP source port = 67/68 (one of them) && UDP dest port = 67/68 (one of them)

Bit[5] DHCPv6	UDP/IP packet && UDP source port = 546/547 (one of them) && UDP dest port = 546/547 (one of them)
Bit[6] ARP Request	Ethertype = 0x0806 && ARP.OPCode = 1 (request)
Bit[7] ARP Reply	Ethertype = 0x0806 && ARP.OPCode = 2 (reply)

2.4 Egress VLAN translation

The Egress VLAN conversion is executed after forwarding and before transmitting. It can translate the VLAN information based on the internal status (internal tag status, internal vid, internal priority). The system provides a 1K entries Egress VLAN conversion table, where INT_OTAG_IF/INT_ITAG_IF is decided by the possible sources: original tag status, VACL assignment and IVC/MAC-based/IP subnet-Based VLAN assignment if matched, INT_OVID/INT_IVID is referred to the ingress I-VID/O-VID which descripted in VLAN assignment chapter, INT_OPRI/INT_IPRI is decided by the possible sources: original tag priority, port-based, VACL assignment and IVC assignment if matched.

Table 2-20 EVC Search Key

Fields	Description
VALID	Valid bit
PORT_TYPE	Type of the ingress port (0 for individual port, 1 for trunk port) This is used as search key for ingress packets to determine the converted VLAN info.
PORT_ID	For individual port, to indicate the port id For trunk port, to indicate the trunk group
INT_OTAG_IF	Internal (pre-egress) outer tag status
INT_OPRI	Internal (pre-egress) outer priority (PCP)
INT_OVID	Internal (pre-egress) outer VLAN ID
INT_ITAG_IF	Internal (pre-egress) inner tag status
INT_IPRI	Internal (pre-egress) inner priority (PCP)
INT_IVID	Internal (pre-egress) inner VLAN ID
INT_VID_RNG_CHK	Range check result given by 32 Internal (pre-egress) VID range check configurations, each bit represents a range check configuration result and can be marked off through mask bits.
BMSK_PORT_TYPE	Care bitmask of PORT_TYPE
BMSK_PORT_ID	Care bitmask of PORT_ID
BMSK_INT_OTAG_IF	Care bitmask of INT_OTAF_IF
BMSK_INT_OPRI	Care bitmask of INT_OPRI
BMSK_INT_OVID	Care bitmask of INT_OVID
BMSK_INT_ITAG_IF	Care bitmask of INT_ITAG_IF
BMSK_INT_IPRI	Care bitmask of INT_IPRI
BMSK_INT_IVID	Care bitmask of INT_IVID
BMSK_INT_VID_RNG_CHK	Care bitmask of INT_VID_RNG_CHK

EVC specifies VID actions below:

Table 2-21 EVC Action List

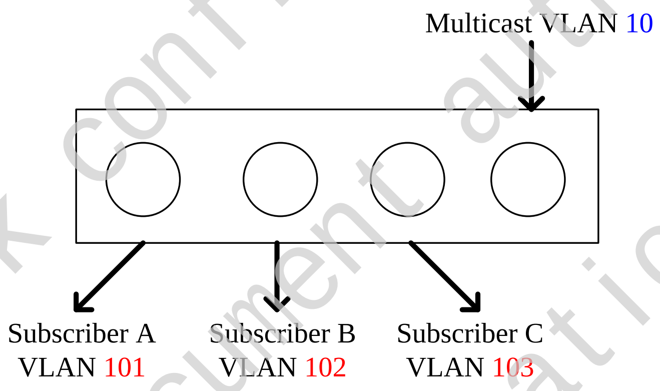
Fields	Description
OTPID_ ASSIGN	Assign egress O-TPID 0: Don't assign 1: Assign the egress O-TPID with OTPID_IDX indexing value
OTPID_IDX	Index of the O-TPID (active when OTPID_ASSIGN = 1)
OTAG_STS_ASSIGN	Assign the O-TAG status 0: Don't assign 1: Assign egress O-TAG status with OTAG_STS value.
OTAG_STS	New outer tag status (active when OTAG_STS_ASSIGN = 1) 0: Assign outer tag status to be untagged 1: Assign outer tag status to be tagged
OPRI_ASSIGN	Assign outer priority (PCP) value 00: Don't assign(null operation) 01: Assign an outer priority with OPRI value 10: Copy from the original Inner priority 11: Reserved Note: that if the packet is inner-untagged and the action is "copy", then the action will be skipped (null operation).
OPRI	New outer priority (active when OPRI_ASSIGN = 1)
OVID_ASSIGN	Assign the egress O-VID 00: Don't assign (null operation) 01: Assign the egress O-VID with OVID directly 10: Shift egress O-VID by adding OVID value 11: Copy from the internal I-VID Note that because internal I-VID is always exist,the "copy" action will never be skipped.
OVID	Outer VID value for OVID_ASSIGN using.
ITPID_ ASSIGN	Assign I-TPID 0: Don't assign 1: Assign new I-TPID with ITPID_IDX indexing value
ITPID_IDX	Index of the I-TPID (active when ITPID_ASSIGN = 1)
ITAG_STS_ASSIGN	Assign the I-TAG status 0: Don't assign(null operation) 1: Assign new I-TAG status with ITAG_STS value.
ITAG_STS	New inner tag status (active when ITAG_STS_ASSIGN = 1) 0: Assign inner tag status to be untagged 1: Assign inner tag status to be tagged
IPRI_ASSIGN	Assign inner priority (PCP) value 00: Don't assign(null operation) 01: Assign an inner priority with IPRI value 10: Copy from the internal Outer priority 11: Reserved Note that because internal Outer priority is not always exist, if internal Outer tag status is untagged, the "copy" action will be skipped
IPRI	New inner priority (active when IPRI_ASSIGN = 1)

IVID_ASSIGN	Assign a new I-VID 00: Don’t assign (null operation) 01: Assign a new I-VID with IVID directly 10: Shift I-VID by adding IVID value 11: Copy from the internal O-VID Note that because internal O-VID is always exist,the “copy” action will never be skipped.
IVID	Inner VID value for IVID_ASSIGN using.

Each EVC entry have one hit indication bit, if packet matches EVC rule, corresponding hit indication bit will be set to 1, if multiple entries hit, only lowest index hit indication bit will be set. When the tagged packet doesn’t hit any EVC rule, it should be dropped. System provides a register to indicate the default lookup-miss action. This register can help us to implement this function easily.

There is a typical application that stream server (Multicast VLAN) and subscribers (Subscriber VLAN) are in different VLANs, and a multicast stream that be forwarded to different subscriber VLANs (could be different service providers) must carry different VLAN tags, the scenario is shown as below diagram:

Figure 2-6 Multicast VLAN Converts to Different Subscriber VLANs



2.4.1 Egress VID range check

The system supports 4 set of EVC VID range check table, and each table has 32 dedicated O-VID/I-VID range check entries. The 4 sets of rang check table can be selected which to use based on egress port. The VID range check entry configuration is as below:

Table 2-22 VLAN_EGR_VID_RNG_CHK_SET Register

Bits	Field	Description
24	TYPE	Range check data type 0: internal inner VLAN ID 1: internal outer VLAN ID
23:12	UPPER	VID range upper bound
11:0	LOWER	VID range lower bound

2.5 N:1 VLAN Aggregation

System supports the N:1 VLAN Aggregation. N:1 VLAN aggregation can be set by per-port and per trunk.

Table 2-23 VLAN_PORT_L2TBL_CNVT_CTRL Register

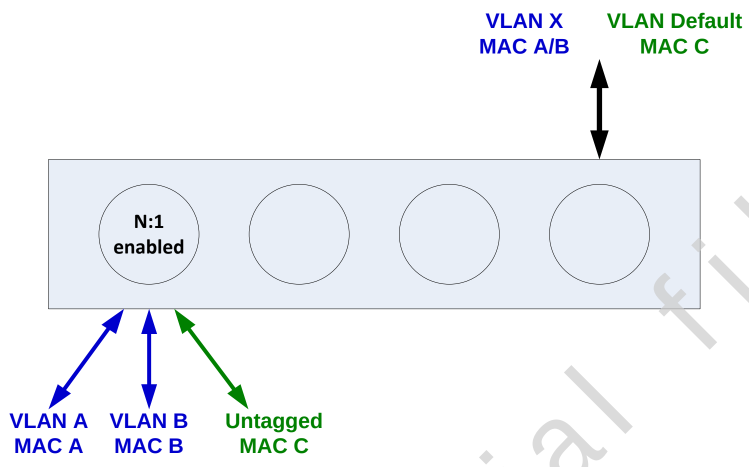
Bits	Field	Description
2	PRI_TAG_ACT	Ingress Port Priority-tagged Packet Learning VID. 0x0: 0 0x1: port-based VID Note: when it is set 1, Port-based VID will based on VID_SEL, if VID_SEL = 0, it is port-based IVID, else it is port-based O-VID.
1	VID_SEL	Port ingress learning into address table and egress conversion VID selection. 0x0: inner VID 0x1: outer VID
0	CNVT_EN	Enable Egress Port take back the VID to replace the VLAN from mac address table 0:disable 1:enable.

Table 2-24 VLAN_TRUNK_L2TBL_CNVT_CTRL Register

Bits	Field	Description
2	PRI_TAG_ACT	Ingress Port Priority-tagged Packet Learning VID. 0x0: 0 0x1: port-based VID Note: when it is set 1, Port-based VID will based on VID_SEL, if VID_SEL = 0, it is port-based IVID, else it is port-based O-VID.
1	VID_SEL	Port ingress learning into address table and egress conversion VID selection. 0x0: inner VID 0x1: outer VID
0	CNVT_EN	Enable Egress Port take back the VID to replace the VLAN from mac address table 0:disable 1:enable.

The N:1 VLAN aggregation is that all the upstream traffic from a N:1 enabled port with different customer VLANs is translated/aggregated to the same VLAN (this can be done through IVC table) before forwarding and the reverse operation is taken for downstream traffic(this can be done through EVC table). For the upstream untagged packet, it is inserted a default VLAN tag before forwarding and the default VLAN tag is removed for downstream packet if the default VLAN is observed.

Figure 2-7 N:1 VLAN Aggregation



How is it possible for a downstream traffic with same VLAN but can be translated back to its original VLAN? The trick is to record the VLAN ID for (inner or outer, based on VID_SEL) tagged unicast upstream packet to L2 table and then retrieve the VLAN ID back for replacing the VLAN for downstream packet.

When the original packet is untagged, L2-table logs the VID with VID=0 .when the original packet is priority-tag, L2-table logs VID based on VLAN_PORT_L2TBL_CNVT_CTRL.PRI_TAT_ACT. When the N-to-1 reversion action is executed, ASIC will take back the VID to replace the VLAN which selected according to VID_SEL.

If logged VID is zero, The N-to-1 reversion action will not be executed.

2.6 Egress VLAN Tagging

2.6.1 Egress VLAN Mode

The system per ingress and egress port provides an inner VLAN mode configuration IGR_ITAG_KEEP & EGR_ITAG_KEEP which are used to assign the inner tag format for the egress traffic.

The packet's inner tag content is kept (includes content and tag format) only if both IGR_P_ITAG_KEEP and EGR_P_ITAG_KEEP are configured to be "keep inner tag content". If IGR_P_ITAG_KEEP or EGR_P_ITAG_KEEP is configured to be "normal mode", the inner tag status of outgoing traffic is then determined by the VLAN untag set and egress port inner tag status configuration.

The packet's outer tag decision flow is same as inner tag status.

Table 2-25 VLAN_PORT_TAG_STS_CTRL Register

Bits	Field	Description
7:6	EGR_OTAG_STS	To indicate whether keep outer tag status on egress port. 0b00: follow the current tag status. 0b01: force to untag packet 0b10: force to tagged packet 0b11: force to priority-tag.

5:4	EGR_ITAG_STS	To indicate whether keep inner tag status on egress port. 0b00: follow the current tag status. 0b01: force to untag packet 0b10: force to tagged packet 0b11: force to priority-tag.
3	EGR_OTAG_KEEP	To indicate whether keep outer tag status on egress port. 0b0: Don't keep. 0b1: Keep.
2	EGR_ITAG_KEEP	To indicate whether keep inner tag status on egress port. 0b0: Don't keep. 0b1: Keep.
1	IGR_OTAG_KEEP	To indicate whether keep outer tag status on ingress port. 0b0: Don't keep. 0b1: Keep.
0	IGR_ITAG_KEEP	To indicate whether keep inner tag status on ingress port. 0b0: Don't keep. 0b1: Keep.

2.6.2 Egress VLAN Tagging

If a packet is NOT determined to be "keep inner tag content", then the egress inner tag status is determined by the VLAN untag set (lookup by forwarding VID and assumes the forwarding VID is based on inner VID) and egress port inner tag status configuration(VLAN_PORT_TAG_STS_CTRL.EGR_ITAG_STS). The packet is determined to be untag if its egress port is in VLAN untag member set OR its egress port inner tag setting is specified to "inner untag"

The outer tag status decision flow is the same as the inner tag status.

2.6.3 Egress VLAN Tagging TPID

In addition to provide four I-TPID values and four O-TPID values for user to define, system per-egress-port provides following configurations to determine which I-TPID/O-TPID to use for the outgoing traffic (this is useful if the switch has multiple uplink port and they are using different O-TPIDs). Per-egress port uses the configured I-TPID/O-TPID to transmit packets, and there is an option (KEEP_TPID) can be enabled to use the ingress original TPID.

Table 2-26 VLAN_PORT_TAG_STS_CTRL Register

Bits	Field	Description
5:4	OTPID_IDX	Index of port n outer TPID Point to VLAN_TAG_TPID_CTRL.OTPID
3	OTPID_KEEP	Keep original O-TPID 0: Use the per-port egress configured O-TPID. 1: Keep using the original O-TPID if the ingress packet is inner tagged.
2:1	ITPID_IDX	Index of port n inner TPID Point to VLAN_TAG_TPID_CTRL.ITPID

0	ITPID_KEEP	Keep original I-TPID 0: Use the per-port egress configured I-TPID. 1: Keep using the original I-TPID if the ingress packet is inner tagged.

3 L2 Switch & Bridge

3.1 L2 Switch

System provides a L2 MAC address table (32K entries) for unicast/multicast (including broadcast in this context) MAC L2 forwarding. Per entry can be used for unicast or multicast MAC. Unicast entry can be learned automatically by ASIC or configured by software, but the multicast entry must be written by software only.

For a specific MAC, which entry of the L2 table to learned into or configured into is depending on the HASH algorithm selected. Our system provides the new 2-left 4-way HASH algorithm, which can reduce the collision possibility.

For security concern, MAC constraint mechanism are provided to limit whole system L2 MAC address learning number, port-based L2 MAC address learning number, and another 8 entries of VLAN-based L2 MAC address learning number. Besides that, SA block and DA block are also supported to implement host block list.

Oure system supports the notification functionality to cover the applications that CPU needs to know the L2 table content change event (entry aging out, new entry learned etc.).

3.1.1 Layer2 Forwarding

For a L2 unicast/multicast packet received, our system will use the SMAC of that packet to search the L2 table (called SA Lookup Process) to check whether SMAC exists in the L2 table or not. If not, SMAC of the packet will be learned into L2 table (called SA Learn Process) automatically if new SMAC learned allowed. If it exists, aging of the entry would be updated and port move may occur. Then our system will use the DMAC of that packet to search the L2 table again to get the final outgoing port or port-mask.

3.1.2 Address Entry Format

As mention above, per L2 table entry can be used for unicast or multicast MAC. They have different entry format below. The detailed content will be introduced in the unicast and multicast section respectively.

Table 3-1 L2 Entry Format

Type	Table Content																
L2 Unicast	Valid (1)	FMT (1)	TYPE (1)	FID (12)	MAC (48)	L2_TNL (1)	Next_hop (1)	IS_TRK (1)	UNIT_ID (4)	PORT_ID (6)	Age (3)	SABLK (1)	DABLK (1)	Static (1)	Suspending (1)	TAGSTS (1)	AGG_PRI (3)	AGG_VID (12)
									Trunk ID									

L2 Multicast	Valid (1)	FMT (1)	TYPE (1)	FID (12)	MAC (48)	LOCAL _FWD (1)	Next_ hop (1)	MC_PMSK_IDX (12)	REMOTE_FWD (1)	L2_TNL_LST_IDX (13)	BPE_TYPE (1)

3.1.2.1 Unicast Entry

The detailed description for L2 unicast entry is as follows.

Table 3-2 L2 Unicast Table Entry

Fiels	Description
VALID	This entry is valid or not: 0:invalid 1:valid
FMT	Indicates entry format 0b00: L2 or port extension CB entry 0b01: OpenFlow
ENTRY_TYPE	Entry type 0b0: L2 unicast/multicast entry 0b1: PE forwarding entry
FID_RVID	FID/RVID(known as forwarding VID)
MAC	MAC address
L2_TNL	L2 tunnel indicator
NEXT_HOP	Next Hop (flag indicating can be used for routing)
IS_TRK	trunk or physical port 0b0: physical port 0b1: trunk

SPA	if IS_TRK=0
	SPA<9:6> is unit, SPA<5:0> is physical port
	if IS_TRK=1, SPA<5:0> is trunk id(0-63)

AGE	Aging Time, Range from 0 to 7
SA_BLK	Source MAC address blocking
DA_BLK	Destination MAC address blocking
STATIC	Static entry
SUSPEND	Suspend (this entry waits to be removed/modified)
TAGSTS	VLAN tag status (used by VLAN aggregation)
AGG_PRI	Aggregating priority (used by VLAN aggregation)
AGG_VID	Aggregating VLAN ID (used by VLAN aggregation)

3.1.2.2 L2 Multicast Entry

The detailed description for L2 multicast entry is as follows.

Table 3-3 L2 Multicast Table Entry

Fiels	Description

VALID	This entry is valid or not: 0:invalid 1:valid
FMT	Indicates entry format 0b00: L2 or port extension CB entry 0b01: OpenFlow
ENTRY_TYPE	Entry type 0b0: L2 unicast/multicast entry 0b1: PE forwarding entry
FID_RVID	FID/RVID(known as forwarding VID)
MAC	MAC address
LOCAL_FWD	Forward packet to the local portmask indexed by MCAST_PMSK_IDX.
NEXT_HOP	Next Hop (need to routing)
MC_PMSK_IDX	Multicast port-mask Index pointing to the port-mask table
REMOTE_FWD	Forward packet to the tunnel(s) indexed by L2_TNL_LST_IDX.
L2_TNL_LST_IDX	L2 Tunnel List Index
BPE_TYPE	BPE type indicator

The field: MC_PMSK_IDX is used to point to the port-mask table (4096 entries). This port-mask table could be referred to by other modules such as ACL/ Routing and so on. Per port-mask table entry content is just the 57 bits port mask as follows.

Table 3-4 Port-mask Table Entry

Fiels	Description
PORTMASK	57 bits port mask

3.1.3 SA Look Up & SA Learning

While a packet is received, SA Look Up process will be executed using Search KEY: (SMAC+ Forwarding VID) to verify the SMAC of the packet has already learned or not. Furthermore, if not learned, it is thought as a new SMAC packet and will do the SA Learning Process if configured to enable to learn. If learned already, system will continue checking the L2 entry content, such as SA Block field and so on.

System supports the new 2-left 4-way HASH algorithm, which can reduce the collision possibility. The 2-left means that the whole 32k L2 table is divided into two blocks, per block can select its own HASH algorithm from global two HASH algorithms independently. The control register field is as below.

Table 3-5 L2_CTRL Register

Bits	Field	Description
1:1	L2_HASH_ALGO_BLK1	L2 hash algorithm for block 1 of lookup table 0b0: algorithm 0 0b1: algorithm 1
0:0	L2_HASH_ALGO_BLK0	L2 hash algorithm for block 0 of lookup table 0b0: algorithm 0 0b1: algorithm 1

3.1.3.1 New SMAC Learning

While packet is thought as a NEW SMAC packet, system will refer to both the following per-port and per-VLAN profile setting to control learning behavior. In total, there are three types of learning behavior: [not learn], [learn as a suspend entry] and [ASIC auto learn], where [learn as a suspend entry] means that the suspend bit of the entry to be learned will be set. The "suspend" bit has effects on the following SA Look Up/DA Look UP behavior comparing with the [ASIC Auto learn] entry, which will be introduced in detail in the corresponding chapter.

Per port setting:

Table 3-6 L2_PORT_SALRN Register

Bits	Field	Description
1:0	SALRN	L2 table SA learning behavior 00b: ASIC Auto Learn 01b: learn as a suspend entry 10b: not Learn 11b: reserved

Per VLAN profile setting:

Table 3-7 VLAN_PROFILE_SET Register

Bits	Field	Description
207:206	L2_NEW_SA_LRN	L2 table learning behavior when receives a packet with new SMAC. 0b00: ASIC Auto Learn 0b01: learn as a suspend entry 0b10: not Learn 0b11: reserved

Maybe there are learning (disable/enable) settings collision between the per-port and the per-VLAN profile setting. If that occurs, the arbitration logic is as follows.

- Either of them set to [not learn], system decides not to learn;
- If both allows learning, either of them set to [learn as a suspend entry], system decides to suspend learn;
- Otherwise, ASIC will do the [ASIC auto learn] operation.

3.1.3.2 New SMAC Action

While packet is thought as a NEW SMAC packet, system also will refer to either the following per-port or per-VLAN profile setting to control NEW SMAC forwarding action.

Per port uses the field L2_SA_ACT_REF.SRC to select the NEW SMAC forwarding action from the per-port or per VLAN settings.

Table 3-8 L2_SA_ACT_REF Register

Bits	Field	Description
0:0	SRC	New SA action source select:. 0b0: reference port configure 0b1: reference vlan profile configure

Per port setting:

Table 3-9 L2_PORT_NEW_SA_FWD Register

Bits	Field	Description
2:0	NEW_SA_FWD	Packet forwarding behavior when receives a packet with new SMAC. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6 0x7: reserved

Per VLAN profile setting:

Table 3-10 VLAN_PROFILE_SET Register

Bits	Field	Description
205:203	L2_NEW_SA_ACT	Packet forwarding behavior when receives a packet with new SMAC. 0x0: Forward 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved

If SMAC entry has been found already learned, packet is not taken as NEW SMAC packet, and L2_PORT_NEW_SA_FWD/ VLAN_PROFILE_SET.L2_NEW_SA_ACT register will not be referred to naturally. But something is special if the learned entry hit is with "suspend" bit set. Either L2_PORT_NEW_SA_FWD or VLAN_PROFILE_SET.L2_NEW_SA_ACT register still will be used, but the action behavior will be changed to the following definition.

- NEW_SA_FWD/ L2_NEW_SA_ACT == 0x0, Forward;
- NEW_SA_FWD/ L2_NEW_SA_ACT == 0x1, Drop;
- NEW_SA_FWD/ L2_NEW_SA_ACT == 0x2, Drop;
- NEW_SA_FWD/ L2_NEW_SA_ACT == 0x3, Forward;
- NEW_SA_FWD/ L2_NEW_SA_ACT == 0x4, Drop;
- NEW_SA_FWD/ L2_NEW_SA_ACT == 0x5, Forward;

3.1.3.3 Port Move

System provides two independently settings to control the static entry port move and non-static(dynamic) entry port move action and learning behavior, where static entry means the entry with "Static" bit set.

For dynamic entry port move, per port setting provided is as follows.

Table 3-11 L2_ PORT_DYN_MV_ACT Register

Bits	Field	Description
2:0	ACT	Dynamic port move action. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6 0x7: reserved

Table 3-12 L2_PORT_DYN_MV_LRN Register

Bits	Field	Description
0:0	LRN	The learning behavior when dynamic port move. 0b0: not learn 0b1: learn

For example, if entry hit is learned in port1, now the packet with the same SMAC coming from port2, The L2_PORT_ DYN_MV_ACT/L2_PORT_DYN_MV_LRN of Port2 will be referred to instead of Port1. And if learned is enabled, the fields “SPA”,”AGE” and “AGG_VID” of the existed entry will be updated.

For static entry port move, system provides a global port move and learning control setting, relative to per-port settings for dynamic entry port move.

Table 3-13 L2_GLB_STT_PORT_MV_ACT Register

Bits	Field	Description
2:0	ACT	Dynamic port move action. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6 0x7: reserved

Table 3-14 L2_GLB_STT_PORT_MV_LRN Register

Bits	Field	Description
0:0	LRN	The learning behavior when dynamic port move. 0b0: not learn 0b1: learn

Different from the dynamic port move case, if learning is enabled, these fields ”AGE” and “AGG_VID” of the existed entry will be updated, but the filed “SPA” will kept still the original value.

Besides that above, system provides per-port (0-56) and per-trunk (0-25) port move forbidden enabled/disabled settings below. Port move forbidden feature is used to enable/disable the port move event happening or not.

Table 3-15 L2_PORT_MV_FORBID Register

Bits	Field	Description

0:0	EN	The port move forbidden state. 0b0: disable 0b1: enable

Table 3-16 L2_TRK_MV_FORBID Register

Bits	Field	Description
0:0	EN	The trunk port move forbidden state. 0b0: disable 0b1: enable

Table 3-17 L2_PORT_MV_FORBID_CTRL Register

Bits	Field	Description
1:0	FORBID_ACT	Port move forbidden action. 0x0: forward 0x1: drop 0x2: trap to CPU 0x3: copy to CPU

For example, if entry hit is learned (ASIC auto learnt, not suspended learnt) in port1, now the packet with the same SMAC coming from port2, if either of port1 and port2's L2_PORT_MV_FORBID/ L2_TRK_MV_FORBID setting is configured ENABLE, the L2_PORT_MV_FORBID_CTRL.FORBID_ACT will be referred to, and the existed entry will not be updated any field, including the "AGE" field.

The port move forbidden setting is not applicable to the three cases below while the learnt entry has one of the following features.

- Learnt entry with "suspend" bit set;
- Learnt entry with (SABLK=1 || DABLK=1) && Port=63;
- Learnt entry with NextHop=1 && Port=63;

3.1.3.4 SA Block

In order to implement the function of blocking some illegal SMAC address, L2_PORT_SABLK_CTRL register provides per-port option to enable or disable the SA blocking state. If configured to enabled, and the packet matched an existed entry with "SABLK" bit set, system will drop that packet but the SMAC still may be learned into the L2 table depending on the learnt related settings.

Table 3-18 L2_PORT_SABLK_CTRL Register

Bits	Field	Description
0:0	EN	SA blocking state of the port. 0b0: disable 0b1: enable

3.1.3.5 Hash Full

Although system adopts the new 2-left 4-way HASH algorithm, but the collision cannot be avoid. If that event occurs, one per-port register is provided to control trap/drop/copy that packet to the CPU or not.

Table 3-19 L2_CTRL Register

Bits	Field	Description
10:8	HASH_FULL_ACT	SA learning hash full action. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved

3.1.4 DA Look Up

System will do the DA Look Up process to find the outgoing port or port-mask. The HASH algorithm and the HASH Key may be different from SA Look Up and Learning process, depending on the packet DMAC type and switch settings. The DA Look Up process is different from each other while the DMAC is unicast, multicast and broadcast.

3.1.4.1 Unicast

While a unicast packet received, DA Look Up process will be executed always using Search KEY: (DMAC+ Forwarding VID) to verify the DMAC of the packet has already existed or not, and the HASH algorithm is the same as the SA Look Up and Learning process. Moreover, If the result is not hit, that packet is thought as a lookup missed packet and will refer to the unicast lookup missed flooding port-mask and action.

Table 3-20 L2_PORT_UC_LM_ACT Register

Bits	Field	Description
2:0	ACT	Per port L2 unicast packet lookup miss action. Note: A L2 unicast packet is deemed to KNOWN if its DMAC and forwarding VID are equal to switch's MAC and the PVID of CPU port respectively. 0x0: forward (Flooding to L2_UNKN_UC_FLD_PMSK) 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved

Table 3-21 L2_UNKN_UC_FLD_PMSK Register

Bits	Field	Description
56:0	PMSK	Flooding port mask for unknown unicast packet.

If hit, system will continue parsing the hit L2 unicast entry content, such as DA Block/Nexthop/Suspend field and so on to do the forwarding operation. For convenience of reference, here list the unicast entry format again.

Table 3-22 L2 Entry Format

Type	Table Content

L2 Unicast	Valid (1)	FMT (1)	TYPE (1)	FID (12)	MAC (48)	L2_TNL (1)	Next_hop (1)	IS_TRK (1)	UNIT_ID (4)	PORT_ID (6)	Age (3)	SABLK (1)	DABLK (1)	Static (1)	Suspending (1)	TAGSTS (1)	AGG_PRI (3)	AGG_VID (12)
									Trunk ID									

If RX port is set to enable DA block feature (L2_PORT_DABLK=0x1), and the hit entry is with “DABLK” bit set, the packet will be dropped.

Table 3-23 L2_PORT_DABLK_CTRL Register

Bits	Field	Description
0	EN	DA blocking state of the port. 0b0: disable 0b1: enable

If the hit entry with “Suspend” bit set, the packet is also thought as lookup missed and system will refer to the lookup missed action and port-mask above to forward.

If the hit entry with “Nexthop” bit set, that means the hit entry is a next hop entry used for routing, and the last 12bits field (MAC_IDX) is used as the pointer to the SMAC Table instead of AGG_VID. The more detailed information can be found in the L3 develop guide.

3.1.4.2 L2 Multicast

The DA Look Up process are different for pure L2 multicast, IPv4 multicast and IPv6 multicast.

For pure L2 multicast packet (non IP multicast), system will always use the HASH key (DMAC+ Forwarding VID) to search the L2 table, and the HASH algorithm is the same as the L2 unicast.

But for IPv4/IPv6 multicast, the HASH key is depending on the VLAN profile settings IP4_MC_BDG_MODE/ IP6_MC_BDG_MODE. If it is set to MAC-Based, the HASH key and HASH algorithm are the same as L2 multicast packet, If it is set to IP-Based, system will not search the L2 table and directly search the multicast routing table using the HASH key ({FVID+SIP+GIP},{FVID+GIP}), which will be introduced in L3 develop guide.

If the corresponding L2 entry has been hit, system will continue parsing the hit L2 multicast entry content, such as FWD_IDX/NextHop field and so on to do the forwarding operation. For convenience of reference, here list the multicast entry format again too.

Table 3-24 L2 Entry Format

Type	Table Content										
L2 Multicast	Valid (1)	FMT (1)	TYPE (1)	FID (12)	MAC (48)	LOCAL_FWD (1)	Next_hop (1)	MC_PMSK_IDX (12)	REMOTE_FWD (1)	L2_TNL_LST_IDX (13)	BPE_TYPE (1)

If the hit entry with “Nexthop” bit set, that means the hit entry is a next hop entry used for routing.

Moreover, no matter whether the packet is L2 unicast,IPv4 multicast or IPv6 multicast, if no multicast entry hit, that packet is thought as a lookup missed packet and will refer to the multicast lookup missed flooding port-mask and action field stored in VLAN profile. The more detailed information, please refer to the VLAN develop guide.

Table 3-25 VLAN_PROFILE_SET Register

Bits	Field	Description

200:198	L2_MC_BDG_LU_MIS_ACT	L2 Multicast Packet look up miss action 0x0: Forward (Flooding to L2_UNKN_MC_FLD_PMSK) 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved
197:195	IP4_MC_L2BDG_LU_MIS_ACT	IPv4 Multicast Packet lookup miss action 0x0: Forward (Flooding to IP4_UNKN_MC_FLD_PMSK) 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved
194:192	IP6_MC_L2BDG_LU_MIS_ACT	IPv6 Multicast Packet lookup miss action 0x0: Forward (Flooding to IP6_UNKN_MC_FLD_PMSK) 0x1: Drop 0x2: Trap to local CPU 0x3: Copy to local CPU 0x4: Trap to master CPU 0x5: Copy to master CPU 0x6~0x7: Reserved
184:128	L2_UNKN_MC_FLD_PMSK	Unknown L2 multicast flooding portmask
120:64	IP4_UNKN_MC_FLD_PMSK	Unknown IP4 multicast flooding portmask
56:0	IP6_UNKN_MC_FLD_PMSK	Unknown IP6 multicast flooding portmask

3.1.4.3 Broadcast

If received packet belongs to broadcast will DMAC=0xFFFFFFFFFFFF, the DA Look Up process is similar to the multicast flow. But if lookup missed, system will refer to the independently broadcast lookup missed flooding port-mask. System does not provide the lookup missed action for broadcast packet.

Table 3-26 L2_BC_FLD_PMSK Register

Bits	Field	Description
56:0	PORTMASK	Flooding port mask for broadcast packet.

3.1.5 Age & Flush

3.1.5.1 Age

Per L2 entry has 3 bits Age field to indicate the entry is aged out or not. Per port/trunk can be controlled to enable aging out or not, the corresponding registers are L2_PORT_AGE_CTRL/ L2_TRK_AGE_CTRL.

If it is enabled, the field "Age" of each entry will decrease by 1 in the unit of AGE_UIT * 0.12024 seconds. The field "Age" of new entry learnt into L2 table will always be 7, but for suspended entry learnt it will be the value of L2_AGE_CTRL.SUS_AGE_MAX, which can be used to reduce the aging out time if that set to less than 7. When "Age" is decreased to 0, the next round would flush the entry (set valid bit to 0), so the aging time of a non-suspend entry is 7* AGE_UIT * 0.12024 ~ 8* AGE_UIT * 0.12024.

The aged out entry (next period after "Age" becoming 0) with "SABLK", DABLK", "Nexthop" and "Static" bit not set will become invalid. Otherwise, the aged out entry still will be valid, but the "SPA" will be updated to 63. If hit entry with the field SPA equal to 63 while forwarding, packet is thought as the lookup missed packet and will refer to the lookup missed action and flooding port mask.

Table 3-27 L2_PORT_AGE_CTRL Register

Bits	Field	Description
0:0	AGE_EN	The port age state. 0b0: disable 0b1: enable

Table 3-28 L2_TRK_AGE_CTRL Register

Bits	Field	Description
0:0	AGE_EN	The trunk age state. 0b0: disable 0b1: enable

Table 3-29 L2_AGE_CTRL Register

Bits	Field	Description
24:22	SUS_AGE_MAX	The Age Value when learning a suspend entry : 0b0: not do suspend learning 0b1-7: age value when learning a suspend entry
20:0	AGE_UNIT	The granularity of aging bits in L2 table. Aging bits reduce one every AGEUNIT* 0.1024 seconds Disable aging when AGE_UNIT == 0.

3.1.5.2 Link Down Flush

If the designated port is linked down, the entries learnt on that port are invalid now. In order to offload CPU, system provides the help to automatically invalidate the L2 entry with SPA becoming linked down.

The global setting enabling the linked down flush feature or not is as follows.

Table 3-30 L2_CTRL Register

Bits	Field	Description
7:7	LINK_DOWN_P_INVLD	Invalidates the L2 entries whose associated ports are link down 0b0: disable 0b1: enable

3.1.5.3 Table Flush

Besides the link down flush feature, system also supports another more flexible mechanism to flush/replace the entries by port, by VLAN and so on. The control register is L2_TBL_FLUSH_CTRL.

Table 3-31 L2_TBL_FLUSH_CTRL Register

Bits	Field	Description
60:60	STS	It is used to trigger ASIC to start the flush/replace action. It will be auto cleared after the flush/replace process is finished. 0b0: finish 0b1: busy
59:59	ACT	Flush or replace of entry. 0b0: flush 0b1: replace
58:58	FVID_CMP	Whether compare FID/VID. 0b0: disable 0b1: enable
57:57	AGG_VID_CMP	Whether compare AGG_VID 0b0: disable 0b1: enable
56:56	PORT_CMP	Whether compare PORT_ID. 0b0: disable 0b1: enable
55:54	ENTRY_TYPE	The detail action type to take. 0x0: all L2 Unicast Entry 0x1: L2 dynamic and SABLK & DABLK Entry 0x2: NextHop Etnry 0x3: clear AGG_VID
53:43	PORT_ID	The port ID to be compared. <10:10>: trunk or physical port: 0b0: physical port 0b1: trunk In physical port, <5:0> is port ID and <9:6> is unit ID. In trunk port, <6:0> is trunk ID and <9:7> is reserved.
42:32	REPLACING_PORT_ID	Replacing port ID. <10:10>: trunk or physical port. 0b0: physical port 0b1: trunk In physical port, <5:0> is port ID and <9:6> is unit ID. In trunk port, <6:0> is trunk ID and <9:7> is reserved. Note: This field is meaningful only when ACT = "replace".
31:20	FVID	The FID/VID to be compared and replaced/ flushed.
19:0	AGG_VID	The AGG_VID to be compared.

3.1.6 MAC Constraint

System provides system, per-port based and 32 VLAN based learning constraint, When a port receives a packet with new SMAC, ASIC will check all the learned numbers of the system, per-port, and VLAN at the same time. When any one of them reaches its own constraint number first, the ASIC will stop learning this MAC. Then apply the corresponding constraint action to this packet, if more than one constraint number overflow at the same time, the action priority will be: Drop > Trap > Copy > Forward. It should be noted that the global/per-port/per-VLAN learning constraint only takes effect while the packet is allowed to learn by the switch settings.

3.1.6.1 System Mac Constraint

System provides global register L2_LRN_CONSTRT_CTRL & L2_LRN_CONSTRT_CNT. If the L2_LRN_CONSTRT_CNT.LRN_CNT counter counts over the L2_LRN_CONSTRT_CTRL.CONSTRT_NUM, the packet with new SA should not be learned and refer to the action of L2_LRN_CONSTRT_CTRL.ACT.

Table 3-32 L2_LRN_CONSTRT_CTRL Register

Bits	Field	Description
18:3	CONSTRT_NUM	Learning constraint number of System. 0x0: never learning 0x1_0xFFFE: learning constraint number 0xFFFF: unlimited
2:0	ACT	Learning constraint action of system. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved

Table 3-33 L2_LRN_CONSTRT_CNT Register

Bits	Field	Description
15:0	LRN_CNT	Learnt counter of system.

3.1.6.2 Port Mac Constraint

System provides per-port/per-trunk 15 bit CONSTRT_NUM and 15 bits LRN_CNT register to limit and display L2 leant number. Whenever ASIC learns a SMAC from a port, it increases LRN_CNT counter of the port. If the counter counts over the CONSTRT_NUM, the new SMAC will not be learnt and takes action according to L2_PORT_LRN_CONSTRT.ACT or L2_TRK_LRN_CONSTRT.ACT.

Table 3-34 L2_LRN_PORT_CONSTRT_CTRL & L2_LRN_TRK_CONSTRT_CTRL Register

Bits	Field	Description
18:3	CONSTRT_NUM	Learning constraint number of System. 0x0: never learning 0x1_0x7FFE: learning constraint number 0x7FFF: unlimited
2:0	ACT	Learning constraint action of system. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved

Table 3-35 L2_LRN_PORT_CONSTRT_CNT/ & L2_LRN_TRK_CONSTRT_CNT Register

Bits	Field	Description
15:0	CONSTRT_NUM	Learnt counter of system.

3.1.6.3 VLAN Mac Constraint

System also provides a table (32 entries) for learning constraint based on VLAN (or VLAN + port). The table entry definition is as follows. For each packet, if the CONSTRT_NUM is counted more than the CONSTRT_NUM, the global L2_VLAN_CONSTRT_CTRL.ACT action will be referred to.

Table 3-36 L2_LRN_VLAN_CONSTRT_ENTRY Register

Bits	Field	Description
43:32	FVID	The FVID to be compared.
22:16	PORT_ID	The port ID to be compared. <6:6>: trunk or physical port. 0b0: physical port 0b1: trunk port <5:0>: port Id or trunk ID which is depends on <6:6>
15:0	CONSTRT_NUM	Learning constraint number of VLAN learning constraint entry. 0x0: never learning 0x1_0x7FFE: learning constraint number 0x7FFF: unlimited

Table 3-37 L2_LRN_VLAN_CONSTRT_CNT Register

Bits	Field	Description
15:0	CONSTRT_NUM	Learnt counter of system.

Table 3-38 L2_VLAN_CONSTRT_CTRL Register

Bits	Field	Description
2:0	ACT	Learning constraint action of system. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved

3.1.7 Source Port Filter

Source port filter check is to prevent the L2 traffic forwarding back to the RX port. Per port register is provided to enable the filter check or not. But the filter check has no effect on the routing packet.

Table 3-39 L2_SRC_P_FLTR Register

Bits	Field	Description
0:0	SRC_FLTR_EN	Decides whether filter the ingress port from outgoing Port-mask. 0b0: disable 0b1: enable

3.1.8 L2 Notification

For applications that CPU needs to know the L2 table on-the-fly, L2 notification is supported to notify CPU when the table is adding or deleting entry. Two methods: NIC mode and packet mode are provided for L2 notification. To enable L2 notification function, NTFY_EN should be configured as enabled. NIC mode and packet mode are selected by NTFY_DST_TYPE.

The events on L2 table that would trigger notifications are:

- Aging-out
- Link-down flush (if LD_FLUSH_NTFY_EN = Enabled)
- New learn
- Port move
- TAGSTS/AGG_PRI/AGG_VID changed (if TAGSTS_NTFY_EN = Enabled)
- Table hash full (8 entries full, if HASHFULL_NTFY_EN = Enabled)

The L2 entry types to trigger notifications are select-able:

- Dynamic entries (if DYN_NTFY_EN = Enabled)
- Static entries (if STTC_NTFY_EN = Enabled)
- DA/SA block entries (if DASABLK_NTFY_EN = Enabled)
- Suspend entries (if SUS_NTFY_EN = Enabled)

Table 3-40 L2_NTFY_CTRL Register

Bits	Field	Description
20:19	NTFY_DST_TYPE	Type of destination CPU 0b00: send to local CPU 0b01: send to master CPU 0b10: send to all CPU 0b11: NIC mode
18:18	HASHFULL_NTFY_EN	Hash full event notification enabled or disabled. 0: disable 1: enable
17:17	LD_FLUSH_NTFY_EN	Link down flush entry event notification enable or disable. 0: disable 1: enable
16:16	TAGSTS_NTFY_EN	Tag status or AGG_PRI or AGG_VID modification event notification enable or disable. 0: disable 1: enable
15:15	SUS_NTFY_EN	Enables suspend entry new learn event notification. 0b0: disable 0b1: enable
14:14	DASABLK_NTFY_EN	Enables SA/DA BLK entry notification. 0b0: disable 0b1: enable
13:13	STTC_NTFY_EN	Enables static entry notification. 0b0: disable 0b1: enable
12:12	DYN_NTFY_EN	Enables dynamic entry new learn event notification. 0b0: disable 0b1: enable
0:0	NTFY_EN	Enable entry notification. Defaultly, dynamic entry only. (include NEXTHOP, exclude SUSPEND entry) 0b0: disable 0b1: enable

3.1.8.1 NIC Mode

When NTFY_DST_TYPE is configured to 3, L2 notification would work in NIC mode. In NIC mode, hardware would write L2 events to a memory buffer which could be read by CPU. CPU should pre-allocate a descriptor ring and Notification buffer (NBuf) in memory.

The content of a descriptor entry is as follows:

Table 3-41 Descriptor Entry Format

Name	Description	Bits
ADDRESS	The pointer to the NBuf. It must be 4 Byte alignment.	30
WRAP	Indicating the end of the descriptor ring.	1
OWN	Indicating this entry is owned by CPU or Switch.	1

The content of a NBuf entry is as follows:

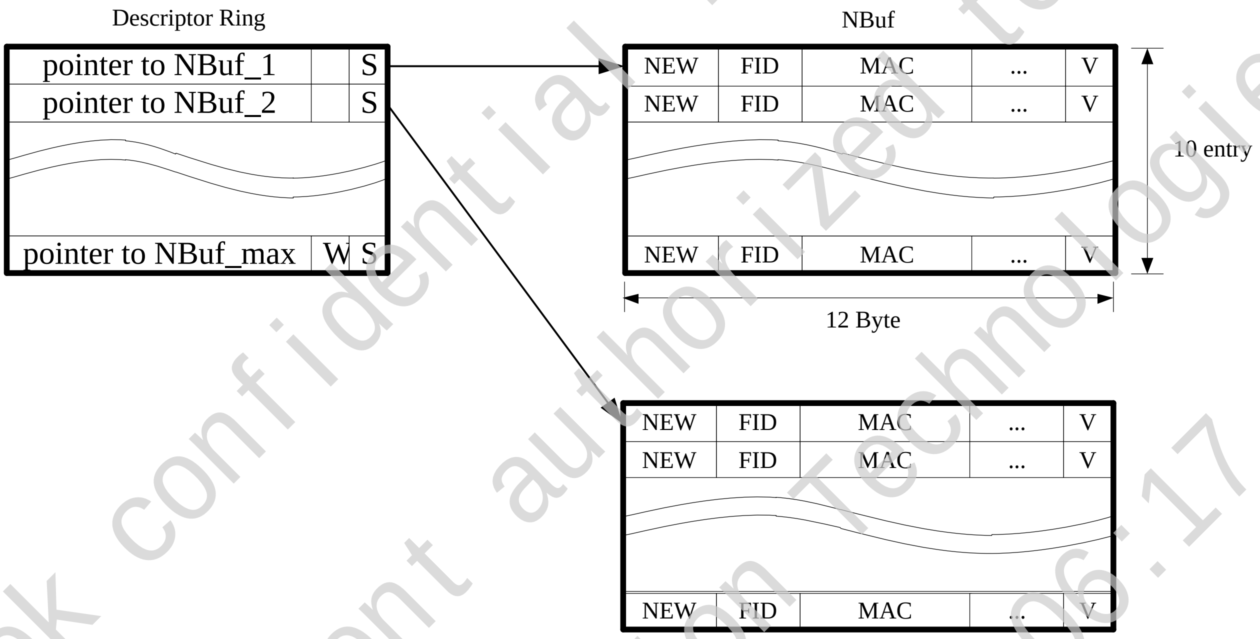
Table 3-42 NBuf Entry Format

Name			Description			Bits		
TYPE			0: Aging out/link-down flush 1: New learn 2: Port Move/TAGSTS/AGG_PRI/AGG_VID 3: Hash Full			2		
FID			The FID of adding/deleting entry			12		
MAC			The MAC address of adding/deleting entry			48		
IS_TRK			0: normal port 1: trunk port			1		
SPA			NEW = 0 or 1 or 3: The source logical port of adding/ deleting entry NEW = 2: The source logical port after moving			10		
SUS			The Suspend bit of adding/deleting entry			1		
STATIC			The Static bit			1		
SABLK			The SABLK bit			1		
DABLK			The DABLK bit			1		
NEXTHOP			The NEXTHOP bit			1		
L2_TNL			Indicates whether this event content is L2 tunnel unicast message 0: L2 unicast or CB unicast message 1: L2 tunnel unicast message			1		
TAGSTS	ECID_EXT	RSVD	The TAGSTS of adding/deleting entry	E-CID extension	Reserved	1	8	4
AGG_PRI			The AGG_PRI of adding/deleting entry			3		
AGG_VID	ECID_BAS	L2_TNL_IDX	The AGG_VID of adding/deleting entry	E-CID base	L2 tunnel index	12	12	12

Reserved	E	RSVD	Reserved		Reserved	4		4
VALID			Indicating this is a valid entry			1		
Reserved			Reserved			5		

When initializing, the descriptor points to the NBuf and the starting address of descriptor ring is pointed by L2_NTFY_RING_BASE_ADDR.

Figure 3-1 Descriptor Ring And NBuf of L2 Notification NIC Mode



Whenever L2 table has changed and need to notify CPU, ASIC writes the modification into the NBuf according to the pointer in the descriptor and change the OWN bit in descriptor to CPU own, then interrupt CPU and update L2_NTFY_RING_CUR_ADDR to next available ring entry. Once CPU notices this interrupt, it reads the Ring and NBuf and modifies the Ring.OWN bit to Switch own. If ASIC has written the last entry (WRAP = 1) and still have notifications to write, it will find the next available one from the top of ring. If all descriptor in ring are used up, ASIC should make an NBuf-run-out interrupt.

Table 3-43 L2_NTFY_RING_CUR_ADDR Register

Bits	Field	Description
31:0	CUR_ADDR	Current entry address of descriptor ring

There is a detail that before notifications are DMA to NBuf, it is stored in a local FIFO with 2K entries size. If ASIC's local buffer runs out, it makes a LocalBuf-run-out interrupt. If software wants to recycle the memory space used by descriptor ring and NBuf, it should wait local FIFO empty which could be confirmed by checking by FIFO_EMPTY register.

Table 3-44 L2_NTFY_IF_INTR_STS Register

Bits	Field	Description

1:1	NTFY_BUF_RUN_OUT	Pending status of the NTFY_BUF_RUN_OUT interrupt.
0:0	LOCAL_NTFY_BUF_RUN_OUT	Pending status of the LOCAL_NTFY_BUF_RUN_OUT interrupt.

Table 3-45 L2_NTFY_NIC_FIFO_STS Register

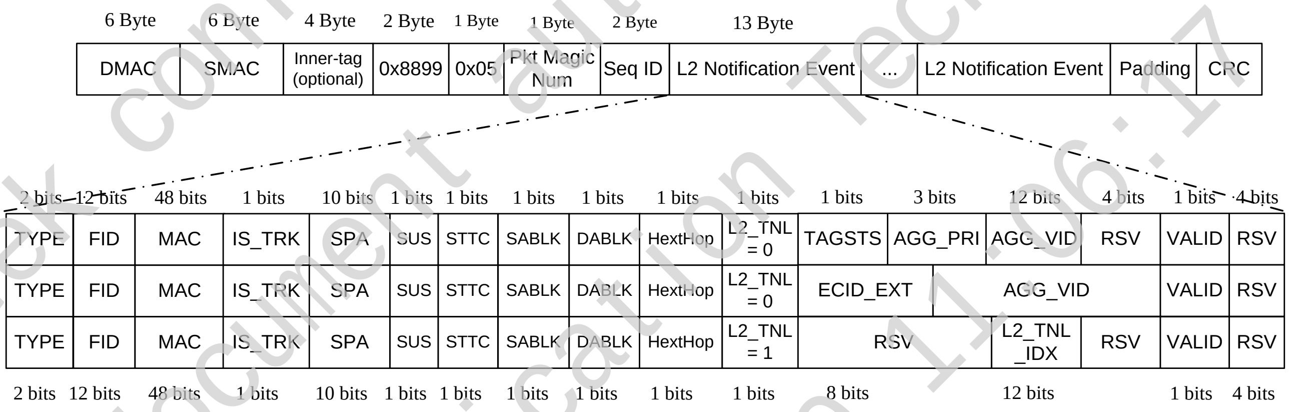
Bits	Field	Description
24:24	FIFO_EMPTY	Decides whether filter the ingress port from outgoing Port-mask. 0b0: disable 0b1: enable

3.1.8.2 Packet Mode

When NTFY_DST_TYPE is configured to 0 to 2, L2 notification would work in packet mode. In packet mode, event contents would be packaged to a packet and send to local or remote CPU (for stacking system).

The frame format is as follows:

Figure 3-2 Frame Format Of L2 Notification Packet Mode



DMAC and SMAC are decided by L2_NTFY_PKT_MAC configuration. VLAN tag status is optional which refers to L2_NTFY_PKT_CTRL.CTAGIF. Packet magic number field is controlled by L2_NTFY_PKT_MAGIC_NUM.NUM, and if the magic number if packet is different from the register, the packet would be dropped. Sequence number is incremental every packet sent, and the value is reset to 0 when NTFY_EN is set to disabled.

In packet mode, when a L2 event occurs, the event would be store in a MAC FIFO buffer with 1K event entries size. Like NIC mode, if FIFO is full, NBuf-run-out interrupt would be triggered. FIFO empty status could be read from L2_NTFY_PKT_FIFO_STS.FIFO_EMPTY.

If one of the following rules is satisfied, a notification packet would be generated and sent to CPU:

1. No event occurs for L2_NTFY_PKT_TIMEOUT.TIMEOUT (us) after previous event. In this situation, the frame size (include CRC) would be padding to L2_NTFY_PKT_CTRL.MIN_PKT_LEN if necessary.

2. The number of accumulated event entries is larger or equal to L2_NTFY_PKT_CTRL.MAX_EVENT. In this situation, the frame is sent without padding. Software should make sure the max event size is larger than MIN_PKT_LEN.

Table 3-46 L2_NTFY_PKT_MAC Register

Bits	Field	Description
111:64	DMAC	The DMAC address in a notification packet.
47:0	SMAC	The SMAC address in a notification packet.

Table 3-47 L2_NTFY_PKT_MAGIC_NUM Register

Bits	Field	Description
7:0	NUM	L2 notification magic number in packet mode.

Table 3-48 L2_NTFY_PKT_CTRL Register

Bits	Field	Description
14:14	ITAG_IF	Decides whether contains inner tag.
13:7	MAX_EVENT	The max events contained in a notification packet.
6:0	MIN_LEN	The min length of a notification packet.

Table 3-49 L2_NTFY_PKT_TIMEOUT Register

Bits	Field	Description
31:0	TIMEOUT	Timeout to send notification packet. (unit: us).

3.2 Spanning Tree

Spanning tree is a Layer 2 link management protocol that provides path redundancy while preventing loops in the network. For a Layer 2 network to function properly, only one active path can exist between any two stations. The system provides a number of hardware features to support IEEE 802.1D STP, IEEE 802.1W RSTP, and IEEE 802.1s MSTP which includes per port per instance spanning port state configuration, the VLAN ID to instance mapping in 4K VLAN table.

3.2.1 Spanning Tree State Configuration

The system provides 128 instances of spanning tree port state configuration. Each instance contains port states of port 0 to port 55 (STATE_PORT0 to STATE_PORT55), and the state value could be disabled(0), blocking/listening(1), learning(2), forwarding(3) which can map to STP/ RSTP/ MSTP states as following table.

Table 3-50 Definition of FCS

STP State Mapping	RSTP State Mapping	MSTP State Mapping	STATE_PORT Configuration
Disabled	Disabled	Disabled	Disabled(0)
Blocking/Listening	Discard	Discard	Blocking/Listening(1)

Learning	Learning	Learning	Learning(2)
Forwarding	Forwarding	Forwarding	Forwarding(3)

The port states have the following behaviors:

Table 3-51 Port State Description

Port State	Rx Packet	Rx BPDU (RMA Forward)	Rx BPDU (RMA Trap)	CPU Tx Packet(without bypass-STP)	CPU Tx Packet (bypass-STP)	Learn Address	Forward Frame
Disabled	No	No	No	No	No	No	No
Blocking/ Listening	No	No	Yes	No	Yes	No	No
Learning	No	No	Yes	No	Yes	Yes	No
Forwarding	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Note

”bypass-STP” is a field from CPU tag (from CPU_TX_TAG.BP_STP). Detail please refers to CPU tag developer guide.

- 00b – Disabled:
  - All packets (including RMA/BPDUs) must be dropped on ingress/egress port.
  - Do not learn the SA address.
  - Do not transmit any packets on the egress port even with bypass-STP.
- 01b – Blocking/Listening
  - Only RMA/BPDU and RLDP/RLPP packets may/could be sent to the CPU, the other packets are dropped.
  - Do not learn the SA address.
  - Do not transmit the packet out without bypass-STP.
  - If RMA configuration specifies the particular packet to bypass the STP port state, then the packet can pass the STP ingress filter. Detail please refers to RMA developer guide. If RLDP/RLPP set trap to CPU, it can always bypass ingress STP filter.
- 10b – Learning
  - Only RMA/BPDU and RLDP/RLPP packets may/could be sent to the CPU, the other packets are dropped after Learning Process.
  - Do not transmit the packet out without bypass-STP.
  - If RMA configuration specifies the particular packet to bypass the STP port state, then the packet can pass the STP ingress filter. Detail please refers to RMA developer guide. If RLDP/RLPP set trap to CPU, it can always bypass ingress STP filter.

- 11b – Forwarding
  - All packets can be forwarded, and the SA address would be learnt.

Table 3-52 MSTI Table

Field	Description
STATE_PORT0 ~ STATE_PORT55	STP Port State 0b00: Disabled State 0b01: Blocking/Listening State 0b10: Learning State 0b11: Forwarding State

3.2.2 Spanning Tree Instance

The system supports 128 instances of spanning tree port states, and the instances could be mapped by VLAN to support MSTP per port per VLAN configuration. The field MSTI of VLAN entry in 4k VLAN table decides the mapped instance ID (detail of configuring MSTI please refer to VLAN developer guide). For example, if MSTI of VLAN 100 is configured value 1, the spanning tree port states of instance 1 would take effect to affect the packets learning, receiving, and forwarding behavior of VLAN 100.

For the VLAN blink protocol STP and RSTP, only instance 0 (CIST) is needed to configured and the port state should take effect to all VLANs. To reduce the cost when the system switching between VLAN aware spanning tree and VLAN blink spanning tree, global register field MSTI_MODE is provided to decide the mapped instance ID of VLANs. If MSTI_MODE is normal mode (value=0), the mapped instance ID would follow MSTI value which suits the behavior of MSTP; if MSTI_MODE is force 0 mode (value=1), the mapped instance ID of all VLANs would be forced 0 which suits the behavior of STP and RSTP. By configuring MSTI_MODE, software needs not modifying MSTI of 4k VLAN table when user changes spanning tree mode from MSTP to STP or RSTP.

Table 3-53 ST_CTRL Register

Bits	Field	Description
0:0	MSTI_MODE	Indicates that instance ID is either from FID_MSTI of VLAN table or force using 0 (CIST). 0b0: normal mode (from VLAN table) 0b1: force mode (always be 0) Note: The configuration only affects MSTI but not FID.

3.3 Traffic Isolation

3.3.1 Traffic Isolation Overview

Traffic isolation separates all the traffic (unicast, multicast, and broadcast) in layer 2 to provide traffic interference and bandwidth utilization. The system provides two type isolation modules, Port-based Isolation and VLAN-based Isolation. The details are as below.

3.3.2 Port-based Isolation

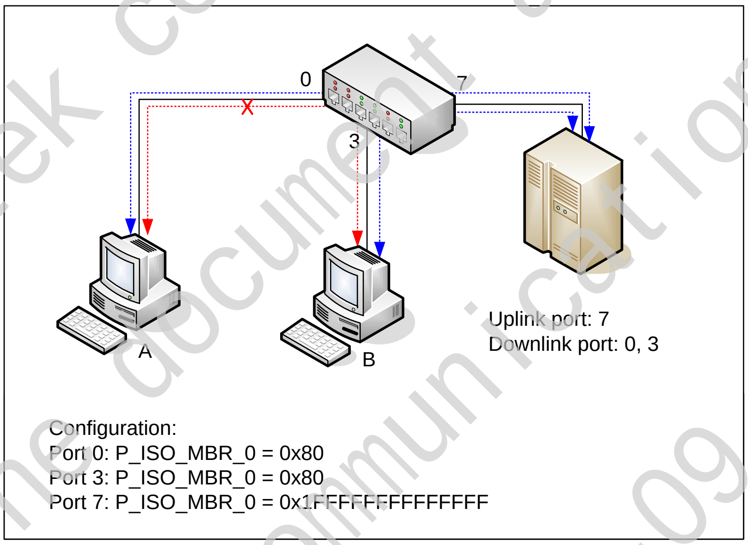
When hosts are connected to router through switch ports, they can utilize Port Isolation to force the communication between hosts through the router. Port isolation is a mechanism to provide a CPU configurable destination port mask for an ingress port. For each ingress port, the device provides a port isolation port mask configuration defined in Figure 3-3. Bit value 1 of the port mask means the port can communicate with each other. Bit value 0 is that the port can't forward any packet. No matter the received packets are known/unknown Unicast, known/unknown Multicast, Broadcast packets. All of the packets are limited by the configuration of Port Isolation port mask in forwarding.

Table 3-54 PORT_ISO_CTRL Register

Bits	Field	Description
63:7	P_ISO_MBR_0	Port isolation configuration that each port can specify a port list to communicate with.

For example as Figure 3-3, hosts A and B are connected to ports 0 and 3 separately while router is connected to port 7. The traffic between A and B must be forwarded by router. Therefore, user can configure the 7th bit of Port isolation port mask of ports 0 and 3 to be 1 while other bits should be configured to 0, and Port Isolation port mask of port 7 still remains all ports. Then, direct communication between downlink ports, 0 and 3, is forbidden and only traffic between uplink port and downlink port can be forwarded.

Figure 3-3 Example of Uplink Port and Downlink Port for Port Isolation


0
7
3
X
A
B
Uplink port: 7
Downlink port: 0, 3
Configuration:
Port 0: P_ISO_MBR_0 = 0x80
Port 3: P_ISO_MBR_0 = 0x80
Port 7: P_ISO_MBR_0 = 0x1FFFFFFFFFFFFFF

The device provides a control setting RESTRICT_ROUTE, which indicates whether the routed packet is restricted by Port-based port isolation.

Table 3-55 PORT_ISO_RESTRICT_ROUTE_CTRL Register

Bits	Field	Description
0:0	RESTRICT_ROUTE	Indicates whether the routed packet is restricted by Port-based port isolation. 0b0: not restricted 0b0: restricted

3.3.3 VLAN-based Isolation

Sometimes, uplink ports communicate with uplink port and downlink ports, but downlink ports don't be allowed to communicate with each other in VLAN domain. Customer can use VLAN-based Isolation to achieve the application. The device provides 32 sets VLAN-based Isolation configurations in Table 3-56.

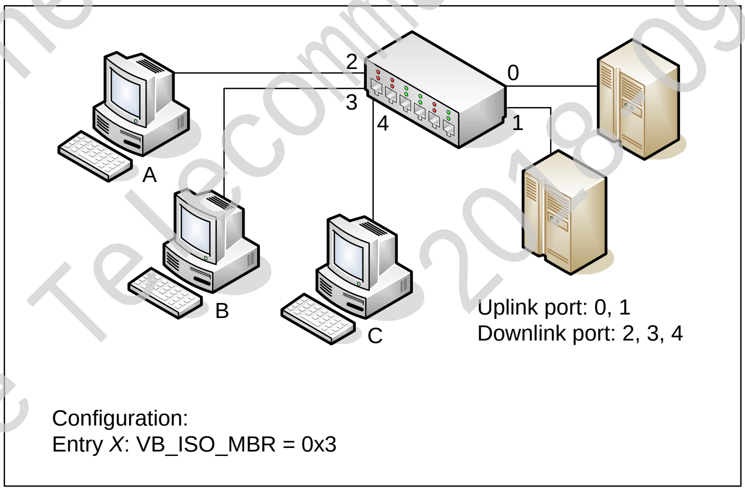
Table 3-56 PORT_ISO_VB_ISO_PMSK_CTRL Register

Bits	Field	Description
88:32	VB_ISO_MBR	VLAN-based isolated port mask (port 56 to 0). Ports which have corresponding port mask clear(i.e. 0) in this mask could not communicate with each other, while ports which have corresponding port mask set could communicate with any other ports.
24:13	VID_UPPER	Upper bound VLAN ID
12:1	VID_LOWER	Upper bound VLAN ID
0:0	VALID	Indicates whether this entry is valid 0: entry invalid; 1: entry valid.

For every packet, system will check all the 32 sets if matched or not. The matched condition is: the VALID field is valid and forwarding VID in the range [VID_UPPER, VID_LOWER]. If matched (multiple hit, select the lowest index set), the field VB_ISO_MBR will be referred.

The ports with bit value 0 in VB_ISO_MBR can't communicate with other ports having same bit value 0. However, they can communicate with the other port with bit value 1. If bit value of ports is set to 1, the port can communicate to all ports in the VLAN. Sometimes, bit value of uplink port is set to 1 and bit value of downlink port is set to 0 in network environment. For example as Figure 3-4, ports 0 and 1 are uplink ports and other ports are downlink ports. Only configure (VB_ISO_MBR) of VLAN-based Isolation entry to be (0x3), then uplink ports can communicate to all ports, but downlink ports only communicate to uplink port.

Figure 3-4 Example of Uplink Port and Downlink Port for VLAN-based Isolation



Unlike the port-based isolation, the VLAN based isolation always has no effect on the routing packet, including unicast routing and multicast routing packet. But both the unicast and multicast bridged packet are still restricted.

3.4 Trunk

3.4.1 Overview

Trunk also named link aggregation is used to utilize bandwidth effectively and improve network resilience by aggregating several physical ports into a logical port.

The system supports max 26 trunk groups in stand-alone mode and max 128 trunk groups in stacking mode for normal ports. Each of the group can aggregate at most 8 physical ports across stacking devices.

Besides that, System also supports link aggregation of stacking ports to at most 8 groups with maximum 8 member ports for each group.

3.4.2 Stand-Alone And Stacking Trunk Mode

System supports 2 mode of trunk operation: stand-alone and stacking mode. TRK_STAND_ALONE_MODE register field globally decides if trunks work in stand-alone or stacking mode.

Table 3-57 TRK_CTRL Register

Bits	Field	Description
2:2	TRK_STAND_ALONE_MODE	Configure the enable status of stand-alone mode for normal port link aggregation. In stand-alone mode, hardware would skip the link down trunk member ports. Bit value: 0b0: disable stand-alone mode 0b1: enable stand-alone mode

The characteristics of two modes are:

Table 3-58 Differences of Stand-Alone and Stacking Mode

Trunk mode	Number of trunks	Port Link Change Fail-Over	Other Difference
Stand-Alone	26	Hardware Supported	-
Stacking	128	Software Handle	Support local first function

In stand-alone mode, the link down ports of trunk egress ports would be excluded from traffic TX ports by hardware. However, in stacking mode, since hardware could not detect link status of ports on remote switch, hardware would disable trunk member link change fail-over mechanism and software should handle it.

3.4.3 Normal Port Trunk

Packet received from one physical port will be processed by the ingress process phase, to decide whether the RX port belongs to a single trunk or not which would affect L2 learnt port/trunk ID and source port filtering. While packet is going out through one trunk, the egress process phase will do the load sharing operation to decide the finally outgoing physical port.

While setting a normal port trunk, both the ingress and egress member ports should be set right. The ingress member ports settings locate in the *Source Port Mapping Table* (also known as *Ingress Trunk Table*) and local trunk portmask. The egress member set is fulfilled by the *Egress Trunk Table.*

In general, for the same trunk group, member ports in the Egress Trunk Table entry should be identical to the Ingress Trunk Table entry, but for some protocol application, our system supports that egress member set is less than ingress member set.

3.4.3.1 Ingress Process

System provides 1024 *Source Port Mapping Table (Ingress Trunk Table)* entries, which is used to configure different device and port belongs to a trunk or not. The format of the entry is as follows.

Table 3-59 Source Port Mapping Table Entry

Fields	Description
TRK_ID_VALID	Specify if TRK_ID is valid or not. If TRK_ID is valid, the port is a trunk member port of the trunk.
TRK_ID	ID of link aggregation group of source port.

While a packet is received from normal or stacking port, system will use KEY <DEV+PORT> to decide whether if it is from trunk and the corresponding TRK_ID. The KEY <DEV> represents the stacking device ID (more detailed information, please refer to stacking develop guide). The entry index would be DEV*64 + PORT. The result TRK_ID can be used to learn into L2 table as the source port and so on. For example, if a packet is received from local port 3, and the DEV configuration of the switch is 0, it would lookup Source Port Mapping Table of index (0*64 + 3) = 3 to decide if it is a trunk member port and the trunk ID.

Table 3-60 TRK_ID_CTRL Register

Bits	Field	Description
7:7	TRK_ID_VALID	Specify if TRK_ID is valid or not. If TRK_ID is valid, the TRK_ID decides the trunk ID and TRK_PPM decides the local member ports.
6:0	TRK_ID	ID of link aggregation group of source port.

Table 3-61 TRK_MBR_CTRL Register

Bits	Field	Description
55:0	TRK_PPM	Local member portmask of link aggregation group. Bit value: 0b0: individual port 0b1: individual port

System supports 52 local trunk group member configurations which are used for source port filtering and flooding frame handling. Since there are 128 trunk groups supported in stacking mode but local trunk supports at most 52 trunks (which is enough since one port can join at most one trunk), TRK_ID is required to decide the trunk ID and TRK_PPM would decide the local trunk member ports.

3.4.3.2 Egress Process

When packets are going out through one trunk, packets will do load balancing within member ports. The final outgoing port is decided by the trunk member port set, load sharing algorithm selected, traffic separation and local first if set.

Egress Member Port

System provides *Egress Trunk Table* to configure egress member port. Per Egress Trunk Table entry has 8 (TRK_DEV, TRK_PORT) configurations that denotes trunk TX candidate ports. At most 8 ports can be configured as member port.

The *Egress Trunk Table* has 128 entries in total. Per Entry format is as follows.

Table 3-62 Egress Trunk Table Entry

Fields	Description
NUM_TX_CANDI	Number of valid TX candidate ports of this trunk group.
TRK_DEV7	Device ID 7
TRK_PORT7	Trunk port 7
TRK_DEV6	Device ID 6
TRK_PORT6	Trunk port 6
TRK_DEV5	Device ID 5
TRK_PORT5	Trunk port 5
TRK_DEV4	Device ID 4
TRK_PORT4	Trunk port 4
TRK_ DEV3	Device ID 3
TRK_PORT3	Trunk port 3
TRK_ DEV2	Device ID 2
TRK_PORT2	Trunk port 2
TRK_DEV1	Device ID 1
TRK_PORT1	Trunk port 1
TRK_DEV0	Device ID 0
TRK_PORT0	Trunk port 0
L2_HASH_MSK_IDX	Index of hash parameters mask of a trunk of L2 packets. (index to L2_HASH_MASK)
IP4_HASH_MSK_IDX	Index of hash parameters mask of a trunk of IPv4 packets. (index to L3_HASH_MASK)
IP6_HASH_MSK_IDX	Index of hash parameters mask of a trunk of IPv6 packets. (index to L3_HASH_MASK)

SEP_FLOOD_EN	Enable to separate the flooding packets (including unknown L2MC, IPMC, and DLF(destination lookup fail) packets) to a specific port for a trunk group. Bit value: 0b0: Disable the traffic separation. 0b1: Enable the traffic separation.
SEP_MC_EN	Enable to separate the known L2MC and IPMC packets to a specific port for a trunk group. Bit value: 0b0: Disable the traffic separation. 0b1: Enable the traffic separation.

L2_HASH_MSK_IDX points to two sets of L2_HASH_MSK to decide hash key mask for L2 frames. IP4_HASH_MSK_IDX and IP6_HASH_MSK_IDX point to L3_HASH_MSK to decide hash key mask for IPv4 and IPv6 frames. Detail of L2_HASH_MSK and L3_HASH_MSK please refer to 0.

Load Sharing Algorithm

If packet is forwarded into a single trunk, the load sharing algorithm will be executed to decide the final outgoing port chosen from the egress trunk member set. The load sharing algorithm is very simple, just doing the 'XOR' operation for selected Hash Key, which is folded into several 6bits. The Hash Key selected is different depending on the packet type.

For the known L2(non IP) unicast packet whose destination is one trunk, system enables user to select the Hash Key from 2 sets of L2 Hash Key settings, and for the known L3(IP) unicast packet whose destination is a trunk, system also provides 2 sets of L3 Hash Key settings.

Per L2 Hash Key setting is a mask, which allows user to select one or more of SPA, SMAC, DMAC, VLAN ID as the Hash Key. Per L3 Hash Key setting is a mask too, which allows user to select one or more of SPA, SMAC, DMAC, VLAN ID, SIP, DIP, IP PROTO, IPV6 FLOW LABEL, SPORT, DPORT as the Hash Key.

For the other packet, the hash key is fixed.

Method of Hash Key selected can be summarized as below.

Table 3-63 Hash Key

Packet Type	Hash Key
Known Unicast L2 Packet to a single trunk	(SPA,SMAC,DMAC,VLANID)
Known Unicast IPV4 Packet to a single trunk	(SPA,SMAC,DMAC,VLANID, SIP,DIP,IPPROTO , SPORT,DPORT)
Known Unicast IPV6 Packet to a single trunk	(SPA,SMAC,DMAC,VLANID, SIP,DIP,IPPROTO,IPV6 FLOW LABEL, SPORT,DPORT)
Others L2 Packet	(SMAC, DMAC, SPA)
Others L3(IPV4/IPV6) Packet	(SIP, DIP, SPA)

The hash mask register is described in Table 3-64.

Table 3-64 TRK_HASH_CTRL Register

Bits	Field	Description
13:4	L3_HASH_MSK	10 bits mask for link aggregation group hash parameter selection for IPv4 and IPv6 packets. Bit 0: SPP (source physical port) Bit 1: SMAC Bit 2: DMAC Bit 3: VLAN ID Bit 4: SIP Bit 5: DIP Bit 6: L4SPORT Bit 7: L4DPORT Bit 8: Protocol ID Bit 9: IPv6 Flow Label Note that it could be an arbitrary combination
3:0	L2_HASH_MSK	4 bits mask for link aggregation group hash parameter selection for L2 packets. Bit 0: SPP (source physical port) Bit 1: SMAC Bit 2: DMAC Bit 3: VLAN ID Note that it could be an arbitrary combination

Besides that, Each Hash Key can be shifted several bits before 'XOR' operation to avoid traffic unbalanced because of the way of folding the keys. For example, if hash key *PROTO* listed in Table 3-65 is shifted 2 bits, the key folded result is in Table 3-66.

Table 3-65 PROTO Hash Key Original

Key	HASH Bit 5	HASH Bit 4	HASH Bit 3	HASH Bit 2	HASH Bit 1	HASH Bit 0
PROTO	0	0	0	0	Bit 7	Bit 6
	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Table 3-66 PROTO Hash Key Shifted

Key	HASH Bit 5	HASH Bit 4	HASH Bit 3	HASH Bit 2	HASH Bit 1	HASH Bit 0
PROTO	0	0	Bit7	Bit6	Bit 5	Bit 4
	Bit 3	Bit 2	Bit 1	Bit 0	0	0

Shift configuration is global and register is as Table 3-67.

Table 3-67 TRK_SHFT_CTRL Register

Bits	Field	Description
29:27	SHIFT_BITS_FLOW_LABEL	Shift bits of IPv6 flow label hash key
26:24	SHIFT_BITS_PROTO	Shift bits of protocol ID hash key
23:21	SHIFT_BITS_L4DPO RT	Shift bits of L4DPORT hash key

20:18	SHIFT_BITS_L4SPORT	Shift bits of L4SPORT hash key
17:15	SHIFT_BITS_DIP	Shift bits of DIP hash key
14:12	SHIFT_BITS_SIP	Shift bits of SIP hash key
11:9	SHIFT_BITS_VLAN	Shift bits of VLAN ID hash key
8:6	SHIFT_BITS_DMAC	Shift bits of DMAP hash key
5:3	SHIFT_BITS_SMAC	Shift bits of SMAC hash key
2:0	SHIFT_BITS_SPP	Shift bits of SPP hash key

Traffic Separation

System provides a traffic separation mechanism to separate known multicast traffic or L2 lookup miss traffic to a Trunk largest index of *Egress Member* port.

Per trunk group support *SEP_DLF_BCAST_EN* and *SEP_KWN_MC_EN* separate types, which could be referred from Table 3-62. When *SEP_DLF_BCAST_EN* and *SEP_KWN_MC_EN* are both enabled, global support *SEP_PORT_SEL* to decide whether send Know Multicast Packet to second largest index Port.

Table 3-68 TRK_CTRL Register

Bits	Field	Description
1:1	SEP_PORT_SEL	If SEP_KWN_MC_EN and SEP_FLOOD_EN are both enabled, this register decides the separated port. 0b0: FLOOD known MC packets are transmitted to member port with largest ID. 0b1: FLOOD packets are transmitted to member port with largest ID; known MC packets are transmitted to member port with second largest ID.

The selected port would not be a hash candidate port if traffic separation is enabled. For example, if link up trunk member ports are port 1,3,5,7, and traffic separation is configured to separate known multicast traffic, port 7 would be dedicated to output known multicast traffic, and other traffic would output through port 1,3,5 only.

Note

1. *SEP_DLF_BCAST_EN*, *SEP_KWN_MC_EN* and *SEP_PORT_SEL*, different devices should have same configuration when work in Stacking system.
2. When *SEP_DLF_BCAST_EN* and *SEP_KWN_MC_EN* are both enabled, If only 2 members are in trunk group, known MC packets are still transmitted to member port with largest port, no matter what *SEP_PORT_SEL* value is.
3. Traffic separation mechanism would stop if there is 0 or only 1 link up trunk member port.

Local First Forwarding

For trunk stacking mode, system provides the local first mechanism to reduce the bandwidth through stacking port. After local first feature is enabled, for the known-unicast packet destined to a single trunk will be forwarded to my own device uplink port instead of remote device uplink port as possible.

Table 3-69 TRK_CTRL Register

Bits	Field	Description
4:4	LOCAL_FIRST	Decide if trunk load balancing runs on local first mode Bit value: 0b0: normal load balancing 0b1: local first load balancing

Bear in mind that only known-unicast packet would do local first load balancing. Non-unicast packet and separated packet would always follow hash result to forward.

3.4.3.3 Trunk Configuration in LACP State Machine

In LACP collection state definition, a port can only receive data traffic but can't transmit. Transmission is enabled when it enters distribution state. To satisfy the state definition, the configuration of trunk when a port enters the two states are:
Collection State: Bind the port to the trunk in source port mapping table and add it to local trunk portmask. Detail could refer to 3.4.3.1.

Distribution State: Add the port to trunk group in egress trunk table. Detail could refer to 0.

3.4.4 Stacking Port Trunk

Stacking port trunk provides a means of bundling multiple stacking ports together to provide higher bandwidth between devices. Two stacking trunk groups are supported, with each up to four stacking member ports.

Stacking port trunk also will do the load sharing like the normal trunk. System provides the option *STK_HASH_CAL* to enable stacking port trunk to follow the normal trunk hash value or recalculate the hash value itself. While recalculated enabled, Hash Key selected is decided by the L2/IPV4/IPV6 index pointed the two L2 or L3 Hash Key Mask Settings which is identical to the normal trunk.

Table 3-70 TRK_STK_CTRL Register

Bits	Field	Description
18:18	L2_HASH_MSK_IDX	Index of hash parameters mask of a trunk for L2 packets. (index to L2_HASH_MASK)
17:17	IP4_HASH_MSK_IDX	Index of hash parameters mask of a trunk for IPv4 packets. (index to L3_HASH_MASK)
16:16	IP6_HASH_MSK_IDX	Index of hash parameters mask of a trunk for IPv6 packets. (index to L3_HASH_MASK)

15:0	STK_TRK_PPM	Member port mask of link aggregation group for stacking ports or EST ports Bit value: 0b0: individual port 0b1: aggregation port

Note

1. *SEP_DLF_BCAST_EN*, *SEP_KWN_MC_EN* and *SEP_PORT_SEL* are no use in stacking ports trunk, but hash key shift is still referred to.

3.5 Reserved Multicast Address (RMA)

3.5.1 RMA Overview

IEEE defines some Reserved Multicast Address (RMA) for 802.1 serials protocols. System supports RMA from 01-80-C2-00-00-00 to 01-80-C2-00-00-2F and some RMA with special ether type. Each RMA also has an individual setting for the following action: forward, drop, trap to local CPU or trap to master CPU (some even have "flood to all port").

3.5.2 RMA Entry

System supports control protocol BPDU, PTP, LLDP, EAPOL and the others RMA from 01-80-C2-00-00-00 to 01-80-C2-00-00-2F.

If multiple RMA hit at the same time, the priority is: User define 0 > User define 1 > User define 2 > User define 3 > LLDP > RMA_00(BPDU) > RMA_02 > PTP > EAPOL > RMA class 0 > RMA Class 1 > RMA class 2.

3.5.2.1 BPDU

System can forward, drop, trap and flood BPDU packet. The device supports per port registers to control the behavior for BPDU.

Table 3-71 RMA_PORT_BPDU_CTRL

Bits	Field	Description
2:0	ACT	BPDU (01-80-C2-00-00-00) Configuration. 0x0: Forward 0x1: Drop 0x2: Trap to local CPU 0x3: Trap to master CPU 0x4: Flood(limited by RMA_BPDU_FLD_PMSK) 0x5 0x7: Reserved Note: When it is set to Trap, BPDU packet should be able to bypass VLAN.

3.5.2.2 PTP

PTP packet includes two types: PTP Ethernet II and PTP UDP. The ether type of PTP Ethernet II is 0x88F7, the UDP destination port of PTP UDP is 319 or 320. System can forward, drop or trap PTP packet. The device supports per port registers to control the behavior for PTP.

Table 3-72 RMA_PORT_PTP_CTRL

Bits	Field	Description
3:2	UDP_ACT	The PTP UDP packet takes configured action. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU Note: When it is set to Trap, PTP packet should be able to bypass STP.
1:0	ETH2_ACT	The PTP Ethernet II packet takes configured action. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU Note: When it is set to Trap, PTP packet should be able to bypass STP.

3.5.2.3 LLDP

The ether type of LLDP is 0x88CC. System can forward, drop, trap or flood LLDP packet. The device supports per port registers to control the behavior for LLDP.

Table 3-73 RMA_PORT_LLDP_CTRL

Bits	Field	Description
2:0	ACT	The packet whose ethertype is 0X88CC is treated as LLDP and takes configured action. 0x0: Forward 0x1: Drop 0x2: Trap to local CPU 0x3: Trap to master CPU 0x4: Flood(limited by RMA_LLDP_FLD_PMSK) 0x5 0x7: Reserved Note: When it is set to Trap, LLDP packet should be able to bypass STP and VLAN drop ("VLAN ingress filter", "VLAN error", "VLAN accept frame type" and "CFI = 1").

3.5.2.4 EAPOL

The ether type of EAPOL is 0x888E. System can forward, drop, trap or flood EAPOL packet. The device supports per port registers to control the behavior for EAPOL.

Table 3-74 RMA_PORT_EAPOL_CTRL

Bits	Field	Description

2:0	ACT	The packet whose ethertype is 0X888E is treated as EAPOL and takes configured action. 0b00: Forward 0b01: Drop 0x2: Trap to local CPU 0x3: Trap to master CPU 0x4: Flood(limited by RMA_EAPOL_FLD_PMSK) 0x5 0x7: Reserved Note: When it is set to Trap, EAPOL packet should be able to bypass STP and VLAN drop ("VLAN ingress filter", "VLAN error", "VLAN accept frame type" and "CFI = 1").

3.5.2.5 Other RMA

Besides BPDU, PTP, LLDP and EAPOL, the device also supports the following RMA in Table 3-75, Table 3-76 and Table 3-77.

Packet with DMAC (01-80-C2-00-00-01) includes pause frame and others (i.e. MPCP). In Table 3-75, RMA_01_ACT takes action on the packet with DMAC 01-80-C2-00-00-01 but excluding pause frame. RMA_02_ACT takes action on the packet with DMAC 01-80-C2-00-00-02 but excluding OAM packet.

Table 3-75 RMA_CTRL_0

Bits	Field	Description
31:30	RMA_0F_ACT	Reserved Multicast Address 01-80-C2-00-00-0F configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
29:28	RMA_0E_ACT	Reserved Multicast Address 01-80-C2-00-00-0E configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
27:26	RMA_0D_ACT	Reserved Multicast Address 01-80-C2-00-00-0D configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
25:24	RMA_0C_ACT	Reserved Multicast Address 01-80-C2-00-00-0C configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
23:22	RMA_0B_ACT	Reserved Multicast Address 01-80-C2-00-00-0B configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
21:20	RMA_0A_ACT	Reserved Multicast Address 01-80-C2-00-00-0A configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU

19:18	RMA_09_ACT	Reserved Multicast Address 01-80-C2-00-00-09 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
17:16	RMA_08_ACT	Reserved Multicast Address 01-80-C2-00-00-08 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
15:14	RMA_07_ACT	Reserved Multicast Address 01-80-C2-00-00-07 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
13:12	RMA_06_ACT	Reserved Multicast Address 01-80-C2-00-00-06 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
11:10	RMA_05_ACT	Reserved Multicast Address 01-80-C2-00-00-05 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
9:8	RMA_04_ACT	Reserved Multicast Address 01-80-C2-00-00-04 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
7:6	RMA_03_ACT	Reserved Multicast Address 01-80-C2-00-00-03 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
5:4	RMA_02_ACT	Reserved Multicast Address 01-80-C2-00-00-02 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU Note: When it is set to Trap, this type packet should be able to bypass STP and VLAN drop ("VLAN ingress/egress filter", "VLAN error", "VLAN accept frame type" and "CFI = 1").
3:2	RMA_01_ACT	Reserved Multicast Address 01-80-C2-00-00-01 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU

Table 3-76 RMA_CTRL_1

Bits	Field	Description

31:30	RMA_1F_ACT	Reserved Multicast Address 01-80-C2-00-00-1F configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
29:28	RMA_1E_ACT	Reserved Multicast Address 01-80-C2-00-00-1E configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
27:26	RMA_1D_ACT	Reserved Multicast Address 01-80-C2-00-00-1D configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
25:24	RMA_1C_ACT	Reserved Multicast Address 01-80-C2-00-00-1C configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
23:22	RMA_1B_ACT	Reserved Multicast Address 01-80-C2-00-00-1B configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
21:20	RMA_1A_ACT	Reserved Multicast Address 01-80-C2-00-00-1A configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
19:18	RMA_19_ACT	Reserved Multicast Address 01-80-C2-00-00-19 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
17:16	RMA_18_ACT	Reserved Multicast Address 01-80-C2-00-00-18 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
15:14	RMA_17_ACT	Reserved Multicast Address 01-80-C2-00-00-17 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
13:12	RMA_16_ACT	Reserved Multicast Address 01-80-C2-00-00-16 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU

11:10	RMA_15_ACT	Reserved Multicast Address 01-80-C2-00-00-15 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
9:8	RMA_14_ACT	Reserved Multicast Address 01-80-C2-00-00-14 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
7:6	RMA_13_ACT	Reserved Multicast Address 01-80-C2-00-00-13 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
5:4	RMA_12_ACT	Reserved Multicast Address 01-80-C2-00-00-12 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
3:2	RMA_11_ACT	Reserved Multicast Address 01-80-C2-00-00-11 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
1:0	RMA_10_ACT	Reserved Multicast Address 01-80-C2-00-00-10 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU

Table 3-77 RMA_CTRL_2

Bits	Field	Description
31:30	RMA_2F_ACT	Reserved Multicast Address 01-80-C2-00-00-2F configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
29:28	RMA_2E_ACT	Reserved Multicast Address 01-80-C2-00-00-2E configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
27:26	RMA_2D_ACT	Reserved Multicast Address 01-80-C2-00-00-2D configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU

25:24	RMA_2C_ACT	Reserved Multicast Address 01-80-C2-00-00-2C configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
23:22	RMA_2B_ACT	Reserved Multicast Address 01-80-C2-00-00-2B configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
21:20	RMA_2A_ACT	Reserved Multicast Address 01-80-C2-00-00-2A configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
19:18	RMA_29_ACT	Reserved Multicast Address 01-80-C2-00-00-29 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
17:16	RMA_28_ACT	Reserved Multicast Address 01-80-C2-00-00-28 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
15:14	RMA_27_ACT	Reserved Multicast Address 01-80-C2-00-00-27 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
13:12	RMA_26_ACT	Reserved Multicast Address 01-80-C2-00-00-26 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
11:10	RMA_25_ACT	Reserved Multicast Address 01-80-C2-00-00-25 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
9:8	RMA_24_ACT	Reserved Multicast Address 01-80-C2-00-00-24 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
7:6	RMA_23_ACT	Reserved Multicast Address 01-80-C2-00-00-23 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU

5:4	RMA_22_ACT	Reserved Multicast Address 01-80-C2-00-00-22 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
3:2	RMA_21_ACT	Reserved Multicast Address 01-80-C2-00-00-21 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU
1:0	RMA_20_ACT	Reserved Multicast Address 01-80-C2-00-00-20 configuration. 0b00: Forward 0b01: Drop 0b10: Trap to local CPU 0b11: Trap to master CPU Note: When the RMA within 01-80-C2-00-00-20~2F is set to Trap, this kind of packet should be able to bypass STP and VLAN drop ("VLAN ingress/egress filter", "VLAN error", "VLAN accept frame type" and "CFI = 1").

3.5.3 User Defined RMA

The device supports 4 user defined RMA settings. Using them, you can configure specific MAC address as RMA. The fields of each user defined RMA setting are shown as Table 3-78. The whole 48 bits can be configured in user defined RMA.

Table 3-78 RMA_USR_DEF_CTRL Register

Bits	Field	Description
143:128	ADDR_MAX_HI	The address [47:32] of maximum boundary of user-defined RMA.
127:96	ADDR_MAX_LO	The address [31:0] of maximum boundary of user-defined RMA.
79:64	ADDR_MIN_HI	The address [47:32] of minimum boundary of user-defined RMA.
63:32	ADDR_MIN_LO	The address [31:0] of minimum boundary of user-defined RMA.
24:9	ETHER_TYPE	Ether type value
8	BYPASS_VLAN	User-defined RMA bypass VLAN drop ("VLAN ingress filter", "VLAN error", "VLAN accept frame type" and "CFI = 1"). This works only if EN is set as enabled. 0b0: Follow VLAN decision 0b1: Bypass (works only if the ACT field is trapping to CPU or forward)
7	LRN	0b0: Not learn the SAMC 0b1: Learn the SMAC
6	BYPASS_STP	User-defined RMA bypass "Blocking" and "Learning" port state of spanning tree. This works only if EN is set as enabled. 0b0: Drop by STP 0b1: Bypass (works only if the ACT field is trapping to CPU)

5:3	ACT	User-defined RMA action. 0x0: Forward 0x1: Drop 0x2: Trap to local CPU 0x3: Trap to master CPU 0x4: Flood (limited by RMA_USR_DEF_FLD_PMSK) 0x5 0x7: Reserved
2:1	TYPE	Compare field. 0b00: Compare ADDR only 0b01: Compare EtherType only 0b10: Both ADDR and EtherType 0b11: Reserved
0	EN	Enable this User-defined RMA rule. 0b0: Disable 0b1: Enable

3.5.4 RMA Entry Action

The device supports actions for handling behavior of RMA.

The forward action is to lookup L2 table first. If lookup hit, the packet will be forwarded to the port mask of hit entry. Otherwise, the packet will be flooding to RMA_FLD_PMSK.

For BPDU, LLDP, EAPOL and user-defined, they support flood action. The flood action will bypass egress VLAN filtering. The action is to do L2 table lookup first. If lookup hit, the packet will be forwarding to the port mask of hit entry. Else the packet will be flooding related RMA_BPDU_FLD_PMSK, RMA_LLDP_FLD_PMSK , RMA_EAPOL_FLD_PMSK and RMA_USR_DEF_FLD_PMSK .

The drop action will drop packet.

The trap action will trap to CPU.

Note

For BPDU, LLLDP, LACP and EAPOL etc. protocols, trap action should be higher priority than New SA Drop.

3.5.5 STP Ingress Check Bypass

The following protocols are able to bypass STP "Blocking" and "Learning" port state. The bypass action only works if the action of RMA is trap to CPU.

EAPOL

01-80-C2-00-00-02(Not include OAM)

LLDP

01-80-C2-00-00-00(BPDU)

PTP

01-80-C2-00-00-2X (ex,GVRP)

If STP port state is disabled, system still drops the packet.

For user-defined RMA, each entry can bypass STP ingress check by configured BYPASS_STP when action is trap.

3.5.6 VLAN Check Bypass

The following protocols are able to bypass VLAN drop which is decided by VLAN ingress/egress filter, VLAN accept frame type, or CFI = 1. The bypass action only works if the action of RMA is trap to CPU.

EAPOL

01-80-C2-00-00-02(Not include OAM)

LLDP

01-80-C2-00-00-00(BPDU)

01-80-C2-00-00-2X (ex,GVRP)

For user-defined RMA, each entry can bypass VLAN check by configured BYPASS_VLAN when action is trap or forward.

3.5.7 Cancel Mirror

Considering the requirement that some vendors do not mirror control packets, system provides below configuration:

Table 3-79 RMA_MIRROR_CTRL Register

Bits	Field	Description
0	MIRROR_ACT	Decides whether RMA packets could be mirrored. 0b0: whether mirroring packet depends on mirror setting 0b1: Cancel Rx/Tx mirror configuration

3.5.8 SA Learning

When RMA action is drop, the source MAC address wouldn't be learned. Otherwise, RMA_SMAC_LRN_CTRL in Table 3-80 specifies RMA learning behavior of each port when action is set as Forward or Trap.

Some protocols, such as PTP, LLDP and EAPOL which use port MAC as the source MAC of the control frame, it is useless to learn the port MAC. Thus, the device can specify whether to learn the source MAC address of each protocol, see Table 3-81.

Table 3-80 RMA_SMAC_LRN_CTRL Register

Bits	Field	Description
0	LRN	Learning behavior for Reserved Multicast Address 01-80-C2-00-00-00 01-80-C2-00-00-2F. 0b0: Not learn the SAMC 0b1: Learn the SMAC

Table 3-81 RMA_MGN_LRN_CTRL Register

Bits	Field	Description
2	PTP_LRN	Learning behavior for PTP. 0b0: Not learn the SAMC 0b1: Learn the SMAC
1	LLDP_LRN	Learning behavior for LLDP. 0b0: Not learn the SAMC 0b1: Learn the SMAC
0	EAPOL_LRN	Learning behavior for EAPOL. 0b0: Not learn the SAMC 0b1: Learn the SMAC

Note

The learning behavior specified here might violate the other modules. For example, when a packet should be filtered by ingress VLAN filter or STP state is not Learning or Forwarding, SMAC cannot be learned even the RMA learning control register is set as Learn.

4 L3 Feature

4.1 Routing

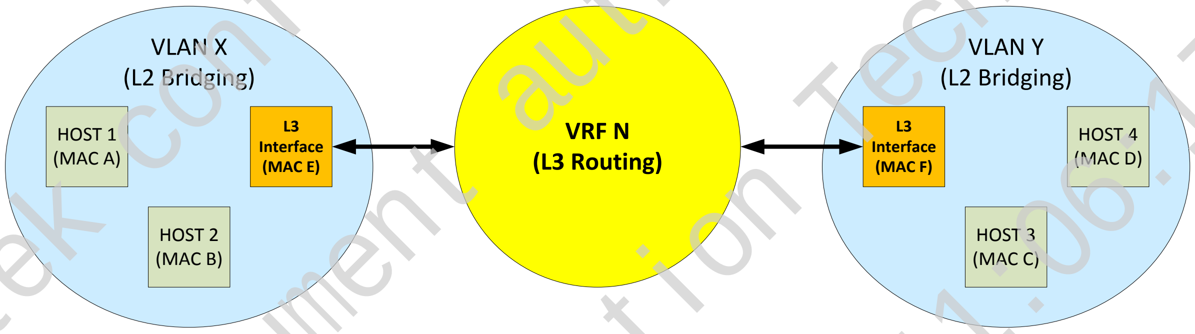
4.1.1 Routing Overview

L3 is the Network Layer in OSI (Open Systems Interconnection) model. The network switching device routes and forwards packets between different networks (VLANs) as a router. It's a common approach to make hosts in different networks able to communicate with each other.

Routing is the process of forwarding packets hop by hop on L3. It primarily includes finding an outgoing interface and a next hop, modifying the packets' SMAC, DMAC, VID (if provided), TTL and L3 header checksum (for IPv4) and forwarding.

The device supports IPv4/IPv6 unicast and multicast routing. The system provides global options for each type of packets to determine whether to route or discard those L3 packets.

Figure 4-1 Classic Forwarding Model of L2 Bridging and L3 Routing



4.1.2 L3 Interface

An L3 interface is conceptually where packets enter and leave a network (VLAN). It's used to route packets between VLANs, so it is also called VLAN interface.

Normally, L3 routing is receiving a packet from an L3 interface and transmitting the packet to another L3 interface.

4.1.2.1 Overview

The device supports 1024 L3 interfaces and up to 16 different MTU values used by all L3 interfaces.

4.1.2.2 Attribute

There are bunch of attributes belong to an interface, the details are described as below.

(Physically, these attributes are stored into L3_IGR_INTF and L3_EGR_INTF separately.)

VID	VLAN ID of an interface Each interface is associated with a specific VLAN which the interface belongs to.
MAC Address	MAC address of an interface If a packet's *destination MAC* is matched with the MAC address and its incoming VLAN is the interface's VID, it means the packet is sent to the L3 interface. When a packet is routed and sent out from an interface, its *source MAC* will be replaced by the MAC address. Note: If a packet's DMAC is multicast or broadcast, the interface should take the packet as well. Especially, those RFC reserved multicast address of certain protocols.
VRF ID	Virtual Routing and Forwarding Identifier Used to logically separate the physical L3 table into many different logical L3 tables, and each interface can only be associated with one logical L3 table.
MTU	Maximum Transmission Units To indicate the largest data unit the interface can pass onwards (including L3 IP headers). By default, it's 1500 bytes. If the data unit length of the routed packet is over the MTU, it will be discarded on the egress L3 interface. The system has 1024 L3 interfaces, and can have 16 different MTU values at the most.
TTL	Time-To-Live Scope It's a threshold associated with the interface, and only applied to multicast packets. If the routed IP multicast packet has a smaller TTL (than the interface's), the packet will be discarded.
Routing Enable	Individual routing options for each type of IP packets: • *IPUC_ROUTE_EN* – IPv4 unicast routing enable • *IP6UC_ROUTE_EN* – IPv6 unicast routing enable • *IPMC_ROUTE_EN* – IPv4 multicast routing enable • *IP6MC_ROUTE_EN* – IPv6 multicast routing enable

uRPF configurations	Unicast Reverse Path Forwarding (uRPF) option For IPv4 unicast packet: • *IP_URPF_CHK_EN* – uRPF check enable • *IP_URPF_MODE* – uRPF check mode • *IP_URPF_DFLT_ROUTE_EN* – default route enable • *IP_URPF_FAIL_ACT* – action for uRPF check failed packets For IPv6 unicast packet: • *IP6_URPF_CHK_EN* – uRPF check enable • *IP6_URPF_MODE* – uRPF check mode • *IP6_URPF_DFLT_ROUTE_EN* – default route enable • *IP6_URPF_FAIL_ACT* – action for uRPF check failed packets *For further details, refer to 4.1.3.4.*
Multicast Lookup Key	IP multicast L3 lookup key selection for 2 rounds For a normal L3 interface, there are 2 options: 1. VID: 1st – (SIP, GIP, VID) 2nd – (*, GIP, VID) 2. Interface: 1st – (SIP, GIP, INTF) 2nd – (*, GIP, INTF) For an L3 tunnel interface, it is fixed to \`Interface\` mode. By default, it's configured to \`VID\` mode when an interface has been created.

Reserved Multicast Address Action	For well-known reserved-multicast-address packets, there are dedicated forwarding actions for each scope of IP multicast addresses.

For IPv4 multicast, the certain scopes are:

- 224.0.0.X/24 – Local Network Control Block
- 224.0.1.X/24 – Internetwork Control Block
- 239.X.X.X/8 – Organization-Local Scope

For IPv6 multicast the certain scopes are:

- FF0X::0000:00XX – the last 32 bits is 0000:00XX (Protocols)
- FF0X::YYYY:XXXX – the last 32 bits is YYYY:XXXX (YYYY != 0)
- FF0X::DB8:0:0/96 – Documentation Addresses
- Neighbor Discovery packet – MLD
    - IPv6 without any error header
    - IPv6.Protocol = 0x3A (ICMPv6)
    - ICMPv6.Type = 133 ~ 137
        - Router Solicitation (NDP)
        - Router Advertisement (NDP)
        - Neighbor Solicitation (NDP)
        - Neighbor Advertisement (NDP)
        - Redirect Message (NDP)
    - Packet without PPPoE or MPLS header

For each type of packets, the action could be:

- Forward – follow the action of hit entry if it's hit an existing entry (higher priority), or flood in VLAN if it looks up miss
- Drop – directly discard the packets (no matter the lookup result)
- Trap to CPU – directly trap the packets to CPU (no matter the lookup result)
- Copy to CPU – forward and copy the packets to CPU

Multicast Lookup Miss Action	Forwarding action for L3 lookup-miss packets

It's optional to drop those packets or trap to CPU for further process. This action will NOT apply to well-known reserved-multicast-address packets.

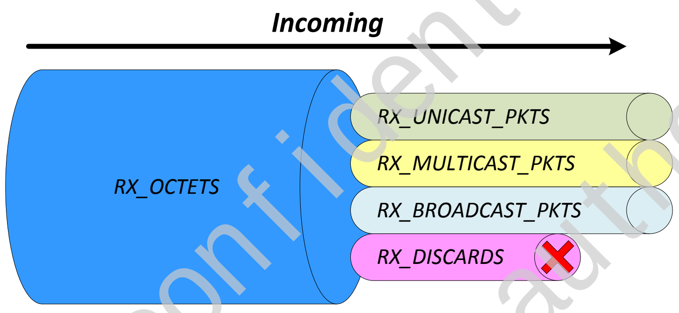
For IPv4 and IPv6 packets, they have individual options. They are configurable independently.

4.1.2.3 Counters

Ingress Counters

When packets arrive an L3 interface, the ingress counters of the L3 interface should be updated. If a packet has been counted in *RX_OCTETS* and one of the packet counters (*RX_UNICATS_PKTS*, *RX_MULTICAST_PKTS*, *RX_BROADCAST_PKTS*, *RX_DISCARDS*) will also be updated. The relationship is shown as below.

Figure 4-2 Ingress Counters of L3 Interface

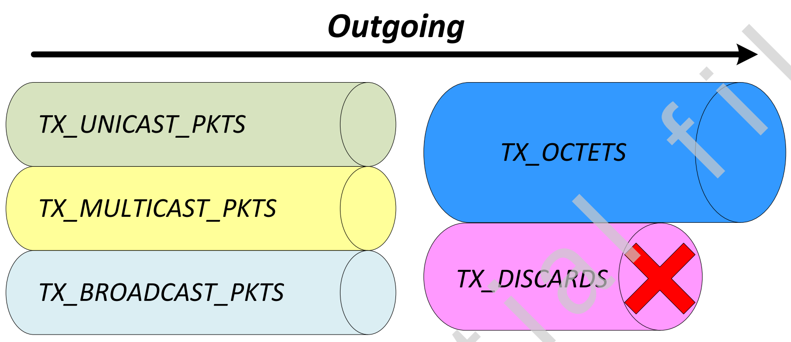


RX_OCTETS	The total bytes of all received packets (including unicast, multicast, broadcast and error packets) The packets counted in this counter are counted into one of the packet counts as well.
RX_UNICAST_PKTS	The number of the received unicast DIP packets Exclusive with other packet counts
RX_MULTICAST_PKTS	The number of the received multicast DIP packets Exclusive with other packet counts
RX_BROADCAST_PKTS	The number of the received broadcast DIP (255.255.255.255) packets Exclusive with other packet counts
RX_DISCARDS	The number of the received error packets (ex. checksum error) Exclusive with other packet counts

Egress Counters

When packets routed and sent out on an L3 interface, they might be counted into the following counters.

Figure 4-3 Egress Counters of L3 Interface



TX_OCTETS	The total bytes of all successfully transmitted packets (excluding error packets) Exclusive with *TX_DISCARDS* counter
TX_UNICAST_PKTS	The number of the transmitted unicast DIP packets Exclusive with *TX_MULTICAST_PKTS* and *TX_BROADCAST_PKTS* counters
TX_MULTICAST_PKTS	The number of the transmitted multicast DIP packets Exclusive with *TX_UNICAST_PKTS* and *TX_BROADCAST_PKTS* counters
TX_BROADCAST_PKTS	The number of the transmitted broadcast DIP (255.255.255.255) packets Exclusive with *TX_UNICAST_PKTS* and *TX_MULTICAST_PKTS* counters
TX_DISCARDS	The number of the dropped packets on this egress interface (ex. filtered by MTU, TTL scope checks)

4.1.3 IP Unicast Routing

Unicast IP packet is a packet with unicast DIP (destination IP) address.

IP Unicast routing is the process of finding a path to reply a packet based on its DIP address, and forward it to the next hop router or destination host.

Once the packets are determined to be routed, their DMAC/SMAC/VID/TTL (and DSCP) will be modified based on configured information and the IP checksum field in the IP header will be recalculated as well.

The VID information in the outgoing L3 Interface entry is used to determine the destination VLAN and modify the packets. Once routed, the packets should be forwarded within the destination VLAN.

If the length of packets' L3 payload is larger than the value in the MTU entry (global 8-entry for IPv4, 8-entry for IPv6) indexed by IP_MTU_IDX / IP6_MTU_IDX, then packets should be trapped or dropped, A value of 65535 of the MTU is equivalent to MTU check disabled.

If the TTL/hoplimit decrement is required, then ASIC would first check the packet's TTL/hoplimit. If the value is larger than 0, then it will be decremented by 1, otherwise it will be kept as 0.

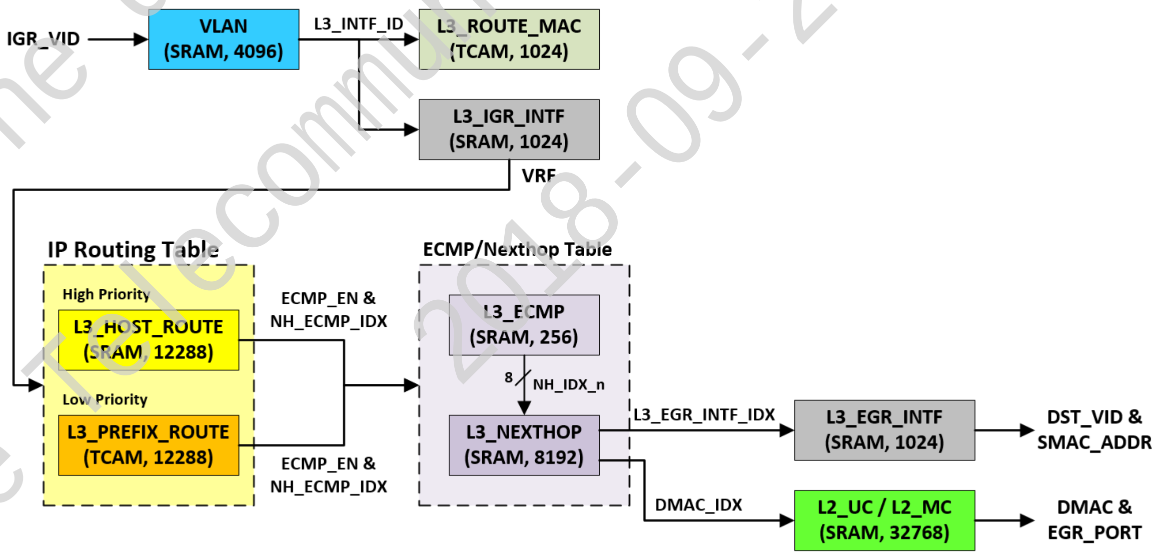
4.1.3.1 Overview

The system supports:

256 VRF	Derived from ingress L3 interface
12K IPv4 L3 host table	Contains IPv4 host routing, shared with IPv6
2K IPv6 L3 host table	Contains IPv6 host routing, shared with IPv4
12K IPv4 L3 LPM table	Contains IPv4 subnets for LPM (Longest Prefix Match) routing, shared with IPv6 and Openflow
6K IPv6 L3 LPM table	Contains IPv6 subnets for LPM (Longest Prefix Match) routing, shared with IPv4 and Openflow
256 ECMP group	Each group can contain at most 8 next hops (paths)
8K next hop table	L3 next hop table

Figure 4-4 displays the paradigm of IP unicast routing.

Figure 4-4 Paradigm of IP Unicast Routing



Each VLAN can associate with an L3 interface by VLAN.L3_INTF_ID index.

Each L3 interface has its own MAC address which is used to represent the Router MAC in the associated VLAN and VRF ID which logical routing table the interface is belong to.

While doing L3 routing, the system performs table lookup with { VRF, IP } as key.

An L3 entry (host route or prefix route) can specify a nexthop with path ID which can represent an ECMP group or a nexthop. Each ECMP group can index to 8 nexthop entries at the most, and selects one of them by hashing with the packet significant header info.

Each nexthop entry can index to an egress L3 interface and an L2 MAC entry. The egress L3 interface is used to derive the destination VLAN and SMAC address, and the L2 MAC entry is used to derive DMAC address and destination port.

The system supports unicast routing to an L2 multicast entry, which can be used to implement MNLB (Microsoft Network Loading Balancing) multicast and IGMP mode.

4.1.3.2 L3 Routing Table

Routing table is mainly composed of L3 host route table and L3 prefix route (LPM) table.

The L3 Host Route table and L3 Prefix Route table lookup with DIP as key.

L3 host route table is a 2-left 6-way SRAM which is shared by IPUC, IPMC and Openflow entries.

Whole host route table can be separated into 2 blocks (pre-configured in SDK), the top block is used by L3 IPUC and IPMC entries and the bottom block is used by Openflow.

4.1.3.2.1 Host Route Table

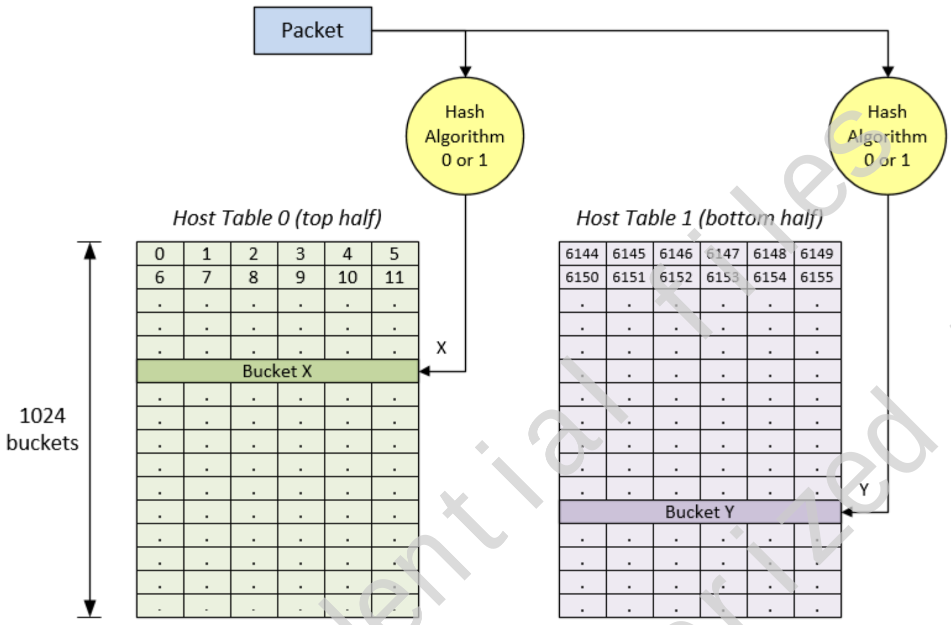
This table is primarily designed for host routing (DIP is fully-match).

L3 host routing table supports 2 hash algorithms. It's configurable for the top and the bottom of the host table.

The hash values are used to address the bucket locations of each half of the host table.

Normally, using different algorithm for each half table could lower the collision rate.

Figure 4-5 L3 Host Table



An L3 table lookup can address to 2 buckets. Each bucket has 6 physical entry slots.

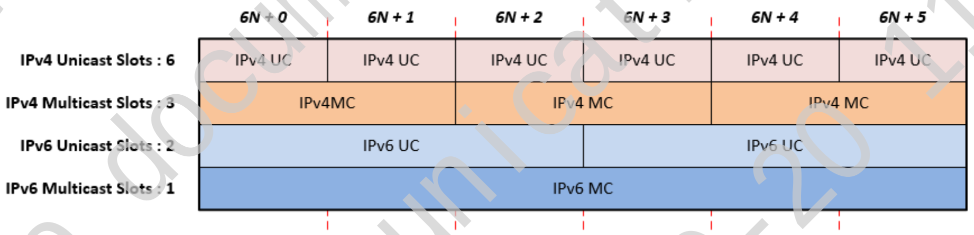
For an IPv4 unicast (IPv4 UC) entry, it costs 1 physical entry slot.

For an IPv6 unicast (IPv6 UC) entry, it costs 3 physical entry slots.

For an IPv4 multicast (IPv4 MC) entry, it costs 2 physical entry slots.

For an IPv6 multicast (IPv6 MC) entry, it costs 6 physical entry slots.

Figure 4-6 L3 Host Table Bucket



Each bucket can populate 6 IPv4 unicast entries, 3 IPv4 multicast entries, 2 IPv6 unicast entries or 1 IPv6 multicast entry.

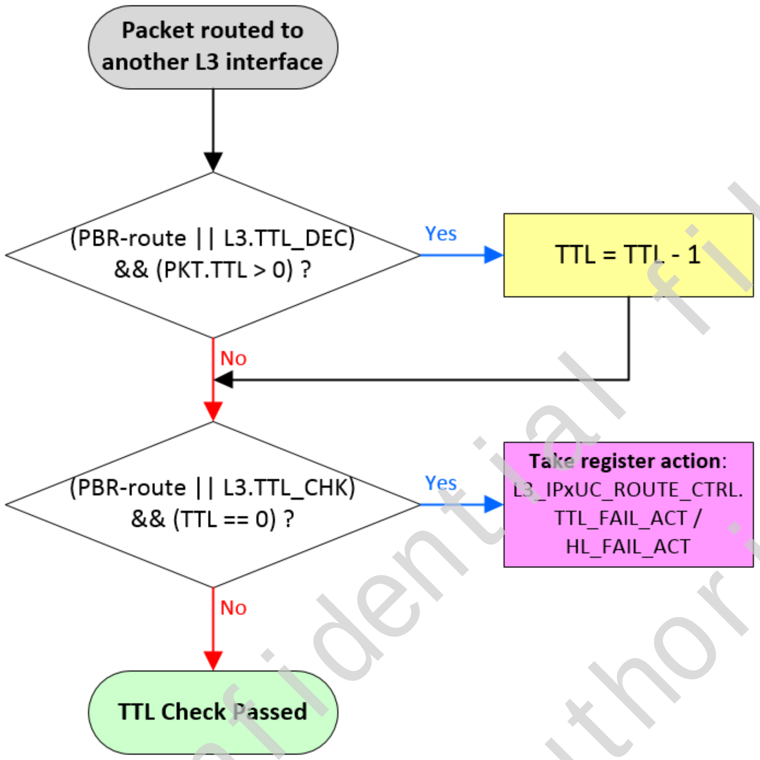
Table 4-1 L3_HOST_ROUTE_IPUC Entry

Field	Description
VALID	Entry valid bit
FMT	KEY: Format type 0x0: L3 entry 0x1: Openflow

ENTRY_TYPE	KEY: Type of the host route entry 0x0: IPv4 Unicast 0x1: IPv4 Multicast 0x2: IPv6 Unicast 0x3: IPv6 Multicast
VRF_ID	KEY: Virtual Route Forwarding database ID
IP	KEY: IPv4/IPv6 Address
ACT	Action performed on DIP lookup (routing): 0x0: route (refer to ECMP_EN/NH_ECMP_IDX) 0x1: trap to local/master CPU (packets destined for CPU) 0x2: copy to local/master CPU (trap and route) 0x3: drop (for NULL interface) 0x4~0x7: Reserved Note: The target CPU is indicated by L3_IPUC_ROUTE_CTRL.PKT_TO_CPU_TARGET.
DST_NULL_INTF	Indicates whether associated destination interface is NULL or not. 0b0: Non-null interface 0b1: Null interface (black hole) Note: It's used for CPU reason resolution. If 0b1 and Act is 0x1 or 0x2(trap to CPU), CPU Reason is "40. L3 IPUC null route". It can be used to identify destination unreachable IP.
ECMP_EN	ECMP enabled or not. (available when DST_NULL_INTF == 0) 0b0: disable 0b1: enable
NH_ECMP_IDX	Next-hop entry / ECMP group index. (available when DST_NULL_INTF == 0)
TTL_DEC	Option: TTL decrease by 1 (only when packet's TTL > 0)
TTL_CHK	Option: TTL check
QOS_EN	Option: Assign a new internal priority with QOS_PRI
QOS_PRI	Internal priority (only available when QOS_AS = 1)
HIT	Hit indication bit of DIP routing lookup (excluding uRPF lookup) write 1 to keep the value, and write 0 to clear.

When a packet is routed to another L3 interface (*not trap to CPU*), the TTL check process is done as below.

Figure 4-7 TTL Check Process of L3 Routing


Packet routed to another L3 interface
(PBR-route || L3.TTL_DEC) && (PKT.TTL > 0) ?
Yes
TTL = TTL - 1
No
(PBR-route || L3.TTL_CHK) && (TTL == 0) ?
Yes
Take register action: L3_IPxUC_ROUTE_CTRL. TTL_FAIL_ACT / HL_FAIL_ACT
No
TTL Check Passed

4.1.3.2.2 Prefix Route Table

L3 prefix routing table is a longest prefix match table, support subnet IP address.

Once a packet is determined to be routed (no matter it is copied to CPU or not), the TCAM-based L3 Prefix Route table will also be searched to find a next hop and an egress L3 interface.

Table 4-2 L3_PREFIX_ROUTE_IPUC Entry

Field	Description
VALID	Entry valid bit
FMT	KEY: Format type 0x0: L3 entry 0x1: Openflow
BMSK_FMT	Care Bitmask: FMT
ENTRY_TYPE	KEY: Type of the prefix route entry 0x0: IPv4 Unicast 0x2: IPv6 Unicast
BMSK_ENTRY_TYPE	Care Bitmask: ENTRY_TYPE
IP	KEY: IPv4/IPv6 address
BMSK_IP	Care Bitmask: IP

DST_NULL_INTF	Indicates whether associated destination interface is NULL or not. 0b0: Non-null interface 0b1: Null interface (black hole) Note: It's used for CPU reason resolution. If 0b1, HOST_ROUTE and Act is 0x1 or 0x2(trap to CPU), CPU Reason is "40. L3 IPUC null route". It can be used to identify destination unreachable IP.
HOST_ROUTE	To indicate if this is a host route (fully-match) Note: It's automatically configured by SDK driver.
DFLT_ROUTE	To indicate if this is a default route Note: It's automatically configured by SDK driver.
ACT	Action performed on DIP lookup (routing): 0x0: route (refer to NH_IDX) 0x1: trap to CPU (packets destined for CPU) 0x2: copy to CPU (trap and route) 0x3: drop (For NULL interface) Note: the target of packet-to-CPU is indicated by L3_IPxUC_ROUTE_CTRL.PKT_TO_CPU_TARGET.
ECMP_EN	ECMP enabled or not. (available when DST_NULL_INTF == 0) 0b0: disable 0b1: enable
NH_ECMP_IDX	Next-hop entry / ECMP group index. (available when DST_NULL_INTF == 0)
TTL_DEC	Option: TTL decrease by 1 (only when packet's TTL > 0)
TTL_CHK	Option: TTL check
QOS_EN	Option: Assign a new internal priority
QOS_PRI	Internal priority (only available when QOS_AS = 1)
HIT	Hit indication bit of DIP routing lookup (excluding uRPF lookup) write 1 to keep the value, and write 0 to clear.

HOST_ROUTE field is used to indicate whether the entry is a network-route (fully match or not). It is also used to encode the CPU reason to help software quickly dispatch the trapped packet for different purpose. If a packet is trapped by a host route, it basically means it's just forwarding to the CPU. If it is trapped by a net route, the software may need to send an ARP request to the destination host which is directly linked and get its MAC address then forward this packet to it.

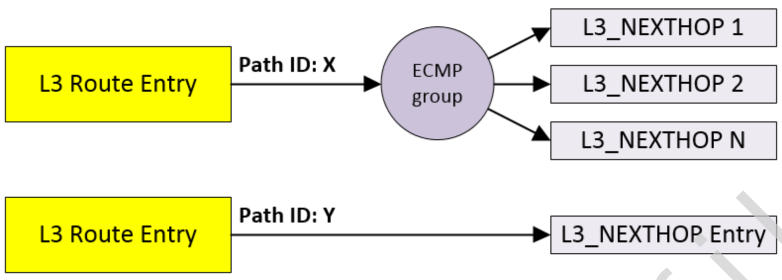
DFLT_ROUTE field is used in the uRPF process or help PBR Default Unicast Routing action, indicating whether this particular routing entry is a default route or no. This field only exists in Prefix Route table (TCAM).

4.1.3.3 L3 Destination

A Path ID(**NH_ECMP_IDX**) is an identifier to indicate a destination of L3 routing which is used by L3 host/prefix route entries.

There are 2 types of destination, nexthop and ECMP. Path ID can represent both of them.

Figure 4-8 L3 Path ID (ECMP / Nextop)



4.1.3.3.1 Nexthop Entry

The device supports 8192 nexthop entries.

The information in L3_NEXTHOP entry is used to modify the packets. DMAC_IDX provides modified DMAC and outgoing port. L3_EGR_INTF_IDX points to L3_EGR_INTF entry and derive the destination VLAN ID from L3_EGR_INTF.DST_VID.

DMAC_IDX support Null interface drop/trap for PBR. See “Unicast Policy-Based Routing” for detail.

Table 4-3 L3_NEXTHOP Table

Field	Description
DMAC_IDX	Index to an L2 entry to obtain the destination MAC address and outgoing port information
L3_EGR_INTF_IDX	Index to L3 outgoing interface entry

4.1.3.3.2 ECMP Group

The device supports 256 ECMP groups, each group can point to 8 nexthop entries at the most.

For an ECMP group, it use a 4-bit hash value to select one of the available nexthop entries in the group. The forwarding module will generate the hash value based on the following information of an ECMP routed packet which is configurable by L3_ECMP_HASH_CTRL register.

Rx Port Number	The incoming physical port which received the packet
Rx Trunk ID	The incoming logical trunk port ID which received the packet If the incoming port is an individual port, then this value is treated as zero.
Source IP Address	Source IPv4/IPv6 address It is able to configure a rotating offset for hashing
Destination IP Address	Destination IPv4/IPv6 address It is able to configure a rotating offset for hashing

IPv4 DSCP/IPv6 TC	IPv4: The 6 most-significant bits (DSCP) of TOS field IPv6: The 6 most-significant bits (Differentiated Services) of Traffic Class field
IPv6 Flow Label	The 20-bit flow label ID in IPv6 header (IPv6 only)
L4 Protocol	8-bit layer-4 protocol identifier
L4 Source Port	16-bit layer-4 source port number It is able to configure a rotating offset for hashing (available for TCP/UDP/UDP-Lite/SCTP, otherwise it's 0)
L4 Destination Port	16-bit layer-4 destination port number It is able to configure a rotating offset for hashing (available for TCP/UDP/UDP-Lite/SCTP, otherwise it's 0)
Universal ID	3-bit universal ID which is used to avoid polarization issue

Figure 4-9 L3 ECMP Group

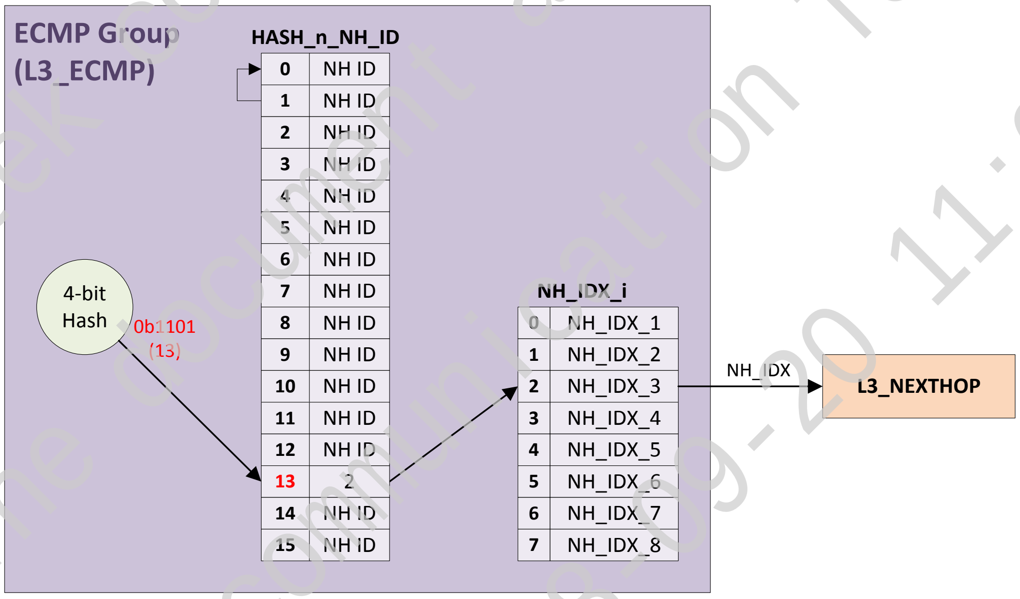


Table 4-4 ECMP Table

Field	Description
HASH_n_NH_ID	Selector of nexthop index for hash value n (n = 0 ~ 15, i-th nexthop index = value + 1)
NH_IDX_i	Index to the i-th next-hop entry (i = 1 ~ 8)

4.1.3.4 Unicast Reverse Path Forwarding (uRPF)

uRPF helps to protect a network against attacks based on source address spoofing. It filters traffic based on the source address. If uRPF is enabled, it discards a received packet if its incoming interface is not the interface on which a packet would be forwarded to reach the address contained in the source address.

System provides a global option (L3_IP_ROUTE_CTRL.URPF_BASE_SEL) to specify using L3-interface-based or port-based uRPF. Only the corresponding uRPF configuration of the specified uRPF base is active. The uRPF configuration for IPv4 and IPv6 are independent, it is able to configure separately.

The uRPF configuration for IPv4/IPv6 of each base includes the following options.

URPF_CHK_EN	Enable/Disable uRPF check for IPv4/IPv6 unicast routing
URPF_CHK_MODE	uRPF check mode selection Strict mode – found route entry and interface match Loose mode – route entry check only
URPF_DFLT_ROUTE_EN	To indicate whether default route is available while performing uRPF check process If this is disabled, the uRPF result is failed when uRPF lookup hits a default route entry
URPF_FAIL_ACT	uPRF check fail action DROP – discard the failed packets TRAP – trap the failed packets to CPU FORWARD – (permit) still forward the failed packets COPY – forward them and copy to CPU as well Note: even the action is configured to FORWARD or COPY, the URPF_RESULT of ACL match key is still indicating as URPF FAILED.

The strict mode of uRPF check requires that the packets must be received on the interface that the router would use to forward the return packet. The loose mode of uRPF check is algorithmically similar to strict uRPF, but differs in that it checks only for the existence of a route (even a default route, if applicable), not where the route points to (unless the route points to the NULL interface, i.e., the DST_NULL_INTF field is set). Practically, this could be considered as a "route presence check".

If the SIP lookup of uRPF check process returns an entry with DST_NULL_INTF field set, it means that the route is associated with NULL interface, thus packets are treated as uRPF failed.

In either mode, if the default route (L3_ENTRY.DFLT_ROUTE field is set) is returned during the uRPF check, the URPF_DFLT_ROUTE_EN is referred to determine the result. If only a default route hit and the URPF_DFLT_ROUTE_EN is enabled, then the uRPF process will continue (i.e., whether success or not depends on strict/loose mode and entry's interface information). Otherwise, packets would be treated as uRPF failed.

The uRPF-failed packets will be trapped (primarily for logging) or dropped according to the URPF_FAIL_ACT (another option is forward and work with ACL rules).

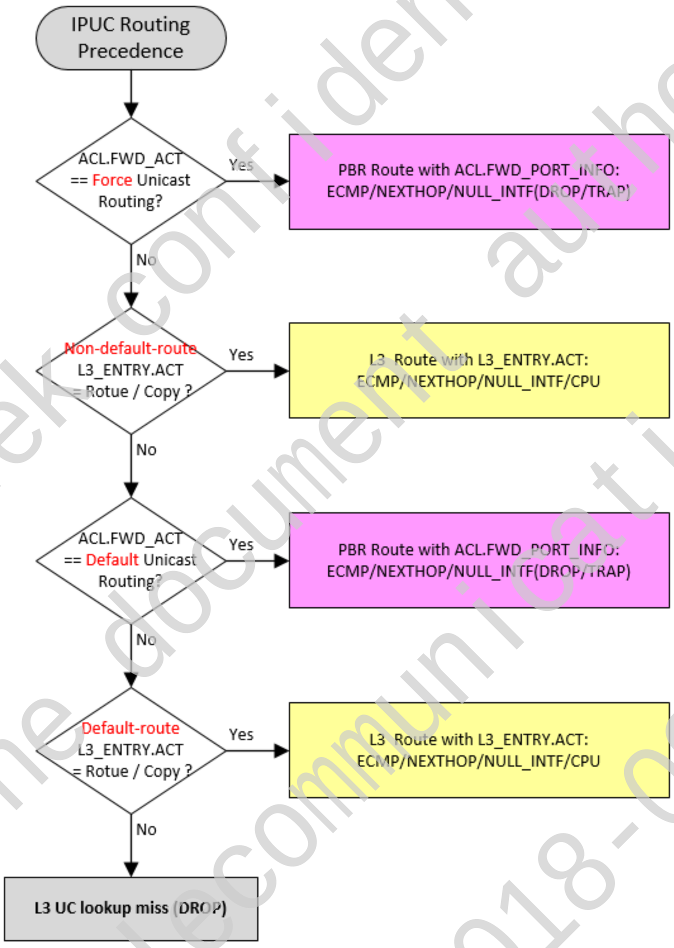
The URPF_FAIL_ACT is usually configured only on downlink interfaces (access side).

4.1.3.5 Policy-Based Routing (PBR)

Unicast PBR (Policy-Based Routing) provides an alternative way for routing unicast packets to the routine destination (IP) based routes derived from routing protocols. Policies can be programed based on source/destination IP address, layer3/4 protocols, packets length, etc.

ACL provides two distinct actions for PBR, the Force Unicast Routing action and the Default Unicast Routing action. The routing precedence is shown as below.

Figure 4-10 Unicast Routing Precedence about PBR


IPUC Routing Precedence
ACL.FWD_ACT == Force Unicast Routing?
Yes
PBR Route with ACL.FWD_PORT_INFO: ECMP/NEXTHOP/NULL_INTF(DROP/TRAP)
No
Non-default-route L3_ENTRY.ACT = Rotue / Copy ?
Yes
L3 Route with L3_ENTRY.ACT: ECMP/NEXTHOP/NULL_INTF/CPU
No
ACL.FWD_ACT == Default Unicast Routing?
Yes
PBR Route with ACL.FWD_PORT_INFO: ECMP/NEXTHOP/NULL_INTF(DROP/TRAP)
No
Default-route L3_ENTRY.ACT = Rotue / Copy ?
Yes
L3 Route with L3_ENTRY.ACT: ECMP/NEXTHOP/NULL_INTF/CPU
No
L3 UC lookup miss (DROP)

ACL PBR routing actions (ACL.FWD_PORT_INFO when ACL.FWD_ACT = Force/Default Unicast Routing):

NEXTHOP_IDX	Index to an L3_NEXTOP entry
ECMP_IDX	Index to an L3 ECMP group entry

NULL_INTF	Null interface (discard packets) Optionally trap the packet to CPU (for ICMP unreachable)

Packets routed by policy always perform the TTL decremented and TTL check, as well as MTU check.

4.1.3.6 IPUC Routing Exceptions

Basically, these checks and exceptions are common for both unicast and multicast.

4.1.3.6.1 IP Header Validation

Once the IP packets are delivered to the L3 engine (hit Router MAC address), they are examined before looking up the routing tables.

The aim of this step is to discard malformed IP packets, or trap these packets to CPU for further processing which beyond the ability of ASIC.

For IPv4 packets, the following conditions are checked for IP Header Validation:

1. Payload length including IPv4 header must be at least 20 bytes.
2. IP checksum must be correct.
3. IP version number must be 0x4.
4. IP header length must be at least 20 bytes (PKT.IP_HDL field >= 0x5) and payload length should be able to cover whole IP header.

Once any of the checks is failed, the packet is forwarded by L3_IP_ROUTE_CTRL.IP_HDR_ERR_ACT.

For IPv6 packets, the following conditions are checked for IP Header Validation:

1. Payload length including IPv6 header must be at least 40 bytes.
2. IP version number must be 0x6.

Once any of the checks is failed, the packet is forwarded by L3_IP_ROUTE_CTRL.IP6_HDR_ERR_ACT.

4.1.3.6.2 Illegal IP Address

Packets with illegal IP (either source IP or destination IP) addresses should be dropped or trapped, and the event could be logged (via trapping).

For IPv4 packets, the illegal source/destination addresses are:

1. Source: 127.0.0.0/8 (localhost)
2. Source: 224.0.0.0/4 (class D, multicast)
3. Source: 255.255.255.255 (broadcast)
4. Destination: 127.0.0.0/8 (localhost)

The action of the exception can be configured by:

L3_IPUC_ROUTE_CTRL.BAD_SIP_ACT (DROP/TRAP)

L3_IPUC_ROUTE_CTRL.ZERO_SIP_ACT (DROP/TRAP)

L3_IPUC_ROUTE_CTRL.BAD_DIP_ACT (DROP/TRAP)

For IPv6 packets, the illegal source/destination addresses are:

1. Source: ::1/128 (localhost)
2. Source: FF::/8 (multicast)
3. Destination: ::1/128 (localhost)
4. Destination: ::/128 (unspecified address)

The action of the exception can be configured by:

L3_IP6UC_ROUTE_CTRL.BAD_SIP_ACT (DROP/TRAP)

L3_IP6UC_ROUTE_CTRL.ZERO_SIP_ACT (DROP/TRAP)

L3_IP6UC_ROUTE_CTRL.BAD_DIP_ACT (DROP/TRAP)

4.1.3.6.3 IP Header Option/Extension Header

For IPv4 unicast packet, IPv4 header with option (header length larger than 5) can be trapped/dropped/routed optionally by L3_IPUC_ROUTE_CTRL.HDR_OPT_ACT (DROP/TRAP/ROUTE/COPY).

For IPv6 unicast packet, verify IP header option and extension header as below.

HBH_ACT	Action for IPv6 packet with Hop-by-Hop extension header (DROP/TRAP/ROUTE/COPY)
HBH_ERR_ACT	Action for IPv6 packet with error HBH extension header which is not present immediately following the common header (DROP/TRAP/ROUTE/COPY)
HDR_ROUTE_ACT	Action for IPv6 packet with Routing extension header (DROP/TRAP/ROUTE/COPY)

The configuration can be programed by: L3_IP6UC_ROUTE_CTRL

4.1.4 IP Multicast Bridging and Routing

The device is a multilayer switch which is able to do L2 bridging and L3 routing at the same time.

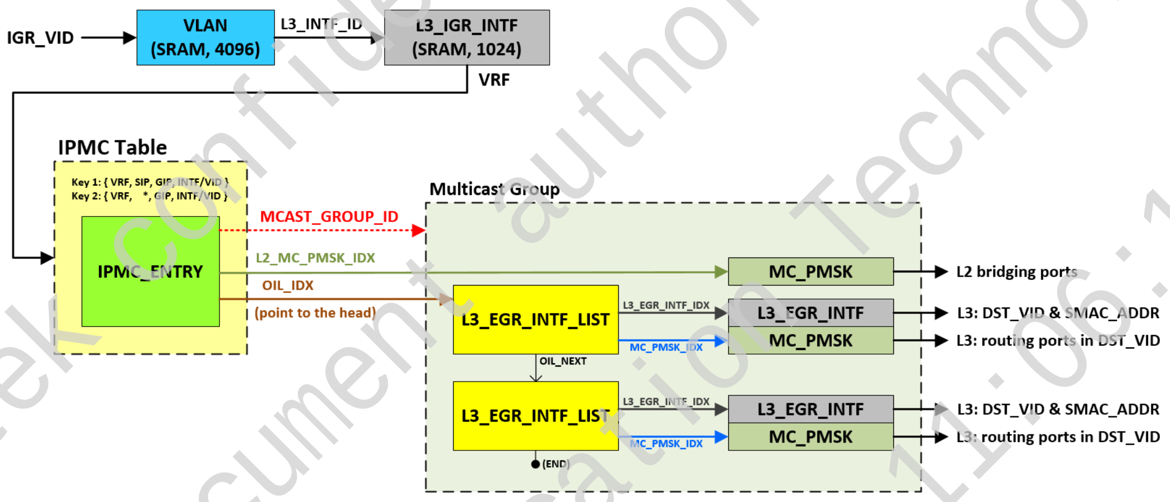
For an IPv4/IPv6 multicast packet, it could be bridged in the ingress VLAN and routed to other VLANs according to the action of IPMC entry which is configured based on the multicast protocol.

4.1.4.1 Overview

The system supports:

256 VRF	Derived from ingress L3 interface
6K IPMC entries	Provide L2 bridging and L3 routing information, shared with L3 host route table For IPv6 multicast packets, it's up to 2K IPv6 IPMC entries.
4K fwd-portmask entries	Support 4096 forwarding portmask entries shared by IP multicast bridging and routing
16K OIL node entries	Outgoing Interface List node for packet replication Can use multiple nodes to make a linked list which is able to be indexed by IPMC entry

Figure 4-11 Paradigm of IP Multicast Bridging and Routing



MCAST_GROUP_ID is an identifier to indicate a multicast group which contains L2 bridging member ports and L3 routing member ports. (It's implemented by SDK driver.)

A multicast group (MCAST_GROUP_ID) can be used by multiple IPMC entries. SDK driver will handle the pointers (L2_MC_PMSK_IDX, OIL_IDX) update of IPMC entry according to its associated multicast group.

4.1.4.2 IP-Based Bridging

The system supports 2 bridging lookup modes. It's belong to the VLAN profile configuration.

Each VLAN can specify a VLAN profile, the device will bridge IPMC packets based on the bridge mode configured in the VLAN profile of the incoming VLAN.

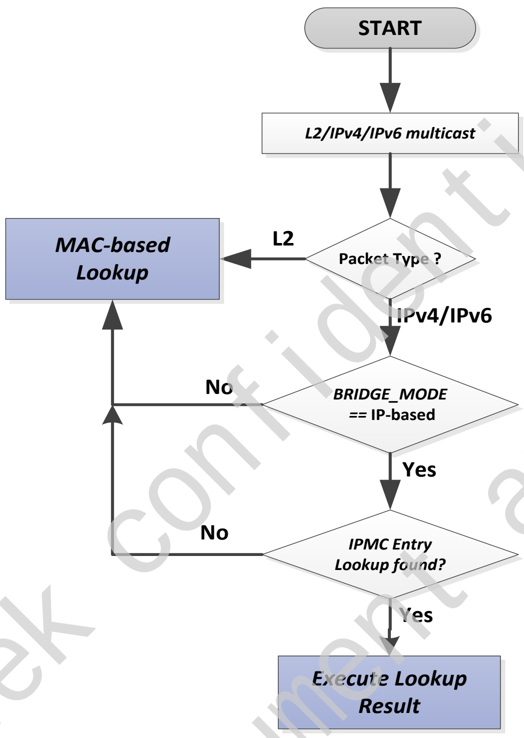
Table 4-5 VLAN_PROFILE_SET Register

Bits	Field	Description

202	IP4_MC_BDG_MODE	Bridging mode for IPv4 multicast packets: 0: MAC-base (forward as L2 multicast) 1: IP-based
201	IP6_MC_BDG_MODE	Bridging mode for IPv6 multicast packets

If the ingress VLAN is configured to use IP-based bridging but the L2 bridging information is invalid (lookup miss), then it will downgrade to MAC-based bridging.

Figure 4-12 IP Multicast Bridge Process flow



Note

The bridge IP-based mode will use the IP multicast route table which is shared with L3 host route table, that means the bridge entry and the route entry can be the same one entry.

4.1.4.2.1 IP-Based Bridging Lookup Miss

IP multicast routing and bridging process independently. When routing and bridging both lookup miss, and the lookup miss action is TRAP, the routing lookup miss reason is prior. Otherwise the reason is bridging lookup miss drop.

Bridging lookup miss action is also configured via VLAN_PROFILE_SET of the ingress VLAN.

Table 4-6 VLAN_PROFILE_SET Register

Bits	Field	Description
200:198	L2MC_BRIDGE_LU_MIS_ACT	L2 Multicast packet lookup miss action 0x0: forward (flooding to L2_UNKN_MC_FLD_PMSK) 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6 ~ 0x7: reserved
197:195	IPMC_BRIDGE_LU_MIS_ACT	IPv4 Multicast packet lookup miss action 0x0: forward (flooding to IP4_UNKN_MC_FLD_PMSK) 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6 ~ 0x7: reserved
194:192	IP6MC_BRIDGE_LU_MIS_ACT	IPv6 Multicast packet lookup miss action 0x0: forward (flooding to IP6_UNKN_MC_FLD_PMSK) 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6 ~ 0x7: reserved

Route lookup miss action setting according to the ingress L3 interface attribute (Multicast Lookup Miss Action).

4.1.4.3 IP Multicast Routing

IP multicast lookup relevant table, it provides the IP multicast information about bridging and routing.

The IP multicast routing function performs based on the global (GLB_EN) and ingress L3 interface (IPMC_ROUTE_EN, IP6MC_ROUTE_EN) configurations.

Table 4-7 L3_IPMC_ROUTE_CTRL Register

Bits	Field	Description
0	GLB_EN	Globally enable IPv4 multicast routing or not. 0x0: disable 0x1: enable

Table 4-8 L3_IP6MC_ROUTE_CTRL Register

Bits	Field	Description
0	GLB_EN	Globally enable IPv6 multicast routing or not. 0x0: disable 0x1: enable

4.1.4.3.1 IPMC Entry (Multicast Address Table)

Table 4-9 IPMC Entry

Field	Description
VALID	Entry valid bit
FMT	KEY: Format type 0x0: L3 entry 0x1: Openflow
ENTRY_TYPE	KEY: Type of the host route entry 0x0: IPv4 Unicast 0x1: IPv4 Multicast 0x2: IPv6 Unicast 0x3: IPv6 Multicast
IPMC_TYPE	KEY: IPMC lookup type 0x0: L2 bridging only 0x1: L2 bridging and L3 routing
MC_KEY_SEL	KEY: lookup key selection (derived from L3_IGR_INTF) 0x0: VLAN ID 0x1: interface ID
VRF_ID	KEY: Virtual Route Forwarding database ID
SIP	KEY: Source IPv4/IPv6 Address
GIP	KEY: Destination/Group IPv4/IPv6 Address
VID_INTF_CMP	OPTION: Compare VID/INTF or not 0x0: ignore VID/INTF 0x1: compare with VID/INTF
VID_INTF	KEY: VLAN ID / Interface ID (use with VID_INTF_CMP)
L2_CHK_VID	Incoming VLAN ID (for L2 information validation) The L2 bridging information is only valid when L2_EN=1 and the incoming VLAN is match with L2_CHK_VID.
L2_EN	Enable L2 bridging information of this entry
L2_ACT	Action performed on L2 bridging: 0x0: bridge (refer to L2_MC_PMSK_IDX) 0x1: trap to local/master CPU 0x2: copy to local/master CPU (bridge and trap) 0x3: drop Note: The target CPU is indicated by L3_IPMC_ROUTE_CTRL.PKT_TO_CPU_TARGET.
L2_MC_PMSK_IDX	Multicast portmask entry index for L2 bridging
L2_MC_TNL_LST_VALID	To indicate whether L2_MC_TNL_LST_IDX is valid
L2_MC_TNL_LST_IDX	Index of L2 tunnel list table entry
L3_EN	Enable L3 routing information of this entry

L3_ACT	Action performed on L3 routing: 0x0: route (refer to OIL_IDX) 0x1: trap to local/master CPU 0x2: copy to local/master CPU (route and trap) 0x3: drop Note: The target CPU is indicated by L3_IPMC_ROUTE_CTRL.PKT_TO_CPU_TARGET.
RPF_CHK_EN	OPTION: Performs RPF check or not. (relevant only for L3_EN is set)
RPF_FAIL_ACT	Action performed for packets failed RPF check (relevant only for RPF_FAIL_EN is set) 0x0: trap to local/master CPU 0x1: drop 0x2: copy to local/master CPU (route and trap) 0x3: ASSERT_CHK: find out whether incoming interface is in the list of outgoing interface.
RPF_ID	RPF VID/INTF_ID for L3 information validation
OIL_IDX_VALID	Valid bit of OIL entry index
OIL_IDX	L3_EGR_INTF_LIST entry index
QOS_EN	Option: Assign a new internal priority with QOS_PRI
QOS_PRI	Internal priority (only available when QOS_EN = 1)
TTL_MIN	Referred TTL minimum, used to compare with the routed packet's TTL value. If the routed packet's TTL is less than TTL_MIN, it could be copied to local/master CPU.
MTU_MAX	Referred MTU maximum, used to compare with the routed packet's L3 payload length. If the payload length is greater than MTU_MAX, it could be copied to local/master CPU.
STK_FWD_PMSK	Forwarding portmask for stacking system (bit[15:0] are mapped to stacking port 15~0) Note: In a stacking system, this portmask is used to indicate which stacking ports want to received one copy of the packet.
HIT	Hit indication bit (write 1 to keep the value, write 0 to clear)

Note

SDK driver will configure the IPMC entry properly and handle the update when multicast group member has changed.

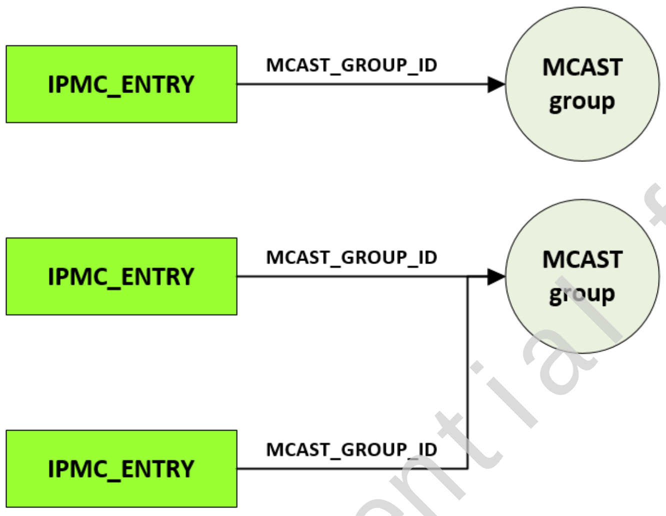
4.1.4.3.2 Multicast Group and Outgoing Interface List (OIL)

A multicast group is a member set which is used by IPMC routing. User can configure the group member by using SDK MCAST APIs.

Before adding an IPMC entry, its multicast group should be created first even the member is empty.

A multicast group (identified by MCAST_GROUP_ID) can be referred by multiple IPMC entries. Before deleting a multicast group, the IPMC entries associated with the group should be removed to prevent from the existing IPMC entry uses the wrong data (garbage).

Figure 4-13 Relationship of IPMC Entry and Multicast Group



A multicast group can contain an L2 bridging member ports and one or many L3 routing members.

An L3 routing member includes an egress L3 interface associated with a VLAN and the member ports belong to this L3 interface.

Figure 4-14 Concept of Multicast Group (MCAST_GROUP)

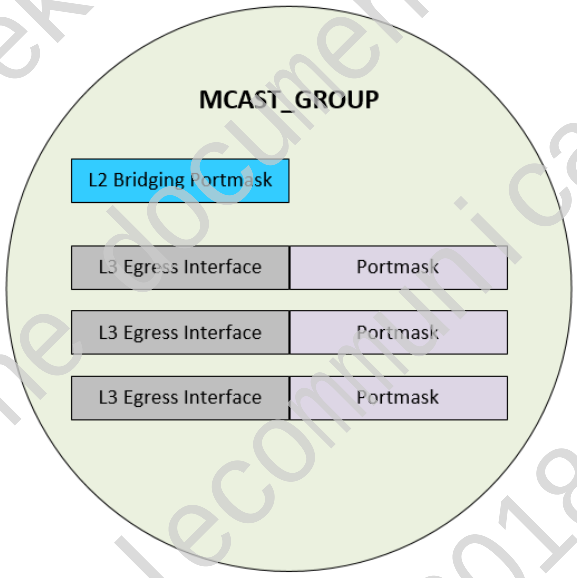
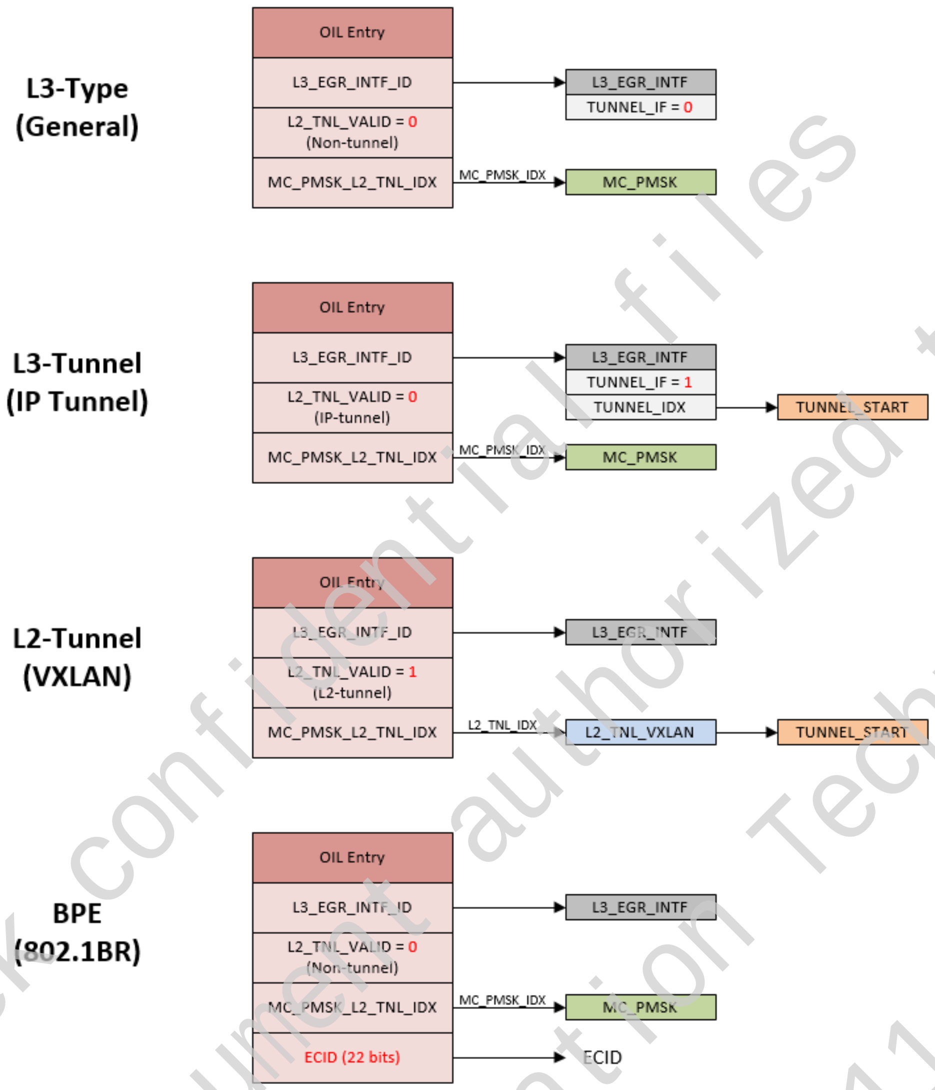


Table 4-10 L3_EGR_INTF_LIST (Multicast Outing Interface List) Entry

Field	Description

OIL_NEXT	OIL pointer to the next OIL entry
TTL_DEC	TTL decrease (should disable for routing to CPU or MVR) 0b0: disable 0b1: enable
TTL_CHK	Enable TTL/HopLimit check 0b0: disable 0b1: enable
SA_REPLACE	Enable the replacement of SMAC with the SMAC_ADDR of L3_EGR_INTF entry (should disable for routing to CPU or MVR) 0b0: disable 0b1: enable
L3_EGR_INTF_ID X	Egress L3 interface index
L2_TNL_VALID	Destination mode: 0b0: Non-tunnel or IP tunnel (refer to L3_EGR_INTF_IDX) 0b1: L2 tunnel (refer to MC_PMSK_L2_TUNNEL_IDX)
MC_PMSK_L2_TN L_IDX	Multicast portmask entry index / L2 tunnel index
DST_PORT_TYPE	Destination Port Type (indicate the function of MC_PMSK_L2_TNL_IDX) 0b0: portmask index 0b1: offset portmask ([10:8] offset, [7:0] 8-bit portmask)
ECID	Outgoing ECID = { GRP [1:0], EXT [7:0], BASE [11:0] }

Figure 4-15 Node Types of OIL Entry



4.1.4.3.3 IP Multicast Route Monitoring Counter

System provides 4 sets of 64-bit counter, each set can specify the counting mode (bytes or packets), as well as an index to the IPMC table, identifying which entry's hit will trigger the counter's update.

If a counter is configured as byte-count, then like MTU processing, the value of total length (L3 payload length) will be added into the counter.

The counters are intended to help the SPT/RPT switchover in PIM-SM, which are generally triggered by the rate of a specific traffic.

Table 4-11 L3_MONT_CNTR_CTRL Register

Bits	Field	Description
17	RST	Write 1 to clear the internal counter then auto clear RST bit.
16	LATCH	Write 1 to trigger H/W latches the value into the COUNT field 0b0: normal idle state 0b1: trigger to latch the counter value

15	MODE	Counter mode: 0b0: packet count 0b1: byte count
14:0	INDEX	Index of the monitored IPMC entry

Table 4-12 L3_MONT_CNTR_DATA Register

Bits	Field	Description
63:32	CNT_HI	MSB 32-bit of the L3 entry counter. User should take care of the wrap around issue.
31:0	CNT_LO	LSB 32-bit of the L3 entry counter.

4.1.4.4 IPMC Routing Exception

4.1.4.4.1 TLL and MTU Check

IP multicast routing supports TTL-based boundary and MTU check. Each IPMC entry provides TTL_MIN and MTU_MAX configuration which can be used to copy the exception packet to CPU for further processing. Once the routed packet TTL value is less than IPMC entry's TTL_MIN or packet's L3 payload length is larger than the value of MTU_MAX, the packet will be copied/trapped to CPU.

Table 4-13 L3_IPMC_ROUTE_CTRL Register

Bits	Field	Description
14:13	TTL_FAIL_ACT	Action for IPv4 multicast packets which fail the TTL boundary check. 0x0: drop 0x1: trap to local CPU 0x2: trap to master CPU 0x3: reserved
12:11	MTU_FAIL_ACT	Action performed on IPv4 multicast packets with MTU check failed. 0x0: drop 0x1: trap to local CPU 0x2: trap to master CPU 0x3: reserved

Table 4-14 L3_IP6MC_ROUTE_CTRL Register

Bits	Field	Description
20:19	HL_FAIL_ACT	Action for IPv6 multicast packets which fail the TTL boundary check. 0x0: drop 0x1: trap to local CPU 0x2: trap to master CPU 0x3: reserved
18:17	MTU_FAIL_ACT	Action performed on IPv6 multicast packets with MTU check failed. 0x0: drop 0x1: trap to local CPU 0x2: trap to master CPU 0x3: reserved

4.1.4.4.2 Source Interface Filter

The routed multicast packet should change the L3 interface normally.

If the ingress L3 interface is equal to the egress L3 interface, the packet may be filtered according to the Source Interface Filter state.

Table 4-15 L3_IPMC_ROUTE_CTRL Register

Bits	Field	Description
10	SRC_INTF_FLTR_EN	Enable source interface filtering for IPv4 multicast routed packets or not. 0x0: disable 0x1: enable

Table 4-16 L3_IP6MC_ROUTE_CTRL Register

Bits	Field	Description
16	SRC_INTF_FLTR_EN	Enable source interface filtering for IPv6 multicast routed packets or not. 0x0: disable 0x1: enable

4.1.4.4.3 Header option

For IPv4 multicast packet, IPv4 header with option (header length larger than 5) can be configured optionally by L3_IPMC_ROUTE_CTRL.HDR_OPT_ACT (DROP/TRAP/FORWARD/COPY).

For IPv6 multicast packet, verify IP header option and extension header as below.

HBH_ACT	Action for IPv6 packet with Hop-by-Hop extension header (DROP/TRAP/ROUTE/COPY)
HBH_ERR_ACT	Action for IPv6 packet with error HBH extension header which is not present immediately following the common header (DROP/TRAP/ROUTE/COPY)
HDR_ROUTE_ACT	Action for IPv6 packet with Routing extension header (DROP/TRAP/ROUTE/COPY)

The configuration can be programed by: L3_IP6MC_ROUTE_CTRL

Table 4-17 L3_IPMC_ROUTE_CTRL Register

Bits	Field	Description

9:7	HDR_OPT_ACT	Action performed on IPv4 multicast packets with IP header options. 0x0: drop 0x1: trap to local CPU and bridge 0x2: route and bridge (i.e., don't care) 0x3: copy to local CPU and route + bridge 0x4: trap to master CPU and bridge 0x5: copy to master CPU and route + bridge 0x6: bridge only 0x7: reserved

Table 4-18 L3_IP6MC_ROUTE_CTRL Register

Bits	Field	Description
15:13	HDR_ROUTE_ACT	Action performed on IPv6 multicast packets with routing extension headers. 0x0: drop 0x1: trap to local CPU and bridge 0x2: route and bridge (i.e., don't care) 0x3: copy to local CPU and route + bridge 0x4: trap to master CPU and bridge 0x5: copy to master CPU and route + bridge 0x6: bridge 0x7: reserved
12:10	HBH_ERR_ACT	Action performed on IPv6 multicast packets with hop-by-hop extension header error. 0x0: drop 0x1: trap to local CPU and bridge 0x2: route and bridge (i.e., don't care) 0x3: copy to local CPU and route + bridge 0x4: trap to master CPU and bridge 0x5: copy to master CPU and route + bridge 0x6: bridge 0x7: reserved
9:7	HBH_ACT	Action performed on IPv6 multicast packets with hop-by-hop extension header. 0x0: drop 0x1: trap to local CPU and bridge 0x2: route and bridge (i.e., don't care) 0x3: copy to local CPU and route + bridge 0x4: trap to master CPU and bridge 0x5: copy to master CPU and route + bridge 0x6: bridge 0x7: reserved

4.1.4.4.4 DIP/DMAC mismatch

IP packets have a standardized mapping between DIP and DMAC. A router MUST NOT forward any packet which the router received as a Link Layer multicast unless the packet's destination address is an IP multicast address. To comply with above requirements, we provide DIP/DMAC address check options.

For an IPv4 multicast packet, the MAC address should be mapped as below:

DMAC [47:24] = 01:00:5E

DMAC [23] = 0

DMAC [22:0] = IPv4 DIP [22:0]

If the IPv4 multicast packet contains a mismatch DMAC, it could only be bridged but not routed.

The action of the exception can be configured by:

L3_IPMC_ROUTE_CTRL.DMAC_MISMATCH_ACT (DROP/TRAP/BRIDGE) for IPv4 multicast packets

For an IPv6 multicast packet, the MAC address should be mapped as below:

DMAC [47:32] = 33:33

DMAC [31:0] = IPv6 DIP [31:0]

If the IPv6 multicast packet contains a mismatch DMAC, it could only be bridged but not routed.

The action of the exception can be configured by:

L3_IP6MC_ROUTE_CTRL.DMAC_MISMATCH_ACT (DROP/TRAP/BRIDGE) for IPv6 multicast packets

Table 4-19 L3_IPMC_ROUTE_CTRL Register

Bits	Field	Description
6:5	DMAC_MISMATCH_ACT	Action performed on IPv4 multicast packets with DMAC mismatch. 0x0: drop 0x1: trap to local CPU and bridge 0x2: trap to master CPU and bridge 0x3: bridge

Table 4-20 L3_IP6MC_ROUTE_CTRL Register

Bits	Field	Description
6:5	DMAC_MISMATCH_ACT	Action performed on IPv6 multicast packets with DMAC mismatch. 0x0: drop 0x1: trap to local CPU and bridge 0x2: trap to master CPU and bridge 0x3: bridge

4.1.4.4.5 Bad source IP address

The system provides some forwarding options about illegal source IP address for IP multicast packets.

For IPv4 multicast packets, the illegal source addresses are:

1. 127.0.0.0/8 (localhost)
2. 224.0.0.0/4 (class D, multicast)
3. 255.255.255.255 (broadcast)

For IPv6 multicast packets, the illegal source addresses are:

1. ::1/128 (localhost)
2. FF::/8 (multicast)

Table 4-21 L3_IPMC_ROUTE_CTRL Register

Bits	Field	Description

4:3	ZERO_SIP_ACT	Action performed on IPv4 multicast packets with zero SIP. 0x0: drop 0x1: trap to local CPU and bridge 0x2: trap to master CPU and bridge 0x3: bridge
2:1	BAD_SIP_ACT	Action performed on IPv4 multicast packets with illegal SIP. 0x0: drop 0x1: trap to local CPU and bridge 0x2: trap to master CPU and bridge 0x3: bridge

Table 4-22 L3_IP6MC_ROUTE_CTRL Register

Bits	Field	Description
4:3	ZERO_SIP_ACT	Action performed on IPv6 multicast packets with zero SIP. 0x0: drop 0x1: trap to local CPU and bridge 0x2: trap to master CPU and bridge 0x3: bridge
2:1	BAD_SIP_ACT	Action performed on IPv6 multicast packets with illegal SIP. 0x0: drop 0x1: trap to local CPU and bridge 0x2: trap to master CPU and bridge 0x3: bridge

4.2 Tunneling

Tunneling can be used to connect two disjoint IP networks that don't have a native routing path to each other, via an underlying routable protocol across an intermediate transport network.

For instance, different branches of a company with using private IP addresses internally want to connect each other via the Internet. It can be achieved by creating an IP-in-IP tunnel to transit the tunneled packet across the Internet.

The device supports IP-in-IP, ISATAP, 6to4, 6rd, GRE, and VXLAN tunnels.

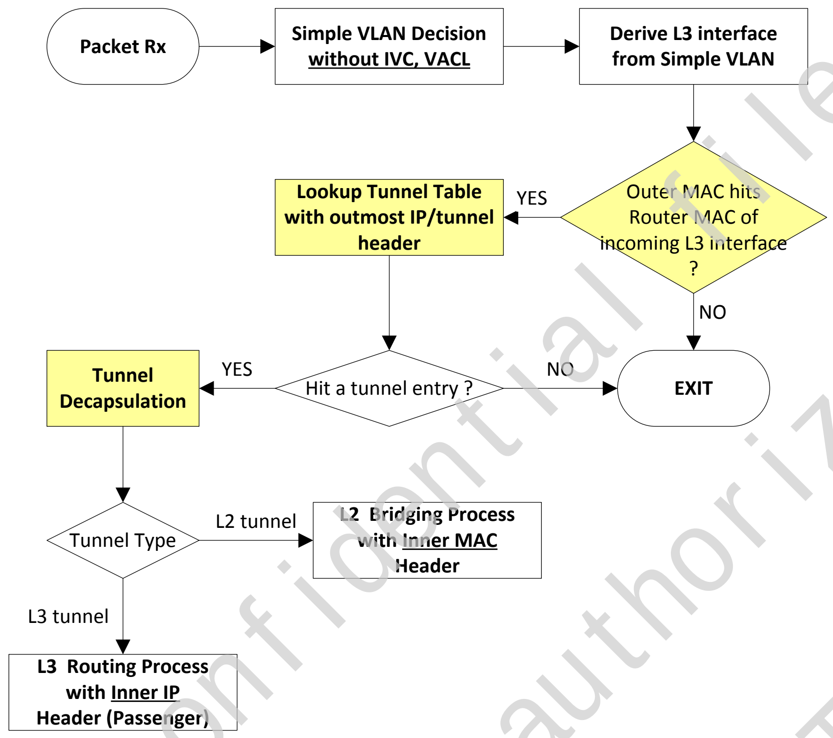
Table 4-23 Supported Tunnel Types

Tunnel Type	Interface Type	References
IP-in-IP	L3 configured tunnel	RFC 1853, RFC 2003, RFC 2893
ISATAP	L3 auto tunnel	RFC 4214
6to4	L3 auto tunnel	RFC 3964
6rd	L3 auto tunnel	RFC 5969
GRE	L3 configured tunnel	RFC 2784, RFC 2890
VXLAN	L2 configured tunnel	RFC 7348, draft-ietf-nvo3-vxlan-gpe-04

Whole system can create up to 384 IPv4 tunnels or 128 IPv6 tunnels.

4.2.1 Tunnel Decapsulation

Figure 4-16 Tunnel Decapsulation Process Flowchart


Packet Rx
Simple VLAN Decision without IVC, VACL
Derive L3 interface from Simple VLAN
Outer MAC hits Router MAC of incoming L3 interface ?
YES
NO
Lookup Tunnel Table with outmost IP/tunnel header
Hit a tunnel entry ?
YES
NO
EXIT
Tunnel Decapsulation
Tunnel Type
L2 tunnel
L2 Bridging Process with Inner MAC Header
L3 tunnel
L3 Routing Process with Inner IP Header (Passenger)

Note

The ingress VLAN decision before tunnel decapsulation is a simplified VLAN decision, which does NOT involve Ingress VLAN Conversion and VLAN ACL in this process

4.2.1.1 Tunnel Interface Configuration

Decapsulating-related configurations are as below：

Remote IP (SIP)	IPv4/IPv6 address of remote tunnel end-point (Received packet's Source IP address)
Local IP (DIP)	IPv4/IPv6 address of local tunnel end-point (Received packet's Destination IP address)
Remote Port (SPORT)	L4 port number of remote tunnel end-point (Received packet's Source Port Number)
Local Port (DPORT)	L4 port number of local tunnel end-point (Received packet's Destination Port Number)

Use Tunnel TTL	To indicate using tunnel header's TTL instead of passenger's. For *Uniform-mode* tunneling, it should be enabled (inherit). For *Pipe-mode* tunneling, it should be disabled.
Use Tunnel DSCP	To indicate using tunnel header's DSCP instead of passenger's. For *Uniform-mode* tunneling, it should be enabled (inherit). For *Pipe-mode* tunneling, it should be disabled.
Keep Passenger DSCP	Force Egress Remarking to keep the original passenger's DSCP value (cannot be modified).
Priority Selection Table	To indicate which set of *Priority Selection Tables* should be used to decide Internal Priority.
Internal Priority Assignment	Tunnel-based internal priority assignment.

Table 4-24 Decapsulation-related Tunnel Interface Configuration

	IP-in-IP	ISATAP	6to4	6rd	GRE	VXLAN
Remote IP (SIP)	●	○	○	●	●	●
Local IP (DIP)	●	●	●	●	●	●
Remote Port (SPORT)	○	○	○	○	○	●
Local Port (DPORT)	○	○	○	○	○	●
Use Tunnel TTL	●	●	●	●	●	○
Use Tunnel DSCP	●	●	●	●	●	●
Keep Passenger DSCP	●	●	●	●	●	●
Priority Selection Table	●	●	●	●	●	●
Internal Priority Assignment	●	●	●	●	●	●

4.2.1.1.1 IPv4/IPv6 Tunneling

IP-in-IP tunnel is a configured tunnel and CHIP supports the following interface types:

IPv4-in-IPv4	The traffic of an IPv4 network is carried by another IPv4 network.
IPv6-in-IPv4	The traffic of an IPv6 network is carried by an IPv4 network.
IPAny-in-IPv4	The traffic of IPv4/IPv6 networks is allowed to be carried by an IPv4 network.
IPv4-in-IPv6	The traffic of an IPv4 network is carried by an IPv6 network.
IPv6-inIPv6	The traffic of an IPv6 network is carried by another IPv6 network.

IPAny-in-IPv6	The traffic of IPv4/IPv6 networks is allowed to be carried by an IPv6 network.

The device supports *Source IPv6 Address Verification* checks to the passenger (inner header), it can be configured globally.

Once the packet has been qualified as one of the following types, the tunnel cannot be terminated while the corresponding check is enabled.

IPv4-compatible IPv6 address	The source address of a passenger IPv6 header is belong to the format: (the prefix 96 bits are all-zeros) ::[IPv4 address]
IPv4-mapped IPv6 address	The source address of a passenger IPv6 header is belong to the format: ::ffff:[IPv4 address]

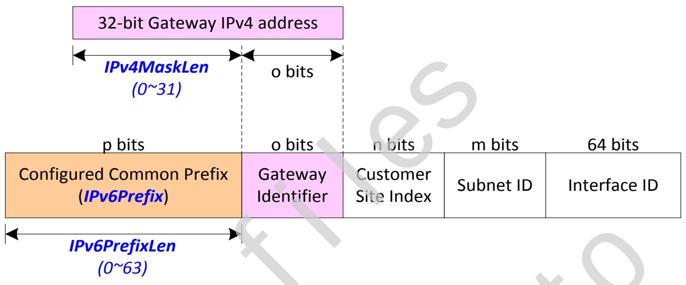
4.2.1.1.2 Automatic Tunneling

The purpose of *IPv6-in-IPv4* auto tunnels are used to connect IPv6 islands (isolated IPv6 networks) across an IPv4 infrastructure.

ISATAP	Intra-Site Automatic Tunnel Addressing Protocol. For a dual-stack node with public/private IPv4 address, the auto-generated IPv6 address will be: fe80::0200:5efe:[public IPv4 address] fe80::0000:5efe:[private IPv4 address] Related configurations: ***ISATAP SIP Type Check*** – Global option to filter the packets which the types (public, private) of source IPv4 and IPv6 addresses are mismatched. ***ISATAP SIP Mapping Check*** – Global option to filter the packets which the source IPv4 address and the source IPv6 address are not mapped.
6to4	IPv6 address for *6to4* should be: 2002:[IPv4 address]::/48 Related configurations: ***6to4 SIP Verification*** – Global option to filter the packets which their source IPv6 addresses are not qualified as 6to4 address or not mapped to the source IPv4 address. ***6to4 DIP Verification*** – Global option to filter the packets which their destination IPv6 address are not qualified as 6to4 address or not mapped to the destination IPv4 address.

6rd

6rd is short for ***IPv6 Rapid Deployment***, the 6rd IPv6 address is configurable which is different from *6to4*.



IPv6Prefix, *IPv6PrefixLen* and *IPv4MaskLen* are configured parameters, which are used to define a 6rd domain (prefix).

The device supports up to 2 6rd domains.

Related configurations:

6rd DIP Verification – Global option to filter the packets with the IPv6 address which is belong to different 6rd domain from IPv4 address (mismatched).

4.2.1.1.3 GRE Tunneling

Generic Routing Encapsulation (GRE) was developed by *Cisco Systems* that can encapsulate a wide variety of network layer protocols inside virtual point-to-point links over an Internet Protocol network.

The device supports IPv4 and IPv6 GRE tunnel (i.e., IP-GRE-IP encapsulation).

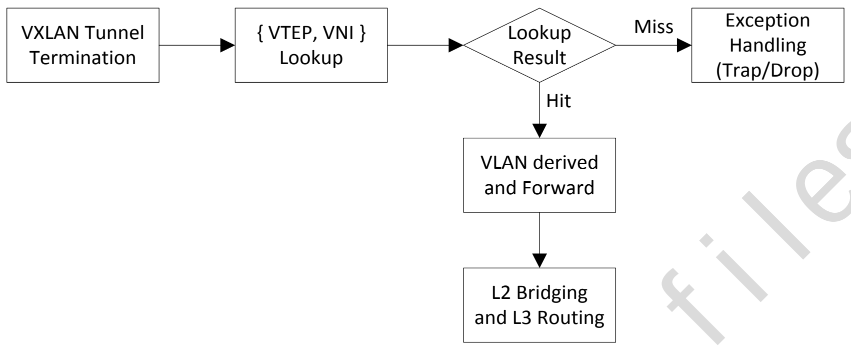
GRE 32-bit Key Check (Optional)

Each GRE tunnel interface is able to check the GRE key (32-bit value) independently.
The KEY values of the received packet and the tunnel interface must match if the check is enabled, otherwise the tunnel won't be terminated.

4.2.1.1.4 VXLAN Tunneling

Virtual Extensible LAN (VXLAN) is a network virtualization technology which is often used in Data Centers. It uses a VLAN-like encapsulation technique to encapsulate Ethernet frames within UDP packets.

Figure 4-17 VXLAN Tunnel Decapsulation Flow

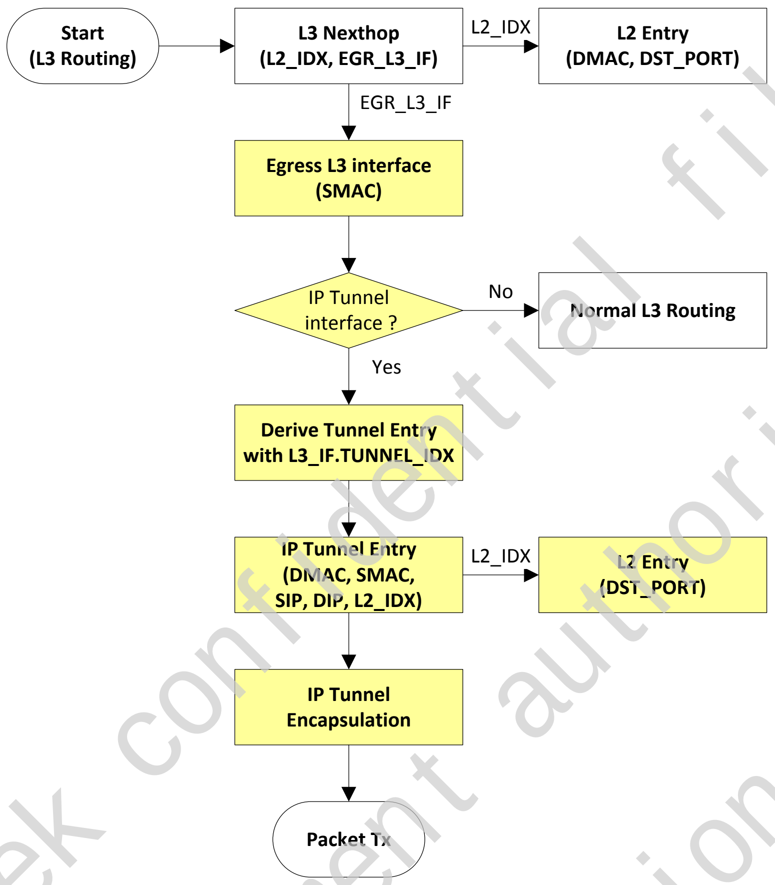


VXLAN Network Identifier (VNI)

The device supports up to 2048 VNI entries, each can be indexed by a combination of a VTEP (VXLAN tunnel interface) and a VNI. While VXLAN decapsulating, the VNI entry is used to derive the corresponding VLAN by { VTEP, VNI }. It is also conceptually a virtual port as an L2 interface.

4.2.2 Tunnel Encapsulation

Figure 4-18 Tunnel Encapsulation Process Flowchart



Note

The final destination port is always derived from L2 table. Packets will be flooded in the destination VLAN if the DST_PORT of the L2 entry has aged out.

4.2.2.1 Tunnel Interface Configuration

Encapsulating-related configurations are as below：

Remote IP (DIP)	IPv4/IPv6 address of remote tunnel end-point (Transmitting packet’s Destination IP address)
Local IP (SIP)	IPv4/IPv6 address of local tunnel end-point (Transmitting packet’s Source IP address)

Remote Port (DPORT)	L4 port number of remote tunnel end-point (Transmitting packet's Destination Port Number)
Local Port (SPORT)	L4 port number of local tunnel end-point (Transmitting packet's Source Port Number)
Flow Label (IPv6 only)	Specify the Flow Label ID of IPv6 tunnel header (Only available for IPv6 tunnels)
Don't Fragment (IPv4 only)	Specify Don't Fragment (DF) bit of IPv4 tunnel header. The DF bit status of IPv4 tunnel header would inherit from the passenger's IPv4 header by default. It also could be assigned with a specified DF status (optional). (Only available for IPv4 tunnels)
Tunnel TTL Assignment	To indicate the tunnel header's TTL. For *Uniform-mode* tunneling, it should be disabled (inherit). For *Pipe-mode* tunneling, it should be enabled and assign a specific TTL value.
QoS Profile Index	To specify a QoS profile for this tunnel interface.

The device provides total 64 sets of QoS profiles, each has the following configurations.

DSCP Assignment	To indicate the source of the DSCP of the tunnel header. ***Specify*** – Using a fixed DSCP value ***Passenger's*** – Inherit from passenger if it's existing ***Internal Priority*** – Remarking based on Internal Priority
Inner Priority Assignment	To indicate the source of the inner tag priority of the tunnel header. ***Specify*** – Using a fixed priority value ***Passenger's*** – Inherit from passenger (for L2 tunnel only) ***Internal Priority*** – Remarking based on Internal Priority
Outer Priority Assignment	To indicate the source of the outer tag priority of the tunnel header. ***Specify*** – Using a fixed priority value ***Passenger's*** – Inherit from passenger (for L2 tunnel only) ***Internal Priority*** – Remarking based on Internal Priority

Table 4-25 Encapsulation-related Tunnel Interface Configuration

	IP-in-IP	ISATAP	6to4	6rd	GRE	VXLAN
Remote IP (DIP)	●	○	○	●	●	●
Local IP (SIP)	●	●	●	●	●	●
Remote Port (DPORT)	○	○	○	○	○	●
Local Port (SPORT)	○	○	○	○	○	●

Flow Label (IPv6 only)	●	○	○	○	●	●
Don't Fragment (IPv4 only)	●	●	●	●	●	●
Tunnel TTL Assignment	●	●	●	●	●	●
QoS Profile Index	●	●	●	●	●	●

4.2.2.1.1 IPv4/IPv6 Tunneling

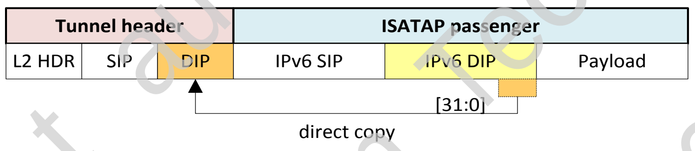
Configured tunnels must indicate a specific remote end-point. Therefore, the remote IP address of the remote end-point has been pre-configured. It's the major difference between *Configured Tunnel* and *Auto Tunnel*.

4.2.2.1.2 Automatic Tunneling

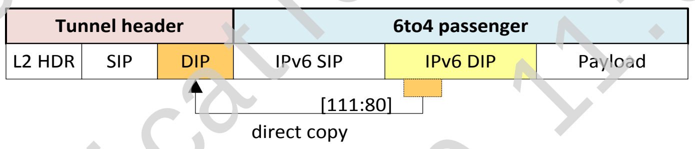
Each *Auto Tunnel* has different mechanism to automatically generate the header's destination IP address (DIP), therefore the high-layer software has to configure the L3 routing table carefully.

The software must guarantee the L3 entry (as destination host or network) is matched with the outgoing L3 interface. (i.e., L3 ISATAP routing entry → ISATAP tunnel interface)

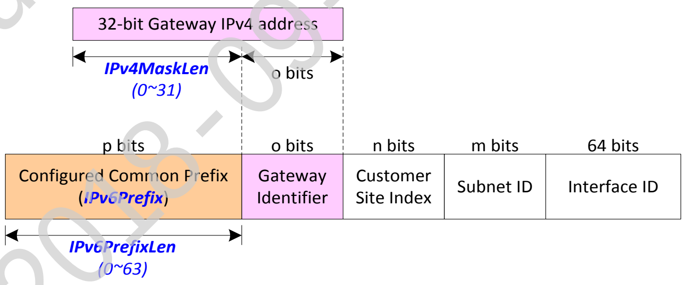
ISATAP The tunnel's DIP is automatically mapped from passenger's DIP (the lowest 32 bits).



6to4 The tunnel's DIP is mapped from passenger's DIP bit[111:80].



6rd The tunnel's DIP is automatically generated with the *6rd domain* associated with the tunnel interface.



4.2.2.1.3 GRE Tunneling

The device supports the additional configuration as below.

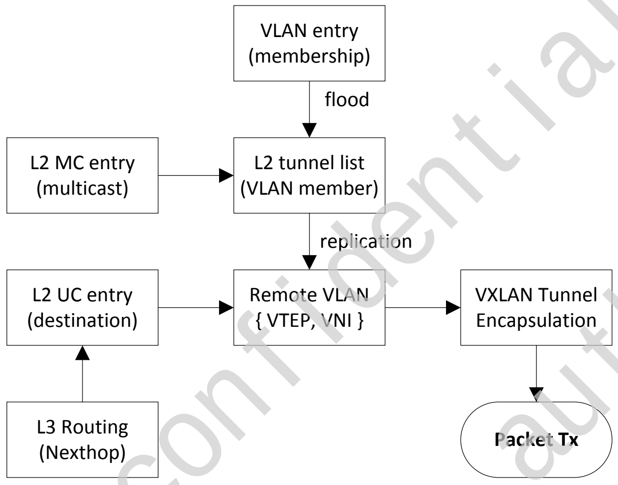
GRE 32-bit Key (Optional)

Each GRE tunnel interface is able to insert a GRE key (32-bit value) independently.

4.2.2.1.4 VXLAN Tunneling

VXLAN tunnel could be initialed by VLAN flooding or DA lookup (Unicast and Multicast).

Figure 4-19 VXLAN Tunnel Encapsulation Flow


VLAN entry (membership)
flood
L2 MC entry (multicast)
L2 tunnel list (VLAN member)
replication
L2 UC entry (destination)
Remote VLAN { VTEP, VNI }
VXLAN Tunnel Encapsulation
L3 Routing (Nexthop)
Packet Tx

VXLAN Network Identifier (VNI)

VNI is used to identify a VLAN (broadcast domain) associated with a VTEP. The VNI could also be globally unique in a management domain across different locations of Data Centers.

5 MPLS

5.1 MPLS Overview

Multi-Protocol Label Switching (MPLS) can be used to carry many different types of traffic such as IP, ATM, SONET and Frame Relay. MPLS offers Enterprises and Service Providers value-added services over a single infrastructure.

Layer2 Header	Top Label	....	Bottom Label	Layer3 Header

MPLS Header

Label (20b)	TC (3b)	S (1b)	TTL (8b)

The system supports MPLS Layer 3 Virtual Private Network (VPN). MPLS L3VPN uses a peer-to-peer model that uses higher layer protocols to distribute VPN-related information. The system also supports MPLS label operation in label switch router (LSR) and label edge router (LER), MPLS DiffServ and MPLS ECMP.

The system supports up to 2 MPLS label lookups per packet. The system supports **PHP**, **POP**, **PUSH** and **SWAP** operation. The system pops and pushes up to 2 labels. And supports swap 1 label and push 1 label. The system supports 2K labels lookup.

5.2 MPLS Decapsulation

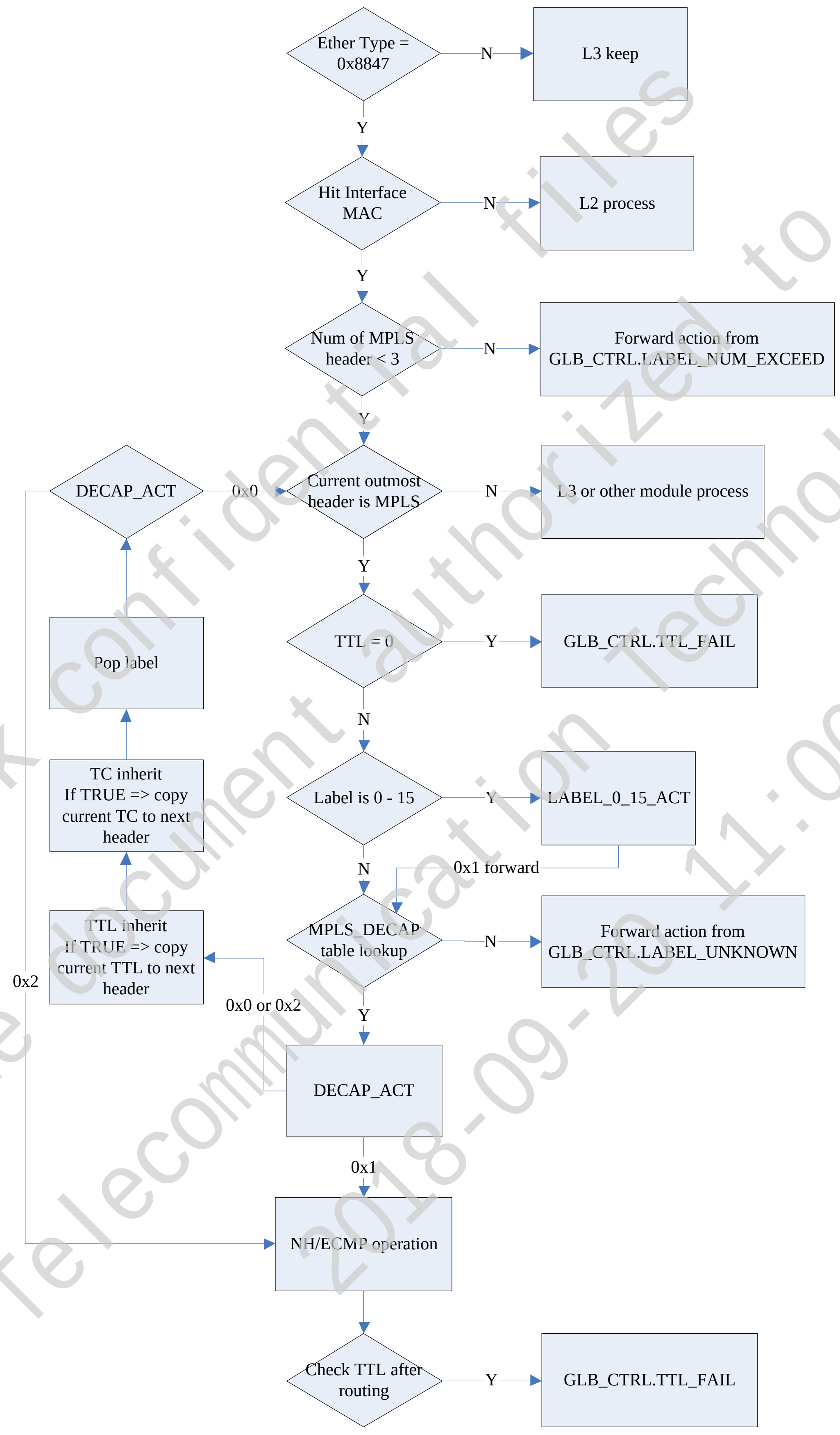
The system supports Ether-Type = 0x8847 to treat as MPLS packet. Before MPLS decapsulate, the system performs tunnel interface MAC lookup to check whether tunnel decapsulation is needed.

The system supports 2K MPLS decapsulate entries. The key of decapsulate entry is **LABEL**. Each entry has **POP**, **SWAP** and **PHP** actions. The system supports up to 2 MPLS label lookups. If the outermost label of packet is hit decapsulate entry, which is configured POP action, the system lookups the next label of packet. When pop MPLS label, entry outputs ingress L3 interface. The configuration of innermost label decapsulation will overwrite the configurations of in front labels.

For SWAP action, user needs to configure MPLS decapsulation and MPLS encapsulation. MPLS decapsulation entry supports next hop/ECMP index which pointer to MPLS encapsulation entry. Both decapsulation and encapsulation entry are configured SWAP action.

For uniform, pipe and short-pipe mode, each entry supports TTL inherit, TC inherit actions. Each entry can select priority selection group for E-LSP. Or each entry can assign internal priority for L-LSP.

5.2.1 Packet Processing Flow



5.2.2 TTL & TC

For uniform mode, the TC and TTL will inherit to next label or IP header.

For pipe or short pipe, the TC and TTL will not change next label or IP header.

Table 5-1 TC behavior

Tunnel mode	IP -> MPLS (Ingress LER)	MPLS -> MPLS (LSR)	MPLS -> IP (Egress LER)
Uniform	Copy IP precedence / DiffServ into MPLS TC	MPLS TC may be changed by service provider.	MPLS TC copied to IP precedence / DiffServ
Pipe	MPLS TC set by service provider QoS policy		Original IP precedence / DiffServ preserved (egress queue based on MPLS TC)
Short-Pipe			Original IP precedence / DiffServ preserved (egress queue based on IP precedence / DiffServ)

Table 5-2 E-LSP and L-LSP behavior

	E-LSP	L-LSP
Class	From TC mapping	depends on label mapping
DP	From TC mapping	From TC mapping

5.2.3 Hash Algorithm

The component of 2K MPLS decapsulation table is 256 buckets, per bucket has 8 entries. The system supports 2 hash algorithms with hash key is label.

Table 5-3 MPLS_GLB_CTRL Register

Bits	Field	Description
3:3	HASH_SEL	Select MPLS decapsulation table hash algorithm 1b0: hash algorithm 0 1b1: hash algorithm 1

5.2.4 Exception Handle

The system supports TTL_FAIL, number of label of packet is over 2 labels and unknown label action for unexpected packet.

Table 5-4 MPLS_GLB_CTRL Register

Bits	Field	Description

4:4	TTL_FAIL	Incoming or outgoing MPLS packet TTL = 0 1b0: drop 1b1: trap to CPU
1:1	LABEL_NUM_EXCEED	Incoming packet has over 2 MPLS label 1b0: drop 1b1: trap to CPU
0:0	LABEL_UNKW	Unknown MPLS label 1b0: drop 1b1: trap to CPU

For special-purpose MPLS label values (0 ~ 15), the system supports to trap to CPU or forward (lookup decapsulation table)

Table 5-5 MPLS_GLB_CTRL Register

Bits	Field	Description
5:5	LABLE_0_15_ACT	Forward action for reserved label 1b0: trap to CPU 1b1:forward

5.3 MPLS Encapsulation

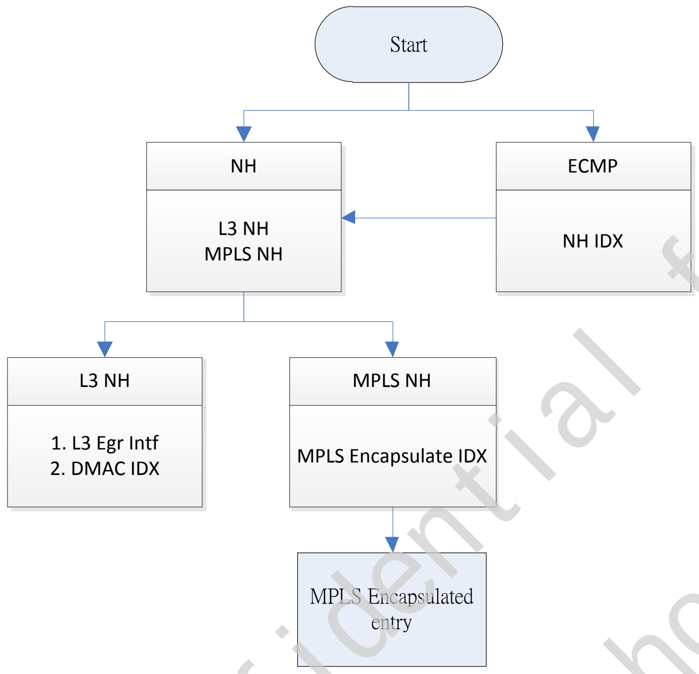
When packet arrives from IP domain to MPLS domain (ingress LER) or MPLS LSR, the system supports MPLS encapsulation to **PUSH** MPLS header or **SWAP** MPLS label.

The encapsulation entry is pointer by next hop (NH) entry. For ingress LER, packet is lookup in L3 table, then L3 entry outputs NH index. The NH index is pointer to L3 NH and MPLS NH.

The system supports up to PUSH 2 labels. The encapsulation entry has next index for PUSH next MPLS label. When packet has two MPLS labels, the encapsulation action will be ignored.

For LSR, packet is lookup in decapsulation table, then decapsulation entry outputs NH for SWAP label.

5.3.1 Packet Processing Flow



5.3.2 TTL & TC

Each encapsulation entry can configured TTL is forced assign or copy from inner MPLS TTL/IP TTL. And supports force TC, copy from inner MPLS TC/IP DSCP or internal priority mapping.

6 BPE

In data center topology, network virtualization reduces the maintenance cost but brings on the management complexity. The traditional network policy is hard to apply on the endpoint (virtual machine). IEEE 802.1BR defines BPE (Bridge Port Extender) which provides the capability to extend MAC service over an extended bridge. All the traffic in the extended bridge is forwarded through E-Channel and centrally managed by controlling bridge (CB).

E-channel is identified by ECID (E-channel Identifier) which is carried in the "E-TAG" of packet. E-TAG is 6 bytes field right after source MAC address, see Figure 6-1. The field definition of E-TAG shows in the Figure 6-2.

Figure 6-1 E-Channel Packet Format

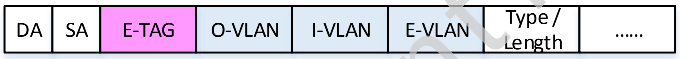
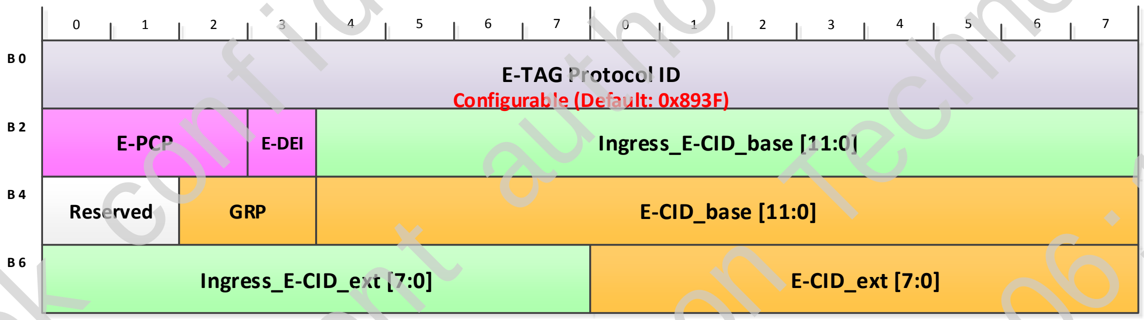


Figure 6-2 E-Tag Format



E-TAG Protocol ID is 2 bytes Ethertype for E-TAG which is configurable (default value is 0x893F). E-PCP/E-DEI is the QoS attribute as those in VLAN tag. Ingress E-CID (ext[7:0], base[11:0]) is used to identify source endpoint for point-to-multipoint downstream. E-CID (GRP[1:0], ext[7:0], base[11:0]) is used to identify E-Channel for this stream. If the GRP is 0, this E-Channel is point-to-point (Unicast). Otherwise, this E-Channel is point-to-multipoint while GPR is 1~3. Base bit of E-CID provides 1~4095 E-Channel. Extension (ext) bit extends the E-Channel up to 1048575.

Table 6-1 PE_ETAG_MAC_CTRL Register

Bits	Field	Description
15:0	TPID	Ethertype of E-TAG, used to parse and encapsulate a packet.

6.1 Extended Bridge

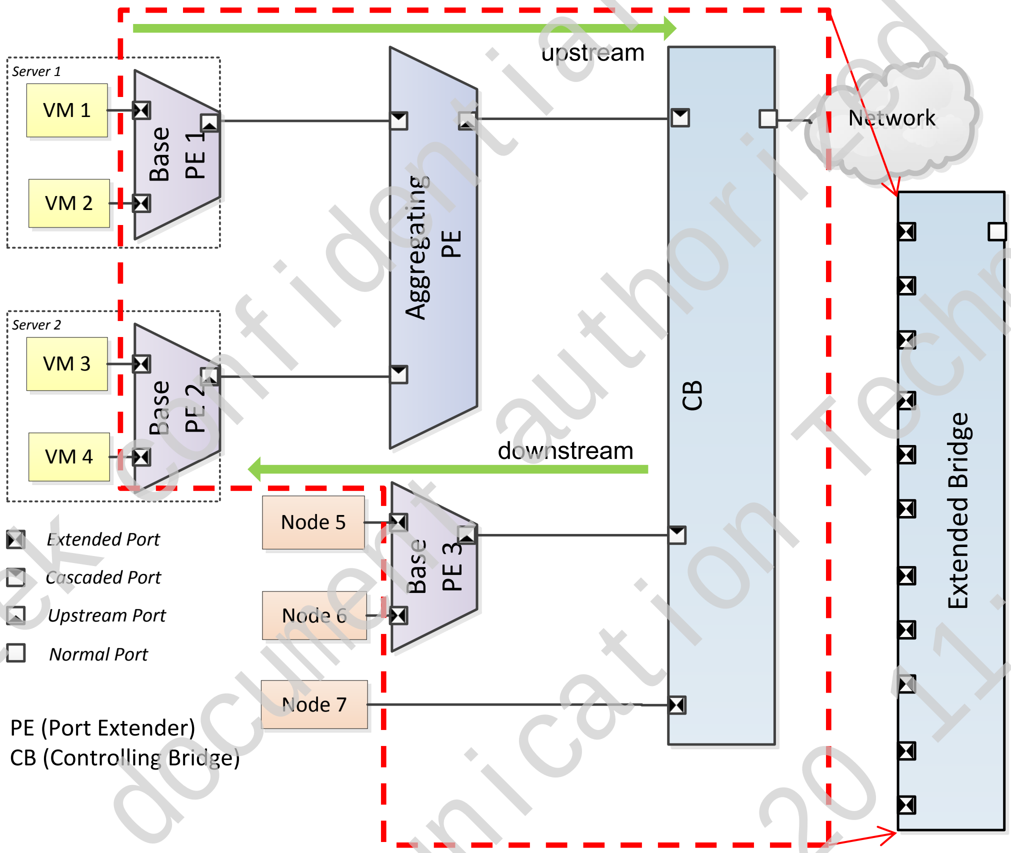
Figure 6-3 shows an example of extended bridge which includes one CB (Controlling Bridge) and four PEs (Port Extenders). PE CSP (Port Extender Control and Status Protocol) runs between CB and PEs to construct the forwarding path. Endpoint node 7 is connected to CB directly. Endpoint node 5 and node

6 access network via base PE 3. The base PEs for endpoint VM 1~4 access network through aggregating PE which can extend the capacity of E-Channel by the extension bits of E-CID.

All the upstream from endpoint are tagged with E-TAG and forwarded to CB for central management. CB forwards these packets based on DA lookup and replaced the E-CID of E-TAG to the corresponding E-Channel. In PEs, the downstream from CB is forwarded by the E-CID in the E-TAG instead of destination MAC address.

The extended bridge could be considered as a single bridge. The physical access ports on the CB are extended as 11 virtual ports (7 for endpoints and 4 for PEs). In another word, the definition of virtual port in a CB is {PORT_ID, ECID}.

Figure 6-3 Extended Bridge



There are four types of port role in an Extended Bridge:

Extended port

Endpoint is connected to extended port directly. Extended port assigns the PCID (Port E-CID) as E-CID for upstream without E-TAG. On the other hand, extended port removes the E-TAG of downstream.

Cascaded port

Cascaded port is the access port which is not connected to endpoint directly. More than one E-Channel accesses network through cascaded port. Cascaded port is capable to expand the E-channel capacity. The extension bits of E-CID may be replaced as the extension bits configured at this cascaded port if "USE_DEFAULT" is enabled.

Upstream port

All the upstream received by PE should be forwarded to upstream port which is directed to CB. If the E-CID of the downstream received by upstream port is the same as the PCID, device should trap this packet up the CPU. Otherwise, PE lookups the forwarding table based on the E-CID of E-TAG instead of destination MAC address.

Normal Port

Normal port in the CB represents as the port which is not able to recognize or process E-TAG.

6.1.1 Extended Bridge Port Configuration

System provides control register to achieve the behavior of each port role.

Table 6-2 PE_PORT_ETAG_MAC_CTRL Register

Bits	Field	Description
0	ETAG_EN	Enable to parse and forward E-TAGed packet 0b0: disable (forward E-tagged packet as an unknown ethtype L2 packet) 0b1: enable

Table 6-3 PE_PORT_ETAG_IGR_CTRL Register

Bits	Field	Description
11	ECID_RPF_CHK	Enable bit for ECID RPF check when lookup ECID_PVID table 0b0: Disable 0b1: Enable
10	RSVD_ECID_FLTR	Filter of E-TAGed packet with reserved E-CID (0x0 and 0x3FFFFF). 0b0: forward 0b1: drop the packet with reserved E-CID if received on the port. Note: it should be turned on when it's a Cascaded Port.
9	MC_ECID_FLTR	Filter of E-TAGed packet with multicast E-CID (GRP != 0). 0b0: forward 0b1: drop the packet with multicast E-CID if it's received on the port
8:1	ECID_NSG	Name Space Group (NSG) of E-CID. Use to logically separate the E-channel ID's name space.

0	LU_KEY_SEL	Lookup key of L2 forwarding table while relaying the packet.
		0b0: Normal {FID, DMAC}
		0b1: E-CID {RX_PORT.NSG , E-CID}

Table 6-4 PE_PORT_ETAG_EGR_CTRL Register

Bits	Field	Description
26	VLAN_TAG_STS_MODE	Egress VLAN tag status mode 0x0: normal (follow VLAN table untagged set) 0x1: MC tagged (insert VLAN tag if ECID.GRP!=0); Otherwise, follow VLAN table untagged set.
25	TAG_DEI_RMK_EN	Enable bit of egress E-TAG's DEI remarking. 0b0: disable (should keep the original value if it exists) 0b1: enable
24	TAG_PRI_KEEP	To indicate keeping the E-TAG priority value if the original packet has one. 0b0: from internal priority 0b1: force to use the original value if it's present

23:22	TAG_STS_MODE	Egress E-TAG status mode 0b0: untagged (always remove E-TAG) 0b1: tagged (always insert E-TAG) 0b2: tagged if UC and (ECID != PCID); Otherwise, untagged. 0b3: untagged if UC and (ECID==PCID); Otherwise, tagged.

Table 6-5 PE_PORT_PCID_CTRL Register

Bits	Field	Description
24:23	ECID_EQ_PCID_ACT	Forwarding action of received packet's ECID is equal to Rx Port's PCID. 0x0: forward (do nothing) 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU
22	PE_SRC_PORT_FLTR	Source extended port filtering of Port Extension. 0b0: disabled (forward the packet, don't need to check) 0b1: enabled (filter packet by Ingress E-CID check) Note: It should be enabled on Extended Port.
21	USE_DEFAULT	Used by aggregating bridge port extender mode. Indicate to replace the extension value of packet's E-CID. 0b0: false (use the original extension value) 0b1: true (use the PCID's extension value)
20	BASE_PE	Indicate the E-CID capability for port extender. 0b0: false (aggregating bridge port extender, support full E-CID) 0b1: true (base bridge port extender, do not support extension value at both incoming and outgoing ECID process)
19:12	PCID_EXT	Extension of the port-based default E-channel ID.
11:0	PCID_BASE	Base of the port-based default E-channel ID.

The followings are the suggested configuration for each port role in an extended bridge.

Table 6-6 Example for CB Port Configuration

Feature	Cascaded Port (1BR enabled)	Extended Port (1BR enabled)	Normal Port (1BR disabled)
VLAN member port	True (4k VLANs)	True (4k VLANs)	-
VLAN untag set	False	(Configured by S/W)	-
VLAN ingress filter	Disabled	Disabled	-
VLAN egress filter	Disabled	Disabled	-
Source (Physical) Port Filter	Disabled	Disabled	Enabled
ETAG_EN	Enabled (Parse)	Enabled (Parse)	-
ECID_EQ_PCID_ACT	Forward (bypass)	Forward (bypass)	-

PE_SRC_PORT_FLTR	Disabled	Enabled (Filter)	Disabled
RSVD_ECID_FLTR	Enabled (Filter)	Disabled	-
MC_ECID_FLTR	Enabled (Filter)	Enabled (Filter)	-
LU_KEY_SEL	CB mode (0:Normal)	CB mode (0:Normal)	CB mode (0:Normal)
TAG_STS_MODE	UC check, MC tagged (mode 3)	Untagged (mode 0)	-
VLAN_TAG_STS_MODE	Mcast tag (1)	Normal (0)	Normal (0)
PCID_EXT/ PCID_BASE	Valid Value	Valid Value	0
ECID_RPF_CHK	Enabled	Enabled	Disabled

Table 6-7 Example for Common PE Port Configuration

Feature	**Upstream Port**	**Cascaded Port**	**Extended Port**
VLAN member port	True (4k VLANs)	True (4k VLANs)	True (4k VLANs)
VLAN untag set	False	False	(Configured by S/W)
VLAN ingress filter	Disabled	Disabled	Disabled
VLAN egress filter	Disabled	Disabled	Disabled
Port isolation	All Cascade/Extended ports	Upstream port	Upstream port
Port-based L2 SA learning	Disabled	Disabled	Disabled
Various flooding portmask	True (set)	True (set)	True (set)
ETAG_EN	Enabled (Parse)	Enabled (Parse)	Enabled (Parse)
ECID_EQ_PCID_ACT	Trap (to CPU)	Forward (bypass)	Forward (bypass)
PE_SRC_PORT_FLTR	Disabled	Disabled	Enabled (Filter)
RSVD_ECID_FLTR	Disabled	Enabled (Filter)	Disabled
MC_ECID_FLTR	Disabled	Enabled (Filter)	Enabled (Filter)
LU_KEY_SEL	PE mode (1: ECID)	PE mode (1: ECID)	PE mode (1: ECID)
TAG_STS_MODE	UC check, MC don’t care (mode 2 or 3)	UC check, MC tagged (mode 3)	Untagged (mode 0)

VLAN_TAG_STS_MODE	Normal (0)	Normal (0)	Normal (0)
USE_DEFAULT	Disabled	Enabled	-
ECID_RPF_CHK	Disabled	Disabled	Disabled

In addition to the common configuration for PE, the difference between aggregating PE and base PE is listed in Table 6-8 and Table 6-9.

Table 6-8 Example for Aggregating PE Port Configuration

Feature	Upstream	Cascaded	Extended
BASE_PE	Disabled	Disabled	Disabled
PCID_EXT	Valid Value	Valid Value	Valid Value
PCID_BASE	Valid Value	Valid Value	Valid Value

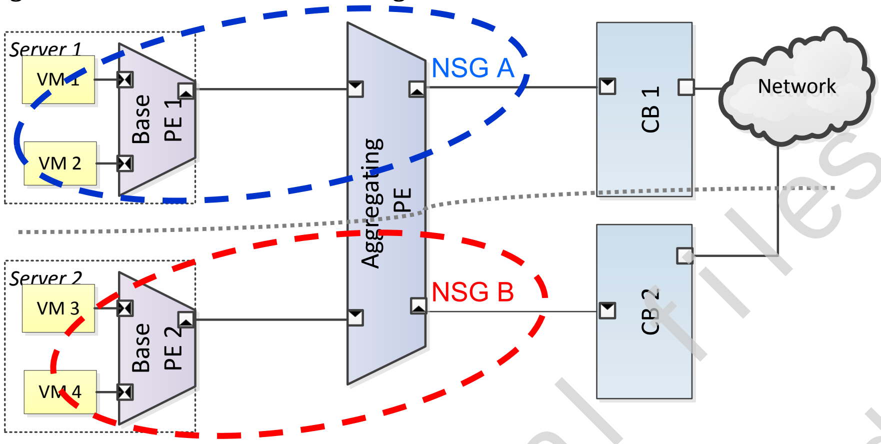
Table 6-9 Example for Base PE Port Configuration

Feature	Upstream	Extended
BASE_PE	Enabled	Enabled
PCID_EXT	0	0
PCID_BASE	Valid Value	Valid Value

6.1.2 Name Space Group

NSG (Name Space Group) provides virtual domain in the PE. Upstream port could be configured with a non-zero NSG value. PE lookups the forwarding table according to {RX_PORT.NSG, E-CID}. Each E-CID should be unique in the same NSG. The example shows in the Figure 6-4.

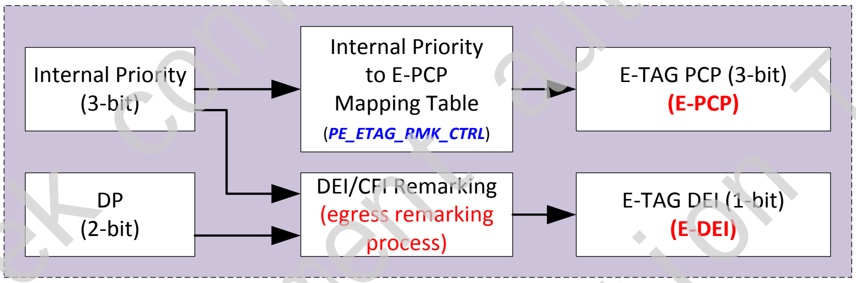
Figure 6-4 NSG of Extended Bridge



6.1.3 Remark E-PCP and E-DEI

E-TAG carries PCP and DEI value those can be remarked or kept as the original value if E-TAG is received. Figure 6-5 shows the remarking decision.

Figure 6-5 Remark Decision for E-PCP and E-DEI


Internal Priority (3-bit)
DP (2-bit)
Internal Priority to E-PCP Mapping Table (PE_ETAG_RMK_CTRL)
DEI/CFI Remarking (egress remarking process)
E-TAG PCP (3-bit) (E-PCP)
E-TAG DEI (1-bit) (E-DEI)

6.1.4 PVID

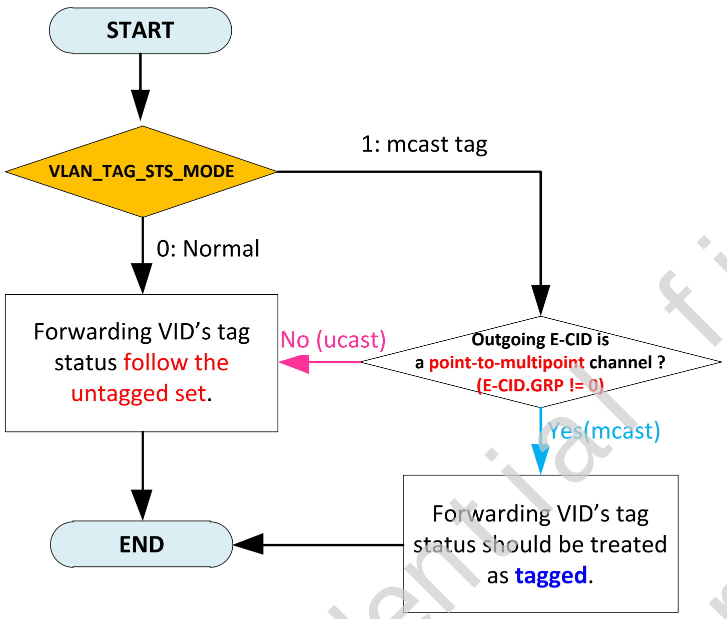
Each cascaded port on a CB is mapped to several external extended ports via various E-Channels. The PVID of physical port is no longer to be represented for related external extended ports. For E-TAGed stream without VLAN tag, CB determines the PVID according to the preconfigured ECID_PVID table. The lookup key of ECID_PVID table is {RX_PORT.NSG, E-CID}. Only the point-to-point (unicast) E-Channel is able to specify corresponding PVID via ECID_PVID table.

The ECID_PVID table may be used for ECID_PVID or MPLS_DECAP. The mode selection of this table should be configured properly before utilizing as ECID_PVID mode.

6.1.5 Egress VLAN Tag Decision

According to the requirement in IEEE 802.1BR, point-to-multipoint (multicast) E-Channel from CB should be always tagged with VLAN tag. Per port is capable to configure VLAN_TAG_STS_MODE to determine the egress VLAN tag decision. NORMAL mode follows the VLAN untagged set. MCAST_TAG mode always inserts VLAN tag for multicast E-Channel and follows the VLAN untagged set for unicast E-Channel.

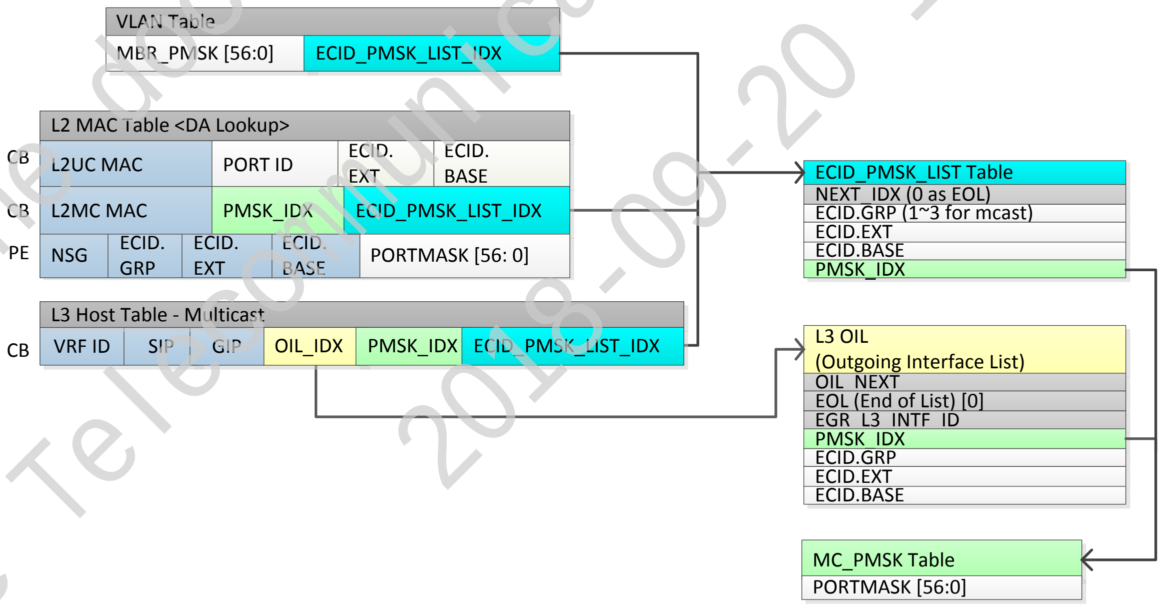
Figure 6-6 VLAN Tag Decision for E-Channel


START
VLAN_TAG_STS_MODE
1: mcast tag
0: Normal
Forwarding VID's tag status follow the untagged set.
No (ucast)
Outgoing E-CID is a point-to-multipoint channel ? (E-CID.GRP != 0)
Yes(mcast)
Forwarding VID's tag status should be treated as tagged.
END

6.2 BPE Packet Process

Figure 6-7 shows the BPE related table. For unknown flooding or multicast stream out from CB, packet is replicated by ECID_PMSK_LIST entry which specifies the outgoing E-Channel and egress port list. The non-zero NEXT_IDX of ECID_PMSK_LIST entry indicates that this packet is allowed to be replicated in more than one E-Channel. If the index of ECID_PMSK_LIST is zero at VLAN/L2MC/L3 Host-Multicast table, device doesn't replicate this packet and only forwards to the normal ports. The normal ports are specified in MBR_PMSK[56:0] field of VLAN entry or PMSK_IDX in L2MC/L3 Host-Multicast entry.

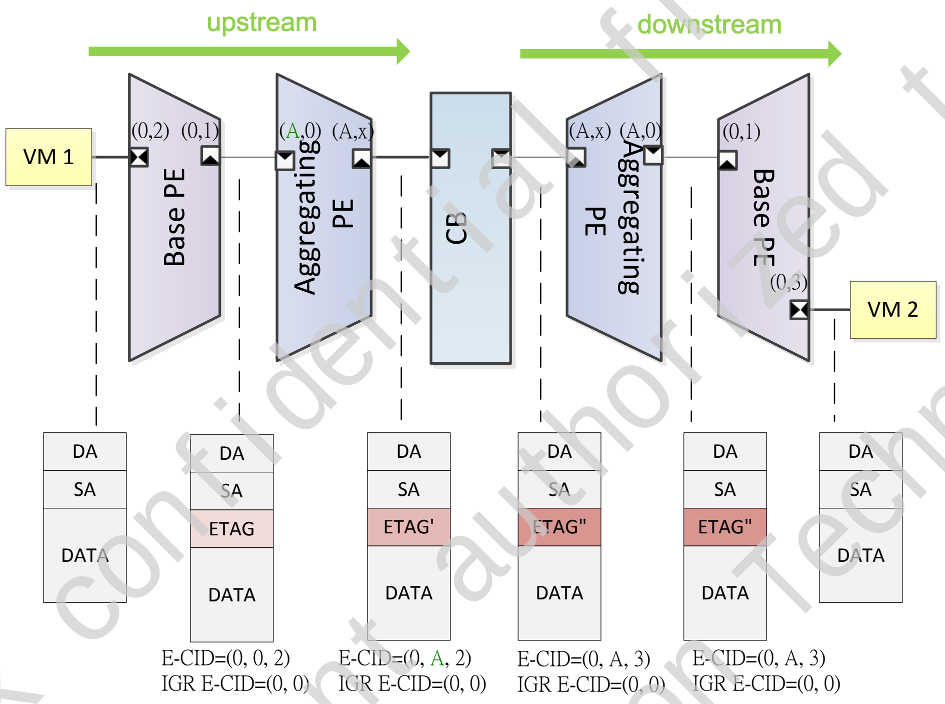
Figure 6-7 BPE Related Table



The ECID_PMSK_LIST table is configurable for ECID port mask list or MPLS next hop. The mode selection of this table should be configured properly before utilizing as ECID_PMSK_LIST mode.

6.2.1 BPE L2 Unicast Flow

Figure 6-8 BPE L2 Unicast Flow



In Figure 6-8 , we assume that all the ports are under the same NSG.

When base PE receives L2 unicast from extended port, this upstream is redirected to uplink port by port isolation and inserted with E-TAG. The E-CID of E-TAG is determined by extended port's PCID if E-TAG of incoming stream is not present.

Before arriving at CB, the aggregating PE may modify the extension bit of E-CID to expand the capacity of the E-Channel managed by CB.

CB receives this E-TAGed stream and forwards it via DA lookup. The MAC address is learned in the L2 table with E-CID if CB enables L2 SA learning. According to DA lookup result, the E-CID of E-TAG is replaced as the value in the L2 table.

The downstream with replaced E-CID is forwarded to the responsible PE. The uplink port of this PE receives downstream and lookups the destination through {RX_PORT.NSG, E-CID} instead of {FID, DMAC}. PE removes E-TAG the egress port's PCID is equal to the E-CID of E-TAG; otherwise, the E-TAG is kept and forwarded to the next PE. The base PE ignores the extension bit of E-CID and uses {RX_PORT.NSG, ECID_BASE} as lookup key.

For both upstream and downstream, the GRP bit of E-CID set as 0 and represents point-to-point E-Channel.

The ingress (IGR) E-CID is useless in unicast stream and set as zero.

6.2.1.1 DLF

CB may flood unknown unicast packet through ECID_PMSL_LIST with specified E-CID and port list. Packet is replicated via ECID_PMSK_LIST entry which is pointed by VLAN table. If packet is not only replicated in E-Channel but also flooded to normal ports, make sure that only the normal ports are configured in L2_UNKN_UC_FLD_PMSK.

Table 6-10 L2_UNKN_UC_FLD_PMSK Register

Bits	Field	Description
56:32	PMSK_1	Bit <56:32> of flooding port mask for unknown unicast packet.
31:0	PMSK_0	Bit <31:0> of flooding port mask for unknown unicast packet.

6.2.1.2 ECID RPF Check

ECID_PVID table provides the PVID for packet with matched {RX_PORT.NSG, E-CID}. Moreover, it could be populated with allowed port ID for ECID RPF check. If ECID_RPF_CHK is enabled at ingress port, the allowed port ID of matched ECID_PVID entry is to be compared with ingress port. ECID_RPF_FAIL_ACT is applied to the packet which E-CID is not found or ingress port is mismatched.

Table 6-11 L2_CTRL Register

Bits	Field	Description
18:16	ECID_RPF_FAIL_ACT	Action for ECID RPF check failed. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved

6.2.1.3 PE Forwarding Lookup Miss Filter

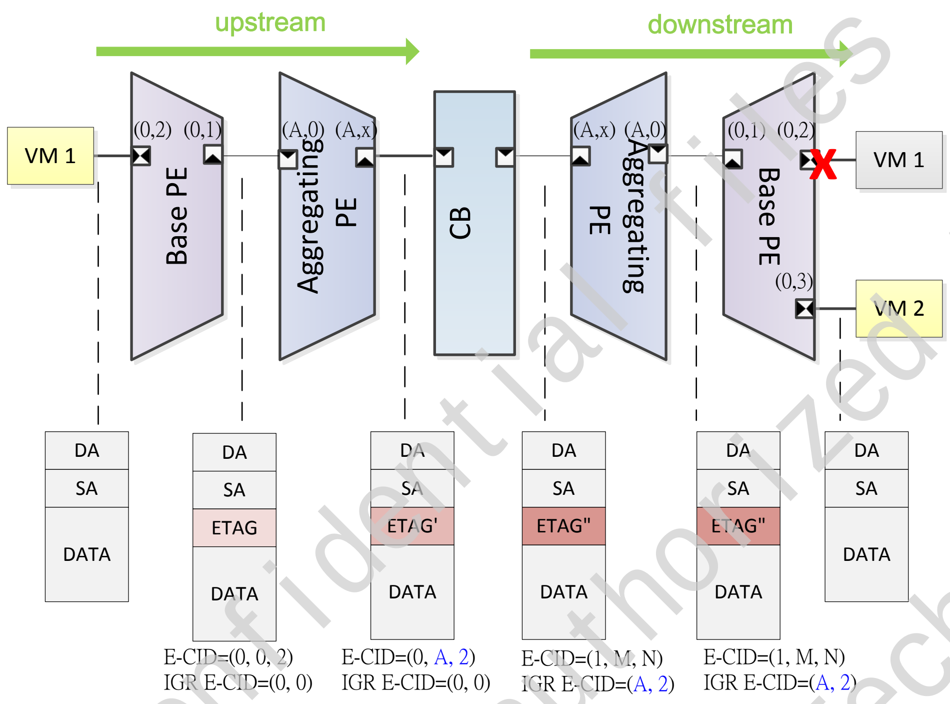
PE lookups the destination via lookup key {RX_PORT.NSG, E-CID}. If lookup miss, the forwarding behavior is determined by PE_LM_FLTR.

Table 6-12 L2_CTRL Register

Bits	Field	Description
15	PE_LM_FLTR	Enable bit to filter packet if PE forwarding table lookup miss 0b0: disable (flood to potential ports) 0b1: enable (drop PE lookup miss packet)

6.2.2 BPE L2 Multicast Flow

Figure 6-9 BPE L2 Multicast Flow with Source Extended Port



In Figure 6-9, we assume that all the ports are under the same NSG.

Similar as the upstream part of BPE L2 unicast flow, multicast upstream is inserted with E-TAG which E-CID is the extended port's PCID. CB replicates packet with replaced multicast E-CID specified in ECID_PMSK_LIST entry which is pointed by L2_CB_MC entry. The ingress E-CID of E-TAG is set as the original E-CID. The downstream with replaced E-CID and non-zero ingress E-CID is forwarded to the responsible PE.

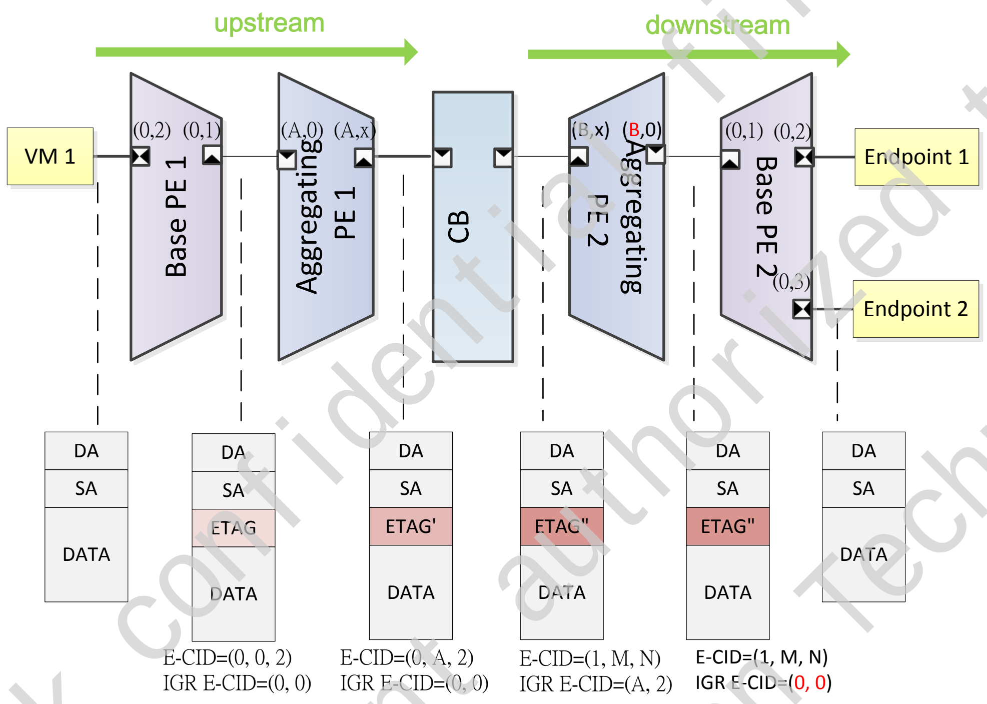
Source Extended Port Filter

Ingress E-CID of E-TAG identifies the source endpoint (extended port). To prevent multicast stream is redirected back to the source extended port, PE_SRC_PORT_FLTR should be enabled at extended port. The uplink port of PE receives downstream and lookups the destination through {RX_PORT.NSG, E-CID} instead of {FID, DMAC}. If the outgoing port enables PE_SRC_PORT_FLTR, packet is filtered when the outgoing port's PCID is equal to the ingress E-CID of E-TAG. Since the base PE can't process the extension bit, only the base bit of ingress E-CID is to be compared. PE_SRC_PORT_FLTR compares none-zero ingress E-CID and outgoing port's PCID when the GRP bit of E-CID is not zero (point-to-multipoint E-Channel).

Multicast stream may be replicated to extended ports and forwarded to normal ports at the same time. L2_CB_MC entry specified the normal ports in MCAST_PMSK_IDX field and extended ports through ECID_PMSK_LST_IDX entry.

The aggregating PE resets ingress E-CID as zero when USE_DEFAULT is enabled at outgoing port and the extension bit of ingress E-CID and PCID are different. In the other words, the different extension bit means that the source extended port is not cascaded under this aggregating PE. The ingress E-CID is reset for early terminated the process of source extended port check. See Figure 6-10.

Figure 6-10 BPE L2 Multicast Flow without Source Extended Port



7 QoS

7.1 Priority and Drop Precedence Decision

Quality of Service (QoS) is a technology for managing network traffic, and provides guarantee bandwidth on ability traffic to network applications or protocols. QoS function includes flow classification, policing (metering), shaping, scheduling and remarking. In flow classification, a received packet may have different priority-based priorities and only one priority is decided by Priority Selection to be unique 3-bit internal priority for queueing. Besides, 2-bit drop precedence is assigned to each received packet for using in policing (metering) module or Simplified Weighted Random Early Detection (SWRED).

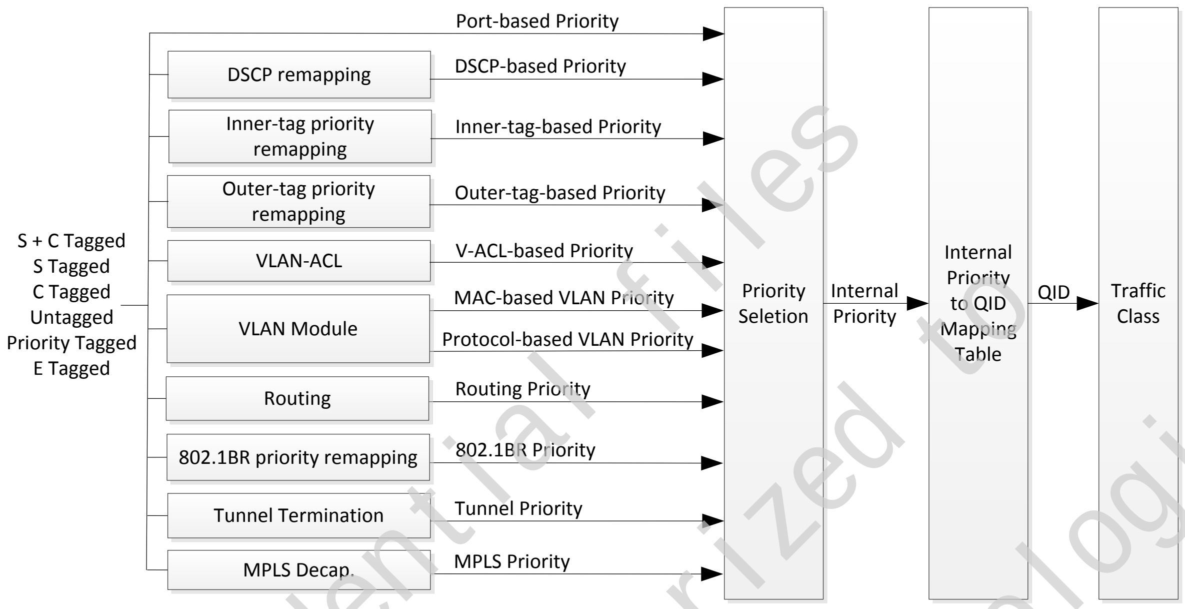
7.1.1 Priority Assignment

Traffic can be classified into traffic classes by different properties or requirements. In the device, 11 priority assignments listing as below are provided. Each priority assignment stands for a special property. Packet can be sent into managed traffic class by the priority assignments for different requirement. The priority assignments supported by the device are below:

- Port-based
- VACL-based
- DSCP-based
- Inner-tag-based
- Outer-tag-based
- Routing-based
- MAC-based/IP-subnet-based
- Protocol-and-port-based
- 802.1BR-based
- Tunnel-based
- MPLS-based

Only one priority assignment is selected to be unique 3-bit internal priority through Priority Selection Table for each forwarding packet. The internal priority is used to translate for getting QID through a global Internal Priority to QID Mapping Table. Figure 7-1 is Priority Decision flow chart.

Figure 7-1 Priority Decision Module



Note

RSPAN packet doesn't use the internal-priority decided by Priority Selection Table. The Internal-priority of RSPAN packet has different generation method. The detail can refer to section 7.1.3.

7.1.1.1 Port-based Priority Assignment

System supports 3-bit port-based priority for each RX port and the per-port register is PRI_SEL_REMAP_PORT.INTPRI_PORT in Table 7-1. Besides, system supports port-based priority to be treated as 802.1p Tag priority feature, and the feature can be set up by a global register PRI_SEL_CTRL.PORT_PRI_REMAP_EN in Table 7-2.

If PRI_SEL_CTRL.PORT_PRI_REMAP_EN=enabled, Inner-priority remapping table or Outer-priority remapping table is selected to translate port-based priority which is according to per-port register PRI_SEL_PORT_CTRL.PORT_PRI_REMAP_TBL_SEL configuration in Table 7-3.

Table 7-1 PRI_SEL_REMAP_PORT Register

Bits	Field	Description
2:0	INTPRI_PORT	Port-based internal priority of specified port.

Table 7-2 PRI_SEL_CTRL Register

Bits	Field	Description

11	PORT_PRI_REMAP_EN	Remap port-based internal priority by PRI_SEL_REMAP_IPRI_CFI0 or PRI_SEL_REMAP_OPRI_DEI0. 0b0: disable 0b1: enable (Remap)

Table 7-3 PRI_SEL_PORT_CTRL Register

Bits	Field	Description
0	PORT_PRI_REMAP_TBL_SEL	Select remapping table for port-based internal priority remapping. 0b0: use inner-priority with CFI 0 remapping table(PRI_SEL_REMAP_IPRI_CFI0) 0b1: use outer-priority with DEI 0 remapping table(PRI_SEL_REMAP_OPRI_DEI0)

7.1.1.2 VLAN-ACL-based Assignment

Incoming packets are checked by VLAN-ACL rules before priority decision module. If packets hit VLAN-ACL rule assigning 3-bit internal-priority, ingress ACL-based priority is assigned to it. Otherwise, ingress VLAN-ACL-based priority is NULL.

7.1.1.3 DSCP-based Priority Assignment

When IP packet is received, 6-bit DSCP value, left 6-bit of TOS field, is taken to transfer for getting DSCP-based priority by global DSCP Remapping Table as Table 7-4. DSCP-based priority is only applicable to IPv4 or IPv6 packets. For other packets, DSCP-based priority is NULL. For IPv6 packets, Traffic Class filed of IPv6 header is an 8-bit field as Figure 7-3, and it's equal to the type of services (TOS) byte in IPv4 header as Figure 7-2.

System provides an invalid DSCP value configuration in Table 7-5. If PRI_SEL_CTRL.DSCP_INVLD_EN=enabled, since DSCP value of received packet is equal to PRI_SEL_CTRL.DSCP_INVLD_VAL, the DSCP-based priority of the packet is treated as NULL.

Table 7-4 PRI_SEL_REMAP_DSCP Register

Bits	Field	Description
0	INTPRI_DSCP	Remap DSCP to internal priority.

Table 7-5 PRI_SEL_CTRL Register – Invalid DSCP Relative Configuration

Bits	Field	Description
8	DSCP_INVLD_EN	Invalid DSCP function state. 0b0: disable 0b1: enable
5:0	DSCP_INVLD_VAL	Invalid DSCP value. If DSCP of received packet matches the configuration, system treats DSCP-based priority as NULL.

Figure 7-2 IPv4 Header Format

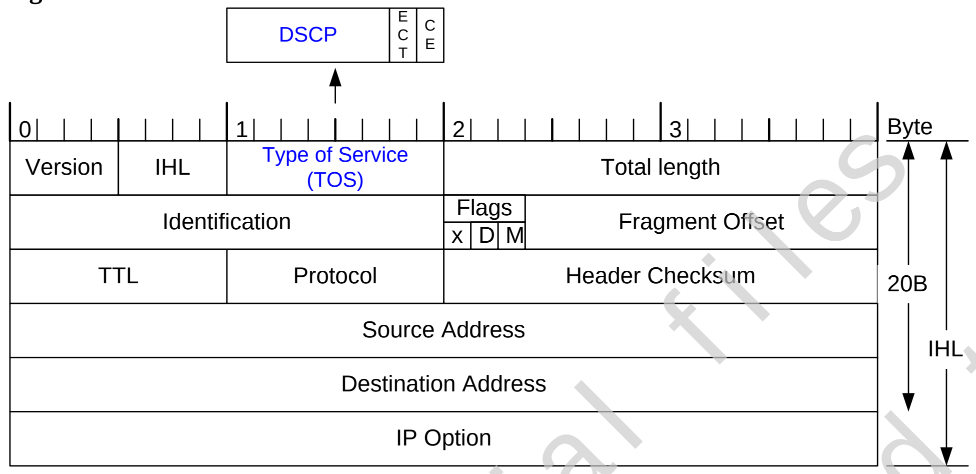
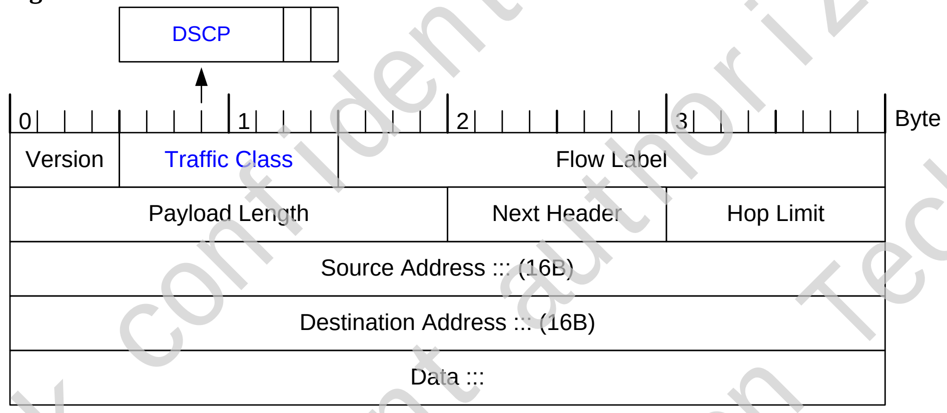


Figure 7-3 IPv6 Header Format



7.1.1.4 Inner-tag-based Priority Assignment

If received packet has inner-tag, system takes the 1-bit DEI and 3-bit priority of the inner-tag to translate for getting inner-tag priority value.

Two Inner-tag Priority Remap Tables are supported by system and they are showing in Table 7-6 and Table 7-7. The table index is 1-bit DEI and 3-bit priority. If DEI of the received packet is equal to 0, PRI_SEL_REMAP_IPRI_CFI0.INTPRI_CFI0_IPRI of Table 7-6 is used to translate inner-tag priority for getting Inner-tag-based priority. Similarly, if DEI is equal to 1, Table 7-7 is used.

For inner-untagged packet, inner-tag-based priority is NULL.

Table 7-6 PRI_SEL_REMAP_IPRI_CFI0 Register

Bits	Field	Description
2:0	INTPRI_CFI0_IPRI	Remap inner-tag priority n with CFI 0 to internal priority.

Table 7-7 PRI_SEL_REMAP_IPRI_CFI1 Register

Bits	Field	Description
2:0	INTPRI_CFI1_IPRI	Remap inner-tag priority n with CFI 1 to internal priority.

7.1.1.5 Outer-tag-based Priority Assignment

Outer-tag-based priority is similar to inner-tag-based priority. It's only working in outer-tag packets. For outer-untagged packet, outer-tag-based priority is NULL.

Two Outer-tag Priority Remap Tables are also supported by system and they are showing in Table 7-8 and Table 7-9. 1-bit outer-tag DEI and 3-bit outer-tag priority of receiving packet are used to be table index. If outer-tag DEI of the received packet is equal to 0, PRI_SEL_REMAP_OPRI_DEI0.INTPRI_DEI0_OPRI of Table 7-8 is used to translate outer-tag priority for getting Outer-tag-based priority. Similarly, if outer-tag DEI is equal to 1, Table 7-9 is used.

Table 7-8 PRI_SEL_REMAP_OPRI_DEI0 Register

Bits	Field	Description
2:0	INTPRI_DEI0_OPRI	Remap outer-tag priority n with DEI 0 to internal priority.

Table 7-9 PRI_SEL_REMAP_OPRI_DEI1 Register

Bits	Field	Description
2:0	INTPRI_DEI1_OPRI	Remap outer-tag priority n with DEI 1 to internal priority.

7.1.1.6 Routing-based Priority Assignment

When packet hit L3 IP unicast or IP multicast route entry, and the hit entry assigns internal priority (entry's QOS_AS=TRUE), the assigned priority of the entry (entry's QOS_PRI) is the Routing-based priority. Otherwise, the Routing-based priority is NULL.

7.1.1.7 MAC-based/IP-subnet-based Priority Assignment

Since packet hits MAC-based/IP-subnet-based VLAN configuration, it has MAC-based/IP-subnet-based VLAN priority. For other non-hit packets, MAC-based/IP-subnet-based VLAN priority is NULL. (Please refer VLAN Spec. for more details)

Similar to port-based priority, MAC-based/IP-subnet-based VLAN priority can be treated as inner-priority or outer-priority. System provides global 1-bit control register PRI_SEL_CTRL.MAC_IP_VLAN_PRI_REMAP_EN in Table 7-10for MAC-based/IP-subnet-based VLAN priority remapping. If PRI_SEL_CTRL.MAC_IP_VLAN_PRI_REMAP_EN is enabled and VLAN_TYPE of the hitting MAC-based/IP-subnet-based VLAN entry is set to Inner VLAN, Inner-tag Priority Remapping Tables in Table 7-6 and Table 7-7 are used to translate MAC-based/IP-subnet-based VLAN priority. If VLAN_TYPE of the hitting MAC-based/IP-subnet-based VLAN entry is set to Outer VLAN, Outer-tag Priority Remapping Tables in Table 7-8 and Table 7-9 are used to translate MAC-based/IP-subnet-based VLAN priority.

For tagged or priority-tagged packet, DEI of tag is used to select one of DEI-0 or DEI-1 Remapping Tables for remapping MAC-based/IP-subnet-based VLAN priority. For untagged, DEI-0 Remapping Table is used to translate MAC-based/IP-subnet-based VLAN priority.

Table 7-10 PRI_SEL_CTRL Register - MAC-based/IP-subnet-based Priority Relative Configuration

Bits	Field	Description

9	MAC_IP_VLAN_PRI_REMAP_EN	Enable or disable MAC-based/IP-subnet-based VLAN priority to remap by Inner-tag-based Priority Remapping Table or Outer-tag-based Priority Remapping Table. 0b0: disable 0b1: enable

7.1.1.8 Protocol-and-port-based Priority Assignment

If packet matches the Protocol-and-port-based VLAN configuration, the VLAN_MAC_BASED table field PRI is the Protocol-and-port-based priority. Otherwise, the Protocol-and-port-based priority is NULL.

As MAC-based/IP-subnet-based VLAN priority, system provides 1-bit global configuration PRI_SEL_CTRL.PROTO_VLAN_PRI_REMAP_EN in Table 7-11 for Protocol-and-port-based priority remapping. The behavior is same as MAC-based/IP-subnet-based VLAN priority. Please refer section 7.1.1.7 MAC-based/IP-subnet-based Priority Assignment to get the details.

Table 7-11 PRI_SEL_CTRL Register – Protocol-and-port-based Priority Relative Configuration

Bits	Field	Description
10	PROTO_VLAN_VLAN_PRI_REMAP_EN	Enable or disable Protocol-and-port-based VLAN priority to remap by Inner-tag-based Priority Remapping Table or Outer-tag-based Priority Remapping Table. 0b0: disable 0b1: enable

7.1.1.9 802.1BR-based Priority Assignment

Since received packet is carried E-Tag, the E-PCP (3-bit priority) and E-DEI (1-bit Drop Eligible Indicator) of E-Tag are taken to remap by global 802.1BR Remapping Table in Table 7-12 for getting 802.1BR-based Priority. If received packet doesn't have E-Tag, the value of 802.1BR-based Priority is NULL.

Table 7-12 PRI_SEL_REMAP_1BR Register

Bits	Field	Description
2:0	INTPRI_1BR	Remap E-PCP and E-DEI of 802.1BR E-TAG to internal priority.

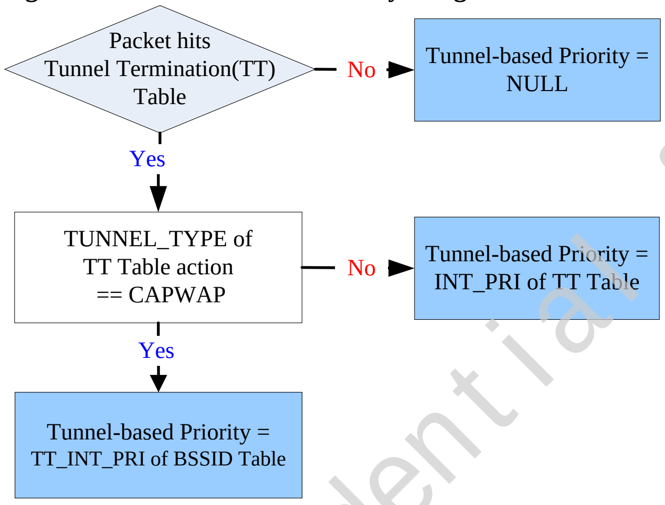
7.1.1.10 Tunnel-based Priority Assignment

Tunnel-based priority has value in packet hitting IPv4/IPv6 Tunnel Termination Table. Otherwise, Tunnel-based priority value is NULL. When packet hits IPv4/IPv6 Tunnel Termination Table, TUNNEL_TYPE of Tunnel Termination Table is used to decide Tunnel-based priority, and there are two situations in TUNNEL_TYPE as below.

- TULLEN_TYPE is equal to CAPWAP: TT_INT_PRI of BSSID Table is taken to be Tunnel-based priority.
- TULLEN_TYPE isn't equal to CAPWAP: INT_PRI of Tunnel Termination Table is treated as Tunnel-based priority.

Since packet hits Tunnel Termination Table, not only Tunnel-based priority is assigned, but also the index of Priority Selection Table is given. The priority of the given index of Priority Selection Table is higher than port-based index of Priority Selection Table described in section 7.1.2.

Figure 7-4 Tunnel-based Priority Assignment Decision Flow


Packet hits Tunnel Termination(TT) Table
No
Tunnel-based Priority = NULL
Yes
TUNNEL_TYPE of TT Table action == CAPWAP
No
Tunnel-based Priority = INT_PRI of TT Table
Yes
Tunnel-based Priority = TT_INT_PRI of BSSID Table

7.1.1.11 MPLS-based Priority Assignment

When packet hits MPLS Decapsulate Table and INT_PRI_EN of MPLS Decapsulate Table is equal to 0b1 (assign internal priority), 3-bit INT_PRI of MPLS Decapsulate Table is used to be MPLS-based priority. If packet hit MPLS Decapsulate Table and INT_PRI_EN of MPLS Decapsulate Table is equal to 0b0 (use TC_PRI_MAP priority remapping), traffic class (TC) of received MPLS packet is taken to remap by global MPLS TC Remapping Table in Table 7-13 for getting MPLS-based priority. In other situation, MPLS-based priority is NULL.

Table 7-13 PRI_SEL_REMAP_MPLS Register

Bits	Field	Description
2:0	INTPRI_MPLS_TC	Remap MPLS traffic class (TC) to internal priority.

7.1.2 Priority Selection

A packet may have different priorities from port-based, VLAN-ACL-based, DSCP-based, Inner-tag-based, Outer-tag-based, Routing-based, MAC-based/IP-subnet-based, Protocol-and-port-based, 802.1BR-based, Tunnel-based and MPLS-based assignments. One of these priorities is selected to be unique 3-bit internal priority for incoming packet according to one of Priority Selection Table. System supports global 4 Priority Selection Tables in Table 7-15 and per-ingress-port table index register PRI_SEL_PORT_TBL_IDX_CTRL.IDX in Table 7-14.

Weight value 0~12 can be configured to each priority assignment in Priority Selection Table. Weight 0 stands for ignoring the priority assignment. If weights of all priority assignments are configured to 0, system assigns the internal priority of the packet to be 0. Valid range of weight is 1 to 12, and 12 is the highest. Highest priority in priority assignments which have the same weight value is chosen to be internal priority for the received packet.

If the weights of all priority assignments are the same, the selected order from high to low of internal priority is as below: VLAN-ACL-based, Routing-based, MAC-based/IP-subnet-based, Protocol-and-port-based, DSCP-based, Outer-tag-based, Inner-tag-based, 802.1BR-based, Tunnel-based, MPLS-based and Port-based.

Table 7-14 PRI_SEL_PORT_TBL_IDX_CTRL Register

Bits	Field	Description
1:0	IDX	The index of priority selection table for a port.

Table 7-15 PRI_SEL_TBL_CTRL Register

Bits	Field	Description
63:48	RESERVE	
47:44	WT_1BR	Priority selection weight of 802.1BR-based Priority.
43:40	WT_MPLS	Priority selection weight of MPLS-based Priority.
35:32	WT_TUNNEL	Priority selection weight of Tunnel-based Priority.
31:28	WT_ROUT	Priority selection weight of Routing-based priority assignment
27:24	WT_PROTO_VLAN	Priority selection weight of Protocol-and-port-based VLAN Priority.
23:20	WT_MAC_VLAN	Priority selection weight of MAC-based VLAN Priority.
19:16	WT_OTAG	Priority selection weight of Outer-tag-based Priority.
15:12	WT_ITAG	Priority selection weight of Inner-tag-based Priority.
11:8	WT_DSCP	Priority selection weight of DSCP-based Priority.
7:4	WT_IGR_ACL	Priority selection weight of Ingress-ACL-based Priority.
3:0	WT_PORT	Priority selection weight of Port-based Priority. 0x0: ignore the priority source 0x1~0xC: weight value. 0xC is highest. If weights of all priority sources are 0(ignore), the internal priority is 0.

Figure 7-5 Example 1 of Priority Selection Weight Rule

(weight)	12	11	10	9	8	7	6	5	4	3	2	1	0
Port-based Priority Assignment:											1		
VLAN-ACL-based Priority Assignment:										1			
DSCP-based Priority Assignment:									1				
Inner-tag-based Priority Assignment:								1					
Outer-tag-based Priority Assignment:							1						
Tunnel-based Priority Assignment:						1							
Routing-based Priority Assignment:					1								
MAC-based/IP-subnet-based VLAN Priority Assignment:				1									
Protocol-and-port-based VLAN Priority Assignment:			1										
MPLS-based Priority Assignment:		1											
802.1BR-based Priority Assignment:	1												

Figure 7-6 Example 2 of Priority Selection Weight Rule

	(weight)	12	11	10	9	8	7	6	5	4	3	2	1	0
Port-based Priority Assignment:	(3)			1										
VLAN-ACL-based Priority Assignment:	(2)			1										
DSCP-based Priority Assignment:	(N)			1										
Inner-tag-based Priority Assignment:	(5)									1				
Outer-tag-based Priority Assignment	(N)								1					
Tunnel-based Priority Assignment:	(N)						1							
Routing-based Priority Assignment:	(N)						1							
MAC-based/IP-subnet-based VLAN Priority Assignment:	(N)					1								
Protocol-and-port-based VLAN Priority Assignment:	(7)				1									
MPLS-based Priority Assignment:	(N)						1							
802.1BR-based Priority Assignment:	(6)					1								

Take Figure 7-6 and Figure 7-6 to be examples. In Figure 7-6, weight of 802.1BR-based is 12, and it's the highest weight. The internal priority is selected from 802.1BR-based priority.

In Figure 7-6, weights of Port-based, VLAN-ACL-based and DSCP-based are 10, and the weight of them are highest. Due to the weight values are the same, system uses the weight priority to decide internal-priority, and VLAN-ACL-based which has highest priority is selected. VLAN-ACL-based priority 2 is selected to be the internal-priority.

Note

If the weights of all priority assignments are 0, system assigns internal-priority 0 to the transmitted packet.

7.1.3 RSPAN Priority

Since received packet is identified as RSPAN packet, tagged priority of the RSPAN packet is directly taken to remap by Inner-tag Priority Remap Tables or Outer-tag Priority Remap Tables for getting its internal-priority, and the result of Priority Selection described in section 7.1.2 isn't to be the internal-priority of the RSPAN packet.

If the VID of RSPAN packet is equal to inner-VID of system configuration, Inner-tag Priority Remap Tables in Table 7-6 and Table 7-7 are used for priority remapping. If the VID of RSPAN packet is equal to outer-VID of system configuration, Outer-tag Priority Remap Tables in Table 7-8 and Table 7-9 are used for priority remapping. The CFI/DEI of RSPAN packet is used to select which one of Inner-tag Priority Remap Tables or Outer-tag Priority Remap Tables.

7.1.4 Internal Priority to QID Mapping

The device supports 8 egress queues to each normal port, 12 egress queues to XG Stacking port and 32 egress queues to CPU port. However, the egress queue can be treated as traffic class. Each priority can be configured to map a traffic class by global 4 mapping registers as follows:

- Mapping Internal Priority to Queue ID Control Register in Table 7-16
- Mapping CPU QID to Normal Port QID Control Register in Table 7-17
- Mapping CPU QID to XG Stacking Port QID Control Register Mapping in Table 7-18
- CPU Trap Reason to CPU QID Control Register in Table 7-19

Table 7-16 QM_INTPRI2QID_CTRL Register

Bits	Field	Description
2:0	QID	Assign normal port queue ID (0~7) of specified internal-priority.

Table 7-17 QM_CPUQID2QID_CTRL Register

Bits	Field	Description
2:0	QID	Assign normal port queue ID (0~7) of specified CPU queue ID (0~31).

Table 7-18 QM_CPUQID2XGSQID_CTRL Register

Bits	Field	Description
3:0	QID	Assign XG Stacking port (port 52~55) queue ID (0~11) of specified CPU queue ID (0~31). 0x0~0xB: queue ID 0~11. 0xC~0xF: invalid value.

Table 7-19 QM_RSN2CPUQID_CTRL_*N* Register

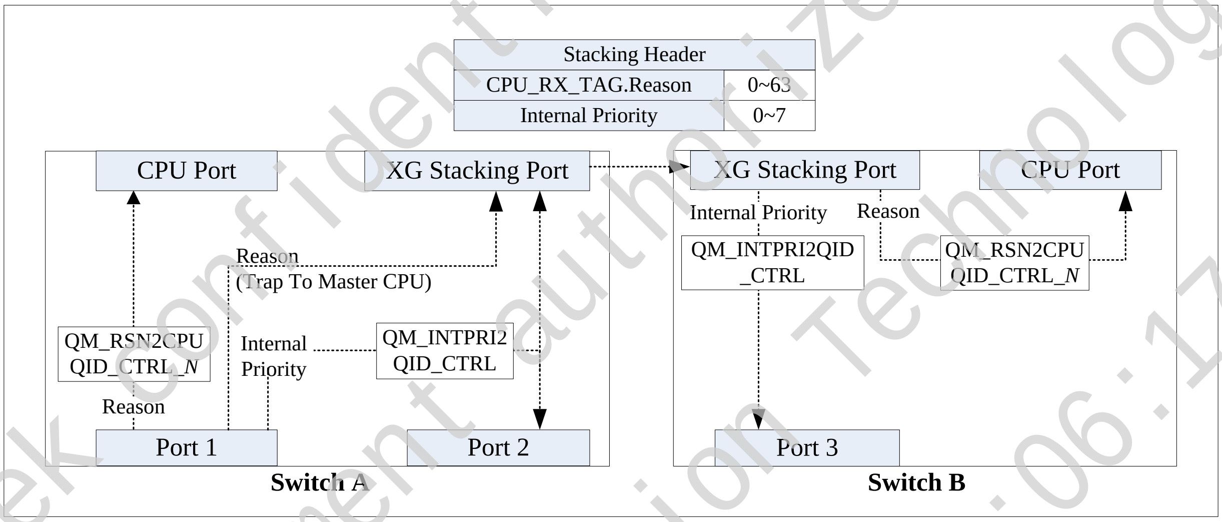
Bits	Field	Description
4:0	REASON_*N*	CPU queue ID (0~31) for reason *N*.

Take Figure 7-7 to explain some traffic flows and how to use the mapping registers. In port 1 of Switch A, there are two receiving traffic flows: one is trapping or forwarding to CPU Port, the other is forwarding to normal ports Port 2 of Switch A and Port 3 of Switch B.

For the traffic trapping or forwarding to CPU Port, trap reason is given to each packet. System assigns a unique CPU QID for each trap reason in Table 7-19, and the packet can be forward to correctly CPU TX Queue. If packet is trap to master CPU in another switch as Switch B, the trap reason is stored in stacking header and it is also transferred by Table 7-19 to get the TX QID of CPU Port in Switch B.

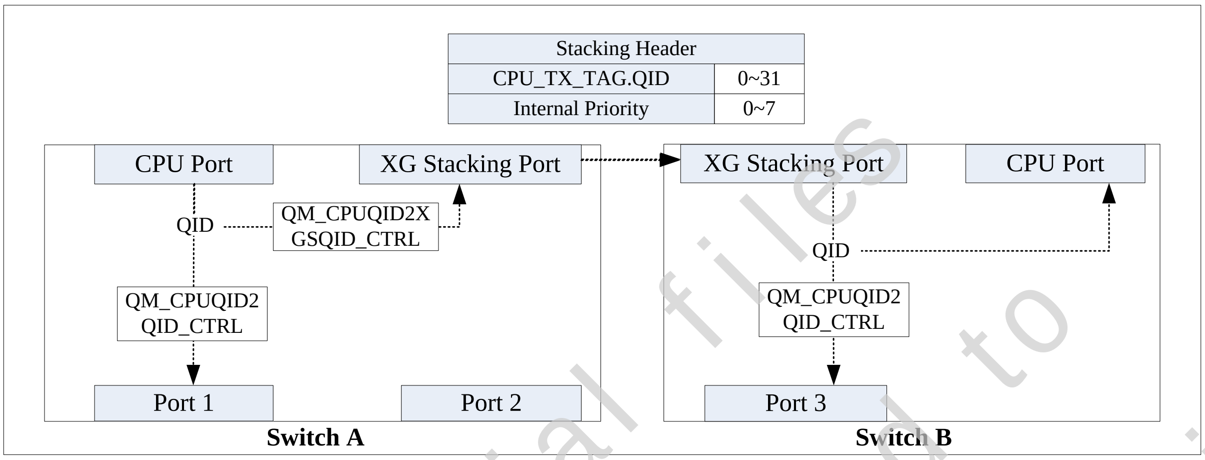
For the traffic forwarding to normal port, the internal priority of packet is taken to remap by Table 7-16 for getting TX QID of normal port. Even traffic is sent to another switch, the getting TX QID is also used to en-queue in XG Stacking Port. If packet is sent to another switch, Switch B, the internal priority of packet is also stored in stacking header and it is used to get the TX QID of Port 3 of Switch B by Table 7-16.

Figure 7-7 Internal Priority, Trap Reason and Egress QID Transformation



In Figure 7-8, CPU Port of Switch A sends packets to normal port and CPU Port of Switch B. At the same time, CPU assigns a CPU QID (value 0~31) in CPU TX Tag. If destination port of the packet sent by CPU is Port 1 of Switch A, CPU QID is transferred by Table 7-17 for getting TX QID of Port 1 of Switch A. If CPU of Switch A talks to CPU of Switch B, the CPU QID is transferred by Table 7-18 QM_CPUQID2XGSQID_CTRL Register to get TX QID of XG Stacking Port and it is saved in stacking header. Since XG Stacking Port of Switch B receives the stacking header, CPU QID is taken directly to be the CPU QID of CPU Port of Switch B. If destination port is Port 3 of Switch B, CPU QID is transferred by Table 7-17 QM_CPUQID2QID_CTRL Register for getting TX QID (0~8) of Port 3 of Switch B.

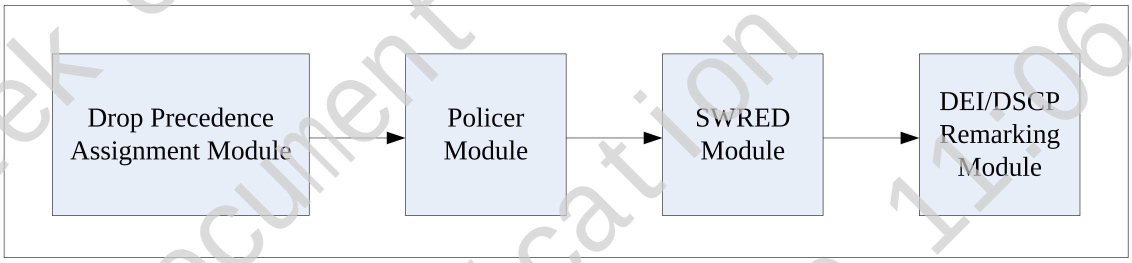
Figure 7-8 CPU QID of CPU TX Tag and Egress QID Transformation



7.1.5 Drop Precedence Assignment

For each transmitted packet, not only 3-bit internal-priority is assigned, but also 2-bit drop precedence (DP) is given. In 2-bit DP value, the value of valid range is from 0 to 2. In this section, DP assignment is introduced. After DP assignment module, assigned DP value will be used in Policer, SWRED and DEI/DSCP Remarking modules in Figure 7-9. In Policer and SWRED modules, the values of DP from 0 to 2 indicate three colors, Green, Yellow and Red, respectively.

Figure 7-9 Drop Precedence in System



As priority assignment, different assigning DP approaches are provided in system as below, and the details of the four DP assignments are described later.

- DSCP-based
- Inner-tag-based
- Outer-tag-based
- MPLS-based

7.1.5.1 DSCP-based DP Assignment

DSCP-based DP assignment only works in IP packet. DSCP value of received IPv4/IPv6 packet is remapped by DSCP to DP Remapping Table in Table 7-20 to get DSCP-based DP value. The method of getting DSCP from IPv4 and IPv6 can refer section 7.1.1.3 DSCP-based Priority Assignment. However, DSCP-based DP value is NULL in non-IP packet.

Table 7-20 DP_SEL_REMAP_DSCP Register

Bits	Field	Description
1:0	DP_DSCP	DP value of specified DSCP. 0x0~0x2: valid DP value. 0x3: reserved

7.1.5.2 Inner-tag-based DP Assignment

Inner-tag-based DP assignment takes CFI and priority of received packet's inner-tag to remap by Inner-tag to DP Remapping Table in Table 7-21 and Table 7-22 for getting inner-tag-based DP value. If CFI of inner-tag is equal to 0, Inner-tag to DP Remapping Table *DP_SEL_REMAP_ITAG_CFI0* is used. In the same way, when CFI of inner-tag is equal to 1, Inner-tag to DP Remapping Table *DP_SEL_REMAP_ITAG_CFI1* is used. If received packet is inner-untagged, inner-tag-based DP of it is NULL.

Table 7-21 DP_SEL_REMAP_ITAG_CFI0 Register

Bits	Field	Description
1:0	DP_CFI0_IPRI	DP value of specified inner-tag priority with CFI 0. 0x0~0x2: valid DP value. 0x3: reserved

Table 7-22 DP_SEL_REMAP_ITAG_CFI1 Register

Bits	Field	Description
1:0	DP_CFI1_IPRI	DP value of specified inner-tag priority with CFI 1. 0x0~0x2: valid DP value. 0x3: reserved

7.1.5.3 Outer-tag-based DP Assignment

Outer-tag-based DP assignment is similar to inner-tag-based DP assignment. If received packet is outer-untagged, outer-tag-based DP of it is NULL. Only outer-tagged packet has outer-tag-based DP value, and DEI and priority of outer-tag are taken to remap for getting outer-tag-based DP value. If DEI of outer-tag is equal to 0, Outer-tag to DP Remapping Table DP_SEL_REMAP_OTAG_DEI0 in Table 7-23is used. If DEI of outer-tag is equal to 1, Outer-tag to DP Remapping Table DP_SEL_REMAP_OTAG_DEI1 in Table 7-24 is used.

Table 7-23 DP_SEL_REMAP_OTAG_DEI0 Register

Bits	Field	Description
1:0	DP_DEI0_OPRI	DP value of specified inner-tag priority with CFI 0. 0x0~0x2: valid DP value. 0x3: reserved

Table 7-24 DP_SEL_REMAP_OTAG_DEI1 Register

Bits	Field	Description

1:0	DP_DEI1_OPRI	DP value of specified inner-tag priority with CFI 1. 0x0~0x2: valid DP value. 0x3: reserved

7.1.5.4 MPLS-based DP Assignment

If received packet is MPLS packet and the packet hits MPLS Decapsulate Table, traffic class (TC) of the packet is taken to remap by global MPLS TC to DP Remapping Table in Table 7-25 for getting MPLS-based DP value. Otherwise, MPLS-based DP value is NULL.

Table 7-25 DP_SEL_REMAP_MPLS Register

Bits	Field	Description
1:0	DP_TC	DP value of specified TC. 0x0~0x2: valid DP value. 0x3: reserved

7.1.6 Drop Precedence Selection

Selection method of DP from different DP assignments is similar to Priority Selection in section 7.1.2. System supports a DP Selection Table for each port, and the configuration is DP_SEL_PORT_TBL_CTRL in Table 7-26. In the DP Selection Table, four DP assignments DSCP-based, Inner-tag-based, Outer-tag-based and MPLS-based have individual 2-bit weight value. Weight value 3 is highest and weight value 0 means ignoring the DP source.

If weights of the four DP assignments are the same, DP decision priority is DSCP > Outer-tag > Inner-tag > MPLS-based.

Table 7-26 DP_SEL_PORT_TBL_CTRL Register

Bits	Field	Description
13:12	WT_MPLS	DP selection weight of MPLS-based DP assignment of specified port n. 0x0: ignore the DP source 0x1~0x3: weight value, 0x3 is highest
9:8	WT_OTAG	DP selection weight of outer-tag-based DP assignment of specified port n. 0x0: ignore the DP source 0x1~0x3: weight value, 0x3 is highest
5:4	WT_ITAG	DP selection weight of inner-tag-based DP assignment of specified port n. 0x0: ignore the DP source 0x1~0x3: weight value, 0x3 is highest
1:0	WT_DSCP	DP selection weight of DSCP-based DP assignment of specified port n. 0x0: ignore the DP source 0x1~0x3: weight value, 0x3 is highest

7.2 Scheduling Algorithm

After Priority Selection function of QoS module, unique 3-bit internal priority is given to the processing packet. System uses the internal priority for queueing packet to traffic class. Then, queueing packet is selected by Packet Scheduler for transmitting.

Packet scheduler architecture is supported to each egress port. It can be used to manage egress port bandwidth, traffic class bandwidth (queue bandwidth), counting packet in packet or byte, and weight configurations between traffic classes, etc.

System supports three algorithms as below:

- Strict Priority
- Weighted Round-Robin (WRR)
- Weighted Fair-Queuing (WFQ)

Besides, system provides weight configurations used in WRR and WFQ to each traffic class. The register of weight configuration is defined in Table 7-27. The valid range of weight is from 1 to 127 and the value 0 of weight stands for traffic blocking. The details of three algorithms are described in below sections.

7.2.1 Strict Priority

Strict priority algorithm is used to provide low latency service to the traffic in higher priority traffic class. All of traffic classes of egress port can be enabled strict priority ability whose register is defined in Table 7-27. Since multi traffic classes enable strict priority ability, the queue with highest index has highest service priority and will be scheduled with highest priority. If the index of the queue is lowest, it will be scheduled with lowest priority.

For example, a TX port has 500Mbps egress bandwidth and 4 egress strict priority queues Q0, Q1, Q2 and Q3. Four traffic, 100Mbps, 200Mbps, 300Mbps and 400Mbps respectively, are received from 4 RX ports and inject to the 4 egress strict priority queues. Only Q3 with 400Mbps and Q2 with 100Mbps can be transmitted.

Table 7-27 SCHED_PORT_Q_CTRL_SETn Register

Bits	Field	Description
31:9	RESERVE	
8	STRICT_EN	Strict priority ability of specified queue of a port. 0b0: disable 0b1: enable
7	RESERVE	
6:0	WEIGHT	Weight value of specified queue of a port. 0x0: blocking. 0x1~0x7F: weight 1~127.

Note

SCHED_PORT_Q_CTRL_SETn (n=0~1): when n is equal to 0, it is used for normal port 0~51. If n is equal to 1, it is for XG Stacking port 52~55.

7.2.2 Weighted Round Robin

WRR is the packet level granularity scheduler and uses weighted access to egress port and egress queue bandwidth control. Per egress port can control the scheduling type by register SCHED_PORT_ALGO_CTRL.SCHED_TYPE in Table 7-28. The scheduler use the weight value of each queue in round robin for de-queue packets. The weight values stand for the number of permitting transmitted packets. For example, the weights of Q7 and Q6 are 5 and 2, respectively. The ratio of transmitted packet number from Q7 and Q6 are {5:2} and no matter what the packet lengths are different. The sequence of packet transmitted from Q7 and Q6 in WRR is {Q7, Q7, Q7, Q7, Q7, Q6, Q6, Q7, Q7, … }. Shared Round Robin (SRR) is better averagely transmitting packet out than the WRR. SRR sends 1 packet from each queue until the all packet send out. In the same example described above, the sequence of packet transmitted from Q7 and Q6 in SRR is {Q7, Q6, Q7, Q6, Q7, Q7, Q7, Q7, Q6, … }. Actually, system implements SRR instead of WRR.

Table 7-28 SCHED_PORT_ALGO_CTRL Register

Bits	Field	Description

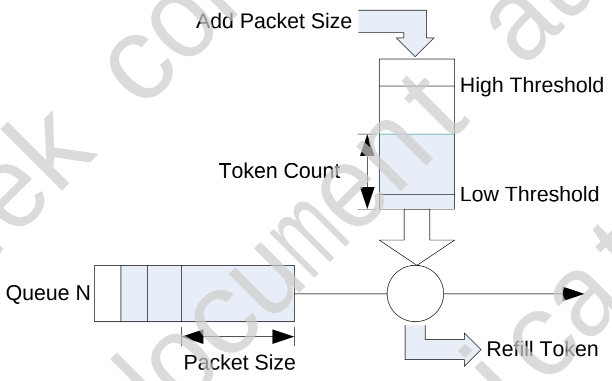
1	SCHED_TYPE	Scheduling algorithm for specific egress port. 0b0: WFQ (Weighted Fair-Queuing) 0b1: WRR (Weighted Round-Robin)

7.2.3 Weighted Fair Queue

WFQ uses leaky bucket (LB) mechanism to ensure the fairness between multiple traffic classes with different packet lengths. LB mechanism is as Figure 7-10, and there are high and low thresholds of bucket. High threshold is used in transmitting packets checking, and low threshold is used in subtract token checking. If token in LB is less than high threshold, packets can be transmitted. When packets pass through, token which size is same as the passing packet is added into the bucket. On the side, LB token is subtracted by setting rate until the token is less than low threshold. LB mechanism is not only used in WFQ, also be used in bandwidth control modules like as Egress Port Bandwidth, Egress Queue Bandwidth, Assured bandwidth, …, etc.

In WFQ, a quantity token with B-byte is subtracted from one of traffic classes. The weight values in WFQ are used to control the number of times in subtracting token of WFQ LB. The high threshold of all WFQ LBs is controlled by burst size configuration of Egress Port Bandwidth. However, the low threshold can be treated as the zero-line in mathematics. For system implementation, low threshold is fixed in (B-1)-byte and it can't be modulated.

Figure 7-10 Leaky Bucket Mechanism



In WRR, the ratio of transmitting packets number can be controlled by the weight configuration. However, in WFQ each queue can get the expected bandwidth by the weight configuration due to the weight controls the fixed quantity of token in WFQ LB. For example, a TX port configured in WFQ has 500Mbps egress bandwidth and 4 egress queues Q0, Q1, Q2 and Q3 with weights {1:2:3:4}. The number of RX ports is 4 and they receive four wire speed traffic transmitting to the TX port. The transmitting rates of Q0, Q1, Q2 and Q3 would be 50Mbps, 100Mbps, 150Mbps and 200Mbps, respectively.

7.3 Egress Remarking

Network devices usually apply some QoS attributes to decide traffic forwarding speed. Sometimes, traffic transmitting speed is reduced or increased by the attributes remarking. The section describes

QoS remarking module. In the module, packet with QoS attributes as below may be remarked or preserved before packets are actually going to be sent out:

- Inner-tag Priority
- Outer-tag Priority
- DSCP
- DEI flag

System provides individual enable/disable per-egress-port configuration as Table 7-29 for each QoS attributes described as above. Since packet hits one of inner-tag or outer-tag TPID configurations described in VLAN module, the packet is treated as having inner-tag priority or outer-tag priority. The details of each remarking modules are described in below sections.

Table 7-29 RMK_PORT_CTRL Register – Remarking Ability Relative Configuration

Bits	Field	Description
3	DEI_RMK_EN	Enable/Disable DEI remarking for a port. 0b0: Disable 0b1: Enable
2	DSCP_RMK_EN	Enable/Disable DSCP remarking for a port. 0b0: Disable 0b1: Enable
1	ORI_RMK_EN	Enable/Disable outer-tag priority remarking for a port. 0b0: Disable 0b1: Enable
0	IPRI_RMK_EN	Enable/Disable inner-tag priority remarking for a port. 0b0: Disable 0b1: Enable

7.3.1 Inner-tag Priority Remarking

The inner-tag priority remarking module is not only used to remark inner-tag priority but also used to decide a default inner-tag priority for inner-untagged packets. The two sections are discussed as following.

7.3.1.1 Inner-priority Remarking Behavior

System provides a global register RMK_CTRL.IPRI_RMK_SRC in Table 7-30 for selecting inner-tag priority remarking source. There are overall 4 inner-tag priority remarking sources: internal-priority, original inner-tag priority, original outer-tag priority and DSCP. For each remarking source, an individual remarking table is supported. The remarking tables are in Table 7-31, Table 7-32, Table 7-33 and Table 7-34.

In original inner-tag priority, original outer-tag priority, and DSCP remarking sources, if original packet doesn't have inner-tag, outer-tag and DSCP respectively, section 7.3.1.2 Default Inner-priority Assignment is used to decide TX inner priority.

Table 7-30 RMK_CTRL Register – Inner-priority Remarking Relative Configuration

Bits	Field	Description
9:8	IPRI_RMK_SRC	Select inner-priority remarking source. 0b00: Internal priority 0b01: Original inner-tag priority 0b10: Original outer-tag priority 0b11: Original DSCP value
1	IPRI_DFLT_CFG	Select RX port or TX port's configuration (REMARK.IPRI_DFLT_SRC) to decide default inner-priority. 0b0: Use RX port configuration 0b1: Use TX port configuration

Table 7-31 RMK_INTPRI2IPRI_CTRL Register

Bits	Field	Description
2:0	IPRI	New inner-priority for specified internal priority. The default inner-priority remarking value of each internal priority 0~7: {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7}

Table 7-32 RMK_IPR2IPRI_CTRL Register

Bits	Field	Description
2:0	IPRI	New inner-priority for specified original inner-tag priority. The default inner-priority remarking value of each original inner-tag priority 0~7: {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7}

Table 7-33 RMK_OPR2IPRI_CTRL Register

Bits	Field	Description
2:0	IPRI	New inner-priority for specified original outer-tag priority. The default inner-priority remarking value of each original outer-tag priority 0~7: {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7}

Table 7-34 RMK_DSCP2IPRI_CTRL Register

Bits	Field	Description
2:0	IPRI	New inner-priority for specified DSCP.

7.3.1.2 Default Inner-priority Assignment

Since system received inner-untagged packet and transmitted it with inner-tag, the 3-bit priority of the inner-tag is decided according to default inner-priority assignment described in this section.

The device takes global RMK_CTRL.IPRI_DFLT_CFG in Table 7-30 to extract configuration of ingress port or egress port. After getting the target port, per-port 2-bit register REMARK.IPRI_DFLT_SRC defined in Table 7-35 is used to give TX default inner-tag priority. There are two methods of REMARK.IPRI_DFLT_SRC: Copy internal priority and using port-based default inner-priority defined in REMARK.IPRI_DFLT_PRI in Table 7-35.

Table 7-35 REMARK Register – Default Inner-priority Relative Configuration

Bits	Field	Description
10:8	IPRI_DFLT_PRI	Default inner-priority for specific port.
2	IPRI_DFLT_SRC	Default inner-priority source for specific port. 0b0: Copy internal priority 0b1: Use REMARK.IPRI_DFLT_PRI

In another case, if inner-tag priority remarking of egress port is enabled, default inner-priority assignment is also used to procedure TX inner-tag priority in follow three cases:

- In original inner-priority of inner-priority remarking source selected, received packet is inner-untagged packet.
- In original outer-priority of inner-priority remarking source selected, received packet is outer-untagged packet.
- In DSCP of inner-priority remarking source selected, received packet is non-IP packet.

7.3.2 Outer-tag Priority Remarking

In outer-tag priority remarking module, there are also two sections as inter-tag priority remarking: Outer-tag Priority Remarking and Default Outer-tag Priority Assignment. Following are the details.

7.3.2.1 Outer-priority Remarking Behavior

Outer-tag priority remarking is similar to inner-tag priority remarking. System provides a global 2-bit register RMK_CTRL.OPRI_RMK_SRC in Table 7-36 for selecting outer-tag priority remarking source: internal-priority, original outer-priority, original inner-priority and DSCP value. Similarly, an individual remarking table is supported for each remarking source. The remarking tables are defined in Table 7-37, Table 7-38, Table 7-39 and Table 3-6.

In three remarking sources, original inner-tag priority, original outer-tag priority, and DSCP, if original packet doesn't have inner-tag, outer-tag and DSCP respectively, section 7.3.2.2 Default Outer-priority Assignment is used to decide TX outer priority.

Table 7-36 RMK_CTRL Register – Outer-priority Remarking Relative Configuration

Bits	Field	Description
7:6	OPRI_RMK_SRC	Select outer-priority remarking source. 0b00: Internal priority 0b01: Original inner-priority 0b10: Original outer-priority 0b11: Original DSCP value
0	OPRI_DFLT_CFG	Select RX port or TX port's configuration (REMARK.OPRI_DFLT_SRC) to decide default outer-priority. 0b0: Use RX port configuration 0b1: Use TX port configuration

Table 7-37 RMK_INTPRI2OPRI_CTRL Register

Bits	Field	Description

2:0	OPRI	New outer-priority for specified internal priority. The default outer-priority remarking value of each internal priority 0~7: {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7}

Table 7-38 RMK_IPR2OPRI_CTRL Register

Bits	Field	Description
2:0	OPRI	New outer-priority for specified original inner-tag priority. The default outer-priority remarking value of each original inner-tag priority 0~7: {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7}

Table 7-39 RMK_OPR2OPRI_CTRL Register

Bits	Field	Description
2:0	OPRI	New outer-priority for specified original outer-tag priority. The default outer-priority remarking value of each original outer-tag priority 0~7: {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7}

Table 7-40 RMK_DSCP2OPRI_CTRL Register

Bits	Field	Description
2:0	OPRI	New outer-priority for specified DSCP.

7.3.2.2 Default Outer-priority Assignment

For outer-untagged packet, the device also provides default outer-tag priority configuration described in this section. The device takes global RMK_CTRL.OPRI_DFLT_CFG in Table 7-36 to extract configuration of ingress port or egress port. After getting the target port, per-port default outer-priority configuration REMARK.OPRI_DFLT_SRC in Table 7-41 is used and it has three selections: copy internal priority, using port-based default priority REMARK.OPRI_DFLT_PRI in Table 7-41 and copy inner-tag priority.

Similarly, if outer-tag priority remarking of egress port enabled, default outer-priority assignment is also used to procedure TX outer-tag priority in follow three cases:

- In original inner-priority of outer-priority remarking source selected, received packet is inner-untagged packet.
- In original outer-priority of outer-priority remarking source selected, received packet is outer-untagged packet.
- In DSCP of outer-priority remarking source selected, received packet is non-IP packet.

Table 7-41 REMARK Register – Default Outer-priority Relative Configuration

Bits	Field	Description
6:4	OPRI_DFLT_PRI	Default outer-priority for specific port.

1:0	OPRI_DFLT_SRC	Default outer-priority source for specific port. 0b00: Copy internal priority 0b01: Use REMARK.OPRI_DFLT_PRI. 0b10: Copy RX inner-priority. 0b11: Reserved

Note

In REMARK.OPRI_DFLT_SRC = 0b10, if original packet without inner-tag, system follows REMARK.OPRI_DFLT_SRC = 0b01 behavior.

7.3.3 DSCP Remarking

DSCP is the filed in header of IP packet, and it has different levels of service for network traffic. As inner-tag priority remarking and outer-tag priority remarking, DSCP remarking has a global 2-bit register RMK_CTRL.DSCP_RMK_SRC in Table 7-42 for select DSCP remarking source: internal-priority, original inner-priority, original outer-priority, original DSCP, drop precedence, and drop precedence and internal priority. The remarking sources, excluding drop precedence, has its individual remarking tables in Table 7-43, Table 7-44, Table 7-45, Table 7-46 and Table 7-47.

Table 7-42 RMK_CTRL Register – Outer-priority Remarking Relative Configuration

Bits	Field	Description
4:2	DSCP_RMK_SRC	Select DSCP remarking source. 0x0: Internal priority 0x1: Original inner-priority 0x2: Original outer-priority 0x3: Original DSCP value 0x4: Drop precedence(color) 0x5: Drop precedence and internal priority 0x6 ~ 0x7: Reserved

Note

In RMK_CTRL.DSCP_RMK_SRC = 0x4, system is according to RFC2475, only packet with DSCP value AF11~AF43 will be remarked.

AF Group: Low drop(green), Med drop(yellow), High drop(red)

Class 1: AF11(001010), AF12(001100), AF13(001110)

Class 2: AF21(010010), AF22(010100), AF23(010110)

Class 3: AF31(011010), AF32(011100), AF33(011110)

Class 4: AF41(100010), AF42(100100), AF43(100110).

Table 7-43 RMK_INTPRI2DSCP_CTRL Register

Bits	Field	Description
5:0	DSCP	New DSCP value for the specified internal-priority.

Table 7-44 RMK_IPR2DSCP_CTRL Register

Bits	Field	Description
5:0	DSCP	New DSCP value for the specified inner-priority.

Table 7-45 RMK_OPR2DSCP_CTRL Register

Bits	Field	Description
5:0	DSCP	New DSCP value for the specified outer-priority.

Table 7-46 RMK_DSCP2DSCP_CTRL Register

Bits	Field	Description
5:0	DSCP	New DSCP value for the specified DSCP.

Table 7-47 RMK_DPINTPRI2DSCP_CTRL Register

Bits	Field	Description
5:0	DSCP	New DSCP value for the specified drop precedence and internal-priority.

It's special in using drop precedence remarking source. System remarks DSCP according to RFC 2475 "DiffServ", following PHBs have been defined in RFC:

- Default PHB(Best Effort): 000000(0)
- Expedited Forwarding PHB: 101110(46)
- Assured Forwarding PHB group(RFC 2597)
- Class Selector PHB: xxx000

Figure 7-11 Assured Forwarding Behavior Group

Assured Forwarding (AF) Behavior Group				
	Class 1	Class 2	Class 3	Class 4
Low Drop (Green, DP=0)	AF11(001010)	AF21(010010)	AF31(011010)	AF41(100010)
Med Drop (Yellow, DP=1)	AF12(001100)	AF22(010100)	AF32(011100)	AF42(100100)
High Drop (Red, DP=2)	AF13(001110)	AF23(010110)	AF33(011110)	AF43(100110)

Only AF PHBs utilize drop precedence (color) to forward/remark the packet. Therefore, system only applies the remarking on packet with DSCP values described above (AF11~AF43) as Figure 7-11. Take an example: received packet with DSCP value AF11 (001010) and has been colored to be red, then ASIC will automatically remark the DSCP value to be AF13 (001110). Packet with other DSCP value is not remarked even the DSCP remarking is selected to base on DP. The remarking will be done by ASIC automatically and no configurations are needed.

The device can remark DSCP value in IPv4 and IPv6 packet due to Traffic Class filed in IPv6 header is an 8-bit field equal to the type of services (TOS) byte in IPv4 header. Figure 7-12 and Figure 7-13 are the header formats of IPv4 and IPv6. If DSCP remarking enabled, the 6-bit in the byte would be modified no matter what the packet is IPv4 or IPv6.

Figure 7-12 Type of Service (ToS) field in IPv4 header format

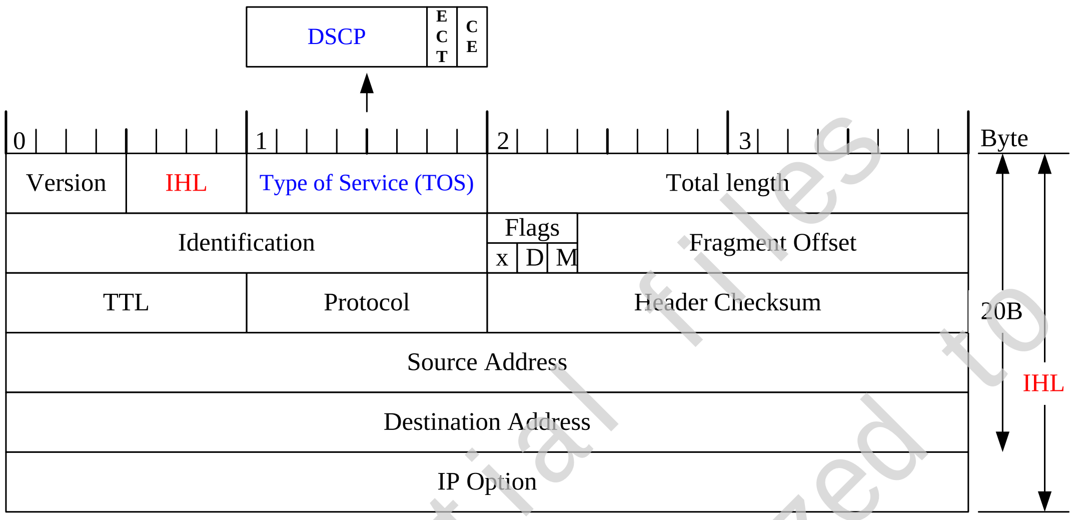
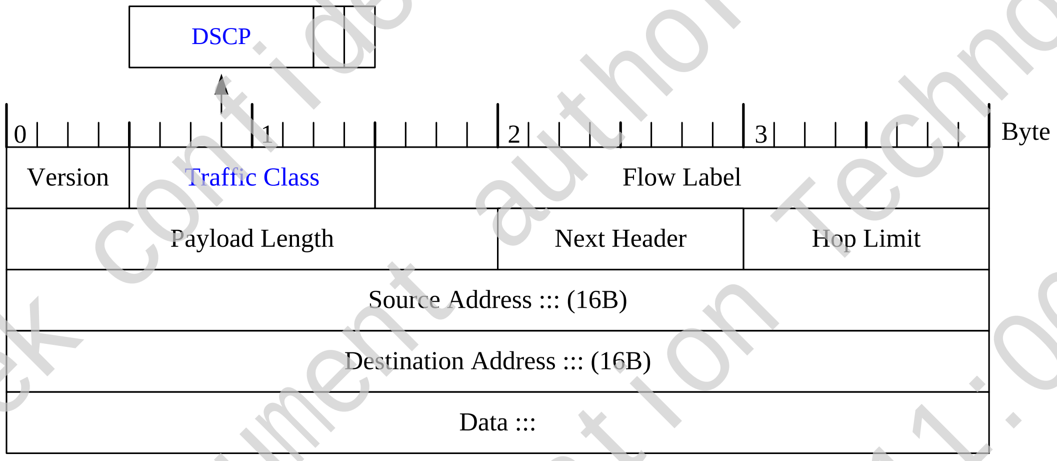


Figure 7-13 Traffic Class field in IPv6 header format



7.3.4 DEI Remarking

DEI remarking can be considered CFI remarking for C-tag due to system remarking DEI bit according to per-egress-port register RMK_PORT_CTRL.DEI_RMK_TAG_SEL in Table 7-48.

Two remarking sources are supported in DEI remarking and are configured in global register Table 7-49. Global two remarking tables in Table 7-50 and Table 7-51 are supported for the DEI remarking sources.

Table 7-48 RMK_PORT_CTRL Register – DEI Remarking Relative Configuration

Bits	Field	Description
4	DEI_RMK_TAG_SEL	Select inner-tag or outer-tag to remark DEI bit for specified port. 0b0: Inner-tag 0b1: Outer-tag

Table 7-49 RKM_CTRL Register – DEI Remarking Relative Configuration

Bits	Field	Description

5	DEI_RMK_SRC	Select DEI/CFI remarking source. 0b0: Internal priority 0b1: DP

Table 7-50 RMK_INTPRI2DEI_CTRL Register

Bits	Field	Description
0	DEI	New DEI/CFI for specified internal priority.

Table 7-51 RMK_DP2DEI_CTRL Register

Bits	Field	Description
0	DEI	New DEI/CFI for specified drop precedence.

8 Traffic Suppression & Rate Limit

8.1 Ingress Bandwidth Control

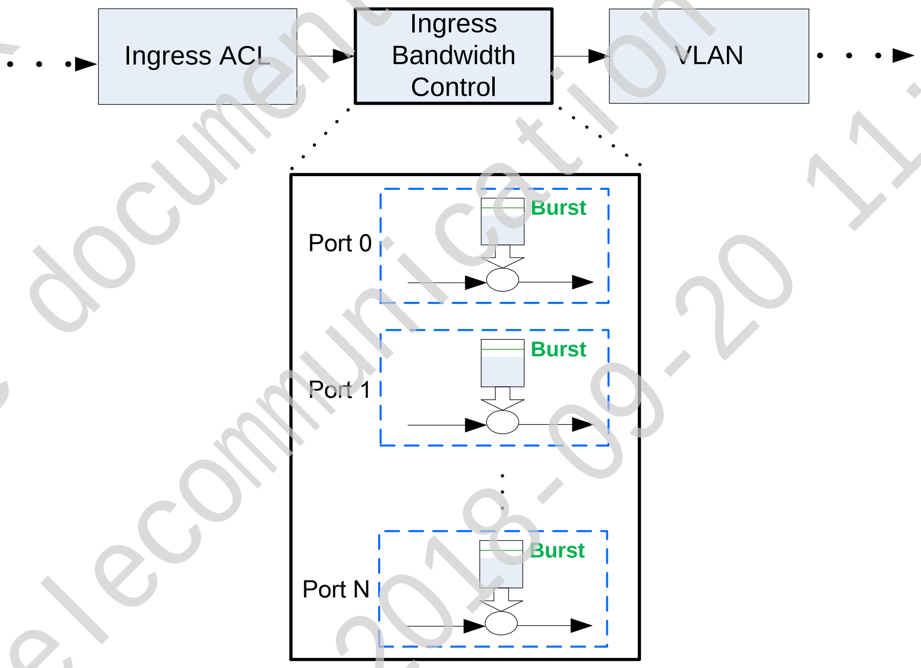
To limit ingress traffic rate of a port, system provides ingress bandwidth control function for every port. When ingress rate exceeds the relevant configured-bandwidth, system could be configured to drop packet.

Some special protocols, including BPDU, RMA, IGMP, ARP, RIP, DHCP and Realtek control packet (Ethertype = 0x8899), could be configured to bypass the bandwidth control. In addition, traffic matching the ingress ACL can also bypass the bandwidth control by configuring ACL bypassing action.

8.1.1 Bandwidth Configuration

Device provides a port-based leaky bucket for each port as shown in Figure 8-1.

Figure 8-1 Ingress Leaky Bucket for Each Port



8.1.1.1 Port-based Configuration

Table 8-1 shows IGBW_PORT_CTRL register which specifies the rate limitation configuration of each port. System provides exceed flag declared at Table 8-2 for each port to indicate the rate exceeding status.

When the traffic rate is over the configured rate of a port, system allows sending pause frames to prevent packet from dropping by configuring IGBW_PORT_FC_CTRL register as Table 8-3 for each port.

Table 8-1 IGBW_PORT_CTRL Register

Bits	Field	Description
52	EN	The state of ingress bandwidth control for port n. 0b0: disable 0b1: enable Note: if leaky bucket is disabled, leaky bucket token counter of the port will be cleared by ASIC.
52:32	RATE	Ingress bandwidth of port n. 0: blocking; Others: controlled rate = RATE * 16kbps Unit: 16kbps
15:0	BURST	Burst size (high threshold) of LB of port n. Unit: Byte

Table 8-2 IGBW_PORT_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	Indicate ingress bandwidth exceed for port n.

Table 8-3 IGBW_PORT_FC_CTRL Register

Bits	Field	Description
0	EN	The state of ingress bandwidth control flow-control ability for port n. 0b0: disable 0b1: enable

8.1.2 Protocol Bypass

System provides global options to allow below protocols to bypass the ingress bandwidth module.

- ARPREQ
- RMA
- BPDU
- RTKPKT
- IGMP
- RIP
- DHCP

The IFG and the bypassed traffic can also be configured to or not to be included in bandwidth counting, and can be configured by IGBW_CTRL register as Table 8-4.

Table 8-4 IGBW_CTRL Register

Bits	Field	Description
8	INC_BYPASS_PKT	Ingress bandwidth control includes or excludes bypass packets. It applies to all ports. 0b0: Exclude bypass packets (default). 0b1: Include bypass packets.
7	INC_IFG	Ingress bandwidth control includes or excludes the Preamble & IFG (20 Bytes). 0b0: exclude 0b1: include
6	ADMIT_RIP	Admit RIP(Routing Information Packet) and OSPF(Open Shortest Path First) packet bypassing ingress bandwidth control. 0b0: disable 0b1: enable
5	ADMIT_DHCP	Admit DHCP packet bypassing ingress bandwidth control. 0b0: disable 0b1: enable
4	ADMIT_ARPREQ	Admit ARP Request (and Neighbor Solicitation for ICMPv6 (Type:135)) packet bypassing ingress bandwidth control. 0b0: disable 0b1: enable
3	ADMIT_RMA	Admit RMA (exclude BPDU) packet bypassing ingress bandwidth control. 0b0: disable 0b1: enable
2	ADMIT_BPDU	Admit BPDU packet bypassing ingress bandwidth control. 0b0: disable 0b1: enable
1	ADMIT_RTKPKT	Admit Realtek control packet (ethertype=0x8899) bypassing ingress bandwidth control. 0b0: disable 0b1: enable

8.2 Egress Bandwidth

Egress bandwidth module is used for buffer the burst incoming packet and rate limited the packet flow. The egress bandwidth function includes egress port bandwidth and egress queue bandwidth described in this chapter. The position is before packet modify model (priority remark/egress VLAN Translation).

The device supports a global EGBW_CTRL.INC_IFG register in Table 8-5 to compute inter-frame gap (IFG) or not. In including IFG configuration, 20Bytes IFG is computed to all egress port bandwidth and egress queue bandwidth. In excluding IFG configuration, only packet size is added to all egress port bandwidth and egress queue bandwidth.

Table 8-5 EGBW_CTRL Register

Bits	Field	Description
1	INC_IFG	Egress bandwidth control includes/excludes the Preamble & IFG (20 Bytes). 0b0: Exclude. 0b1: Include.

8.2.1 Egress Port Bandwidth Management

The device supports egress bandwidth configurations including enable/disable state, rate and burst size in Table 8-6 for each port.

Table 8-6 EGBW_PORT_CTRL Register

Bits	Field	Description
63:53	RESERVE	
52	EN	The state of egress bandwidth control for port n. 0b0: disable 0b1: enable
51:32	RATE	Egress bandwidth of port n. 0: blocking. Others: controlled rate = RATE * 16kbps Unit: 16kbps
31:16	RESERVE	
15:0	BURST	Burst size of egress bandwidth for port n. The maximum burst size is 64KByte. Unit: Byte

8.2.2 Egress Queue Bandwidth Management

Not only egress port bandwidth configuration but also egress queue bandwidth configuration is supported by the device. Egress queue bandwidth management can be separated to two parts: Queue Maximum Bandwidth and Assured Bandwidth. The details are discussed in below sections.

8.2.2.1 Egress Queue Maximum Bandwidth

For each egress queue, system supports three configurations: enable or disable state, maximum bandwidth and burst size in Table 8-7. The configurations can be used to limit the maximum traffic boundary of the egress queue.

For example, a port has 500Mbps egress bandwidth and 4 available egress queues Q0, Q1, Q2 and Q3. The weights of 4 queues are {1:2:3:4} in WFQ scheduling algorithm. Four wire speed traffics are sent to 4 queues, respectively. Egress queue bandwidth of Q3 is set to 140Mbps. The transmitting rates of Q0, Q1, Q2 and Q3 will be 60Mbps, 120Mbps, 180Mbps and 140Mbps, respectively. However, if egress queue bandwidth of Q3 is configured to over 200Mbps, the transmitting rates of 4 queues are keeping in 50Mbps, 100Mbps, 150Mbps and 200Mbps due to Q3 is limited by weight value 4 in WFQ module.

Table 8-7 EGR_Q_BW Register

Bits	Field	Description
37	ASSURED_MODE_Qm	Remain bandwidth mode of assured bandwidth of egress queue m. 0b0: Shared bandwidth. 0b1: Fixed bandwidth.
36	ASSURED_BW_EN_Qm	The state of egress queue assured bandwidth control for queue m. 0b0: disable 0b1: enable
35:16	ASSURED_BW_Qm	Max bandwidth of egress queue m. 0: blocking; Others: controlled rate = RATE * 16kbps Unit: 16kbps
15:0	ASSURED_LB_BURST_Qm	Max bandwidth burst size of egress queue m. Unit: Byte
36	MAX_BW_EN_Qm	The state of egress queue max bandwidth control for egress queue m. 0b0: disable 0b1: enable
35:16	MAX_BW_Qm	Max bandwidth of egress queue m. 0: blocking; Others: controlled rate = RATE * 16kbps Unit: 16kbps
15:0	MAX_LB_BURST_Qm	Max bandwidth burst size of egress queue m. Unit: Byte

8.2.2.2 Egress Queue Assured Bandwidth

The device supports assured bandwidth to reserve and guarantee bandwidth for each traffic class. The capability is used to retain private bandwidth for the specified traffic class. The relative configurations contain: enable or disable state, assured bandwidth, burst size and remain bandwidth mode. The configurations can refer Table 8-7.

If traffic classes configure assured bandwidth, the egress port bandwidth will reserve the bandwidth. If egress port bandwidth have not enough bandwidth for the totally of assured bandwidth of all egress traffic classes (Total assured bandwidth > TX port bandwidth), the big queue ID have higher priority to get the bandwidth.

Remain bandwidth mode configuration of assured bandwidth has two modes: Shared bandwidth mode and Fixed bandwidth mode. In shared bandwidth mode, the residual bandwidth can be shared to other queues. In fixed bandwidth mode, remaining bandwidth is occupied in the queue and the egress port is maybe not in wire speed.

Take the same example, a port has 500Mbps egress bandwidth and 4 available egress queues Q0, Q1, Q2 and Q3. The weights of 4 queues are {1:2:3:4} in WFQ scheduling algorithm. Q0, Q1 and Q2 have wire speed and Q3 has 20% (200M) traffic. Assured bandwidth of Q3 is configured in 300M. In shared bandwidth mode, the transmitted rates of Q0, Q1, Q2 and Q3 are nearly 50Mbps, 100Mbps, 150Mbps and 200Mbpss, respectively. In fixed bandwidth mode, the transmitted rates of Q0, Q1, Q2 and Q3 are nearly 33Mbps, 67Mbps, 100Mbps and 200Mbpss respectively, and 100Mbps bandwidth of egress port is wasted.

Note

1, Assured bandwidth should less than maximum bandwidth and TX port bandwidth.

2, Shared bandwidth mode of assured bandwidth should not co-work with fixed bandwidth mode in the port.

8.2.3 CPU Port Egress Bandwidth Management

Not only byte per second (BPS) mode but also packet per second (PPS) mode are supported to CPU port due to packet unit is often used in CPU process. System provides EGBW_CTRL.RATE_MODE_CPU to select bandwidth process mode for CPU port. The configuration applies to both egress port rate/burst and egress queue rate/burst of CPU port. In BPS mode, it can refer normal port configuration and normal queue configuration. In PPS mode, the definitions of egress port rate and egress queue rate are in step of 1 pps instead of 16 Kbps and the units of egress port burst size and egress queue burst size become to 1 packet. The CPU port queue does not support assured bandwidth.

Table 8-8 EGBW_CTRL Register

Bits	Field	Description
0	RATE_MODE_CPU	CPU port rate limit mode. 0b0: PPS 0b1: BPS

Note

When PPS mode is used for CPU port bandwidth management, suggest that SCHED_TYPE of CPU port is set as WRR instead of WFQ for getting better weight effort.

8.2.3.1 Egress Port Bandwidth

The CPU port egress bandwidth management configuration is same as the normal port configuration. Please refer Table 8-6.

8.2.3.2 Egress Queue Bandwidth

The CPU port has 32 egress traffic classes and Maximum bandwidth is supported for each traffic class. The queue configuration of CPU port is in Table 8-9.

Table 8-9 EGBW_CPU_MAX_LB_CTRL Register

Bits	Field	Description
63:53	RESERVE	
52	EN	The state of egress queue maximum bandwidth control for CPU port queue m. 0b0: disable 0b1: enable

51:32	RATE	Egress queue maximum bandwidth rate of CPU port queue m. 0: blocking. Others: controlled rate = RATE * 16kbps Unit: 16kbps or 1pps
31:16	RESERVE	
15:0	BURST	Burst size of egress queue maximum bandwidth of CPU port queue m. The maximum burst size is 64KByte. Unit: Byte or pps

8.3 Storm Control

System supports basic storm control to limit rate of malicious traffic listed as below:

- Unknown unicast
- All unicast (known & unknown)
- Unknown multicast
- All multicast (known & unknown)
- Broadcast

To avoid control packets be filtered by storm control, system provides bypass options for protocols , including BPDU, DHCP, IGMP, RIP, ARP request (containing neighbor solicitation) traffic and so on, to bypass storm rate limiting.

System also provides protocol storm control to prevent malicious protocol traffic interfering the network, including DHCP, BPDU, IGMP/MLD and ARP.

8.3.1 Basic Storm Control

System supports storm control in each port for unicast, multicast, and broadcast traffic respectively. Each type of traffic that exceeds the limited rate would be dropped. For each type of traffic, the limited rates can be configured by per port registers: STORM_PORT_UC_CTRL (

Table 8-10) for unicast, STORM_PORT_MC_CTRL (Table 8-11) for multicast, and STORM_PORT_BC_CTRL (Table 8-12) for broadcast traffic. For unicast and multicast traffic, system also supports option to limit both known and unknown traffic or limit unknown traffic only.

The storm control is implemented by leaky bucket mechanism which supports the rate and burst configuration. For each port, the limited rate could be configured as packet-based (pps) or byte-based (bps) as Table 8-13. In bps mode, the granularity of rate setting is 16Kbps.

For example, if user wants to limit unknown multicast and broadcast traffic received from port 1, it could be achieved by below configuration:

■ MODE = 0	To use pps mode for port 1 storm control
■ MC_TYPE = 0	To limit unknown multicast traffic only

■ MC_RATE = 1000	To limit multicast traffic rate as 1000pps
■ MC_BURST_PPS = 10	To allow 10 packet burst
■ BC_RATE = 1000	To limit broadcast traffic rate as 1000pps

Table 8-10 STORM_PORT_UC_CTRL Register

Bits	Field	Description
57	TYPE	Unicast type selection 0b0: unknown unicast 0b1: both known and unknown unicast
56	EN	Unicast storm control status of port. 0b0: disable 0b1: enable
55:32	RATE	Unicast storm control rate of port. In pps mode, the unit is 1pps. In bps mode, the unit is 16kbps. In value 0, the traffic is blocking.
15:0	BURST	Unicast burst size of storm control leaky bucket for specific port. In pps mode, the unit is 1 packet. In bps mode, the unit is 1 byte. Note: The value should be larger than Token value in pps mode.

Table 8-11 STORM_PORT_MC_CTRL Register

Bits	Field	Description
57	TYPE	Multicast type selection 0b0: unknown multicast 0b1: both known and unknown multicast
56	EN	Multicast storm control status of port. 0b0: disable 0b1: enable
55:32	RATE	Multicast storm control rate of port. In pps mode, the unit is 1pps. In bps mode, the unit is 16kbps. In value 0, the traffic is blocking.
15:0	BURST	Multicast burst size of storm control leaky bucket for specific port. In pps mode, the unit is 1 packet. In bps mode, the unit is 1 byte. Note: The value should be larger than Token value in pps mode.

Table 8-12 STORM_PORT_BC_CTRL Register

Bits	Field	Description
56	EN	Broadcast storm control status of port. 0b0: disable 0b1: enable

55:32	RATE	Broadcast storm control rate of port. In pps mode, the unit is 1pps. In bps mode, the unit is 16kbps. In value 0, the traffic is blocking.
15:0	BURST	Broadcast burst size of storm control leaky bucket for specific port. In pps mode, the unit is 1 packet. In bps mode, the unit is 1 byte. Note: The value should be larger than Token value in pps mode.

Table 8-13 STORM_PORT_CTRL Register

Bits	Field	Description
0	MODE	Storm control is based on packet count or byte count of port. 0b0: pps (packet per second) 0b1: bps (byte per second).

For each type of storm control, per port supports exceed flags to indicate if the traffic ever reached the limit rate as Table 8-14, Table 8-15 and

Bits	Field	Description
0	FLAG	Multicast storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

Table 8-16

Table 8-14 STORM_PORT_UC_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	Unicast storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

Table 8-15 STORM_PORT_MC_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	Multicast storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

Table 8-16 STORM_PORT_BC_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	Broadcast storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

8.3.2 Storm Control Bypass Mechanism

As Table 8-17, system allows several type of traffic to bypass storm control, including BPDU, DHCP, IGMP, RIP, ARP request (containing neighbor solicitation) and so on. The bypassed traffic can be configured to be included in rate counting or not. In addition, IFG can be configured to be included or excluded in the storm rate counting.

Table 8-17 STORM_CTRL Register

Bits	Field	Description
10	INC_BYPASS_PKT	Storm control includes or excludes bypass packets. It applies to all storm control leaky buckets of all ports. 0b0: Exclude bypass packets (default). 0b1: Include bypass packets.
9	INC_IFG	Storm control rate includes IFG(inter frame gap) and preamble. 0b0: exclude 0b1: include
6	ADMIT_RIP	Admit RIP(Routing Information Packet) and OSPF(Open Shortest Path First) packet bypassing ingress bandwidth control. 0b0: disable 0b1: enable
5	ADMIT_DHCP	Admit DHCP packet bypassing storm control. 0b0: disable 0b1: enable
4	ADMIT_ARPREQ	Admit ARP Request (and Neighbor Solicitation for ICMPv6 (Type:135)) packet bypassing storm control. 0b0: disable 0b1: enable
3	ADMIT_RMA	Admit RMA (exclude BPDU) packet bypassing storm control. 0b0: disable 0b1: enable
2	ADMIT_BPDU	Admit BPDU packet bypassing storm control. 0b0: disable 0b1: enable
1	ADMIT_RTKPKT	Admit Realtek control packet (ethertype=0x8899) bypassing storm control. 0b0: disable 0b1: enable
0	ADMIT_IGMP	Admit IGMP packet bypassing storm control. Note: This includes IPv4 IGMP and IPv6 MLDv1/v2. 0b0: disable 0b1: enable

8.3.3 Protocol Storm Control

System provides four special protocol storm controls for DHCP, BPDU, IGMP/MLD and ARP, respectively. Per port can configure the rate of each type in PPS mode respectively as Table 8-18, Table 8-19, Table 8-20 and Table 8-21, and also support exceed flag for each type of traffic as Table 8-22, Table 8-23, Table 8-24 and Table 8-25 to record the rate exceeding.

For ARP and DHCP, system also supports options for both type of traffic to take effect on the desired VLAN with corresponding GROUP_MASK bit enabled, which is GROUP_MASK[0] for ARP and GROUP_MASK[1] for DHCP as Table 8-26.

The protocol storm use to limit specific protocol flow. For example, if a port set multicast storm control STORM_PORT_MC_CTRL. RATE = 200pps, bypass BPDU, (global STORM_CTRL. ADMIT_BPDU =enabled) and BPDU protocol storm STORM_PORT_PROTO_BPDU_CTRL. RATE=30pps. Then, the total multicast traffic is limited in 230pps (BPDU is limited in 30pps and the other multicast flow is limited in 200pps).

Table 8-18 STORM_PORT_PROTO_BPDU_CTRL Register

Bits	Field	Description
20	EN	Status of BPDU storm control of port. 0b0: disable 0b1: enable
16:8	RATE	BPDU storm control rate of port. Unit: 1pps
7:0	BURST	Burst size of BPDU storm control of port. The unit is 1 packet. Note: The value should be larger than Token value in pps mode.

Table 8-19 STORM_PORT_PROTO_DHCP_CTRL Register

Bits	Field	Description
20	EN	Status of DHCP storm control of port. 0b0: disable 0b1: enable
16:8	RATE	DHCP storm control rate of port. Unit: 1pps
7:0	BURST	Burst size of DHCP storm control of port. The unit is 1 packet. Note: The value should be larger than Token value in pps mode.

Table 8-20 STORM_PORT_PROTO_IGMP_CTRL Register

Bits	Field	Description
20	EN	Status of IGMP/MLD storm control of port. 0b0: disable 0b1: enable
16:8	RATE	IGMP/MLD storm control rate of port. Unit: 1pps
7:0	BURST	Burst size of IGMP/MLD storm control of port. The unit is 1 packet. Note: The value should be larger than Token value in pps mode.

Table 8-21 STORM_PORT_PROTO_ARP_CTRL Register

Bits	Field	Description
20	EN	Status of ARP storm control of port. 0b0: disable 0b1: enable
16:8	RATE	ARP storm control rate of port. Unit: 1pps

7:0	BURST	Burst size of ARP storm control for specific port. The unit is 1 packet. Note: The value should be larger than Token value in pps mode.

Table 8-22 STORM_PORT_PROTO_DHCP_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	DHCP storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

Table 8-23 STORM_PORT_PROTO_BPDU_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	BPDU storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

Table 8-24 STORM_PORT_PROTO_IGMP_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	IGMP storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

Table 8-25 STORM_PORT_PROTO_ARP_EXCEED_FLAG Register

Bits	Field	Description
0	FLAG	ARP storm control meter exceed status of port n. Write 1 to clear. 0b0: not exceed 0b1: exceed

Table 8-26 STORM_CTRL Register

Bits	Field	Description
8	ARP_VLAN_CONSTRT	The state of ARP protocol storm control constrained by VLAN module. If it's enabled, ARP protocol storm only suppresses the packet hitting the VLAN entry which GROUP_MASK<0> is 0b1. 0b0: disable 0b1: enable
7	DHCP_VLAN_CONSTRT	The state of DHCP protocol storm control constrained by VLAN module. If it's enabled, DHCP protocol storm only suppresses the packet hitting the VLAN entry which GROUP_MASK<1> is 0b1. 0b0: disable 0b1: enable

8.4 Meter

8.4.1 Meter Overview

The meter measures the traffic and determines if it exceeds the contract. When traffic exceeds the contract, the packet is marked to colors (green, yellow and red).

The supported features of meter are:

- Dual leaky bucket (DLB)
- Single Rate Three Color Marker (srTCM)
- Two Rate Three Color Marker (trTCM)

System supports 16 meter blocks, and per meter block has 32 meter entries. As Table 8-27, per meter entry contains two meter bucket configurations, which rate and burst of both buckets can be configured. In addition, meter type (DLB, srTCM or trTCM) and counting mode (packet-based or byte-based) can be specified for each meter entry. Color awareness of receiving traffic can also be specified in each meter entry configuration.

In byte-based metering, preamble and IFG lengths can be globally configured to count in or not as Table 8-28. Traffic policing can also be supported by ACL with meter actions, including color drop or color remarking.

Table 8-27 Configuration of each entry in METER Table

Bits	Field	Description
95:94	TYPE	Meter Type 0x0: Disable 0x1: DLB 0x2: srTCM 0x3: trTCM
93	MODE	Meter Mode 0x0: byte-based 0x1: packet-based
92	COLOR_AWARE	0x0: color-blind 0x1: color-aware
91	RST	Reset Meter leaky bucket
90:71	LB0_RATE	Rate of leaky bucket 0
70:54	LB0_BS	Burst size of leaky bucket 0
53:34	LB1_RATE	Rate of leaky bucket 1
33:17	LB1_BS	Burst size of leaky bucket 1

Table 8-28 METER_GLB_CTRL Register

Bits	Field	Description
0	INCL_PREIFG	Packet length include or exclude preamble and IFG 0x0: Exclude 0x1: Include

Note

The same meter entry cannot be used by different phase.

The same meter entry cannot be used by different PBG in the same phase.

8.4.2 Dual Leaky Bucket (DLB)

DLB contains two independent buckets. When traffic is going through, tokens would be added into both buckets. System leaks tokens of both buckets with individual specified rates at the same time.

If any tokens added to both buckets exceed its bucket thresholds (Burst size), the packet would be marked as Red.

8.4.3 Single Rate Three Color Marker (srTCM)

srTCM is based on RFC2697. Two buckets are required to support in srTCM to handle CBS and EBS. Single rate (CIR) is to leak on both buckets in conditions. srTCM can operate in the color-aware and color-blind modes. In the color-aware mode, the colors of incoming packets affect the egressing color. In the color-blind mode, the colors of incoming packets are ignored.

System leaks tokens of bucket 0 with CIR rate if bucket 0 is not empty. Otherwise, system leaks token in bucket 1 with CIR rate. When traffic is going through, tokens would be added into bucket 0, which the number of tokens in bucket 0 is not over the CBS (Green). Once tokens in bucket 0 are over CBS (Yellow), tokens would be added into bucket 1 until the number of tokens over EBS (Red).

8.4.4 Two Rate Three Color Marker (trTCM)

trTCM is based on RFC2698. Two buckets are required to support trTCM to handle CBS and PBS. Each bucket has a specified leaks rate individually (CIR and PIR). trTCM can operate in the color-aware and color-blind modes. In the color-aware mode, the colors of incoming packets affect the egressing color. In the color-blind mode, the colors of incoming packets are ignored.

System leaks tokens in both buckets with specified rate at the same time, which is, CIR rate for bucket 0 and PIR rate for bucket 1. When traffic is going through, tokens would be added into bucket 0 if the number of tokens in bucket 0 is not over CBS (Green). Otherwise, tokens would be added into bucket 1 (Yellow), until the number of tokens over PBS (Red).

8.4.5 Exceed Flag

Each meter entry contains a flag to indicate if the packet exceeds the contract and is marked as Red.

Table 8-29 METER_LB_EXCEED_STS Register

Bits	Field	Description
31:0	LB_EXCEED	Flag of meter entry exceed the contract to marked RED or drop. 0b0: no packet is marked RED or dropped 0b1: some packet is marked RED or dropped Note: Each bit represents a meter entry flag. Each register contains 32 meter entry flag.

To offload S/W polling effort, system aggregates exceed indications of every 16 meter entries to a bit as Table 8-30. If one of 16 meter entries exceeds the contract, the correspondent bit will be turned on.

Table 8-30 METER_LB_GLB_EXCEED_STS Register

Bits	Field	Description
31:0	LB_EXCEED	Group flag for aggregating the 16 meter entries flag result. 0b0: not hit 0b1: hit Note: Each bit represents a group flag.

9 Buffer Management

9.1 Simplified Weighted Random Early Detection

In the general network devices, if flow control ability is enabled, then ASIC will take some actions likes send the pause frames or back pressure to slow down the transmitting speed of link partner when the resource of packet buffer is insufficient. If flow control ability is disabled, packet will be dropped when the resource of packet buffer is insufficient. However, it will cause deterioration of TCP performance.

Here, not only egress drop mechanism but also Simplified Weighted Random Early Detection (SWRED) is provided for the condition of flow control disabled. The drop algorithm can be configured by per-egress port EGBW_CTRL register in Table 7-29. SWRED can randomly early drop packet to improve TCP performance. In addition, SWRED statistically drops more packets from large users than small. Therefore, traffic sources that generate the most traffic are more likely to be slowed down than traffic sources that generate little traffic. SWRED provides individual thresholds and weights for different drop precedence, allowing user to provide different qualities of service for different traffic. The details will be discussed below.

Table 9-1 FC_PORT_EGR_DROP_CTRL Register

Bits	Field	Description
1	EGR_DROP_ALGO	Select egress drop algorithm for specified port (CPU port only supports Egress Drop). 0b0: Egress Drop 0b1: SWRED

Similar with egress tail drop mechanism, SWRED still reference ingress port rx_congestion by default and also supports the configurations of Force Drop and Flooding Traffic HOL Prevention mentioned in Egress Tail Drop.

9.1.1 Threshold Configuration

The device supports three sets of threshold configurations for drop precedence 0~2 to provide different qualities of service for different traffic. After packets incoming switch, all packets are given a drop precedence value. The details can refer Drop Precedence Assignment. In generally, the Red color of packet has higher drop probability than Yellow/Green.

System provides global 12 set configurations for different queue ID. One set configuration contains MIN and MAX in Table 9-2 and RATE in Table 9-3 for three drop precedence n (n=0~2 stand for Green, Yellow and Green). Queue m (m=0~11) in all ports uses one set registers.

Table 9-2 SWRED_Q_THR Register

Table 9-3 SWRED_Q_DROP_RATE Register

Bits	Field	Description

7:0	RATE	Drop rate of egress queue m (0~11) for drop precedence n (0~2). Drop probability = RATE/1023. Suggest: drop probability < 0.1

Bits	Field	Description
31:29	RESERVE	
28:16	MAX	Maximum drop threshold of egress queue m (0~11) for drop precedence n (0~2). Unit: page.
15:13	RESERVE	
12:0	MIN	Minimum drop threshold of egress queue m (0~11) for drop precedence n (0~2). Unit: page.

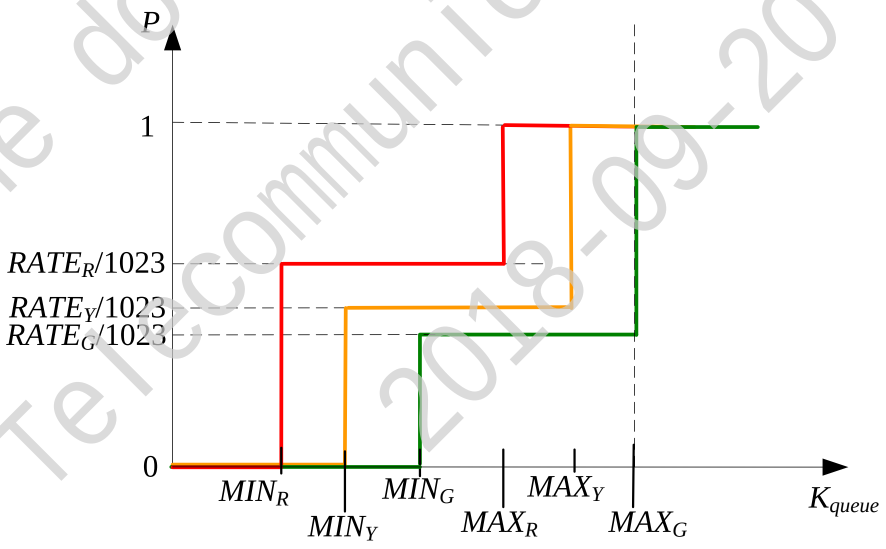
9.1.2 Drop Mechanism

The device checks three conditions before packets are sent into the output queue of the destination egress port:

- Flow control ability of the source ingress port is disabled.
- Currently egress queue length (K_{queue}) is greater than minimum drop threshold MIN.
- System resource is in congestion state.

If the three conditions are all met, the packets will be dropped or forwarded depends on the packet-drop probability (*P*). Figure 9-1 is the relationship of packet-drop probabiltiy *P* and egress queue length K_{queue} with three drop precedence Red, Yellow and Green.

Figure 9-1 SWRED with Different Precedence Setting



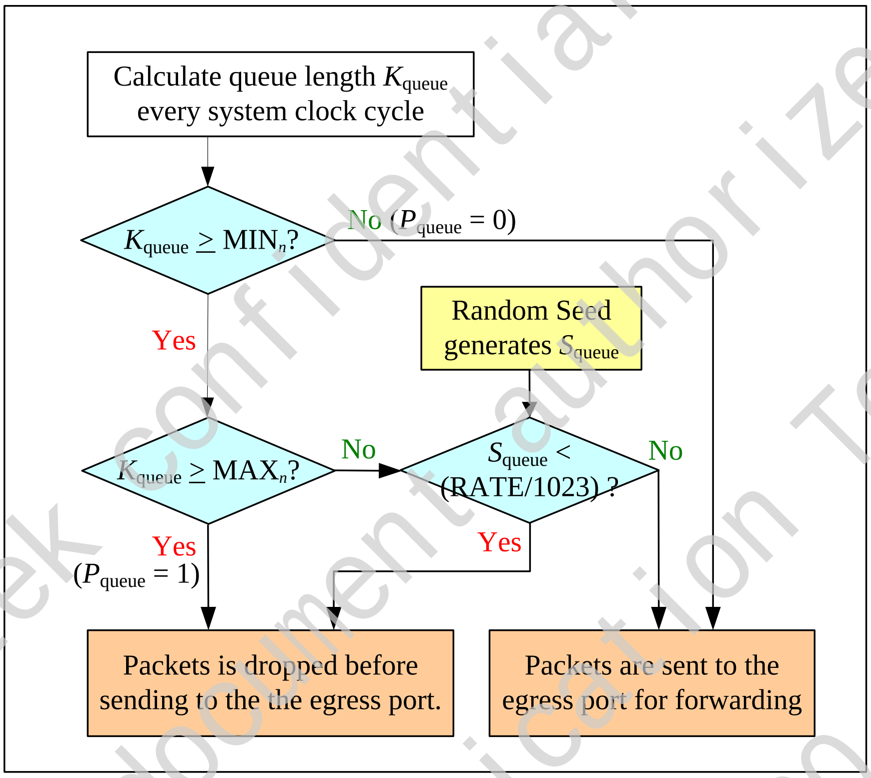
The device calculates parameters K_{queue} every system clock cycle. Packet-drop probabilities of queue-based P_{queue} is updated every system clock cycle by following formulas. The SWRED flow chart is as Figure 9-2.

$$P_{queue} = 1, \quad if \quad K_{queue} \geq \text{MAX} \; ....(a)$$

$$P_{queue} = \frac{\text{RATE}}{1023}, \quad if \quad \text{MIN} \leq K_{queue} < \text{MAX} \; ....(b)$$

$$P_{queue} = 0, \quad if \quad others \; ....(c)$$

Figure 9-2 SWRED Flow Chart


Calculate queue length K_{queue} every system clock cycle
$K_{queue} \geq \text{MIN}_n$?
No (P_{queue} = 0)
Yes
Random Seed generates S_{queue}
$K_{queue} \geq \text{MAX}_n$?
No
S_{queue} < (RATE/1023) ?
No
Yes
Yes (P_{queue} = 1)
Packets is dropped before sending to the the egress port.
Packets are sent to the egress port for forwarding

10 Access Control List

10.1 ACL

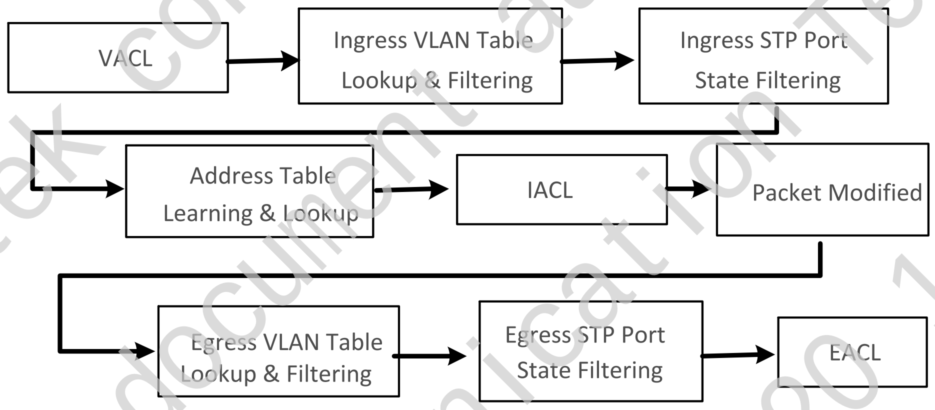
10.1.1 ACL Overview

The Access Control List (ACL) processes and classifies packets at wire speed. ACL is used for operating and managing traffic for flow-based applications such as flow-based ACL, flow-based Quality of Service (QoS), flow-based rate limiting, and flow-based mirroring, and so on.

There are three types of ACL, VLAN-based ACL (VACL), Ingress ACL (IACL) and Egress ACL (EACL). Figure 10-1 shows the position of each types of ACL in forwarding pipeline.

VACL	For ingress packets, classify before L2 lookup and L3 routed.
IACL	For ingress packets, classify after L2 and L3 lookup and before L3 routed.
EACL	For Egress packets, classify Tx packet content.

Figure 10-1 ACL relation pipeline

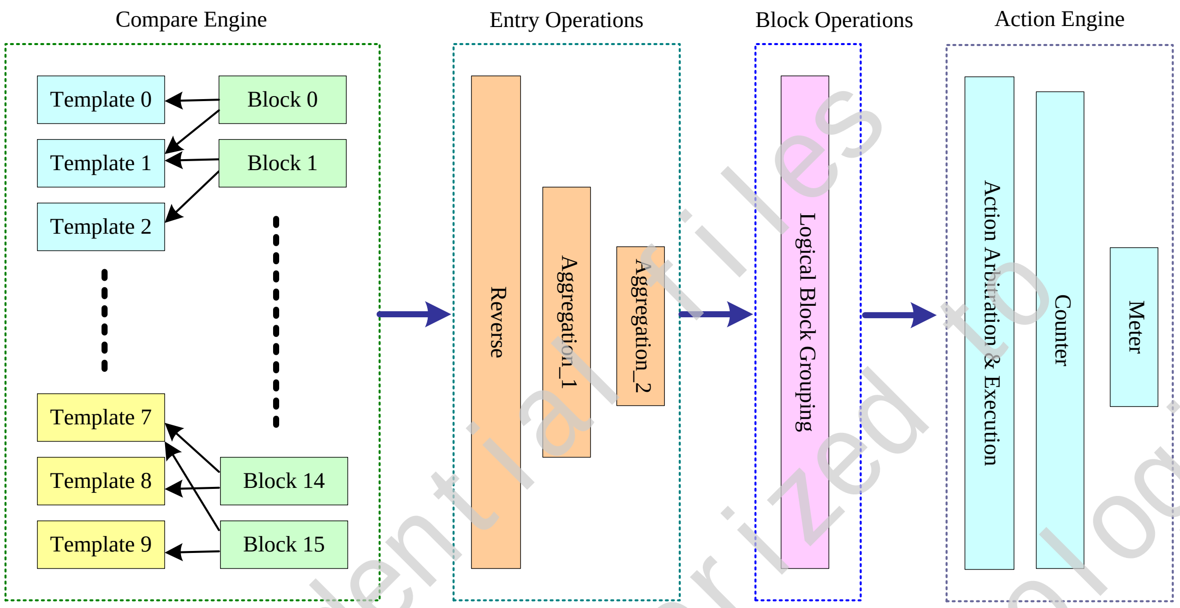

VACL
Ingress VLAN Table Lookup & Filtering
Ingress STP Port State Filtering
Address Table Learning & Lookup
IACL
Packet Modified
Egress VLAN Table Lookup & Filtering
Egress STP Port State Filtering
EACL

ACL is mainly composed of Compare engine and Action engine. The ACL Compare engine exam packet data down to bit level and the Action engine executes the command for the classified packet for specific applications.

The receiving packet is first processed by Compare Engine and outputs hit results. The results are then further processed by ACL entry operations and block operations. Once the final hit results are extracted, the Action Engine executes the actions for the hits entries after running action arbitration.

Same process will be performed in each phase, Figure 10-2 shows the functional block of ACL.

Figure 10-2 ACL Function Block



The system supports:

32 ACL blocks	Each block contains 128 entries.
4096 ACL entries	Each entry includes 240-bit data, with 240-bit care mask and 1-bit valid fields.
10 templates	5 pre-defined templates and 5 configurable templates.
4096 log entries	Each entry contains 36-bits packet-based and 42-bits byte-based counter and binds to each rule.
512 meter entries	Dual leaky bucket, srTCM and trTCM are supported.

10.1.2 PIE Block

Each block can be configured to one of the phases: VACL, IACL or EACL.

Table 10-1 PIE_BLK_PHASE_CTRL Register

Bits	Field	Description
2:0	Phase	Block phase

Each PIE Block group (PBG) can belong to a logical block or adjust the priority by configuration of Group ID (0 ~ 15) and Logic ID (0 ~ 15). Through the configuration, system can support flexible block grouping and dynamic block resource allocation according to application.

Group ID

By configuring the same Group ID, multiple physical blocks can combine to a logical block, and extract only one hit result among combined physical blocks. If packet hits multiple entries in different logical blocks with conflict action, the action will be taken from the logical block with the lowest Group ID.

Logic ID

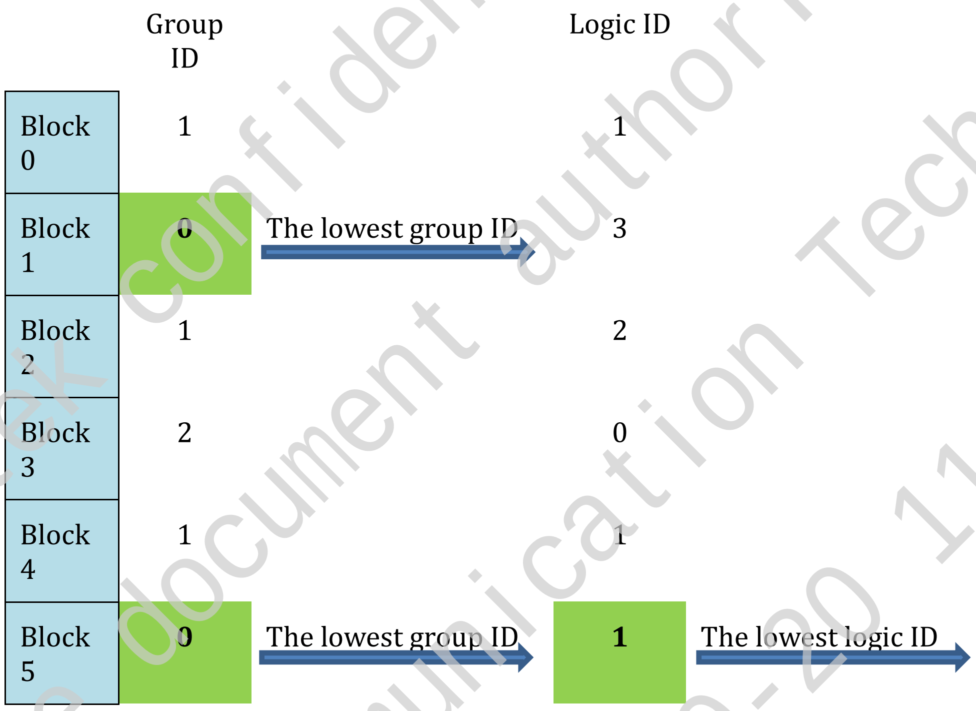
Logic ID determines the priority among physical blocks with the same Group ID. The block with the lowest Logic ID has the highest priority. If several blocks have the same Logic ID and the same Group ID, block with the lowest physical block ID has the highest priority. The hit entry will be determined by compare sequence: Group ID > Logic ID > Physical ID.

Table 10-2 PIE_BLK_GROUP_CTRL Register

Bits	Field	Description
7:4	GRP_ID	Block group ID
3:0	LOGIC_ID	Block logical ID

E.g. in Figure 10-3, Block 5 with the lowest group ID and Logic ID will be extracted for priority comparison.

Figure 10-3 ACL Block Grouping



10.1.3 PIE Rule Template

The system supports 5 pre-defined templates and 5 configurable templates.

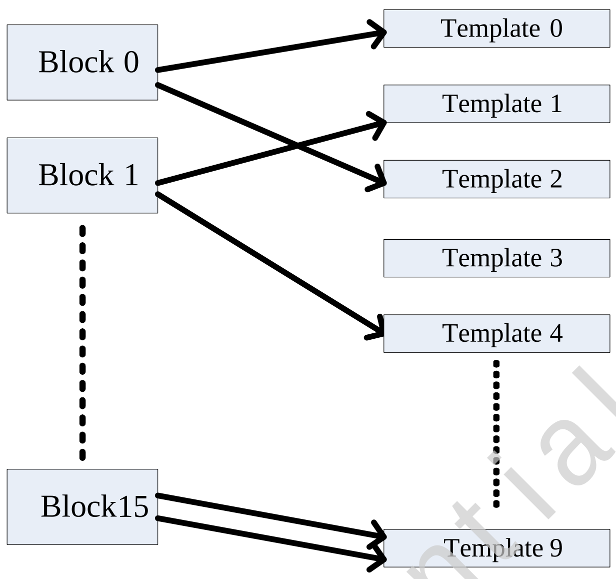
For VACL and IACL, each block can binds to 2 different templates.

For EACL, each block can binds to only one template.

Table 10-3 PIE_BLK_TMPLTE_CTRL Register

Bits	Field	Description
7:4	BLK_TMPLTE2	Maps block to the second template.
3:0	BLK_TMPLTE1	Maps block to the first template.

Figure 10-4 ACL Block and Template Mapping Relationship



10.1.3.1 Pre-defined Template

System supports 5 pre-defined templates with template ID 0~4. They are shown as below：

Table 10-4 Pre-defined Template 0

Field 0	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
DMAC0	DMAC1	DMAC2	SMAC0	SMAC1	SMAC2	VLAN
Field 7	Field 8	Field 9	Field 10	Field 11	Field 12	Field 13
IPTOSPROTO	DSAP_SSAP	ETHER TYPE	SPM0/DPM0	SPM1/DPM1	SPM2/DPM2	SPM3/DPM3

Table 10-5 Pre-defined Template 1

Field 0	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
SIP0	SIP1	DIP0	DIP1	IPTOSPROTO	TCPINFO	L4SPORT
Field 7	Field 8	Field 9	Field 10	Field 11	Field 12	Field 13
L4DPORT	VLAN	RANGECHK	SPM0/DPM0	SPM1/DPM1	SPM2/DPM2	SPM3/DPM3

Table 10-6 Pre-defined Template 2

Field 0	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
DMAC0	DMAC1	DMAC2	VLAN	ETHER TYPE	IPTOSPROTO	SIP0
Field 7	Field 8	Field 9	Field 10	Field 11	Field 12	Field 13
SIP1	DIP0	DIP1	L4SPORT	L4DPORT	MetaData	SLP/DLP

Table 10-7 Pre-defined Template 3

Field 0	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
DIP0	DIP1	DIP2	DIP3	DIP4	DIP5	DIP6
Field 7	Field 8	Field 9	Field 10	Field 11	Field 12	Field 13

DIP7	IPTOSPROTO	TCP_INFO	L4SPORT	L4DPORT	RANGECHK	SLP/DLP

Table 10-8 Pre-defined Template 4

Field 0	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
SIP0	SIP1	SIP2	SIP3	SIP4	SIP5	SIP6
Field 7	Field 8	Field 9	Field 10	Field 11	Field 12	Field 13
SIP7	MetaData	VLAN	SPM0/DPM0	SPM1/DPM1	SPM2/DPM2	SPM3/DPM3

Note

SPM and SLP are applicable for VACL and IACL.

DPM and DLP are applicable for EACL.

10.1.3.1.1 Pre-defined Template VLAN Selection

VLAN information in pre-defined templates 0, 1, 2 and 4 for each phase can be further configured to match IVLAN, OVLAN or FVLAN as Table 10-9 for applications.

Table 10-9 PIE_CTRL Register

Bits	Field	Description
23:22	E_TMPLTE0_IOTAG_SEL	Configure the template 0 VLAN for EACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)
21:20	E_TMPLTE1_IOTAG_SEL	Configure the template 1 VLAN for EACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)
19:18	E_TMPLTE2_IOTAG_SEL	Configure the template 2 VLAN for EACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)
17:16	E_TMPLTE4_IOTAG_SEL	Configure the template 4 VLAN for EACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)
15:14	V_TMPLTE0_IOTAG_SEL	Configure the template 0 VLAN for VACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based and exclude IACL assignment)
13:12	V_TMPLTE1_IOTAG_SEL	Configure the template 1 VLAN for VACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based and exclude IACL assignment)

11:10	V_TMPLTE2_IOTAG_SEL	Configure the template 2 VLAN for VACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based and exclude IACL assignment)
9:8	V_TMPLTE4_IOTAG_SEL	Configure the template 4 VLAN for VACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based and exclude IACL assignment)
7:6	I_TMPLTE0_IOTAG_SEL	Configure the template 0 VLAN for IACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)
5:4	I_TMPLTE1_IOTAG_SEL	Configure the template 1 VLAN for IACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)
3:2	I_TMPLTE2_IOTAG_SEL	Configure the template 2 VLAN for IACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)
1:0	I_TMPLTE4_IOTAG_SEL	Configure the template 4 VLAN for IACL 0x0: IVLAN (ITAG) 0x1: OVLAN (OTAG) 0x2: FVLAN (depends on forward based)

10.1.3.1.2 Template Tag Format Selection

For VACL, each template can select ITAG and OTAG format to exam by configuration of Table 10-10.

Table 10-10 PIE_TAG_CTRL Register

Bits	Field	Description
n	V_TMPLTEn_IOTAG_FMT	Configure the template OTAG/ITAG Format 0b0: original 0b1: translated (exclude VACL)

10.1.3.2 Configurable Template

System supports 5 configurable templates with template ID 5-9. Each configurable template has 14 fields to determine matching characteristics of packet.

Table 10-11 Configurable Template Format

Field 0	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
FCS0	FCS1	FCS2	FCS 3	FCS 4	FCS5	FCS6
Field 7	Field 8	Field 9	Field 10	Field 11	Field 12	Field 13
FCS7	FCS8	FCS9	FCS10	FCS11	FCS12	FCS13

10.1.3.3 Field Content Selection (FCS)

FCS definitions are as below：

Name	FCS name
Bits of value	Define each bit of this FCS
Bits Allocation	Define the allocation in template. This FCS only can be configured in specified field of template.
VACL	This FCS is support for VLAN ACL.
IACL	This FCS is support for Ingress ACL
EACL	This FCS is support for Egress ACL

Note

Due to the template is not dedicated for a specific ACL phase, S/W must avoid a block binding to the template, which contains the un-supported Field Type for the phase.

Table 10-12 Definition of FCS

Name	Bits of value	Bits Allocation	VACL	IACL	EACL
SPM0	[15:0] Source Physical Port Mask[15:0]	F10	O	O	
SPM1	[15:0] Source Physical Port Mask[31:16]	F11	O	O	
SPM2	[15:0] Source Physical Port Mask[47:32]	F12	O	O	
SPM3	[15] loopback packet [14:9] VRF-ID [8:0] Source Physical Port Mask[56:48]	F13	O	O	
DPM0	[15:0] Destnation Physical Port Mask[15:0]	F10			O
DPM1	[15:0] Destnation Physical Port Mask[31:16]	F11			O
DPM2	[15:0] Destnation Physical Port Mask[47:32]	F12			O
DPM3	[15] loopback packet [14:9] VRF-ID [8:0] Destination Physical Port Mask[56:48]	F13			O
DMAC0	DMAC[15:0] For ARP/RARP packet, If *PIE_MISC.ARP_MAC_CTRL* = 1, this is the target hardware address. Else this is the destination MAC address.	F0/F3/F6/F9	O	O	O

DMAC1	DMAC[31:16] For ARP/RARP packet, If *PIE_MISC.ARP_MAC_CTRL* = 1, this is the target hardware address. Else this is the destination MAC address.	F1/F4/F7/F10	O	O	O
DMAC2	DMAC[47:32] For ARP/RARP packet, If *PIE_MISC.ARP_MAC_CTRL* = 1, this is the target hardware address. Else this is the destination MAC address.	F2/F5/F8/F11	O	O	O
SMAC0	SMAC[15:0] For ARP/RARP packet, If *PIE_MISC.ARP_MAC_CTRL* = 1, this is the sender hardware address. Else this is the source MAC address.	F0/F3/F6/F9	O	O	O
SMAC1	SMAC[31:16] For ARP/RARP packet, If *PIE_MISC.ARP_MAC_CTRL* = 1, this is the sender hardware address. Else this is the source MAC address.	F1/F4/F7/F10	O	O	O
SMAC2	SMAC[47:32] For ARP/RARP packet, If *PIE_MISC.ARP_MAC_CTRL* = 1, this is the sender hardware address. Else this is the source MAC address.	F2/F5/F8/F11	O	O	O
ETHERTYPE			O	O	O
OTAG	[15:13] OPRI/Port-based Priority [12] DEI [11:0] OVID/O-PVID		O	O	O
ITAG	[15:13] IPRI/Port-based Priority [12] CFI [11:0] IVID/I-PVID		O	O	O
SIP0	For IPv4/IPv6, source IP[15:0] For ARP/RARP, sender protocol address	F0/F2/F4/F6/F8/F10/F12	O	O	O
SIP1	For IPv4/IPv6, source IP[31:16] For ARP/RARP, sender protocol address	F1/F3/F5/F7/F9/F11/F13	O	O	O
DIP0	For IPv4/IPv6, destination IP[15:0] For ARP/RARP, target protocol address	F0/F2/F4/F6/F8/F10/F12	O	O	O
DIP1	For IPv4/IPv6, destination IP[31:16] For ARP/RARP, target protocol address	F1/F3/F5/F7/F9/F11/F13	O	O	O
IPTOSPROTO	[15:8] IPv4 TOS/IPv6 Traffic Class [7:0] IPv4 Protocol/IPv6 Next Header [15:0] ARP/RARP op-code For Ethernet and SNAP packet exclude IPv4, IPv6, ARP/RARP, OAM, PPPoE, packet [15:0] After ethertype byte 0,1		O	O	O

L4SPORT	[15:0]TCP/UDP/SCTP Source Port [15:8] ICMP Type/ICMPv6 Type/IGMP Type [7:0] ICMP Code/ICMPv6 Code/IGMP MAX Resp Code [15:0] L4 header byte 0 and 1 [15:0] For ARP/RARP packet, sender hardware address [15:0]	F0/F2/F4/F6/F8/F10/F12	O	O	O
L4DPORT	[15:0] TCP/UDP/SCTP Destination Port [15:0] L4 header byte 2 and 3 [15:0] For ARP/RARP packet, sender hardware address [31:16]	F1/F3/F5/F7/F9/F11/F12	O	O	O
L34HEADER	[15] UNSEQ (Hop-by-Hop header position error) [14] UNREP (Unexpected repeats extesnsion header) [13] NONEXT (Ipv6 packet with No Next header) [12] IPv6 packet with Mobility header [11] IPv6 packet with ESP header [10] IPv6 packet with authentication header [9] IPv6 packet with destination option header [8] IPv6 packet with fragment header [7] IPv6 packet with routing header [6] IPv6 packet with hop-by-hop option header [5:3] IPv4 Flags/IPv6 Rsvd+M-Flag [2] IP fragment packet [1:0] IPv4 TTL/IPv6 Hop Limit 2b'00: TTL = 0 2b'01: TTL = 1 2b'10: 2<=TTL<255 2b'11: TTL = 255 [15:0] For ARP/RARP packet, target hardware address [15:0]		O	O	O
TCPINFO	[15:14] DP (color) [13] Extra-tagged [12:9] flow label[19:16] [8] TCP Non Zero Sequence [7:6] TCP ECN([7]: ECE, [6]: CWR) [5:0] TCP Flag([5]: URG, [0]: FIN) [15:0] For ARP/RARP packet, sender hardware address [47:32]	F1/F3/F5/F7/F9	O	O	O
FIELD_SELECTOR_VALID	[15] IPv4 Header Error [14] L2 Error [13:0] Field Selector Valid Mask		O	O	
FIELD_SELECTOR0	User defined 16-bit field selector 0		O	O	
FIELD_SELECTOR1	User defined 16-bit field selector 1		O	O	
FIELD_SELECTOR2	User defined 16-bit field selector 2		O	O	

FIELD_SELECTOR3	If *PIE_MISC.FILTER_8021BR_EN* = 1, [15:13] E-PCP [12] E-DEI [11:0] ingress ECID base Else User defined 16-bit field selector 3		O	O	
FIELD_SELECTOR4	If *PIE_MISC.FILTER_8021BR_EN* = 1, [14] PE-TAG exist [13:12] ECID GRP [11:0] ECID base Else User defined 16-bit field selector 4		O	O	
FIELD_SELECTOR5	If *PIE_MISC.FILTER_8021BR_EN* = 1, [15:8] ingress ECID ext [7:0] ECID ext Else User defined 16-bit field selector 5		O	O	
FIELD_SELECTOR6/ SIP2	User defined 16-bit field selector 6	F2	O	O	
	IPv6 SIP[47:32]		O	O	O
FIELD_SELECTOR7/ SIP3	User defined 16-bit field selector 7	F3	O	O	
	IPv6 SIP[63:48]		O	O	O
FIELD_SELECTOR8/ SIP4	User defined 16-bit field selector 8	F4	O	O	
	IPv6 SIP[79:64]		O	O	O
FIELD_SELECTOR9/ SIP5	User defined 16-bit field selector 9	F5	O	O	
	IPv6 SIP[95:80]		O	O	O
FIELD_SELECTOR10/ SIP6	User defined 16-bit field selector 10	F6	O	O	
	IPv6 SIP[111:96]		O	O	O
FIELD_SELECTOR11/ SIP7	User defined 16-bit field selector 11	F7	O	O	
	IPv6 SIP[127:112]		O	O	O
FIELD_SELECTOR12	User defined 16-bit field selector 12	F8	O	O	
FIELD_SELECTOR13	User defined 16-bit field selector 13	F9	O	O	
DIP2	IPv6 DIP[47:32] [15:0]: For ARP/RARP packet, target hardware address [47:32]	F2	O	O	O
DIP3	IPv6 DIP[63:48]	F3	O	O	O
DIP4	IPv6 DIP[79:64]	F4	O	O	O
DIP5	IPv6 DIP[95:80]	F5	O	O	O
DIP6	IPv6 DIP[111:96]	F6	O	O	O
DIP7	IPv6 DIP[127:112]	F7	O	O	O

PKT_INFO	[15:10] Source Physical Port [9:8]: DMAC Type 2b'00: unicast 2b'01: broadcast 2b'10: Reserved 2b'11: multicast For VACL: [7:3]: Reserved [2]: IP subnet based hit [1]: MAC-based hit [0]: IVC hit For IACL: [7]: VACL drop action hit [6]: VACL copy action hit [5]: VACL redirect action hit [4]: Attack packet [3]: SMAC hit (L2 unicast) (0: new SA) [2]: DIP hit L3 host routing entry [1]: DIP hit L3 prefix routing entry [0]: DMAC hit is destination MAC address exists in route MAC, L2 unicast or L2 multicast table		O	O	
FLOW LABEL	[15:0]: flow label[15:0] [15:0]: For ARP/RARP packet, target hardware address [31:16]	F0/F2/F4/F6/F8	O	O	O
DSAP_SSAP	[15:8]: DSAP for LLC/SNAP packet [7:0]: SSAP for LLC/SNAP packet For GRE packet, [15:0] GRE key[15:0] For MPLS packet, [15:0] second MPLS label [15:0] For GTP packet, [15:0] TEID[15:0]	F0/F2/F4/F6/F8	O	O	
SNAP OUI	[15:0]: OUI[15:0] in SNAP header For GRE packet, [15:0] GRE key [31:16] For MPLS packet, [15:8] Reserved [7] second MPLS BoS bit [6:4] second MPLS TC [3:0] second MPLS label [19:16] For GTP packet, [15:0] TEID[31:16]	F1/F3/F5/F7/F9	O	O	

FWDVID	[12] Content Too Deep [11:0] Forwarding VID For IACL, [15] Reserved [14] IPUC routing: packet to do IPUC routing [13] IPMC routing: packet to do IPMC routing For VACL and EACL [15:13]: Reserved		O	O	O
RANGECHK	[15:0]: VID/L4 port/Packet Length/L3 Packet Length Range Check Mask *EACL is not support packet length and L3 packet length*		O	O	O
VLAN_GMSK	Forward VLAN group mask From VLAN table, Per VLAN 16 bits		O	O	
SLP	[10] trunk bit. If RX port is trunk logic If RX port is trunk member port, [9:7] Reserved [6:0] trunk ID Else [9:6] device ID [5:0] ingress port ID For IACL, [15] Reserved [14] uRPF check fail [13] PORT_MV [12] IGR_VLAN_FILTER_DROP [11] STP_DROP		O	O	O
DLP	For VACL and IACL [15:8] Reserved [7:0] SNAP_OUI[23:16] For EACL [15:11] Reserved [10] trunk bit. If TX port is trunk logic port. If TX port is trunk member port: [9:7] Reserved [5:0] trunk ID Else [9:6] device ID [5:0] egress port ID		O	O	O
META_DATA	[15:12] Loopback times [11:0] META_DATA		O	O	O

TTID	[15] second MPLS label exist [14] first MPLS label exist [13:11] application type 0x0: MPLS 0x1: GRE 0x2: GTP 0x3: VxLAN 0x4: CAPWAP 0x5: Diameter 0x6: other else [10] tunnel terminated packet If *PIE_MISC.INTF_SEL* = interface, [9:0] ingress interface index Else [9] Reserved [8:0] tunnel terminate index	F0/F2/F4/F6/F8/F10/F12	O	O	
TSID	[15:11] Reserved [10] tunnel start packet [9] Reserved [8:0] tunnel start index	F1/F3/F5/F7/F9/F11/F13		O	O
First_MPLS1	[15:0] first MPLS label[15:0] For VxLAN, [15:0] VNI[15:0]	F0/F2/F4/F6/F8/F10/F12	O	O	
First_MPLS2	[15:8] IP range check [7] first MPLS BoS bit [6:4] first MPLS TC [3:0] first MPLS label[19:16] For VxLAN, [7:0] VNI[23:16]	F1/F3/F5/F7/F9/F11/F13	O	O	

Note

Reserved bits in some field must configured as don't care bits to prevent the matching.

10.1.3.3.1 SPM

SPM (Source physical Port bit Map) can be applied to match packet from multiple ingress ports with only one ACL rule.

E.g. configuring a rule to apply on port 1, 3 or 5 using SPM

Data bits of SPM = all zero
Care bits of SPM for port 1, 3, 5 = 0'b
Care bits of SPM for other ports = 1'b

10.1.3.3.2 DMAC/SMAC

DMAC/SMAC stands for destination/source MAC address. For ARP/RARP packets, DMAC/SMAC can stand for destination/source MAC address or target/sender hardware address by configuration as Table 10-13.

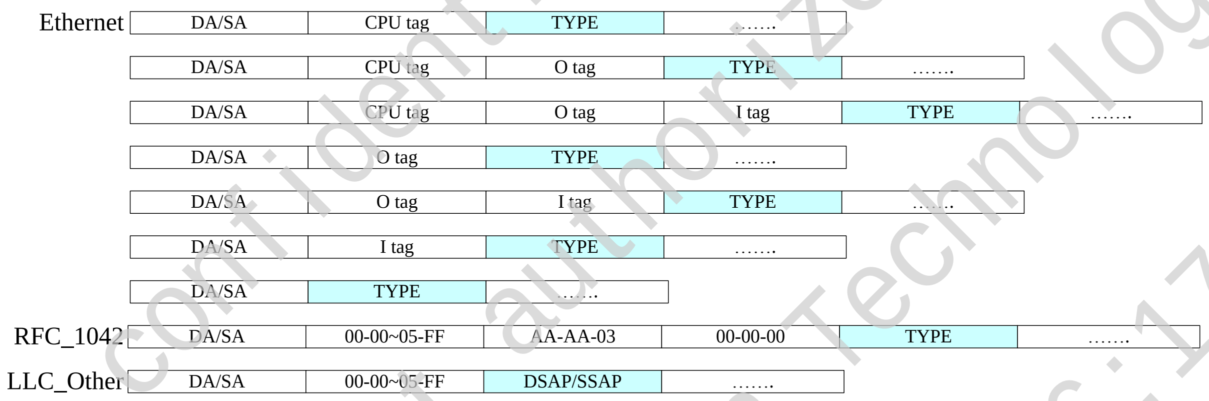
Table 10-13 PIE_MISC Register

Bits	Field	Description
0:0	ARP_MAC_CTRL	Configure SMAC/DMAC address select from: 1'b0: L2 source/destination MAC address 1'b1: sender/target hardware address in payload.

10.1.3.3.3 ETHERTYPE

The position of ETHERTYPE in recognized packets are as Figure 10-5.

Figure 10-5 Ethertype Field Position


Ethernet
DA/SA | CPU tag | TYPE | …….
DA/SA | CPU tag | O tag | TYPE | …….
DA/SA | CPU tag | O tag | I tag | TYPE | …….
DA/SA | O tag | TYPE | …….
DA/SA | O tag | I tag | TYPE | …….
DA/SA | I tag | TYPE | …….
DA/SA | TYPE | …….
RFC_1042
DA/SA | 00-00~05-FF | AA-AA-03 | 00-00-00 | TYPE | …….
LLC_Other
DA/SA | 00-00~05-FF | DSAP/SSAP | …….

For parsing the RFC_1042 header, system provides a configuration as Table 10-14 to omit comparing the OUI (00-00-00), i.e. a frame with format AA-AA-03-XX-XX-XX could be deemed a RFC_1042 frame if the configuration is enabled.

Table 10-14 PARSER_CTRL Register

Bits	Field	Description
0:0	RFC1042_OUI_IGNORE	Ignore comparing the OUI while parsing the RFC1042 header. That is, a frame with format "AA-AA-03-XX-XX-XX" could be deemed a RFC1042 frame. 0b0: don't ignore 0b1: ignore

10.1.3.3.4 OTAG/ITAG

OVID/IVID

For VACL, depends on the setting of PIE_TAG_CTRL.V_TMPLTEn_IOTAG_FMT to fetch VID. If it is configured as original, then the VID is from VLAN tag. When packet is priority-tagged or untagged, the VID is from PVID. If it is configured as translated, then fetch the VID after ingress VLAN translation.

For IACL, fetch VID after VACL phase.

OPRI/IPRI

Defined as O-PRI/I-PRI for the tagged or priority-tagged packet and defined as port-based priority for the untagged packet.

DEI/CFI

Defined as DEI/CFI for the tagged or priority-tagged packet and defined as 0 for the untagged packet.

10.1.3.3.5 IPTOSPROTO

IPv6 Next Header

IPv6 Next Header (IPTOSPROTO [7:0]) extracted from the last known Next Header if there are multiple Next Headers appeared in a packet. If a packet contains too many next headers to parse NH due to over parser length, the NH field is 0x00 and CONTENT_TOO_DEEP is set to 1.

10.1.3.3.6 L4SPORT/L4DPORT

L4SPORT is byte 0 and 1 in L4 header, L4DPORT is byte 2 and 3 in L4 header.

10.1.3.3.7 FIELD_SELECTOR

Field Selector 6-11 are defined as combo fields, which are interpreted as IPv6 SIP [127:32] if the received packet is an IPv6 packet.

i.e. If the filed selectors would be applied on both IPv4 and IPv6 traffic, user must use Field Selector 0-5.

Note

Field Selector 0-5, 12 and 13 cannot be used for matching the packet of stacking ports.

10.1.3.3.8 PKT_INFO

SMAC hit

The bit stands for the source MAC of packet hit the MAC entry learnt in the L2 unicast table.

DMAC hit

The bit stands for the destination MAC of packet hit the MAC entry in route MAC, L2 unicast or L2 multicast tables.

10.1.3.3.9 FWDVID

FWDVID stands for the forwarding VLAN ID determined by VLAN assignment function.

Note

When using VACL, the FWDVID field cannot match the VID assigned by VACL VID assignment action.

10.1.3.3.10 RANGE_CHECK

System supports 16 range check for L4_SPORT/L4_DPORT/L4_PORT/OVID/IVID/PKT_LEN/L3_PKT_LEN. And system supports 8 IP range check.

10.1.3.3.11 VLAN_GMSK

VLAN_GMSK stands for VLAN group mask and is useful for discrete VLAN-based application.

10.1.3.3.12 META_DATA

META_DATA can be used for co-work with previous ACL phase. Default META_DATA of packet is 0 if not being assigned. By using META_DATA, ACL can filter packet characteristic more width and flexible in more than one phase for the same packet, and can also support to build hierarchy mechanism.

10.1.4 PIE Rule

10.1.4.1 Rule Structure

Each PIE rule contains 14 user defined fields (map to selected template) and 1 Fixed Field as Figure 3-1, and is totally 240-bits width. The user defined field and the fixed field are all 16-bits width. Figure 10-7 shows the format of Fix Field.

Figure 10-6 PIE Rule Structure

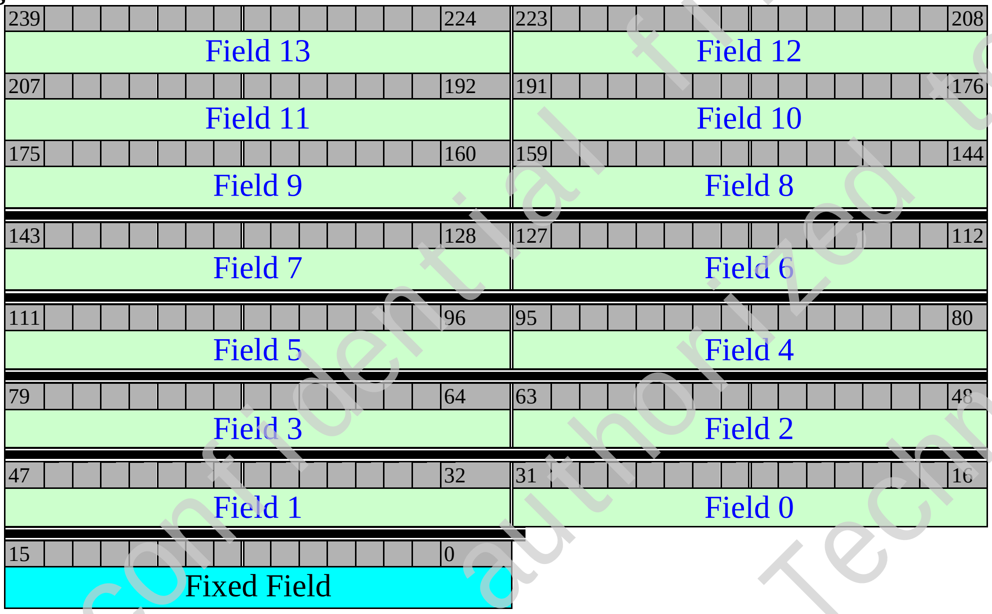


Figure 10-7 Fixed Field Format

Bit 0	Template ID
Bit 1	IGR_NML_PORT/EGR_NML_PORT
Bit 2, 3	2b00: Reserved 2b01: Ethernet 2b10: LLC other 2b11: LLC SNAP
Bit 4	Inner tag or inner priority tag packet
Bit 5	Outer tag or outer priority tag packet
Bit 6	Inner untag or inner priority tag packet
Bit 7	Outer untag or outer priority tag packet
Bit 8, 9	2b00: ARP packet 2b01: L2 packet 2b10: IPv4 packet 2b11: IPv6 packet
Bit 10 ~ 12	0x0: UDP packet 0x1: TCP packet 0x2: ICMP/ICMPv6 packet 0x3: IGMP packet 0x4: MLD packet 0x5: SCTP packet 0x6: L4 unknown

Bit 13	IPv4/IPv6 Non-Zero offset packet
Bit 14	Content too deep/DEV_DMAC
Bit 15	MGMT_VLAN

Bit 1

IGR_NML_PORT bit, for VACL and IACL, is asserted if the receiving port of packet is not stacking and CPU port.

EGR_NML_PORT bit, for EACL, is asserted if the sending port of packet is not stacking and CPU port.

Bit 14

For VACL, the bit represents Content Too Deep, and is asserted if some fields over the limited packet length of parser. The field is excluded status of field selector; field selector is handled by field selector valid.

For IACL and EACL, the bit represents DEV_DMAC, and is asserted when the DMAC of packet is one of the router MAC entries.

Bit 15

MGNT_VLAN is asserted when FWDVID is the same with PVID of CPU port.

10.1.4.2 Rule Operation

NOT

Each PIE rule has a corresponding **NOT** operation which can be used to reverse the PIE lookup result to support more flexible applications.

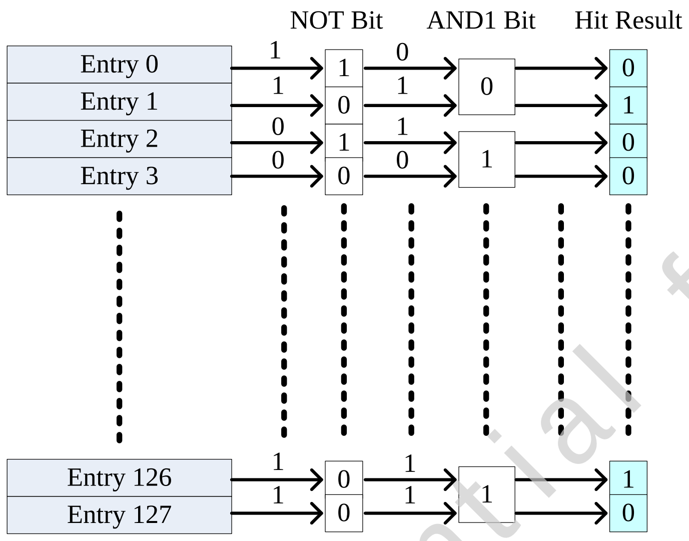
AND1

Two adjacent rules (index 0&1、2&3 ⋯ 2n&2n+1) can combine to inspect more information of a packet with two templates. To combine the rules, even rule (2n) can be configured with the **AND1** operation to indicate doing the AND operation for the adjacent rule (2n + 1). The hit result of the combined rule will be extracted to the rule with lower index as Figure 10-8.

Note

1. AND1 operation only takes effect for two adjacent entries within the same block.
2. AND1 operation cannot be supported by EACL.

Figure 10-8 PIE Rule Hit Result-AND1



AND2

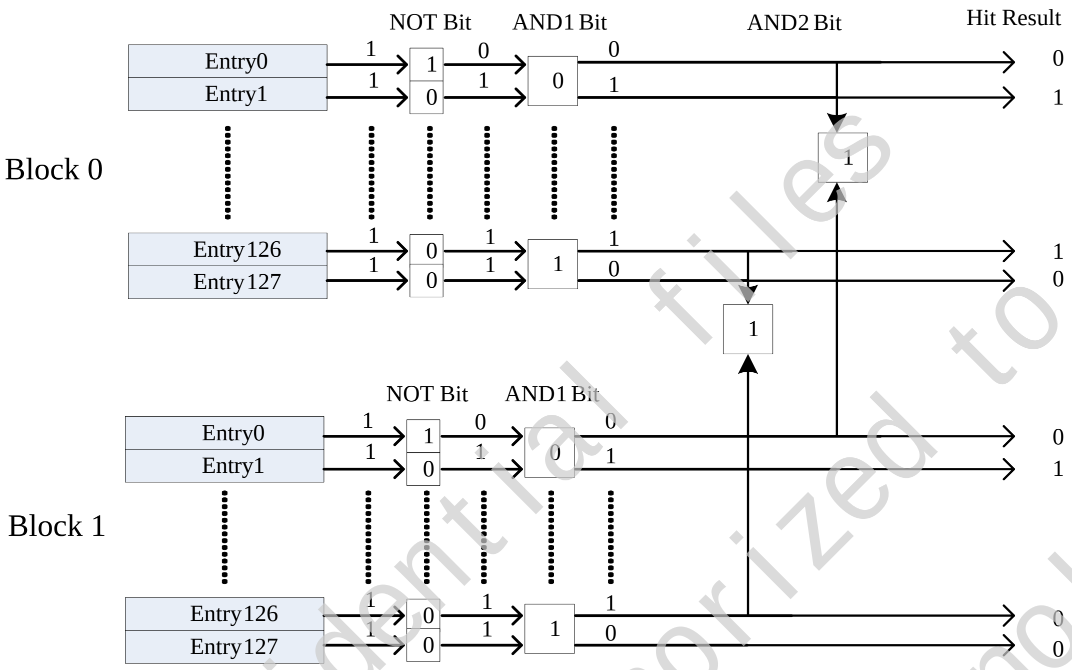
System also support combining 2 rules separated in adjacent block (0&1, 2&3, ..., 2n&2n+1) by **AND2** operation. After **AND2** operation is configured in a rule, the other rule with the same index in adjacent block would be combined. The hit result would be extracted to the rule in the block with lowest ID.

Furthermore, with **AND1** and **AND2** operation enabled concurrently, at most 4 rules can be combined to inspect a packet for more information with 4 different templates, i.e. two **AND1** rules of the same index in adjacent blocks can be combined by **AND2** operation. The hit result of the combined rules would be extracted to the rule with the lowest index in the block with lowest block ID as Figure 7-11.

Note

AND2 operation cannot apply to the rules in different phases.

Figure 10-9 PIE Rule Hit Result-AND2



10.1.4.3 Rule Hit Indication

Each PIE rule supports a hit indication which is asserted if the rule matched the packet. Within a block group, only one entry with the lowest index assert the hit indication.

Table 10-15 PIE_RULE_HIT_INDICATION Register

Bits	Field	Description
31:0	RULE_INDICATION	PIE rule hit indication. Write 1 to clear. 0b0: not hit 0b1: hit Note: Each bit stands for a PIE rule hit indication. Each register contains 32 PIE rules hit indication.

In order to offload S/W polling effort, system supports 4 global hit indication registers to aggregate the hit results. Each global indication bit aggregates 32 PIE rules hit results. Any rule among the 32 rules hit, the aggregated bit will be asserted as well.

Table 10-16 PIE_GLB_HIT_INDICATION Registers

Bits	Field	Description
31:0	GLB_INDICATION	Global indication. Each bit presents 32 PIE entries hit results. 0b0: not hit 0b1: hit

10.1.5 ACL Action

Each ACL entry maps to an action mask register and an action table entry, action mask register is used to indicate the actions to be taken while the entry is hit, and the action table entry stores the data that is needed for executing the actions.

10.1.5.1 VACL Action

Action mask register of VACL is shown as below:

- Drop
- Yellow drop
- Red drop
- Forward
- Log
- Mirror
- Meter
- IVLAN
- IPRI
- OVLAN
- OPRI
- Tag status
- Priority
- Bypass
- Meta data
- QID
- Remark
- Yellow remark
- Red remark
- Invert reserve flag of IPv4 header

The format of action table entry is shown as below:

Table 10-17 VACL Action Table Entry

Fields	Description

DROP_ACT	The action is for packet which is mark Green. 0: Permit(Do Nothing) 1: Drop
YELLOW_DROP_ACT	The action is for packet which is mark Yellow. 000: Permit(Do Nothing) 001: Drop
RED_DROP_ACT	The action is for packet which is mark Red. 000: Permit(Do Nothing) 001: Drop
FWD_ACT	000: Permit(Do Nothing) 001: Drop 010: Copy frame to FWD_PORT_INFO 011: Copy frame to port mask pointed by FWD_PORT_INFO 100: Redirect frame to FWD_PORT_INFO 101: Redirect frame to port mask pointed by FWD_PORT_INFO 110: Force Unicast Routing 111: Default Unicast Routing. For unicast packet if there is no explicit route for this destination.
FWD_CPU_PKT_FMT	Only for FWD_ACT = 010 ~ 101 and packet to CPU 00: original packet 01: modified packet
FWD_SA_LRN	0: Null operation 1: Not learn
FWD_SEL	If DROP_ACT and FWD_ACT is conflict. 0: select DROP_ACT 1: select FWD_ACT Ex1. DROP_ACT = drop and FWD_ACT = copy are take at the same time: If FWD_SEL = 0, packet is drop. Else FWD_SEL = 1, packet is copy and drop. Ex2. DROP_ACT = drop and FWD_ACT = redirect/routing are take at the same time: If FWD_SEL = 0, packet is drop. Else FWD_SEL = 1, packet is redirect/routing.

FWD_PORT_INFO	FWD_ACT=010/100，[10]: Trunk bit, [9:6]: unit ID, [5:0]: Forwarding Port ID/Trunk ID
	FWD_ACT=011/101，[9:0]: index to multicast table for retrieving the port mask
	FWD_ACT=110/111,
	  [14:13] Type
	    0x0: [12:0] = Next Hop index
	    0x1: [7:0] = ECMP index
	    0x2: Null interface
	      [1:0]:
	        0x0 = drop
	        0x1 = trap to local CPU
	        0x2 = trap to master CPU

MIRROT_ACT	0: mirror 1: Cancel mirror

MIRROR_INDEX	For MIRROR_ACT=0, Mirror set index
METER_INDEX	Meter index
IVLAN_VID_ACT	00: Assign new I-VID 01: Shift I-VID 10: Shift from O-VID 11: Reserved

IVLAN_DATA	IVID_ACT=0，[11:0]: New I-VID
	IVID_ACT=1 and 2, [11:0]: Value to shift

IPRI_ACT	Inner tag priority 00: Assign inner tag priority 01: Copy from outer tag priority 10: Keep inner tag priority 11: Reserved
IPRI_DATA	
OVLAN_VID_ACT	00: Assign new O-VID 01: Shift O-VID 10 : Shift from I-VID 11 : Reserved

OVLAN_VID_DATA	OVID_ACT=0, [11:0]: New O-VID
	OVID_ACT=1 and 2, [11:0]: Value to shift

OPRI_ACT	Outer tag priority 00: Assign outer tag priority 01: Copy from inner tag priority 10: Keep outer tag priority 11: Reserved
OPRI_DATA	
TAG_STATUS_INNER	00: Inner untag 01: Inner tag 10: Keep inner tag status 11: NOP(don’t touch)
TAG_STATUS_OUTER	00: Outer untag 01: Outer tag 10: Keep outer tag status 11: NOP(don’t touch)
INTERNAL_PRI	Internal priority
BYPASS_DATA	[0]: bypass ingress bandwidth control and storm control modules [1]: bypass STP source checking for blocking and learning state [2]: bypass ingress VLAN filter.
META_DATA	Co-operate with next phase ACL
QID_ACT	CPU Queue ID
QID_DATA	CPU Queue ID
REMARK_ACT	The action is for packet which is mark Green. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS 11: Remark EAV

REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value [1:0]: 0 is non-EAV, 1 is class A, 2 is class B.
YELLOW_REMARK_ACT	The action is for packet which is mark Yellow. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
YELLOW_REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value
RED_REMARK_ACT	The action is for packet which is mark Red. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
RED_REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value

10.1.5.1.1 Drop/Yellow Drop/Red Drop Action

Depend on packet Drop Precedence (color) to take the drop action

10.1.5.1.2 Forward Action

Permit (FWD_ACT = 000'b)

Default forward action

Drop (FWD_ACT = 001'b)

Drop the packet

Copy packet to single port (FWD_ACT = 010'b)

Forward a copied packet to a specific port or trunk according to the FWD_PORT_INFO. If trunk bit = 0 and port ID = 62, the packet will be copied to master CPU port and ignore unit configuration.

Copy packet to multiple ports (FWD_ACT = 011'b)

Forward copied packets to multiple ports according to the port mask indexed by FWD_PORT_INFO.

Note

The copied packets, which are not destined to CPU, would further process by forwarding decision module such as STP, 802.1X, VLAN egress filtering and port isolation as original packets.

Redirect packet to single port (FWD_ACT = 100'b)

Redirect packet to another port or trunk according to the FWD_PORT_INFO. If trunk bit = 0 and port ID = 62, redirect packet to master CPU port and ignore unit configuration.

Redirect packet to multiple ports (FWD_ACT= 101'b)

Redirect packets to multiple ports according to the port mask indexed by FWD_PORT_INFO.

Note

Packets redirected by ACL are regarded as known packet, i.e. unknown traffic action would take no effect on ACL redirected packets.

Force unicast route (FWD_ACT= 110'b)

The routed packet would be routed to next hop indexed by FWD_PORT_INFO. If the next hop entry is invalid, the packet would be drop.

Default unicast route (FWD_ACT= 111'b)

The routed packet, which doesn't hit route entry or hit the default route, would be routed to the next hop indexed by FWD_PORT_INFO.

Note

The routing related forward actions (FWD_ACT= 110'b and 111'b) in VACL only take effect on the packets which are been routed.

FWD_SEL

When actions are conflict between Drop and Forward action, use FWD_SEL to determine the final action.

10.1.5.1.3 Log Action

For each ACL, there is one 36-bits packet-based counter, and one 42 bits byte-based counter. Only one log action would be taken in a PBG. Log actions in different PBGs can be taken concurrently.

Note

Log action would be executed no matter the packet is dropped.

10.1.5.1.4 Mirror Action

Mirror action in VACL is considered as Ingress Mirror, and the mirrored packet is the original packet content. Packet is mirrored by the configuration of mirroring set, which is indexed by MIRROR_INDEX.

10.1.5.1.5 Meter Action

Meter Action can mark the packet with color and take the color action by the configuration of meter, which is indexed by METER_INDEX.

Note

Entry of different PBG MUST NOT be pointer to the same meter block.

10.1.5.1.6 Ingress IVID Action

Ingress IVID action can change the IVID of the packet before L2 lookup. IVID of packets can be determined by a **new IVID** or **shifting from current IVID or OVID.**

10.1.5.1.7 Ingress IPRI Action

Ingress IPRI Action is for 1P priority remarking and always takes effect on tagged or untagged packets when enabled. This action only impacts the tag value of packet, but cannot impact the internal forwarding and Tx queuing.

Assign inner tag priority (IPRI_ACT = 00'b)

Remark inner priority of incoming packet with IPRI_DATA.

Copy from outer tag priority (IPRI_ACT = 01'b)

Remark inner priority of incoming packet by copying outer tag priority. Action would be ignored if there is no outer tag in original packet.

Keep inner tag priority (IPRI_ACT = 10'b)

Keep inner tag priority. Action would be ignored if packet is inner untagged.

10.1.5.1.8 Ingress OVID Action

The behavior is similar to Ingress IVID action.

10.1.5.1.9 Ingress OPRI Action

The behavior is similar to IPRI action.

10.1.5.1.10 Priority Action

Assign a 3-bit internal priority (INTERNAL_PRI), this priority is one of the sources for priority selection. The final internal priority is made by priority selection module

10.1.5.1.11 Bypass Action

Bypass action only takes effect on packets destined to CPU. The classified packet can be configured to bypass:

1) Ingress bandwidth control and storm control
2) VLAN ingress filter and accept frame type
3) STP ingress filtering.

10.1.5.1.12 Meta Data Action

Meta data is used to co-work with next phase ACL. The function supports filter packet characteristic more width and flexible with more than one phase ACL, and can also be applied to build hierarchy ACL mechanism.

10.1.5.1.13 QID Action

QID Action takes effect on packets destined to local CPU, and assigns 5-bit CPU QID for TX queuing. If packet is copied to CPU, the original one, which is not destined to CPU, would not be affect.

10.1.5.1.14 Remark Action

Remark DSCP (REMARK_ACT = 00'b)

DSCP of classified packets would be remarked by new DSCP.

Remark IP precedence (REMARK_ACT = 01'b)

IP precedence of classified packets would be remarked by new IP Precedence value.

Remark TOS (REMARK_ACT = 10'b)

TOS of classified packets would be remarked by new TOS value.

Note

Remarking DSCP/IP precedence/TOS would only take effect on classified IP packets.

10.1.5.1.15 IP_RSV Action

Invert reserved flag of IPv4 header.

10.1.5.2 IACL Action

Action mask register of IACL is shown as below：

- Drop
- Yellow drop
- Red drop
- Forward
- Log
- Mirror
- Meter
- IVLAN
- IPRI
- OVLAN
- OPRI
- Tag status
- Priority
- Bypass
- Meta data
- CPU QID
- Remark
- Yellow remark
- Red remark
- Invert reserve flag of IPv4 header

The format of action table entry is shown as below：

Table 10-18 IACL Action Table Entry

Fields	Description
DROP_ACT	The action is for packet which is mark Green. 0: Permit(Do Nothing) 1: Drop
YELLOW_DROP_ACT	The action is for packet which is mark Yellow. 0: Permit(Do Nothing) 1: Drop
RED_DROP_ACT	The action is for packet which is mark Red. 0: Permit(Do Nothing) 1: Drop
FWD_ACT	000: Permit(Do Nothing) 001: Drop 010: Copy frame to FWD_PORT_INFO 011: Copy frame to port mask pointed by FWD_PORT_INFO 100: Redirect frame to FWD_PORT_INFO 101: Redirect frame to port mask pointed by FWD_PORT_INFO 110: AND 111: Loopback
FWD_CPU_PKT_FMT	Only for FWD_ACT = 010 ~ 101 and packet to CPU 00: original packet 01: modified packet
FWD_SEL	If DROP_ACT and FWD_ACT is conflict. 0: select DROP_ACT 1: select FWD_ACT
FWD_PORT_INFO	FWD_ACT=010/100，FWD_ACT=010/100，[10]: Trunk bit, [9:6]: unit ID, [5:0]: Forwarding Port ID/Trunk ID FWD_ACT=011/101/110，[11:0]: index to multicast table for retrieving the port mask
MIRROT_ACT	Mirror original packet 00: RX Mirror packet 01: TX Mirror packet 10: Mirror cancel 11: Reserved
MIRROR_INDEX	For MIRROR_ACT=0, Mirror set index
METER_INDEX	Meter index
IVLAN_TPID_ACT	0: Don't assign new I-TPID 1: Assign new I-TPID
IVLAN_TPID_IDX	index to I-TPID list
IVLAN_VID_ACT	00: Assign new I-VID 01: Shift I-VID 10: Shift from O-VID 11: Don't assign new I-VID

IVLAN_DATA	IVID_ACT=0, [11:0]: New I-VID
	IVID_ACT=1 or 2, [11:0]: Value to shift

IPRI_ACT	Inner tag priority 00: Assign inner tag priority 01: Copy from outer tag priority 10: Keep inner tag priority 11: Reserved
IPRI_DATA	
OVLAN_TPID_ACT	0: Don't assign new O-TPID 1: Assign new O-TPID
OVLAN_TPID_IDX	Index to O-TPID list
OVLAN_VID_ACT	00: Assign new O-VID 01: Shift O-VID 10 : Shift from I-VID 11 : Don't assign new O-VID

OVLAN_VID_DATA	OVID_ACT=0, [11:0]: New O-VID
	OVID_ACT=1 or 2, [11:0]: Value to shift

OPRI_ACT	Outer tag priority 00: Assign outer tag priority 01: Copy from inner tag priority 10: Keep outer tag priority 11: Reserved
OPRI_DATA	
TAG_STATUS_INNER	00: Inner untag 01: Inner tag 10: Keep inner tag status 11: NOP(don't touch)
TAG_STATUS_OUTER	00: Outer untag 01: Outer tag 10: Keep outer tag status 11: NOP(don't touch)
PRI_DATA	
BYPASS_DATA	[0]: bypass ingress bandwidth control and storm control modules [1]: bypass egress VLAN filter
META_DATA	Co-operate with next phase ACL
CPU_QID	Only for packet-to-CPU packet
REMARK_ACT	The action is for packet which is mark Green. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value
YELLOW_REMARK_ACT	The action is for packet which is mark Yellow. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
YELLOW_REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value

RED_REMARK_ACT	The action is for packet which is mark Red. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
RED_REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value

10.1.5.2.1 Drop Action

The behavior is similar to VACL.

10.1.5.2.2 Forward Action

Similar forwarding actions in IACL and VACL behave the same. In addition, IACL forward action is the final forward decision module in ingress, which has the highest priority to impact the forwarding ports.

AND (FWD_ACT =110'b)

The original forwarding ports of classified traffic can be "AND" with the port mask indexed by FWD_PORT_INFO to determine the final forwarding ports.

Note

If the forward action of IACL is Permit (Do nothing), the forwarding decision of packet would not be changed in this phase.

10.1.5.2.3 Log Action

The behavior is similar to VACL.

10.1.5.2.4 Mirror Action

The behavior is similar to VACL.

Cancel mirror can abort flow-based RX/TX mirror and port-based mirror in VACL and IACL phase.

Note

If a packet hit both VACL and IACL Mirror action, and both rules point to the same mirror set, only IACL mirror action would be executed, i.e. only one copy of packet would be mirrored. Mirror cancel > IACL mirror > VACL mirror.

10.1.5.2.5 Meter Action

The behavior is similar to as VACL. IACL can further metering the traffic passed through VACL meter.

10.1.5.2.6 Inner TPID action

Assign new inner TPID by IVLAN_TPID_IDX which index to I-TPID list

10.1.5.2.7 Ingress IVID action

The VID operations behave are similar to VACL ingress IVID action. The difference is that IACL only impacts the VLAN ID of TX packets, but not the forwarding VID.

10.1.5.2.8 Ingress IPRI Action

The behavior is similar to VACL .

10.1.5.2.9 Outer TPID action

The behavior is similar to inner TPID action

10.1.5.2.10 Ingress OVID Action

The behavior is similar to Ingress IVID action.

10.1.5.2.11 Ingress OPRI Action

The behavior is similar to Ingress IPRI action.

10.1.5.2.12 Tag Status Action

This action is used to decide tag status of outgoing packet.

Force inner tag status (TAG_STATUS_INNER = 00'b/01'b)

Force inner tag status of outgoing packets to be untagged(00'b) or tagged (01'b) regardless of original inner tag format.

Keep inner tag status (TAG_STATUS_INNER = 10'b)

Keep the original inner tag format.

Force outer tag status (TAG_STATUS_OUTER = 00'b/01'b)

The behavior is similar to "Force inner tag status".

Keep outer tag status (TAG_STATUS_OUTER = 10'b)

The behavior is similar to "Keep inner tag status".

For C → S+C egressing application, UNI port should be forced as inner tagged and outer untagged, while the NNI port should be forced tagged in both inner and outer VLAN tag.

Note

Tag Status action has the highest priority to overwrite the tag status decision made by IVC configuration, VLAN table and egress port tag status configuration.

10.1.5.2.13 Priority Action

Assign a final 3-bit internal priority. The priority action in IACL has the highest priority to overwrite the internal priority generated by priority selection module. The final internal priority then would be mapped to output queue.

10.1.5.2.14 Bypass Action

Bypass action only takes effect on packets destined to CPU. Bypass Action in IACL can bypass

1) Ingress bandwidth and storm control rate limiting.

2) Egress VLAN filter

10.1.5.2.15 Meta Data Action

The behavior is similar to VACL, and would overwrite the action assigned in VACL.

10.1.5.2.16 CPU QID Action

The behavior is similar to VACL, and would overwrite the action assigned in VACL.

10.1.5.2.17 Remark Action

The behavior is similar to VACL, and would overwrite the action assigned in VACL.

10.1.5.2.18 IP_RSV Action

The behavior is similar to VACL, and would overwrite the action assigned in VACL.

10.1.5.3 EACL Action

Action mask register of EACL is shown as below：

- Drop
- Yellow drop
- Red drop
- Log
- Meter
- IVLAN
- IPRI
- OVLAN
- OPRI
- Remark
- Yellow remark
- Red remark

The format of action table entry is shown as below：

Table 10-19 EACL Action Table Entry

Fields	Description
DROP_ACT	The action is for packet which is mark Green. 0: Permit(Do Nothing) 1: Drop
YELLOW_DROP_ACT	The action is for packet which is mark Yellow. 0: Permit(Do Nothing) 1: Drop
RED_DROP_ACT	The action is for packet which is mark Red. 0: Permit(Do Nothing) 1: Drop
METER_INDEX	Meter index
IVLAN_TPID_ACT	0: Don't assign new I-TPID 1: Assign new I-TPID
IVLAN_TPID_IDX	index to I-TPID list

IVLAN_VID_ACT	00: Assign new I-VID 01: Shift I-VID 10: Shift from O-VID 11: Don't assign new I-VID

IVLAN_DATA	IVID_ACT=0, [11:0]: New I-VID
	IVID_ACT=1 or 2, [11:0]: Value to shift

IPRI_ACT	Inner tag priority 00: Assign inner tag priority 01: Copy from outer tag priority 10: Keep inner tag priority 11: Reserved
IPRI_DATA	
OVLAN_TPID_ACT	0: Don't assign new O-TPID 1: Assign new O-TPID
OVLAN_TPID_IDX	Index to O-TPID list
OVLAN_VID_ACT	00: Assign new O-VID 01: Shift O-VID 10 : Shift from I-VID 11 : Don't assign new O-VID

OVLAN_VID_DATA	OVID_ACT=0, [11:0]: New O-VID
	OVID_ACT=1 or 2, [11:0]: Value to shift

OPRI_ACT	Outer tag priority 00: Assign outer tag priority 01: Copy from inner tag priority 10: Keep outer tag priority 11: Reserved
OPRI_DATA	
REMARK_ACT	The action is for packet which is mark Green. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value
YELLOW_REMARK_ACT	The action is for packet which is mark Yellow. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
YELLOW_REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value
RED_REMARK_ACT	The action is for packet which is mark Red. 00: Remark DSCP 01: Remark IP precedence 10: Remark TOS
RED_REMARK_VALUE	[5:0]: DSCP value [2:0]: Precedence value [7:0]: TOS value

10.1.5.3.1 Drop Action

The behavior is similar to VACL.

10.1.5.3.2 Log Action

The behavior is similar to VACL.

10.1.5.3.3 Meter Action

The behavior is similar to IACL.

10.1.5.3.4 Ingress IVID Action

The behavior is similar to IACL.

10.1.5.3.5 Ingress IPRI Action

The behavior is similar to IACL.

10.1.5.3.6 Ingress OVID Action

The behavior is similar to Ingress IVID action.

10.1.5.3.7 Ingress OPRI Action

The behavior is similar to IPRI action.

10.1.5.3.8 Remark Action

The behavior is similar to IACL.

10.1.5.4 Action Priority between phases

- IACL redirect/copy to CPU > Security module drop > VACL redirect/copy to CPU > Port isolation.
- If packet is destined to multiple ports which include CPU port, then the packet destined to CPU has the same priority as redirect/copy to CPU.
- IACL/VACL redirect/copy to normal port is the lowest priority (regarded as known packet, higher than unknown packet drop).

10.1.6 ACL Clearance and Movement

To reduce CPU loading, entry clearance are supported by H/W to clear sequential rules includes field, operation, action and valid bit. The configuration of hardware clearance is as Table 10-20.

Table 10-20 PIE_CLR_CTRL Register

Bits	Field	Description
31:23	RESERVE	
22:12	CLR_TO	Clear to this index.
11:1	CLR_FROM	Clear from this index.
0	CLR	Clear the PIE entries, reset to 0 after clearing is complete

In addition, entry movement is also supported by H/W to adjust the position of sequential rules for priority adjustment.

Table 10-21 PIE_MV_CTRL Register

Bits	Field	Description
31:23	RESERVE	
22:12	MV_TO	Move to this index.
11:1	MV_FROM	Move from this index.
0	MV	Move the PIE entries, reset to 0 after moving is complete

Table 10-22 PIE_MV_LEN_CTRL Register

Bits	Field	Description
31:11	RESERVE	
10:0	MV_LEN	The number of entries to move.

Note

S/W must forbid the movement happened between blocks in different phases or mapped to different templates.

10.2 Field Selector

10.2.1 Overview

Some applications need inspect specific part of packet, which is not covered by pre-defined fields. For flexible inspection of packet, system supports 14 Field Selectors (FS), and each of FS can match 2-bytes of packet content.

Note

FS6 ~ 11 for IPv6 is SIPv6 address.

Each FS provides locator configuration to match the packet content as Table 10-23. System supports 7 types of start position as Table 10-24. Furthermore, each FS can be located from inner or outer payload of tunnel packet.

Table 10-23 PIE_FIELD_SELTOR_CTRL Register

Bits	Field	Description
10	PAYLOAD_SEL	Select inner or outer payload of tunnel packet

9:3	OFFSET	Offset index. Unit is 2-bytes.
2:0	FMT	Type of start position t

Note

Offset calculate is exclude VLAN tag.

Table 10-24 Type of start position

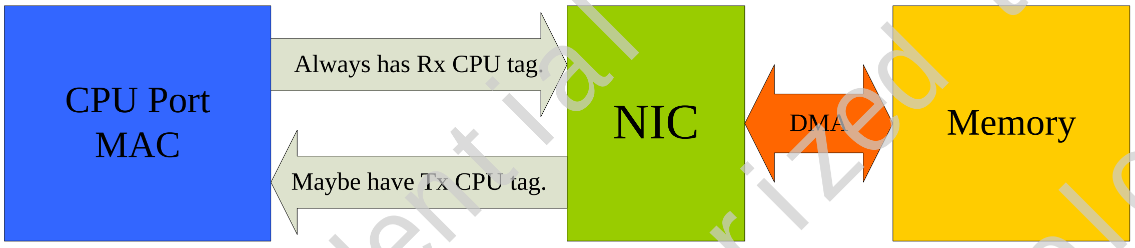
Type	Description
Raw	start from DA
LLC packet	start from length field
L3	start after ether type
ARP/RARP	Start from ARP/RARP header
IP	Start from IPv4/IPv6 header
IP payload	Start from IP payload(L4 header)
L4 payload	Start after L4 header (only support TCP and UDP)

11 CPU Interface

11.1 NIC

NIC connected MAC (MAC 56) with CPU is used for receiving/transmitting packets from/to CPU. NIC is a DMA engine which takes charge of moving packet between CPU (Memory) and Switch Core.

Figure 11-1 NIC and CPU MAC functional blocks



11.1.1 Receive

System provides 32 RX rings mapped to the 32 egress queues of the CPU MAC. When a packet is forwarded to CPU, the NIC performs a DMA Write to move the packet and related information from the CPU MAC to the CPU memory.

Before NIC moving packet to CPU, NIC driver allocates and structures CPU memory as following layout for receiving packets:

Figure 11-2 Structure of Ring-Descriptor-Packet buffer for receiving

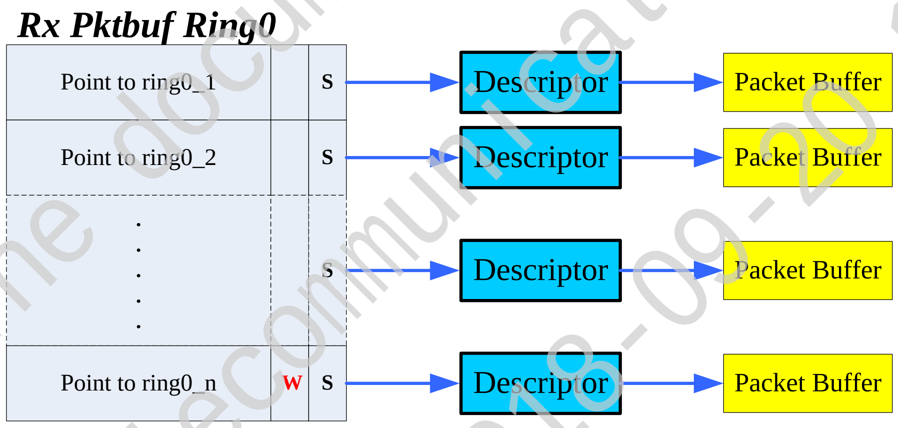


Figure 11-3 Descriptor Format

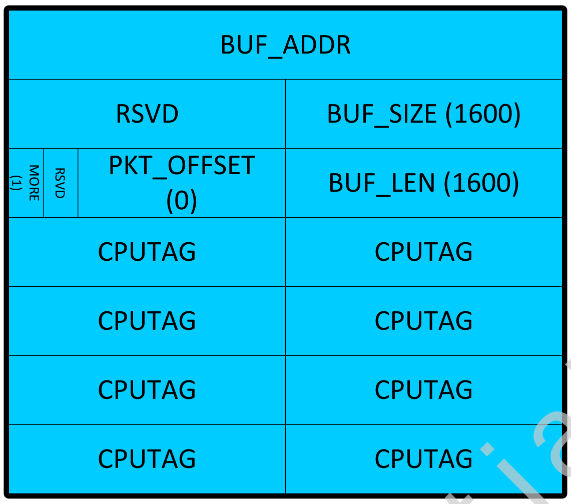


Table 11-1 Descriptor Fields

Name	Bits	Description
BUF_ADDR	32	The start address of the packet buffer. (4-Byte alignment)
RSVD	18	(Reserved)
BUF_SIZE	14	Size of the packet buffer. RX: Sets by NIC driver. Indicates the maximum length that ASIC can write. TX: ASIC don't care this field.
MORE	1	Indicates that the next descriptor describes the remaining part of the current packet.
RSVD	1	(Reserved)
PKT_OFFSET	14	The offset length based on the head of this packet.
RSVD	1	(Reserved)
BUF_LEN	14	Length of actual use of data buffer for RX/TX. RX: Sets by ASIC. Indicates the actual data length received. TX: Sets by Software/Driver.
CPUTAG	1	CPU Tag (Refer to section CPU Tag Spec)

Descriptor (as Table 11-1) stores information for NIC DMA process (e.g. address to packet buffer, buffer length, etc.). NIC DMA finds an available entry by checking the LSB of Ring entry and then writes both packet and CPU RX tag to packet buffer and descriptor respectively. After DMA write process, NIC notifies CPU to receive the packet with an interrupt signal (RX_DONE), which then be handled by NIC driver to perform RX callback function registered by user.

System also supports receiving jumbo frame. If big packet arrived, NIC will set MORE flag of the descriptor and write the remaining data to next packet buffer.

11.1.2 Transmit

For packet transmission, system uses similar ring structure as RX in Figure 11-4. It provides 2 TX rings to transmit packet through CPU MAC to front ports. The queue with higher ID has higher priority than that with lower ID. Unlike the ring structure for receiving packet, packet buffers are not pre-allocated. Whenever the upper layer protocol has a packet to send, NIC driver writes the address of packet to the BUF_ADDR in the TX descriptor and then inform NIC to move the packet by enabling TX_HIGH_FETCH/

TX_LOW_FETCH. Then NIC performs a DMA Read to move the packet from CPU memory to CPU MAC. As RX process, NIC make an interrupt signal to CPU when the moving process is done.

Figure 11-4 Structure of Ring-Descriptor-Packet buffer for transmission

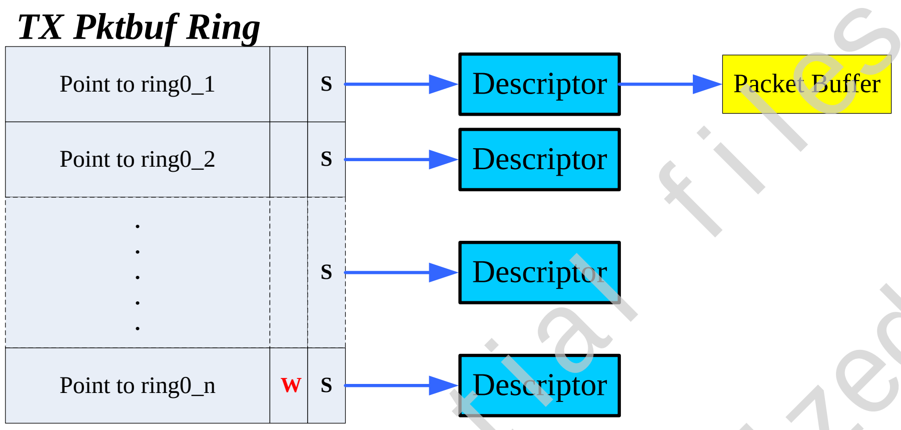


Table 11-2 DMA_IF_CTRL Register

Bits	Field	Description
31:30	RESERVED	
29:16	RX_TRUNCATE_LEN	The length should be truncated in the buffer.
15:7	RESERVED	
6	RX_TRUNCATE_EN	Enable RX truncate jumbo frame. 0b0: disable 0b1: enable
5	TX_EN	Enable TX DMA functionality. 0b0: disabled 0b1: enabled
4	RX_EN	Enable RX DMA functionality. 0b0: disabled 0b1: enabled
3	TX_HIGH_FETCH	TX High ring descriptor fetch notify. (Set this bit to trigger the packet send) Note: Auto clear to 0 by asic.
2	TX_LOW_FETCH	TX low ring descriptor fetch notify. (Set this bit to trigger the packet send) Note: Auto clear to 0 by asic.
1	TX_HIGH_BUSY	TX High Ring DMA status. (for debugging) 0b0: idle 0b1: busy
0	TX_LOW_BUSY	TX Low Ring DMA status. (for debugging) 0b0: idle 0b1: busy

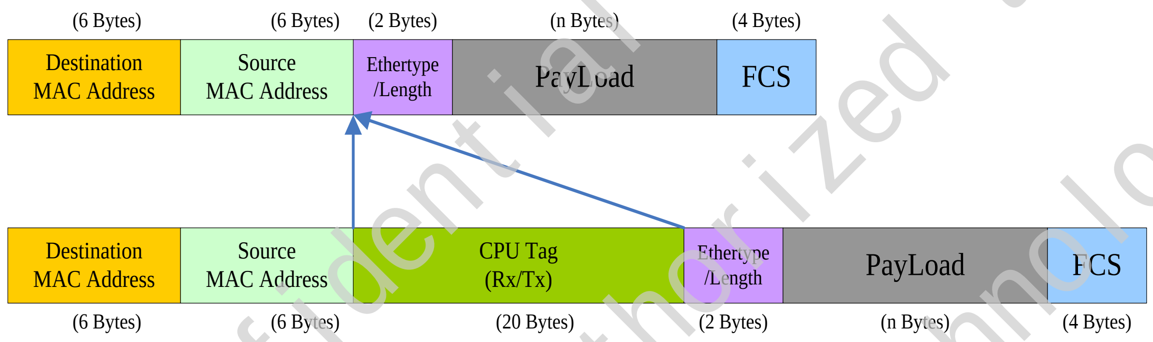
User application can control the forwarding process (e.g. destination ports or bypass some filters) of the sent packets by configuring the CPU TX Tag in Table 11-5.

11.2 CPU Traffic

Every packet destined to CPU can be configured to contain CPU tag. CPU Tag is a 20-Bytes payload to record switch processing information of a packet destined to CPU (CPU RX), and to instruct the switch processing behavior when CPU sending packet (CPU TX).

For embedded CPU, CPU Tag is contained in the *pktbuf* structure. For external CPU application, it needs to insert/extract manually between Source MAC Address and EtherType field.

Figure 11-5 CPU Tag position in an external CPU environment



11.2.1 RX CPU Tag

CPU RX Tag contains information regarding the received packet as Table 11-3.

Table 11-3 Fields of CPU RX Tag

Name	Bits	Description
PROTO	8	Defines the CPU tag. Value is 0x04.
PHYSICAL_RX_PORT	6	Local Physical RX Port.
OF_LU_MIS_TBL_ID	2	The table ID when OpenFlow table lookup miss.
ACL_OF_HIT	2	Indicates whether ACL or OpenFlow hit. 0b00: No hit 0b01: ACL 0b10: OpenFlow 0b11: Reserved
OF_TBL_ID	2	OpenFlow hit table ID
IS_TRK	1	Asserted if this packet is received from trunk port.
TRK_ID	7	If IS_TRK=1, this field indicates the trunk ID.
L2_ERR_PKT	1	The flag would be asserted if the packet is an L2 CRC error packet. Note: The L2 CRC error packet can be still trapped to CPU.
L3_ERR_PKT	1	The flag would be asserted if the packet is an L3 checksum error packet. Note: The L3 checksum error packet can be still trapped to CPU.

ATK_TYPE	5	Indicates which attack type of the received packet: 0b00000: None 0b00001: DA = SA 0b00010: LAND attack (DIP = SIP) 0b00011: UDP Blat attack (DPORT = SPORT) 0b00100: TCP Blat attack (DPORT = SPORT) 0b00101: POD (ping of death) attack 0b00110: IPv6 fragmented that size is less than the minimum. 0b00111: ICMP fragmented packet. 0b01000: ICMPv4 ping that size is over the maximum. 0b01001: ICMPv6 ping that size is over the maximum. 0b01010: Smurf (ICMP ping request that DIP is broadcast IP). 0b01011: TCP header is not complete. 0b01100: TCP SYN/!ACK packet with sport < 1024. 0b01101: TCP NULL scan. 0b01110: TCP XMAS attack (seq=0, FIN,URG,PSH are set) 0b01111: TCP SYN/FIN packet (SYN-FIN scan attack). 0b10000: TCP SYN/RST packet (SYN-RST scan attack). 0b10001: TCP Fragment Offset = 1 0b10010: ARP invalid 0b10011~0b11111: Reserved
QID	5	The queue id that packet is queued in CPU MAC.
SPN	10	Ingress physical port number. SPN[9:6] : source device ID SPN[5:0] : source port ID
ORI_ETAGIF	1	Asserted if the packet contains the ECID tag when received on an ingress port.
IDX	15	For ACL or OpenFlow, IDX indicates the entry index. For S-Flow TX (SFLOW=0b10), IDX[9:6]: indicates TX device, IDX[5:0] : indicates TX port.
MIR_HIT	4	Mirrioring hit indications for 4-sets of configuration. Bit 0: Mirror set 0 hit Bit 1: Mirror set 1 hit Bit 2: Mirror set 2 hit Bit 3: Mirror set 3 hit
SFLOW	2	0b00: Not hit any S-Flow sampling 0b01: S-Flow RX hit 0b10: S-Flow TX hit 0b11: Reserved
TT_HIT	1	Asserted if tunnel terminate entry hit
TT_IDX	9	Tunnel terminate entry index
ETAGIF	1	Asserted if the packet contains the ECID tag.
OTAGIF	1	Asserted if the packet contains the outer tag.
ITAGIF	1	Asserted if the packet contains the inner tag.
FWD_VID_SEL	1	Indicates Forwarding VLAN ID is extracted from: 0: Inner VID 1: Outer VID
FWD_VID	12	Forwarding VLAN ID

MAC_CST	1	Asserted if this packet meets the MAC constraint condition.
DM_RXIDX or OF_LB_TIMES	6	ETH-DM RX Timestamp Index (0~63) Note: The index number comes from the Y.1732 module in ALE. Packet loopback times. Noted: The loopback may result from Openflow entry hit action or Openflow lookup miss.
NEW_SA	1	Asserted if packet is new SA.
PMV_FORBID	1	Asserted if the packet meets port move forbidden
L2_STTC_PMV	1	Asserted if the packet meets L2 Static entry port moving
L2_DYN_PMV	1	Asserted if the packet meets L2 Dynamic entry port moving
OVERSIZE	1	Asserted if the packet is truncated by NIC.
PORT_DATA_IS_TRK	1	Asserted if the PORT_DATA is trunk port.
PORT_DATA	10	This field indicates the original port of learnt SA when port move happened. The updated port would be filled in SPN field. If PORT_DATA_IS_TRK=0, PORT_DATA [9:6]: device ID PORT_DATA [5:0]: port ID, If PORT_DATA_IS_TRK=1, PORT_DATA [6:0]: trunk ID.
HASH_FULL	1	Asserted if the packet meets L2 table hash bucket full.
INVALID_SA	1	Asserted if the packet meets Invalid SA(MC/BC/All Zero)
REASON	6	Reason to CPU: 0. Invalid (reason is recorded in above event flags) 1. CPU to CPU talk (for one hop trap/ unicast to CPU/ broadcast to CPU) 2. OF IP/MPLS TTL exception 3. OF table lookup miss 4 ~ 6. Reserved 7. OAMPDU (OAMPDU, OAM Parser) 8. CFM (CCM Packet, CFM unknown type(opcode), LBM/LBR, LTM/LTR) 9. CFM (ETHDM, It means the field DM_RXIDX is valid) 10. Parser exception (inner/outer frame) 11. Malformed packet 12. IP MAC binding (Match/Mismatch) 13. L3 IPUC RPF (tunnel/route) 14. inner/outer CFI=1 15. IVC (MAC-based/IP-subnet-based) 16. Ingress VLAN filter 17. L2 UC/MC bridge lookup miss 18. IP4/IP6 MC bridge lookup miss 19. PTP 20. RMA (User-defined 0) 21. Reserved 22. RX EST header mismatch 23. ECID equal to RX Port PCID 24. RMA (BPDU 01-80-C2-00-00-00) 25. RMA (LACP 01-80-C2-00-00-02)

		26. RMA (LLDP packet, 0x88CC) 27. RMA (EAPOL packet, 0x888E) 28. RMA (01-80-C2-00-00-xx) 29. Tunnel Interface MAC 30. Tunnel IP address check 31. Router Interface MAC 32. L3 IPUC non-IP packet 33. IP4/IP6 Header Error 34. Routing IP address check 35. L3 routing DIP/DMAC mismatch 36. IP6 UC/MC hop by hop option 37. IP6 UC/MC routing header 38. IP4 IP option (UC/MC) 39. IP4/IP6 MC routing lookup miss 40. L3 IPUC null route (Null interface bit set) (for sending ICMP Unreachable) 41. L3 IPUC PBR null route 42. L3 UC host route (host route bit == 1) (My IP) 43. L3 UC net route (host route bit ==0) (for sending ARP) 44. L3 MC bridge entry 45. L3 MC route entry 46. L3 IPMC RPF (tunnel/route) 47. Routing exception (NOT a Next-Hop Entry) (tunnel/route) 48. Routing exception (Next-Hop Entry Age-out) 49. Routing exception (NH is a tunnel interface while Post Routing) 50. IPUC TTL (tunnel/route) 51. IPMC TTL (tunnel/route) 52. IPUC MTU (tunnel/route) 53. IPMC MTU (tunnel/route) 54. IP4/IP6 ICMP redirect 55. IGMP/MLD 56. DHCP/DHCP6 57. ARP Request/Reply/Gratuitous ARP 58. Neighbor Discovery 59. IP4/IP6 reserved address 60. MPLS exception 61. Reserved 62. L2 Notification 63. Normal forward

Both event flags and REASON of CPU RX tag explains why packet is sent to CPU.

Event Flag

Event flags show the important causes that shall not be ignored.

For the event flags: MAC_CST, NEW_SA, PMV_FORBID, L2_STTC_PMV, L2_DYN_PMV, HASH_FULL, INVALID_SA, ATK_TYPE, OF_HIT, TT_HIT, MIR_HIT and SFLOW, they will be set if the packet cause the corresponding events no matter what action is configured. But for ACL_ HIT (ACL_OF_HIT = 1) and IDX, they will be set only when packet hits ACL rule and the corresponding action is configured as TRAP or COPY.

When TT_HIT is set, it means that packet is a tunneled packet and is terminated at this switch. In such condition, the outer part of packet is stripped before sending to CPU.

REASON

REASON aggregate several trapping/copying reasons in predefined precedence. For example, if a packet is meeting RMA trap and VLAN ingress filter trap at the same time, the REASON field will be 16 due to higher precedence. If packet sent to CPU is merely cause by event (e.g. ACL, NEW_SA…), then only the corresponding event flag will be set and REASON remains 0.

OTAGIF/ITAGIF

Both fields indicate the tag status of packet presented to CPU, but not imply that packet is originally tagged or not tagged.

To keep the original VLAN tag status, user can configure DMA_IF_PKT_CTRL.PKT_VLAN_FMT to 0.

Likewise, the ECID tag status of packet presented to CPU could be decided through TRAP_ETAG_PKT_FMT/FWD_ETAG_PKT_FMT. Other parts of packet modifications could be preserved by configuring TRAP_PKT_FMT/FWD_PKT_FMT to 1.

Table 11-4 DMA_IF_PKT_CTRL Register

Bits	Field	Description
31:7	RESERVED	
6	TRAP_ETAG_PKT_FMT	Specify the ECID tag format trapped/copied to CPU. 0b0: original packet 0b1: modified packet
5	FWD_ETAG_PKT_FMT	Specify the ECID tag format forwarded to CPU. 0b0: original packet 0b1: modified packet
4	TRAP_PKT_TAG	When EXT_CPU_EN=0b1, this bit indicates whether insert the CPU tag into the packets which is trapped to CPU port. 0b0: without CPU tag 0b1: with CPU tag
3	FWD_PKT_TAG	When EXT_CPU_EN=0b1, this bit indicates whether insert the CPU tag into the packets which is forwarded to CPU port. 0b0: without CPU tag 0b1: with CPU tag
2	TRAP_PKT_FMT	Specify the packet format trapped/copied to CPU. 0b0: original packet 0b1: modified packet
1	FWD_PKT_FMT	Specify the packet format forwarded to CPU. 0b0: original packet 0b1: modified packet
0	PKT_VLAN_FMT	Specify the packet's VLAN tag to CPU. 0b0: original packet 0b1: modified packet

11.2.2 TX CPU Tag

The CPU TX Tag provides control information to instruct switch how to process the packet and it is shown as below:

Table 11-5 Fields of CPU TX Tag

Name	Bits	Description
PROTO	8	Defines the CPU tag. Value is 0x04.

FWD_TYPE	4	0: Follow normal forwarding precedure. 1: Uses SW_UNIT and DPM as the destination port mask. (Physical). 2: Uses SW_UNIT and DPM as the destination port mask (Logical). 3: Uses DPM as trunk ID. 4: Sends to One hop of CPU (use DPM as physical TX ports). 5: Sends to One hop of CPU (use DPM as logical TX ports). 6: Unicast to CPU (ignore trunk load balancing). 7: Unicast to CPU (with trunk load balancing). 8: Broadcast to CPU. (all CPUs except local CPU in stacking system would receive)
ACL_ACT	1	Indicates whether take the result of ingress/egress ACL: 0b0: Don't take the ACL effect. 0b1: Takes the ACL effect.
CNGST_DROP	1	Dropable if congestion occurs 0b0: Don't drop it with congestion. 0b1: Can be dropped when the destination port is congestion.
DM_PKT	1	ETH-DM packet indicator: 0b0: This is NOT an ETH-DM packet. 0b1: Notifies TX Modifier to fill the TX Timestamp field based-on OpCode. Note: Only consider that no more than double tag.
DG_PKT	1	Dying Gasp packet 0b0: It's not a dying gasp packet. 0b1: It's a dying gasp packet.
BP_FLTR	1	If BP_FLTR is asserted, below filtering will be bypassed: (1) Egress OAM mux (2) Egress Port Isolation (3) Egress Mirror Isolation
BP_STP	1	Bypass egress STP 0b0: Filter by egress STP port state. 0b1: Bypass egress STP.
BP_VLAN_EGR	1	Bypass Egress VLAN filtering 0b0: Egress VLAN Filtering mode. 0b1: Bypass Egress VLAN filtering.
ALE_AS_TAGSTS	1	Indicates if follows Outer/Inner tagging procedure by ALE. 0b0: DON'T touch (CPU assigns tagging status). 0b1: Follow ALE decision.
L3_ACT	1	Indicates if takes the result of L3 routing and multicast replication: 0b0: Don't apply the L3 effect. 0b1: Applies the L3 effect.
ORI_TAGIF_EN	1	Indicates tag status 0: No reference. 1: Refers to ORI_ITAGIF and ORI_OTAGIF.
AS_QID	1	Indicates the output queue decision 0b0: The output queue is decided by lookup process. 0b1: CPU assigns the output queue directly as QID.
QID	5	Assigned QID
ORI_ITAGIF	1	Indicates whether this packet contains the inner tag.
ORI_OTAGIF	1	Indicates whether this packet contains the outer tag.

FWD_VID_SEL	1	Forwarding VID selection: 0b0: Follows the inner tag. 0b1: Follows the outer tag.
FWD_VID_EN	1	0b0: Follows ALE decision. 0b1: Force using FWD_VID.
FWD_VID	12	Forwarding VID
SRC_FILTER_EN	1	If SRC_FILTER_EN is 0b1, SP should be filtered from outgoing portmask.
SP_IS_TRK	1	0b0: SPN is not trunk ID. 0b1: SPN is trunk ID.
SPN	10	The port ID to be filtered from outgoing portmask. If SP_IS_TRK=0, SPN[9:6] indicates source device ID and SPN[5:0] indicates the port ID to be filtered. If SPN_IS_TRK=1, SPN[6:0] indicates the trunk ID to be filtered.
SW_UNIT	4	If FWD_TYPE is 1 or 2 or 6 or 7 or 9, this field indicates the destination device ID.
DPM	56	DPM is used as portmask when FWD_TYPE is 1, 2, 4, 5. DPM is used as trunk ID when FWD_TYPE is 3.

FWD_TYPE = 1 and 2 are forwarding packet in physical and logical ways. The difference between them is that physical way means packet is transmitted directly according to the SW_UNIT and DPM while the logical should undergo the trunk load-balancing decision if the outgoing port is trunk port.

Although user can specify the outgoing ports by setting FWD_TYPE to 1 or 2, they still filter by the following modules:

- Egress OAM
- Egress Port Isolation
- Egress Mirror Isolation
- Egress spanning-tree port state
- Egress VLAN filtering
- L2 CRC check (if enable check)

If user wants to accurately specify the outgoing ports, he/she should additionally enables BP_FLTR, BP_STP and/or BP_VLAN_EGR.

ORI_ITAGIF, ORI_OTAGIF and ORI_TAGIF_EN are used to specify whether packet contains inner/outer tag. Without enabling them, ASIC will parse packet's tag status according to the VLAN-related configuration of CPU port.

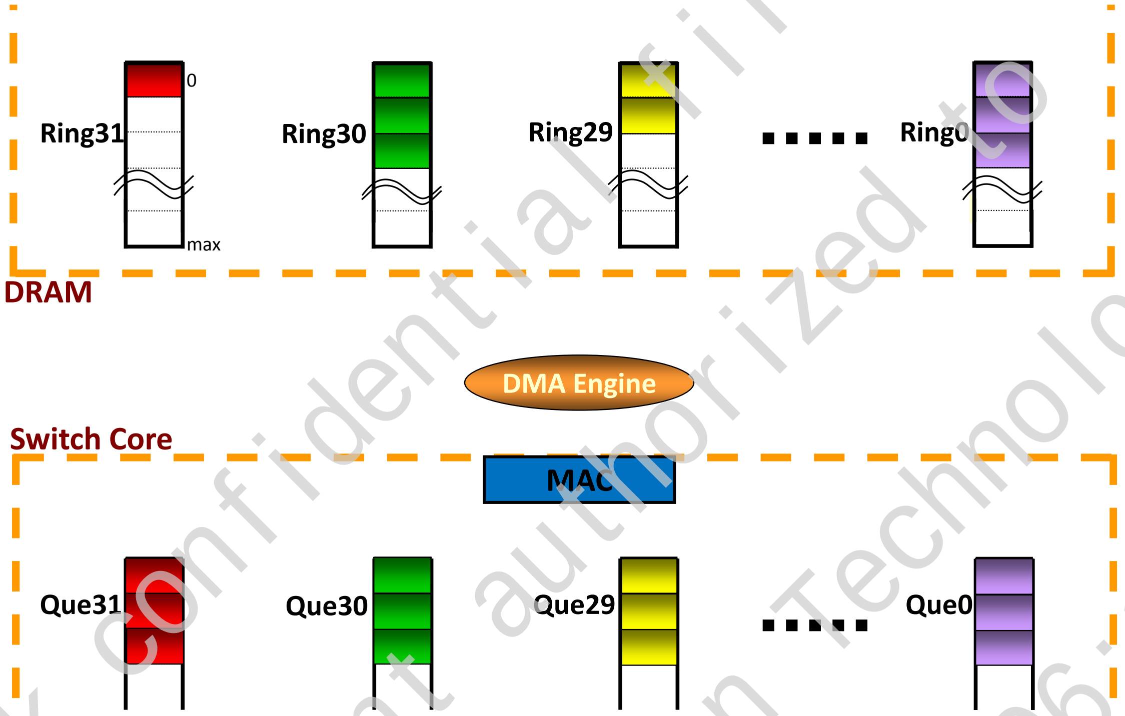
FWD_VID, FWD_VID_SEL and FWD_VID_EN are used to decide the ASIC's forwarding VID. For more information, please refer to VLAN Develop Guide.

SRC_FILTER_EN, SP_IS_TRK and SPN are used to filter SPN from outgoing ports. For example, if CPU send a packet with FWD_TYPE = 0 (ASIC normal lookup and forward), but also would like to filter Port 3, it can be achieved by setting SRC_FILTER_EN = 1, SP_IS_TRK = 0 and SPN = 3.

11.2.3 Queuing to CPU

System provides 32 rings and queues respectively. It is one-to-one relationship between the outgoing queues of CPU MAC and the RX rings in CPU memory. For packet in higher queue, it will be serviced in higher order in both switch and NIC driver.

Figure 11-6 One-to-one mapping between RX rings in CPU memory and switch queues



For each event and reason type, system provides configuration to decide the output queue of packet destined to CPU MAC (MAC 56). For example, if user would like to put the packet that meet MPLS TTL exception into Queue 2, it can be achieved by configuring QM_RSN2CPUQID_CTRL_0 to 0x 200000. For complete reason-to-queue registers, please refer to "RTL9310 Register Profile".

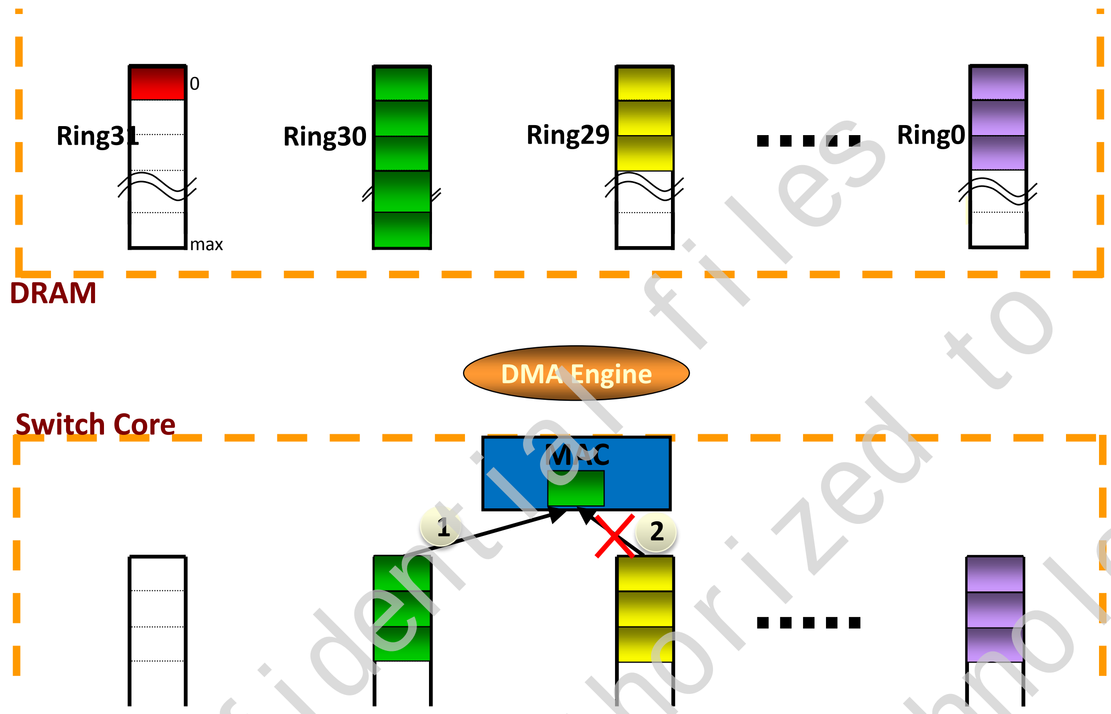
Table 11-6 QM_RSN2CPUQID_CTRL_0 Register

Bits	Field	Description
31:30	RESERVED	
29:25	CPU2CPU_TALK	CPU queue ID (CPU to CPU talk)
24:20	TTL_EXCPT	CPU queue ID (OF IP/MPLS TTL exception)
19:15	OF_TBL_LU_MISS	CPU queue ID (OF table lookup miss)
14:0	RESERVED	

11.2.4 Head of Line Blocking (HOL) Prevention

After packet scheduled from queue and before DMA to CPU memory, packet may be held in switch temporarily due to lack of RX ring space of corresponding queue in CPU memory. In this case, this packet may block other packets in other queues from DMA to corresponding RX rings even if they are available (See Figure 11-7).

Figure 11-7 Head-of-line blocking



To prevent from HOL, system provides *DMA_IF_RX_RING_SIZE* and *DMA_IF_RX_RING_CNTR*.

Table 11-7 DMA_IF_RX_RING_SIZE Register

Bits	Field	Description
9:0	SIZE	The Rx pktbuf descriptor ring size for each queue. 0b00000000: Disable HOL 0b00000001 – 0b11111111: The number of pktbuf descriptor.

Table 11-8 DMA_IF_RX_RING_CNTR Register

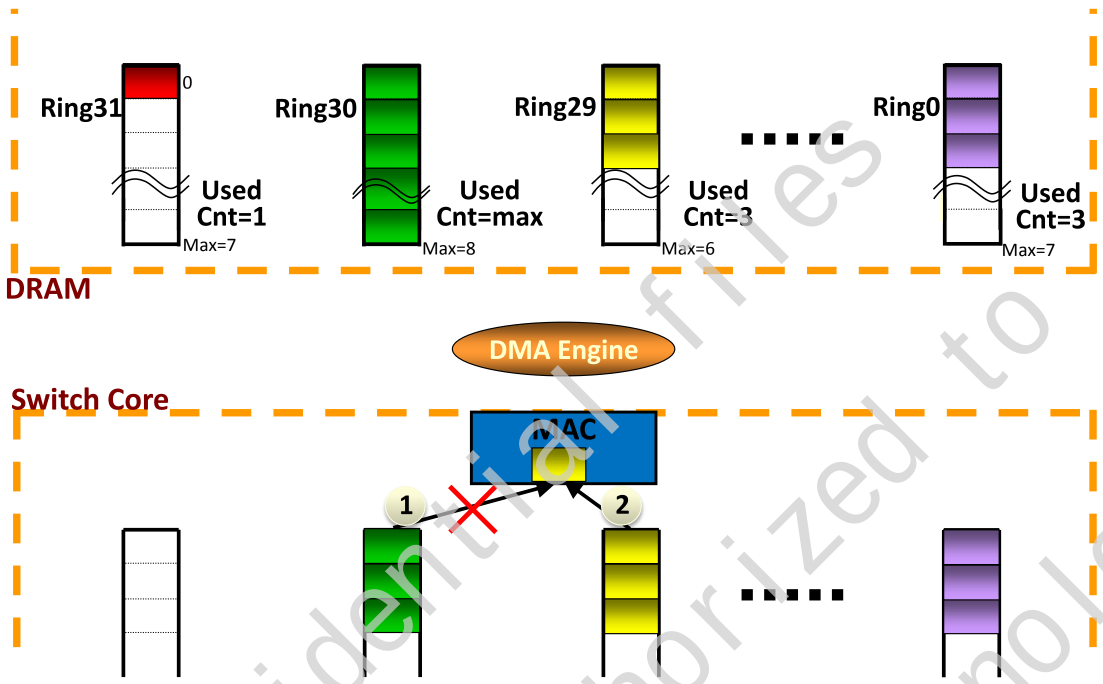
Bits	Field	Description
9:0	CNTR	The Rx pktbuf descriptor ring used count for each queue. For read: get the number of used descriptors. For write: to subtract the value to the counter.

At system initialization, NIC driver will set *DMA_IF_RX_RING_SIZE.SIZE* to the corresponding RX ring size that it allocated in CPU memory. Afterword, switch will increase *DMA_IF_RX_RING_CNTR.CNTR* by one whenever it DMA a packet successfully and NIC driver will decrease *DMA_IF_RX_RING_CNTR.CNTR* by one whenever it has released a space on that RX ring.

If CPU is busy so that *DMA_IF_RX_RING_SIZE.SIZE* equals to *DMA_IF_RX_RING_CNTR.CNTR*, the switch stops taking away packet from this queue. By this way, system guarantees no packet is held after dequeuer and thus no other packets may be blocked.

In Figure 11-8, scheduler gives up taking packet from Queue 30 and change to Queue 29 because *DMA_IF_RX_RING_CNTR.CNTR* of Queue 30 equals to its *DMA_IF_RX_RING_SIZE.SIZE*.

Figure 11-8 Head-of-line blocking Prevention



12 Security

12.1 Attack Prevention

12.1.1 Overview

System supports hardware attack checker to filter the malicious packets. Attacker attempts to make a machine or network resource unavailable to its intended users. The checker can detect these packets and filter them to protect the receiver, or trap these attack packets to CPU for analysis.

12.1.2 Attack Detection

The attack detection types are defined system-wide and each type can define its action as drop or trap. Per port enables attack prevention.

Packet goes over the functional blocks in below sequence.

Figure 12-1 Attack Prevention Packet Flow

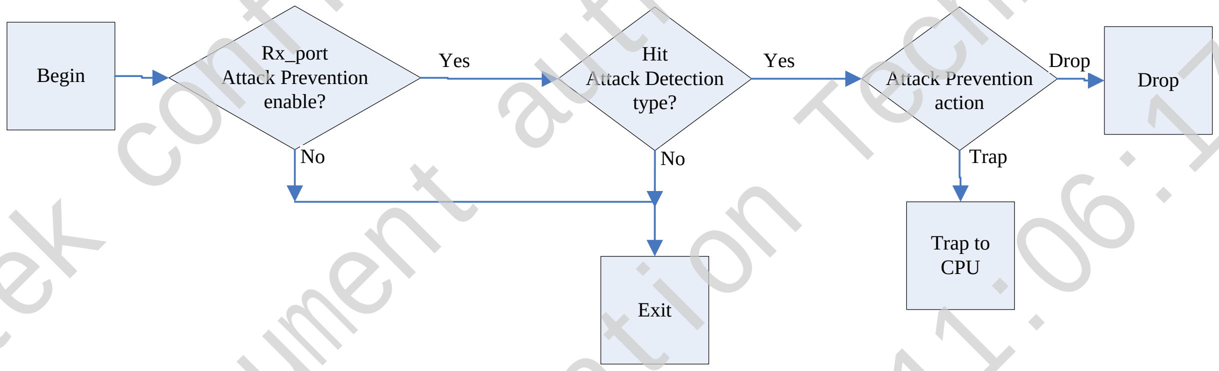


Table 12-1 Attack Detection Type

Type	Description	Action
DA_EQUAL_SA	Filter packets with MACDA = MACSA.	Drop/Trap
LAND	Filter IPv4/IPv6 packets whose SIP = DIP.	Drop/Trap
UDP_BLAT	Filter IPv4/IPv6 UDP packets whose SPORT = DPORT.	Drop/Trap
TCP_BLAT	Filter IPv4/IPv6 TCP packets whose SPORT = DPORT.	Drop/Trap
POD	Filter IPv4 packets that length is larger than 64K bytes through fragments. (Ping of Death)	Drop/Trap
IPV6_FRAG_LEN_MIN	Check for minimum size of IPV6 fragments. (If packet length is larger than ATK_PRVNT_IPV6_CTRL.FRAG_LEN_MIN)	Drop/Trap
ICMP_FRAG_PKT	Filter fragmented ICMP packets (IPv4 & IPv6).	Drop/Trap
ICMPV4_PING_MAX	Filter ICMPv4 ping packets with payload size greater than the programmable value in ATK_PRVNT_ICMP_CTRL.PKT_LEN_MAX.	Drop/Trap
ICMPV6_PING_MAX	Filter ICMPv6 ping packets with payload size greater than the programmable value in ATK_PRVNT_ICMP_CTRL.PKT_LEN_MAX.	Drop/Trap

SMURF	Filter ICMP packets of ping request to a broadcast DIP (x.x.x.255) and the SIP is spoofed (victim).	Drop/Trap
TCP_HDR_LEN_MIN	Check first TCP fragmented packet that don't have the full TCP header. (i.e., data offset field in TCP header should have the value larger than or equal to the configured value in ATK_PRVNT_TCP_CTRL.HDR_LEN_MIN)	Drop/Trap
SYN_SPORT_LESS_1024	Filter TCP packets with SYN bit set and ACK bit not set and sport less than 1024.	Drop/Trap
NULL_SCAN	Filter TCP packets with sequence number is zero and the control bits are zeroes.	Drop/Trap
XMAS	Filter TCP packets with sequence number is zero and the FIN,URG, and PSH bits are set (XMAS Scan attack).	Drop/Trap
SYN_FIN	Filter TCP packets with SYN and FIN set (SYNFIN Scan attack).	Drop/Trap
SYN_RST	Filter TCP packets with SYN and RST set (SYNRST Scan attack).	Drop/Trap
TCP_FRAG_OFF_MIN	Filter TCP non-initial fragments carrying TCP header.	Drop/Trap
INVALID_LEN	Filter header length in ipv4 header is greater than total length in ipv4 header.	Drop/Trap

Note

Any packet dropped/trapped by Attack Prevention is not learned into L2 table.

Table 12-2 ATK_PRVNT_PORT_EN Register

Bits	Field	Description
0	EN	Enable per-port attack prevention

Table 12-3 ATK_PRVNT_CTRL Register

Bits	Field	Description
18	CPU_SEL	Select which CPU attack packets are trapped to. 1b0: local CPU 1b1: master CPU
17	INVALID_LEN	Filter header length in ipv4 header is greater than total length in ipv4 header.
16	TCP_FRAG_OFF_MIN	Filter TCP non-initial fragments carrying TCP header.
15	SYN_RST	Filter TCP packets with SYN and RST set (SYNRST Scan attack)
14	SYN_FIN	Filter TCP packets with SYN and FIN set (SYNFIN Scan attack).

13	XMAS	Filter TCP packets with sequence number is zero and the FIN,URG, and PSH bits are set (XMAS Scan attack).
12	NULL_SCAN	Filter TCP packets with sequence number is zero and the control-bits are zeroes.
11	SYN_SPORT_LESS_1024	Filter TCP packets with SYN bit set and ACK bit not set and sport less than 1024.
10	TCP_HDR_LEN_MIN	Check first TCP fragmented packet that don't have the full TCP header. (i.e., data offset field in TCP header should have the value larger than or equal to the configured value in ATK_PRVNT_TCP_CTRL.HDR_LEN_MIN)
9	SMURF	Filter ICMP packets of ping request to a broadcast DIP (x.x.x.255) and the SIP is spoofed (victim).
8	ICMPV6_PING_MAX	Filter ICMPv6 ping packets with payload size greater than the programmable value in ATK_PRVNT_ICMP_CTRL.PKT_LEN_MAX.
7	ICMPV4_PING_MAX	Filter ICMPv4 ping packets with payload size greater than the programmable value in ATK_PRVNT_ICMP_CTRL.PKT_LEN_MAX.
6	ICMP_FRAG_PKT	Filter fragmented ICMP packets (IPv4 & IPv6).
5	IPV6_FRAG_LEN_MIN	Check for minimum size of IPV6 fragments. (If packet length is larger than ATK_PRVNT_IPV6_CTRL.FRAG_LEN_MIN)
4	POD	Filter IPv4 packets that length is larger than 64K bytes through fragments. (Ping of Death)
3	TCP_BLAT	Filter IPv4/IPv6 TCP packets whose SPORT = DPORT.
2	UDP_BLAT	Filter IPv4/IPv6 UDP packets whose SPORT = DPORT.
1	LAND	Filter IPv4/IPv6 packets whose SIP = DIP
0	DA_EQUAL_SA	Filter packets with MACDA = MACSA.

Table 12-4 **ATK_PRVNT_ACT Register**

Bits	Field	Description
17	INVALID_LEN	Determine the action for the INVALID_LEN packets. 0b0: drop 0b1: trap
16	TCP_FRAG_OFF_MIN	Determine the action for the TCP_FRAG_OFF_MIN packets. 0b0: drop 0b1: trap
15	SYN_RST	Determine the action for the SYNRST packets. 0b0: drop 0b1: trap

14	SYN_FIN	Determine the action for the SYNFIN packets. 0b0: drop 0b1: trap
13	XMAS	Determine the action for the XMAS packets. 0b0: drop 0b1: trap
12	NULL_SCAN	Determine the action for the NULLSCAN packets. 0b0: drop 0b1: trap
11	SYN_SPORT_LESS_1024	Determine the action for the SYN_SPORT_LESS_1024 packets. 0b0: drop 0b1: trap
10	TCP_HDR_LEN_MIN	Determine the action for the TCP header too small packets. 0b0: drop 0b1: trap
9	SMURF	Determine the action for the SMURF attack packets. 0b0: drop 0b1: trap
8	ICMPV6_PING_MAX	Determine the action for the ICMPv6 ping attack packets. 0b0: drop 0b1: trap
7	ICMPV4_PING_MAX	Determine the action for the ICMPv4 ping attack packets. 0b0: drop 0b1: trap
6	ICMP_FRAG_PKT	Determine the action for the ICMP fragmented packets. 0b0: drop 0b1: trap
5	IPV6_FRAG_LEN_MIN	Determine the action for the IPv6 minimum fragmented packets. 0b0: drop 0b1: trap
4	POD	Determine the action for the POD attack packets. 0b0: drop 0b1: trap
3	TCP_BLAT	Determine the action for the TCP blat attack packets. 0b0: drop 0b1: trap
2	UDP_BLAT	Determine the action for the UDP blat attack packets. 0b0: drop 0b1: trap
1	LAND	Determine the action for the LAND attack packets. 0b0: drop 0b1: trap
0	DA_EQUAL_SA	Determine the action for the DA=SA attack packets. 0b0: drop 0b1: trap

Table 12-5 ATK_PRVENT_IPV6_CTRL Register

Bits	Field	Description
15:0	FRAG_LEN_MIN	Value to be used when checking for minimum size of IPV6 fragments. (Checking is enabled by setting APCR.IPV6_MIN_FRAG_LEN)

Table 12-6 ATK_PRVENT_ICMP_CTRL Register

Bits	Field	Description
15:0	PKT_LEN_MAX	Maximum length of ICMP packet allowed before dropping.

Table 12-7 ATK_PRVENT_TCP_CTRL Register

Bits	Field	Description
4:0	HDR_LEN_MIN	Minimum TCP header length allowed.

Table 12-8 ATK_PRVENT_SMURF_CTRL Register

Bits	Field	Description
31:0	NETMASK	Bitmask of Network Mask to check the Smurf.

12.1.3 Validation Check

The device supports to check the validation of ARP packet.

Invalid IPv4 ARP packet, that is matched the one of the following conditions:

- Request packet
  - MAC SA != ARP Sender MAC address.
  - ARP Sender IP = all-zero or multicast or broadcast IP address.
- Reply packet
  - MAC SA != ARP Sender MAC address.
  - ARP Target MAC address = all-zero or multicast address.
  - ARP Target MAC address != MAC DA.
  - ARP Sender IP = all-zero or multicast or broadcast IP address.
  - ARP Target IP = all-zero or multicast or broadcast IP address.

Per port configures the action for invalid ARP packets.

Table 12-9 ATK_PRVENT_ARP_INVLD_PORT_ACT Register

Bits	Field	Description
1:0	ACT	Action for ARP invalid packets. 0b00: Forward 0b01: Drop 0b10: Trap 0b11: Reserved

12.2 IP MAC Binding

For security issue, host IP address may be restricted the access permission via validation. IP source guard checks the source IP address of all IP packets except for DHCP. To prevent ARP spoofing, the sender IP address of ARP packet could be validated in the same way. The validation is checked by the IP MAC binding table which specifies the legal IP addresses with corresponding MAC address in each. Moreover, administrator may configure the VID and ingress port for each binging entry optionally.

12.2.1 Binding Table

System provides a dedicated table with 1024 entries, and an additional table with 2048 entries. The mode selection of this additional table should be configured properly before using as IP MAC binding. The binding factors of each entry could be defined flexibly as followings:

- IP + MAC
- IP + MAC + PORT
- IP + MAC + PORT + VLAN

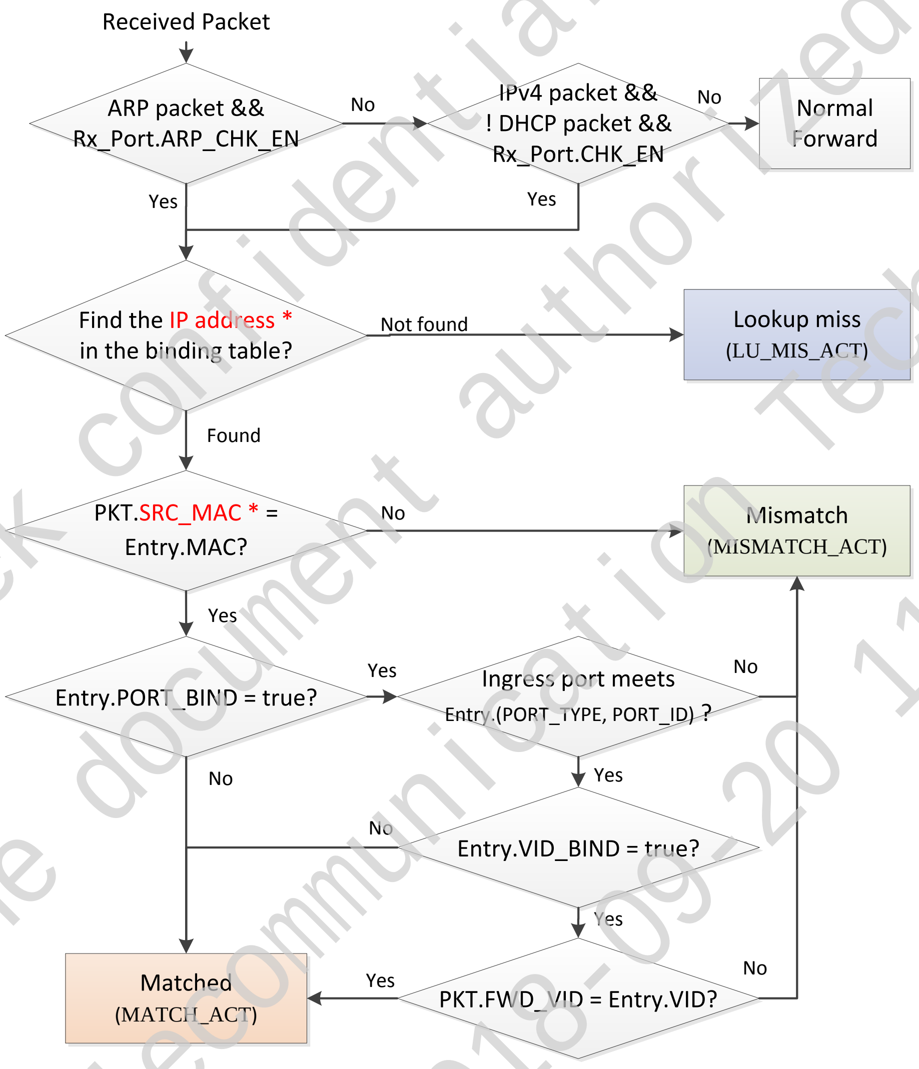
Table 12-10 IP MAC Binding Table Entry

Name	Description
VALID	Valid bit for the entry
SIP [23:0]	Source IPv4 address (Record the last 24 bits of SIP, most 8 bits can be reversed with index) Note: For ARP packet, SIP means Sender IP Address field.
SMAC [48:0]	Source MAC address Note: For ARP packet, SMAC means Sender Hardware Address field.
PORT_BIND [0]	To indicate this packet should be checked the ingress port. 0b0: Don't need to check ingress port 0b1: Need to check ingress port
PORT_TYPE [0]	Type of the port 0b0: Individual port 0b1: Trunk port
PORT_ID [6:0]	For individual port (PORT_TYPE = 0), to indicate the port ID (0~51). For trunk port (PORT_TYPE = 1), to indicate the trunk ID (0~127).
VID_BIND [0]	Bind Status of VLAN 0b0: Don't need to check VLAN 0b1: Need to check VLAN
VID [11:0]	VLAN ID

12.2.2 Binding Validation Process

Figure 4-16 shows the flowchart of IP MAC binding validation. Per port configures whether activate IP MAC binding validation for ARP and IPv4 packet respectively. The lookup miss packet means that the source IP address (sender IP address for ARP) is not existed in the IP MAC binding table. Administrator may determine the behavior via global control register LU_MIS_ACT. Note that the MAC address is always to be compared when IP address is found. The binding factor includes source MAC address at least. The port ID and VLAN ID are furthermore checked if they are specified in the binding entry. Matched action is applied for the packet pass all the comparison. Otherwise, the mismatched action takes effect.

Figure 12-2 Flowchart of IP MAC Binding Validation



Note

For ARP packet, source IP address is the sender IP address, and the source MAC address is the sender hardware address.

Table 12-11 SEC_PORT_IP_MAC_BIND_CTRL Register

Bits	Field	Description
1	ARP_CHK_EN	Enable status of IP-MAC bind checking for ARP packets 0b0: disable 0b1: enable
0	CHK_EN	Enable status of MAC bind checking for non-ARP packets 0b0: disable 0b1: enable

Table 12-12 SEC_IP_MAC_BIND_CTRL Register

Bits	Field	Description
8:6	LU_MIS_ACT	Forwarding action when table looks up miss. 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved
5:3	MISMATCH_ACT	Forwarding action when the bind isn't match 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved
2:0	MATCH_ACT	Forwarding action when the bind is match 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: copy to local CPU 0x4: trap to master CPU 0x5: copy to master CPU 0x6_0x7: reserved

13 Network Monitor

13.1 Mirror

13.1.1 Overview

1. The system supports 4 sets of mirroring. The four mirroring sets are individual.
2. Per entry supports different type mirror: Port Based, Flow Based, or RSPAN Based.
3. If a packet hits multiple mirroring entries, multiple packet copies will be sent to Mirror-To-Port (MTP, also known as sniffer port) no matter MTP in different entries is same or not.

13.1.2 Port Based Mirror

Ingress mirror is to mirror the original ingress packets and egress mirror is to mirror the modified egress packets. For example, a VLAN un-tagged packet received from port 1 outgoes to port 2 with VLAN tagged. If port 1 is ingress mirrored, a copy of the packet without VLAN tag would be sent to mirroring port. If port 2 is egress mirrored, a copy with VLAN tag would be sent to mirroring port.

For ingress mirror, bad packets such as CRC error to be ingress dropped are NOT mirrored; packets dropped by input bandwidth control module are NOT mirrored. If CPU port is ingress mirrored, the CPU TX tag of the packets would not be removed no matter whether MTP and CPU port is on the same device or not. If MTP is remote, packets transmitted from stacking port will keep the CPU TX tag and VLAN tag encapsulated as the payload. SPM is port mask that decides the ingress mirrored ports. For a stacking system, each device configures SPM for local ports. SPM is physical port mask, so for a link aggregation group, each member port is configured individually.

For egress mirror, DPM is port mask that decides the egress mirrored ports. When a non-unicast packet hits multiple ports in DPM, only the packet of the lowest ID port would be mirrored. For example, if a packet hits DPM by port 1, 2, and 3, only packet of port 1 would be egress mirrored. It should be noticed that TX mirror works per device in stacking system, thus each device could TX mirror one packet to MTP if their DPM is hit.

Table 13-1 MIR_CTRL Register

Bits	Field	Description
18:15	MTP_DEV	Device ID of mirror-to-port (MTP).
14:9	MTP_PORT	Port ID of mirror-to-port (MTP).
8:8	MTP_IS_TRK	Speficify if the mirror-to-port (MTP) is a trunk port. 0b0: The MTP is not a trunk port. 0b1: The MTP is a trunk port and MTP_DEV and MTP_PORT are aggregated to identify trunk ID.
1:0	MIR_TYPE	Mirroring set type selected. 0b00: disabled 0b01: port-based mirror 0b10: rspan mirror 0b11: flow-based mirror

Table 13-2 MIR_SPM_CTRL Register

Bits	Field	Description
56:0	SPM	Bitmap to select RX mirrored port. 0b0: disable 0b1: enable

Table 13-3 MIR_DPM_CTRL Register

Bits	Field	Description
56:0	DPM	Bitmap to select TX mirrored port. 0b0: disable 0b1: enable

MTP could be a single physical port or a trunk group. Register field *MTP_IS_TRK* could specify if the MTP is a trunk group or a physical port. If MTP is a single port, *MTP_DEV* and *MTP_PORT* specify the device ID and port ID of MTP. If MTP is a trunk group, (*MTP_DEV*[0:0], *MTP_PORT*[5:0]) are aggregated to a 7 bits trunk ID. Load balancing would be done for a trunk group MTP.

Note

1. For ingress mirror, only packets that are not dropped by MAC can be ingress mirrored; and packets dropped by input bandwidth control module are NOT ingress mirrored.
2. For egress mirror, only packets not dropped by any module can be mirrored. For example, the packets dropped by egress bandwidth control would not be egress mirrored.

13.1.3 Flow Based Mirror

By flow based mirror, a user can mirror traffic of a specific pattern such as mirroring IPv4 traffic only. It is done by configuring VACL or IACL to mirror action. By setting action to mirror and configuring specific mirror entry ID (*MIRROR_INDEX*), the packet that its pattern hits the ACL classification rule would be mirrored to the mirroring port of the specific entry. For details of configuring ACL classifications and actions, please refer to ACL developer guide.

Ingress flow based mirror can be done with VACL and IACL, egress flow based mirror can only be done with IACL. *SPM/DPM/MIR_OP* in mirror entry won't be referred to ingress mirror, but because actual ACL design, *DPM* in mirror entry is referred to egress mirror (egress ACL mirror should hit both ACL entry and *DPM*). If user wants that *DPM* won't be referred to egress mirror, user can configure all DPMs. Same as port based egress mirror, only the packet of the lowest ID port would be egress mirrored when a packets hits multiple DPM ports at the same time .

Following table is a result description when ACL Ingress and Egress mirror work at the same time. Please note that if VACL and IACL actions are in conflict, IACL result will be referred.

Table 13-4 ACL Ingress and Egress Mirror Combination Result

VACL Action	IACL Action	Designated Mirror Entry Index	MTP	Result

Ingress Mirror	Ingress Mirror	Different	Same	2 ingress mirrored packets
			Different	2 ingress mirrored packets
Ingress Mirror	Egress Mirror	Different	Same	1 ingress mirrored packet and 1 egress mirrored packet
			Different	1 ingress mirrored packet and 1 egress mirrored packet
Ingress Mirror	Ingress Mirror	Same	Same	1 ingress mirrored packet
Ingress Mirror	Egress Mirror	Same	Same	1 Egress mirrored packet

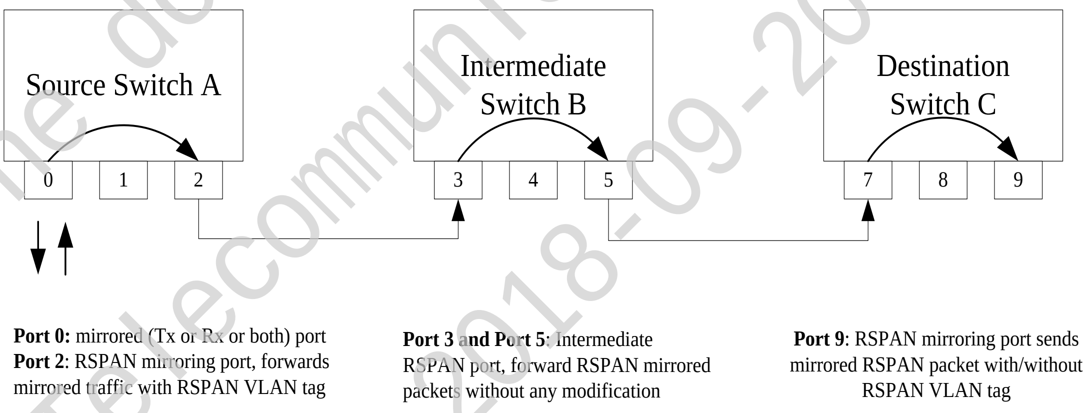
Note

1. Packets won't be mirrored when ACL action assigns one mirror entry but *MIR_TYPE* in the mirror entry isn't flow based.
2. If *DPM* is 0, ACL egress mirror action won't be active.

13.1.4 RSPAN

Remote Switched Port Analyzer (RSPAN) mechanism is to mirror traffic cross network. Switches in RSPAN architecture could play roles as a source switch, an intermediate switch, or a destination switch as Figure 13-1 RSPAN Illustration

Figure 13-1 RSPAN Illustration


Source Switch A
Intermediate Switch B
Destination Switch C
0
1
2
3
4
5
7
8
9
Port 0: mirrored (Tx or Rx or both) port
Port 2: RSPAN mirroring port, forwards mirrored traffic with RSPAN VLAN tag
Port 3 and Port 5: Intermediate RSPAN port, forward RSPAN mirrored packets without any modification
Port 9: RSPAN mirroring port sends mirrored RSPAN packet with/without RSPAN VLAN tag

Source Switch - The switch which has the monitored ports or VLANs on it is the source switch. All packets on the source ports or VLANs are copied and sent to the destination switch. When the mirrored packets are sent out from the source switch, an RSPAN VLAN tag is added to every packet. The incoming port on the source switch for the mirrored packets is referred as the source port.

Intermediate Switch - The function of the intermediate switch is to forward RSPAN traffic in the RSPAN VLAN toward the RSAPN destination. A switch can have the role of an RSAPN VLAN intermediate switch as well as the role of source switch for the same or another RSPAN VLAN. The RSPAN traffic should be treat as mirrored traffic that should not be trapped or learnt. The RSPAN VLAN should be set to be tagged to prevent the RSPAN VLAN tag being removed.

Destination Switch - The port which is directly connected to a network analyzer, other monitoring, or security device is called the destination port. The switch which has a destination port is called the destination switch. The destination switch can keep or removes the RSPAN VLAN tags.

13.1.4.1 Source Switch Functionality Support

RSPAN source switch would do port or flow based mirror, and add an RSPAN tag to the mirrored packets at *MTP* port. The *MIR_TYPE* of mirror entry for RSPAN source switch should be port based mirror or flow based mirror.

The system provides a bit *RSPAN_TX_TAG_ADD* for a mirroring set: If *RSPAN_TX_TAG_ADD* = 0: Don't add RSPAN VLAN tag for mirrored traffic. If *RSPAN_TX_TAG_ADD* = 1: Add RSPAN VLAN tag for mirrored traffic.

Per mirroring set can define the *RSPAN_TAG_TPID*, *RSPAN_TAG_PRI*, *RSPAN_TAG_CFI*, *RSPAN_TAG_VID* of RSPAN VLAN tag which would be added in RSPAN traffic packets. Since a mirrored packet is multicast to all hit MTPs, the RSPAN tag adding is done individually by each *MTP*.

Table 13-5 MIR_RSPAN_TX_CTRL Register

Bits	Field	Description
0:0	RSPAN_TX_TAG_ADD	Add RSPAN VLAN tag for mirrored packets that do not carry RSPAN VLAN tag for mirroring set n. This bit is ignored by packets which already have RSPAN VLAN tag. 0b0: disable 0b1: enable

Table 13-6 MIR_RSPAN_VLAN_CTRL Register

Bits	Field	Description
31:16	RSPAN_TAG_TPID	TPID of RSPAN tag. This is used to identify RX RSPAN traffic and to decide the TPID of TX added RSPAN VLAN tag.
15:13	RSPAN_TAG_PRI	Priority of RSPAN tag. This is used to decide PRI of the TX added RSPAN VLAN tag.
12:12	RSPAN_TAG_CFI	CFI of RSPAN tag. This is used to decide CFI of the TX added RSPAN VLAN tag.
11:0	RSPAN_TAG_VID	VID of RSPAN tag. This is used to identify RX RSPAN traffic and to decide VID of the TX added RSPAN VLAN tag.

13.1.4.2 Intermediate Switch Functionality Support

The intermediate switch receives the RSPAN packet and floods it in RSPAN VLAN. System does not provide specific function to do intermediate switch support and user is supposed to configure VLAN member set to decide the RSAPN traffic forwarding path. The RSPAN VLAN traffic should not be learnt and this could be done by configuring VLAN profile to disable L2 learning for RSPAN VLAN.

To avoid the RSPAN traffic be trapped or dropped by functions such as ACL or RMA, it could be done by setting an ACL rule for RSPAN VLAN (set keys to be 1. VLAN = RSPAN VLAN, 2. VLAN tagged and action to be redirect) to redirect the packet to RSPAN VLAN ports.

13.1.4.3 Destination Switch Functionality Support

The *MIR_TYPE* of mirroring entry for RSPAN destination switch should be configured to RSPAN mirror.

Per mirroring set uses *RSPAN_TAG_VID* to decide the cared *RSPAN VLAN ID* and SPM to decide the RSPAN traffic RX ports. For example, if *RSPAN_TAG_VID* is 100 and *SPM* is port 1 and 2, the packets that received from port 1 or 2 with "first VLAN tag" with VID=100 would be forwarded to the *MTP* of this mirroring set.

RSPAN_RX_TAG_RM decides if the system needs to remove the RSPAN VLAN tag of RX mirrored RSPAN traffic. If *RSPAN_RX_TAG_RM* = 0: Don't remove the RSPAN VLAN tag when the RX RSPAN traffic is TX to MTP, If *RSPAN_RX_TAG_RM* = 1: Remove the RSPAN VLAN tag when the RX RSPAN traffic is TX to MTP.

Table 13-7 MIR_RSPAN_RX_TAG_RM_CTRL Register

Bits	Field	Description
0:0	RSPAN_RX_TAG_RM	If RSPAN_RX_TAG_EN is 1, this bit decides if the RSPAN VLAN tag of RSPAN packet should be removed. This bit is ignored by packets that do not have RSPAN VLAN tag. 0b0: disable 0b1: enable

Note

1. Since mirroring can cross VLAN, and even the packet is VLAN ingress filtered, the packet can be ingress mirrored to the *MTP*. Thus the cared RSPAN ingress ports and MTP are not necessary to join RSPAN VLAN for destination switch configuration.

13.1.5 Mirror Option

Each mirror set supports optional functions to control the mirror traffic.

13.1.5.1 DUP_FILTER

When a single packet is both normal forwarded and mirrored to *MTP* port, analyzer may be confused. System supports *DUP_FILTER* of each mirroring entry set to decide whether to mirror the packet when its normal forwarded destination ports contain *MTP*.

For non-unicast packet, if one of its destination ports of normal forwarded is *MTP* port, it is also affected by *DUP_FILTER*.

Table 13-8 MIR_CTRL Register

Bits	Field	Description
7:7	DUP_FILTER	Decide if port mirror and ACL mirror work for the packet that destination ports of normal forward contain MTP. For non-unicast packet, if one of its destination ports of normal forward is MTP port, it is also affected by this register. 0b0: Packet that destination ports of normal forward contain MTP could be mirrored 0b1: Do not mirror for packet that destination ports of normal forward contain MTP

Note

1. The *DUP_FILTER* function only works when *MTP* and ingress/egress mirrored ports are on the same switch (in stand-alone project, this rule is always matched). For stacking system, *DUP_FILTER* is supposed to be configured as disabled.
2. *DUP_FILTER* is supposed to be configured as disabled if MTP is a trunk group.

13.1.5.2 MIR_OP

For a packet both hits *SPM* and *DPM* in a mirroring entry on a switch, if *MIR_OP* of each mirroring entry is OR, a packet hitting *SPM* or *DPM* would be mirrored. If *MIR_OP* is AND, only packet hitting both *SPM* and *DPM* would be mirrored. In stacking system, *MIR_OP* should be configured to OR since *SPM*/ *DPM* hit in different switches cannot be aware.

Table 13-9 MIR_CTRL Register

Bits	Field	Description
2:2	MIR_OP	Decides the traffic should hit SPM "OR" DPM or hit SPM "AND" DPM to be mirrored. 0b0: or 0b1: and

13.1.5.3 MIR_RX_TX_SEL

If a packet hits both *SPM* and *DPM* on a switch, no matter what *MIR_OP* is, *MIR_RX_TX_SEL* of each mirroring entry decides if ingress or egress mirror should be done.

Table 13-10 MIR_CTRL Register

Bits	Field	Description

4:4	MIR_RX_TX_SEL	If SPM and DPM are both hit, this register decides if ingress mirror or egress mirror is done 0x0: only ingress mirror is done 0x1: only egress mirror is done

Note

1. The *MIR_RX_TX_SEL* function only works when ingress and egress mirrored ports are on the same switch.

13.1.5.4 MIR_SELF_FILTER

MIR_SELF_FILTER of each mirroring entry decides if the packet received by *MTP* should cancel ingress or egress mirror.

Table 13-11 MIR_CTRL Register

Bits	Field	Description
3:3	MTP_SELF_FLTER	Packet transmitted from mirroring port is determined to be mirrored 0b0: it could be mirrored to mirroring port 0b1: the mirrored packet is filtered

13.1.5.5 MTP_TX_ISO

MTP_TX_ISO of each mirroring entry is used to configure whether to isolate the mirroring ports from normal forwarding traffic. If *MTP_TX_ISO* is enabled, the mirroring port would only accept mirrored packets. If *MTP_TX_ISO* is disabled, the mirroring port would accept mirrored packets and normal forwarding packets. The effect of setting *MTP_TX_ISO* is the same as configuring port isolation to remove *MTP* from port mask of all ports. But configuring *MTP_TX_ISO* is more convenient.

In the case that *MTP_TX_ISO* = True, since there would be no normal forward packets sending to *MTP* ports, a packet is free to be mirrored and won't be possible to trigger *DUP_FILTER*=TRUE case.

Table 13-12 MTP_CTRL Register

Bits	Field	Description
6:6	MTP_TX_ISO	Isolate the mirror-to-port from forwarding normal traffic. 0b0: The MTP can forward mirrored and normal traffic. 0b1: The MTP only forward mirrored traffic, but cannot forward normal traffic.

13.1.5.6 VLAN TAG Mode

Analyzer can verify the incoming port of mirrored packets by VLAN ID when VLAN tag status of mirrored packets follows *MTP* port. Per mirroring entry set can configure *MIR _MODE* to make VLAN tag status of mirrored packets (both RX and TX mirror) would follow *MTP* port configuration or Not.

Table 13-13 MIR_MODE Register

Bits	Field	Description
5:5	MIR_MODE	If mirrored packet follows the TX modules of mirroring port (ex: VLAN tag status, remarking, EVC ...). 0b0: do not follow 0b1: follow mirroring port TX modules

Note

1. When mirrored packets cross devices to remote *MTP*, *MIR _MODE* should be set to 0.

13.1.5.7 Mirror VS Link Aggregation

If the mirrored port belongs to a link aggregation, the mirroring would be specific to the physical port instead of the whole link aggregation. If a user wants to mirror all the link aggregation member ports, it can be done by adding all the member ports to *SPM* or *DPM*.

System supports trunk port as *MTP* port of each mirroring entry. For example, if trunk 1 contains (device 1, port 3), (device 2, port 3) as member ports. If user wants to take trunk 1 as *MTP* port, it can be achieved by configuring *MTP_IS_TRK* =1, *MTP_PORT* =1.

Note

1. For stacking system, hash value of mirroring packets is from first device and the value is the same as mirrored frame.

13.1.5.8 Mirror Queue

System supports global control *MIR_QID_EN* and *MIR_QID* to decide if mirroring packet outputs to specific queue. This function could be used to move mirroring packet to a higher or lower priority queue depending on customer requirement.

If a packet is only mirrored to *MTP* (not normal forwarded to *MTP*), the packet would follow *MIR_QID* to enqueue if *MIR_QID_EN* is enabled. If a packet is normal forwarded and mirrored to *MTP*, its output queue would be based on normal forwarding queue instead of following *MIR_QID* configuration. For mirroring packet crossing stacking port, this function would work when it is outgoing to *MTP* port.

Table 13-14 MIR_QID_CTRL Register

Bits	Field	Description
3:1	MIR_QID	Mirrored traffic enter output queue id at mirroring port
0:0	MIR_QID_EN	Specify mirrored traffic output queue id 0b0: disable 0b1: enable

13.1.6 Sampling

The sampling rate *SAMPLE_RATE* is per mirroring set configuration. It is used to support packets sampling mechanisms such as sFlow. If *SAMPLE_RATE* is set to n which is larger than 0, the mirroring action would happen every n+1 mirrored packets. For example, if *SAMPLE_RATE* is 100, the 101th packet that hits the mirroring settings would be mirrored after 100 hit packets skipped.

SAMPLE_RATE configuration also can cooperate with ACL rules to support protocol based or VLAN based sampling.

Table 13-15 MIR_SAMPLE_RATE_CTRL Register

Bits	Field	Description
15:0	SAMPLE_RATE	The sampling rate of mirrored packets. SAMPLE_RATE = 0, just mirror, do not need to care about sampling rate; SAMPLE_RATE = n(n>0),sample the mirrored packets every n + 1 packets (skip n packets)

Note

Packets won't be sampled again if the mirrored packets are from stacking port.

13.2 sFlow

13.2.1 sFlow Overview

sFlow is a count-based (packet-based) technology for monitoring traffic in data networks containing switches and routers. In particular, it defines the sampling mechanisms implemented in a sFlow Agent for monitoring traffic.

13.2.2 sFlow Sampling Method

The system supports ingress and egress sample rate.

Table 13-16 SFLOW_PORT_RATE_CTRL Register

Bits	Field	Description
31:16	EGR_RATE	Packet sampling rate for egress traffic
15:0	IGR_RATE	Packet sampling rate for ingress traffic

If configures rate is N, the system will sample randomly a packet every N packet. The sampled packet will be marked reason "sFlow sampling" in CPU tag.

13.2.3 sFlow Compensation

A packet may be remarked for sampling both ingress and egress at the same time. In that case, SMPL_SEL field becomes the deciding factor, which one has to be sampled. The next packet which satisfies the other direction will be compensated.

Table 13-17 SFLOW_CTRL Register

Bits	Field	Description
1	SMPL_SEL	If any packet is marked for sampling twice for both Ingress & Egress then this bit will dictate which packet will be sampled, either Ingress packet or Egress packet. 0b0: Ingress Sampling 0b1: Egress Sampling

14 Diagnostic

14.1 OAM

14.1.1 OAM Overview

OAM defines the Operations, Administration and Maintenance (OAM) sub-layer, which provides mechanisms useful for monitoring link operation such as remote fault indication and remote loopback control. In general, OAM provides network operators the ability to monitor the health of the network and quickly determine the location of failing links or fault conditions.

OAM information conveys in slow protocol frames called OAM Protocol Data Units (OAMPDUs). OAMPDUs contain the appropriate control and status information used to monitor, test and troubleshoot OAM-enabled links.

Dying Gasp is a message sent by the Data Terminating Entity (DTE) when power outage occurs. It is a point-to-point Ethernet OAM communication. The DTE with dying gasp must derive power for a brief period from another source so that the message can be sent without external power.

14.1.2 Loopback

Loopback mechanism is provided to support a data link layer frame-level loopback mode, which is controlled by remotely. OAM remote loopback can be used for fault localization and link performance testing.

Non-OAMPDU will take the Parser action and the Multiplexer action when enable OAM loopback feature. The MAC address status of loopback packet can be controlled by OAM_CTRL.MAC_SWAP when set the Parser action to loopback. When enable OAM loopback feature, OAMPDU will be trapped to CPU.

Table 14-1 OAM_PORT_ACT_CTRL Register

Bits	Field	Description
2:1	PAR_ACT	Action to be taken by the Parser for non-OAMPDU Rx traffic 0x0: discard; drop all non-OAMPDU Rx traffic 0x1: forward; pass all non-OAMPDU Rx traffic to next module 0x2: loopback; pass all non-OAMPDU Rx traffic to Multiplexer for looping it back to the source port (except Pause Frame & CRC error frame). MAC learning must be OFF for looped back traffic on that port 0x3: trap; just trap the packets to CPU
0:0	MUX_ACT	Action to be taken by the Multiplexer for non-OAMPDU Tx traffic 0b0: discard; drop all non-OAMPDU Tx traffic 0b1: forward; pass all non-OAMPDU Tx traffic to lower layer for transmission

Table 14-2 OAM_CTRL Register

Bits	Field	Description
4:4	LRN	OAMPDU source MAC learning control 0b0: disable 0b1: enable
3:3	CPU_SEL	Select packet trap to local or master CPU 0b0: trap to local CPU 0b1: trap to master CPU
2:2	MAC_SWAP	If OAM Loopback is globally enabled (i.e. EN = 0x1) and parser is in 'Loopback' state (i.e. PAR_ACT = loopback) then this bit takes effect. If this bit is enabled then swap the MAC Addresses (Source MAC & Destination MAC) else do not. 0b0: disable 0b1: enable
1:1	DIS_ACT	If EN is set to disable then this bit takes effect. Such as if EN = disable and DIS_ACT = forward then OAM PDU traffic must be forwarded instead of drop.Value of this bit is forward or drop 0b0: drop 0b1: forward
0:0	EN	Globally enable or disable OAM loopback feature 0b0: disable 0b1: enable

14.1.3 Dying Gasp

Dying Gasp is a message sent by the DTE when a power outage occurs. Critical link events are carried within the Flags field of each OAMPDU. Dying gasp is one of critical link event which means an unrecoverable local failure condition has occurred. The DTE with dying gasp must derive power for a brief period from another source so that the message can be sent without external power.

The user sends packets with dying gasp bit in CPU tag to the system at first. The system will keep the packets in buffer, and then sends out the packets when dying gasp is triggered.

Table 14-3 OAM_GBL_DYING_GASP_CTRL Register

Bits	Field	Description
19:17	DYING_GASP_PKTCNT	Select how many dying gasp packets can be sent when power failure happen or trigger dying gasp.
15:0	TBP_VAL	Time of Voltage below Vref period, represented by setting + 1 Digital Circuit using this register to deglitch. Timer interval depend on system clock.

Table 14-4 OAM_PORT_DYING_GASP_CTRL Register

Bits	Field	Description

0:0	PORT_OAM_DYING_GASP_EN	Enable the specific port to generate dying gasp message when detected power failure. 0b0: disable 0b1: enable

14.2 CFM

Operations, Administration, and Maintenance (OAM), which provides mechanisms useful for monitoring link operation such as remote fault indication and remote loopback control. In general, OAM provides network operators the ability to monitor the health of the network and quickly determine the location of failing links or fault conditions.

Instead of IEEE 802.3 OAM only protects a single link, Connectivity Fault Management (CFM) can further protect a network of bridged networks, over multiple nested ranges.

CFM comprises capabilities for path discovery (Linktrace), fault detecting (Continuity Check), and fault verification (Loopback). These capabilities can be used in networks operated by multiple independent organizations, each with restricted management access to each other's equipment. CFM is designed to be transparent to the customer data transported by a network and to be capable of providing maximum fault coverage.

Table 14-5 CFM_RX_LT_CTRL Register

Bits	Field	Description
2x+1:2x	MD_LV_x	Action on receiving Linktrace frames with MD Level x. (x = 0~7) 0x0: fwd 0x1: drop 0x2: trap to CPU

Table 14-6 CFM_RX_LB_CTRL Register

Bits	Field	Description
2x+1:2x	MD_LV_x	Action on receiving Loopback frames with MD Level x. (x = 0~7) 0x0: fwd 0x1: drop 0x2: trap to CPU

Table 14-7 CFM_RX_CCM_CTRL Register

Bits	Field	Description
2x+1:2x	MD_LV_x	Action on receiving CCM frames with MD Level x. (x = 0~7) 0x0: fwd 0x1: drop 0x2: trap to CPU 0x3: LFD(link fault detection for ring protection)

Ethernet Ring Protection Switching

ITU-T G.8031 and G.8032 are the Ethernet automatic protection switching (APS) mechanisms. The time of protection and recovery switching occupation is within 50ms. It helps achieve highly reliable and stable protection.

G.8031 provides linear protects mechanism in Ethernet layer. It is point-to-point VLAN-based Ethernet sub network connection (SNC). It protects a link between two adjacent nodes using a protection link. It requires a working entity and protection entity. Each bridge selects which link to use to transmit or receive packets.

G.8032 is for Ethernet traffic in a ring topology. It helps achieve never formed loops in the ring. Each Ethernet ring node is connected to adjacent Ethernet ring nodes participating in the same Ethernet ring, using two independent links. The particular link is used to protect the whole ring is called the ring protection link (RPL). Under the normal condition this link is blocked. Under an Ethernet ring failure condition, protection switch blocks traffic on the failure link and the RPL owner node is responsible to unblock RPL, unless the RPL failed.

The device supports link failure detection by sending CFM CCM packet periodically and update keep-alive counter when receive CFM CCM packet by hardware. The mechanism offloads software effort to detect the link failure and therefore gains the time for software to recovery the failed link/ring.

14.2.1 CCM Transmission

The device supports 8 instances with two member ports in each instance. Each instance can be enabled to transmit CCM packet periodically according to the following configuration. The content of VLAN tag is specified in Table 14-8.

Table 14-8 CCM_TX_TAG_CTRL Register

Bits	Field	Description
5:3	PCP	Priority Code Point value of the tag to be used in the generated CCM frames.
2	CFI	Canonical Format Identifier of the tag to be used in the generated CCM frames
1:0	TPID	Index for pointer to outer TPID list for the generated CCM frames

For generating CCM Tx packet, the following registers are instance-based configuration. Each instance can determine different VLAN ID, VLAN tag status, MAID, MEPID, MDL and transmit interval. See Table 14-9 and Table 14-10.

Table 14-9 CCM_TX_INST_CTRL Register

Bits	Field	Description
31	VID_ADD	This bit signifies whether outer VLAN tag would be inserted into the CCM frames generated by PORT0 & PORT1. If enable then VLAN tag would be inserted into the CCM frames else not. 0b0: disable ob1: enable
30:19	VID	Outer VID to be added into the CCM frames if VID_ADD bit is enabled.
18:11	MAID	LSB 1 byte to be inserted into the generated CCM frame as MAID

10	TX	This bit enables/disables CCM frame generation & transmission of corresponding instance. 0b0: disable, do not generate CCM packets 0b1: enable, enable CCM packet generation
9:0	INTLV	CCM frame transmission interval in msec for the corresponding instances.

Table 14-10 CCM_TX_INST_PKT Register

Bits	Field	Description
18:6	MEPID	MEPID to be used for generating CCM frames
5:3	CCM_INTLV	CCM Interval Field to be used for generating CCM frames
2:0	MDL	MD Level to be used for generating CCM frames

In Table 14-11, the index N and N+8 of CFM_TX_INST_MEM.P0 represent the MEMBER0 and MEMBER1 for instance N respectively.

Table 14-11 CCM_TX_INST_MEM Register

Bits	Field	Description
6:0	P0	Port number which would participate in ERP and would generate & send CCM frames.

In Table 14-12, the index N and N+8 of CFM_TX_INST_TRK_PRESENT.TRK_PRESENT represent the MEMBER0 and MEMBER 1 for instance N respectively.

Table 14-12 CCM_TX_INST_TRK_PRESENT Register

Bits	Field	Description
0	TRK_PRESENT	CCM_TX_INST_MEM.P0 is trunk ID or port 0x0: port 0x1: trunk ID

14.2.2 CCM Reception

Configure CFM CCM action to 'LFD' for link failure detect. The device counts down the instance member's keep-alive counter every millisecond and the countdown mechanism doesn't start until the CFM CCM packet is received. When receiving CFM CCM packet, the device finds the corresponding instance member by VID and port, then updates its keep-alive counter to the instance's keep-alive timer (CCM_LIFETIME_CTRL.LIFETIME field). Otherwise when the value of keep-alive counter counts down to 0, the device signals the corresponding instance member's interrupt to CPU to notify the link failure event.

Table 14-13 CCM_RX_LIFETIME_CTRL Register

Bits	Field	Description

9:0	LIFETIME	Lifetime value which must be used to reset the ERP instance counters upon receiving a CCM frame for the corresponding instances. E.g. if a CCM packet satisfies the instance0 then value of lifetime0 must be taken to reset the instance0 counters.

Table 14-14 CCM_RX_INST_CTRL Register

Bits	Field	Description
27:16	VID	VID associated with the corresponding instance. VID = 0 signifies untagged CCM frames are permitted and do not check VID in the frame if the received fame is tagged
15	TRK_PRESENT1	MEMBER1 is trunk ID or port 0x0: port 0x1: trunk ID
14:8	MEMBER1	Port number which would be participated in ERP and would receive CCM frames
7	TRK_PRESENT0	MEMBER0 is trunk ID or port 0x0: port 0x1: trunk ID
6:0	MEMBER0	Port number which would be participated in ERP and would receive CCM frames

Table 14-15 CCM_RX_INST_CNT Register

Bits	Field	Description
19:10	CNTR1	Keep alive counter corresponding to member1, which has configured to receive CCM frames. If this counter reaches 0 then an interrupt must be fired. On receiving CCM packet this counter must be reset to the value configured by the corresponding LIFETIME of the same instance
9:0	CNTR0	Keep alive counter corresponding to member0, which has configured to receive CCM frames. If this counter reaches 0 then an interrupt must be fired. On receiving CCM packet this counter must be reset to the value configured by the corresponding LIFETIME of the same instance

14.3 ETH-DM

14.3.1 Overview

ETH-DM (Ethernet Delay Measurement) can be used for on-demand OAM to measure frame delay and frame delay variation. The measurements are performed by sending and receiving periodic frames with ETH-DM information with the peer MEP (Maintenance End Point) during the diagnostic interval.

ETH-DM can be performed in two ways: one-way ETH-DM and two-way ETH-DM.

14.3.1.1 One-Way ETH-DM

In this case, each MEP sends a frame with one-way ETH-DM information to its peer MEP in a point-to-point ME to facilitate one-way frame delay and/or one-way frame delay variation measurements at the peer MEP. Figure 14-1 shows the frame format of 1DM PDU. Figure 14-2 illustrates the one way delay and delay variation measurement by sending 1DM frame.

Figure 14-1 1DM PDU Format

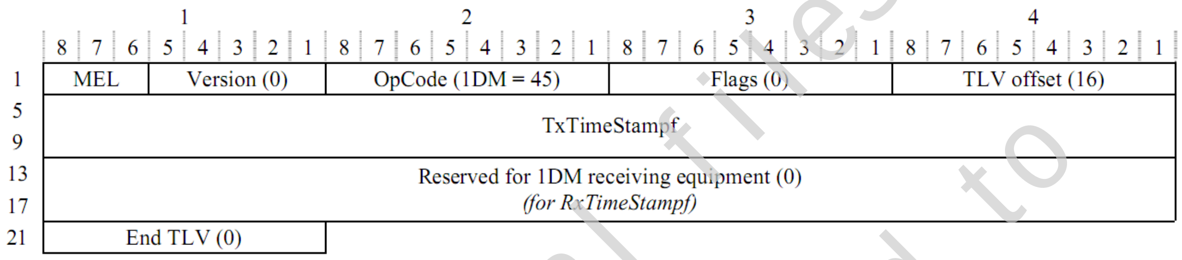
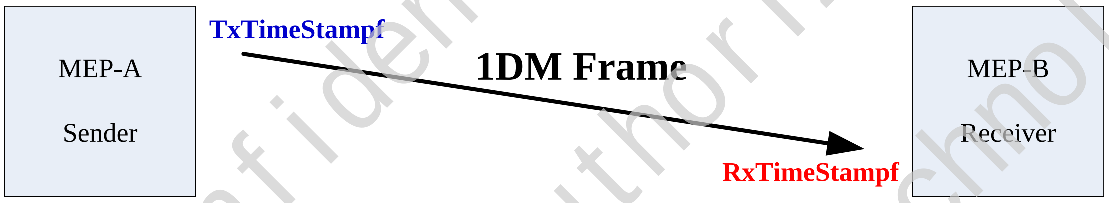


Figure 14-2 One-Way Delay Measurement



One-way Frame Delay = RxTimeStampf – TxTimeStampf
(if the clocks between MEP-A and MEP-B are sychronized)

Frame Delay Variation (n) = | Frame Delay (n) – Frame Delay (n-1) |

14.3.1.2 Two-Way ETH-DM

A MEP sends frames with ETH-DM request information to its peer MEP and receives frames with ETH-DM reply information from its peer MEP to carry out two-way frame delay and two-way frame delay variation measurements.

The PDU used for ETH-DM request is DMM (Delay Measurement Message, Figure 14-3). The PDU used for ETH-DM reply is DMR (Delay Measurement Reply, Figure 14-4). Figure 14-5 illustrates the two way delay and delay variation measurement by DMM and DMR frame exchanges.

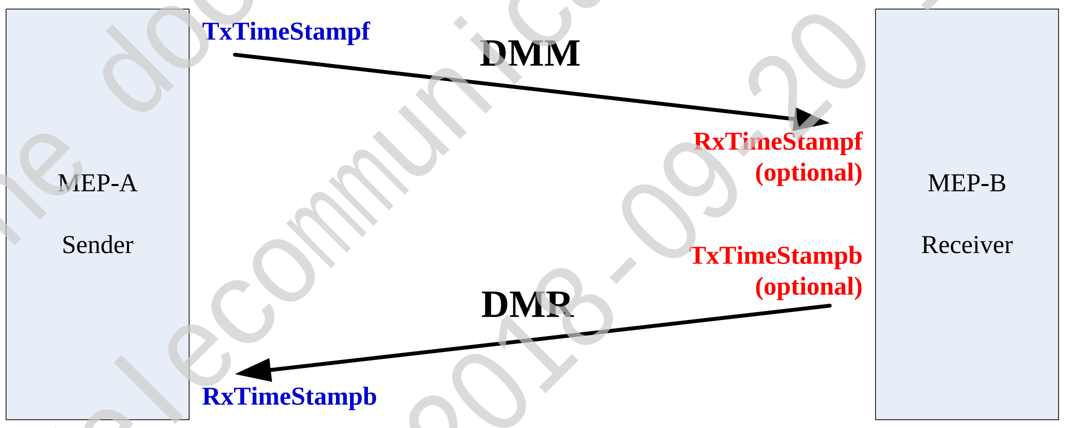
Figure 14-3 DMM PDU Format

	1		2	3	4
	8 7 6	5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
1	MEL	Version (0)	OpCode (DMM = 47)	Flags (0)	TLV offset (32)
5 9	TxTimeStampf				
13 17	Reserved for DMM receiving equipment (0) *(for RxTimeStampf)*				
21 25	Reserved for DMR (0) *(for TxTimeStampb)*				
29 33	Reserved for DMR receiving equipment (0)				
37	End TLV (0)				

Figure 14-4 DMR PDU Format

	1		2	3	4
	8 7 6	5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
1	MEL	Version	OpCode (DMR = 46)	Flags	TLV offset
5 9	TxTimeStampf				
13 17	RxTimeStampf				
21 25	TxTimeStampb				
29 33	Reserved for DMR receiving equipment (0) *(for RxTimeStampb)*				
37	End TLV (0)				

Figure 14-5 DMM PDU Format



Two-way Frame Delay =
(RxTimeStampb – TxTimeStampf) – (TxTimeStampb – RxTimeStampf)

Frame Delay Variation (n) = | Frame Delay (n) – Frame Delay (n-1) |

14.3.2 ETH-DM Time Clock Module

14.3.2.1 Time Clock Control

The system time clock module for ETH-DM is tick in 125MHZ (every 8 ns). The system time is used for RX/ TX timestamping. CLK_EN register field can be used to start/stop clock free run. Access of system time could be achieved by using SEC, NSEC, EXEC, and CMD register fields.

Table 14-16 ETH_DM_CLK_CTRL Register

Bits	Field	Description
0	CLK_EN	Enable the free-running clock of ETH-DM module. 0b0:disable 0b1:enable

Table 14-17 ETH_DM_TIME_CTRL_SEC Register

Bits	Field	Description
63:32	SEC	Second in the reference time clock.
29:0	NSEC	Nanosecond in the reference time clock. (Unit: 1 nanoseconds)

Table 14-18 ETH_DM_TIME_CTRL Register

Bits	Field	Description
1	EXEC	Trigger hardware to execute an indirect access of System Time. (Note: this bit is used by both software and hardware. When hardware completes the table access, it will clear this bit automatically.)
0	CMD	Operation of the indirect access of System Time. 0b0: Read 0b1: Write

14.3.2.2 Frequency Adjustment

A frequency adjustment mechanism is provided for ETH-DM time clock so it could synchronize the frequency with another clock. The default value of frequency configuration register TIME_FREQ is 0x80000000. The frequency of clock is direct proportional to TIME_FREQ. For example, if the clock of DUT is 1 ppb faster than a high accuracy clock. It could be adjusted by configure TIME_FREQ to 0x80000000*(1 – 10^{-9})=0x7FFFFFFD.

Table 14-19 ETH_DM_TIME_FREQ Register

Bits	Field	Description
27:0	TIME_FREQ	Frequency of the system reference time.

14.3.3 RX Timestamp Recording

Hardware based RX timestamping of received 1DM/ DMR/ DMM frames are supported. The RX timestamping would be done if all following conditions are satisfied.

1. Ethertype is 0x8902. (untag and one or two layer VLAN tagged frames are supported)
2. OpCode is 0x2D - 1DM (45), 0x2E - DMR (46), or 0x2F - DMM (47).
3. If ETH-DM EN register field of the receiving port is 0b1.
4. The MD_LV_n of the MEL (Maintenance Entity Group Level) is set to trap action.

Table 14-20 ETH_DM_PORT_EN Register

Bits	Field	Description
0	EN	Enable ETH-DM function. (If an ETH-DM packet comes from an enabled port, ASIC will do the Rx timestaming) 0b0: disable 0b1: enable

Table 14-21 ETH_DM_RX_CTRL Register

Bits	Field	Description
15:14 13:12 11:10 9:8 7:6 5:4 3:2 1:0	MD_LV_n	Action on receiving Ethernet Delay Measurement frames with MD Level n (n ranges from 0 to 7). 0x0: forward 0x1: drop 0x2: trap to local CPU 0x3: trap to master CPU

Total 64 entries of RX timestamp could be stored for software to read. An entry contains SEC and NSEC fields. Hardware would write the RX timestamp to an empty entry (both SEC and NSEC are 0) and increment the current index to be next timestamp index (the next index would wrap to 0 if current index is 63). If the target index is not empty, an interrupt would be triggered (ISR_ETHDM) to inform CPU and the frame would be dropped. The diagram of RX timestamping is illustrated in Figure 15-13.

The RX timestamp index would be written into CPU RX tag field DM_RXIDX[5:0]. Software could base on the index to read the RX timestamp.

Table 14-22 ETH_DM_RX_TIME Register

Bits	Field	Description
61:32	NSEC	The nanosecond of the latched Rx timestamp. Read NSEC would clear both NSEC and SEC.
31:0	SEC	The second of the latched Rx timestamp.

15 Stacking

15.1 Stacking

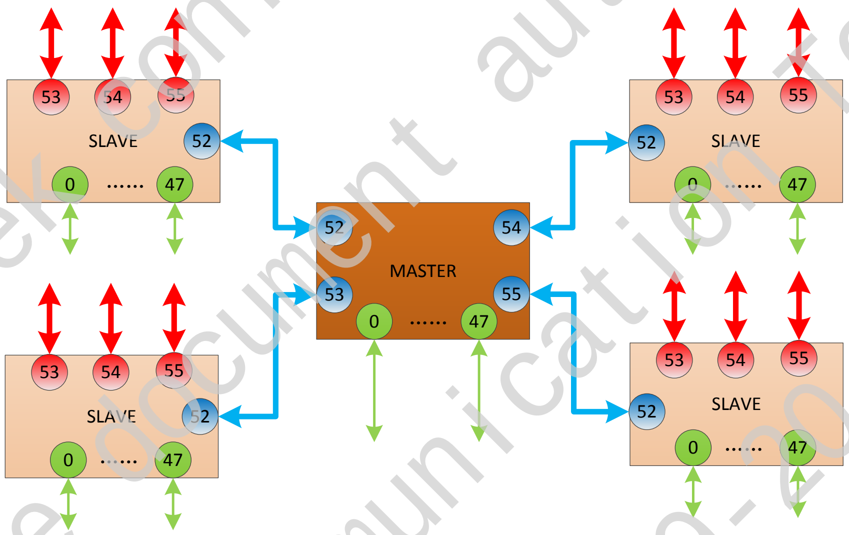
Stackable switch is a network switch that is fully functional operating standalone but which can also be set up to operate together with one or more other network switches, with this group of switches showing the characteristics of a single switch but having the port capacity of the sum of the combined switches.RTL9310 supports the stacking feature completely.

15.2 Stacking Topology

Stacking system can be organized in different topologies, such as STAR, LINE and RING.

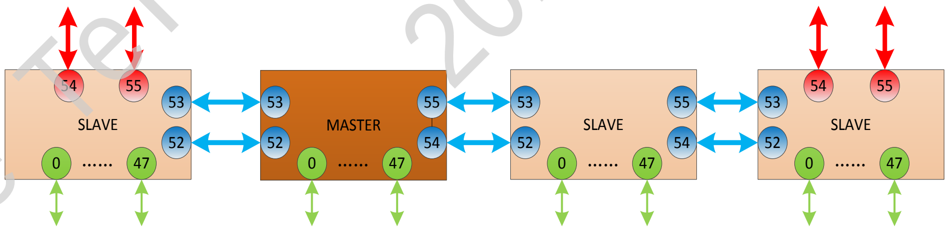
15.2.1 Star Topology

Figure 15-1 Star Topology Diagram



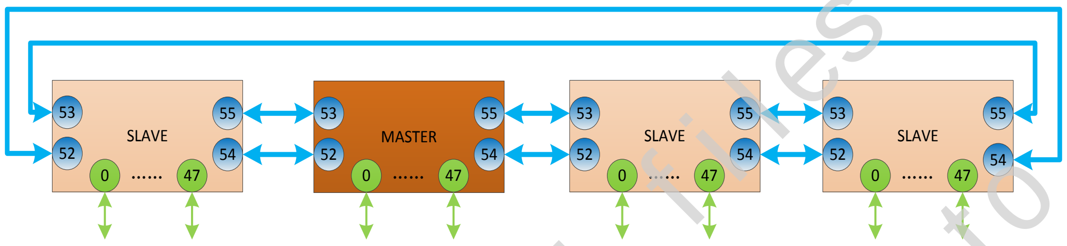
15.2.2 Line Topology

Figure 15-2 Line Topology Diagram



15.2.3 Ring Topology

Figure 15-3 Ring Topology Diagram



15.3 Stacking Configuration

15.3.1 Device ID

Each switch in stacking system should have a unique device ID. One of them will be selected as the MASTER role, and the others are all acted as SLAVE roles.

System supports max to 16 devices stacking together. Each device should be configured with a *MY_DEV_ID* to identify itself, and also a *MASTER_DEV_ID* to recognize the MASTER device.

Table 15-1 STK_GLB_CTRL Register

Bits	Field	Description
7:4	MY_DEV_ID	My Device ID
3:0	MASTER_DEV_ID	Master Device ID

15.3.2 Stacking Port

The port which is used to connect different devices must be configured as stacking port. System supports max to 16 stacking ports which could be selected from any ports of switch. If a stacking port is a 10G fiber port, the IPG/ Preamble and CRC length would be shorten to compensate the enlarged frame size because of attached stacking header. For other speed/ media, the IPG/ Preamble and CRC length would be as normal, and the port could be used as a front stacking port.

Table 15-2 STK_PORT_ID_CTRL Register

Bits	Field	Description
5:0	STK_PORT_ID	Port ID of stacking port. Only valid when the port is stacking port. Value > 55 means the index does not map to any port.

15.3.3 One-Hop Path

One-Hop Path guides the way to other devices. Globally 16 *DEV_PORT_MAP* bitmask settings are provided, which specifies the local device using which stacking ports to forward the packet to the other device. Each bit in bitmask stands for one of the 16 stacking ports.

For example, *STK_DEV_PORT_MAP_CTRL[6].DEV_PORT_MAP = 0b1100* and STK_PORT_ID[2] = 52, STK_PORT_ID[3] = 53 means if packet's destination is Device 6, it should go through stacking port 52 or port 53.

Table 15-3 STK_DEV_PORT_MAP_CTRL Register

Bits	Field	Description
15:0	DEV_PORT_MAP	Specify the target device to stacking port mask mapping for packet forwarding decision. Value: 0b0: Packet forwarding to the device should not forward to the stacking port. 0b1: Packet forwarding to the device should forward to the stacking port.

15.3.4 Stacking Port Trunk

Several stacking ports also can be aggregated into one trunk to utilize the bandwidth effectively. As there are max 16 stacking ports, system supports only 8 stacking trunks settings which are independent of normal trunks. Max 8 stacking ports can be aggregated into single trunk.

Table 15-4 TRK_STK_CTRL Register

Bits	Field	Description
15:0	STK_TRK_PPM	Member portmask of stacking trunk

15.3.5 Loop Prevention

Loop problem maybe occurs in the stacking system, especially in the Ring topology. To prevent this problem, system provides two mechanisms to resolve that.

15.3.5.1 Drop My Device

When receiving a packet with source device ID in stacking header equal to my own device ID, it means the packet comes back to me who once sent it. So the packet should be dropped to prevent loop.

Table 15-5 STK_GLB_CTRL Register

Bits	Field	Description
8:8	DROP_MY_DEV	Drop the packet that source device field in stacking header is the same as MY_DEV_ID. Value: 0b0: do not drop the packet 0b1: drop the packet

15.3.5.2 Non-Unicast Block

Mechanism of DROP_MY_DEV can simply prevent loop storm. But in Ring topology, Non-Unicast traffic, such as broadcast, multicast and unknown unicast, may be received twice from different paths. So our system provides *NON_UNICAST_BLOCK_PM* per device to cut off the redundant path.

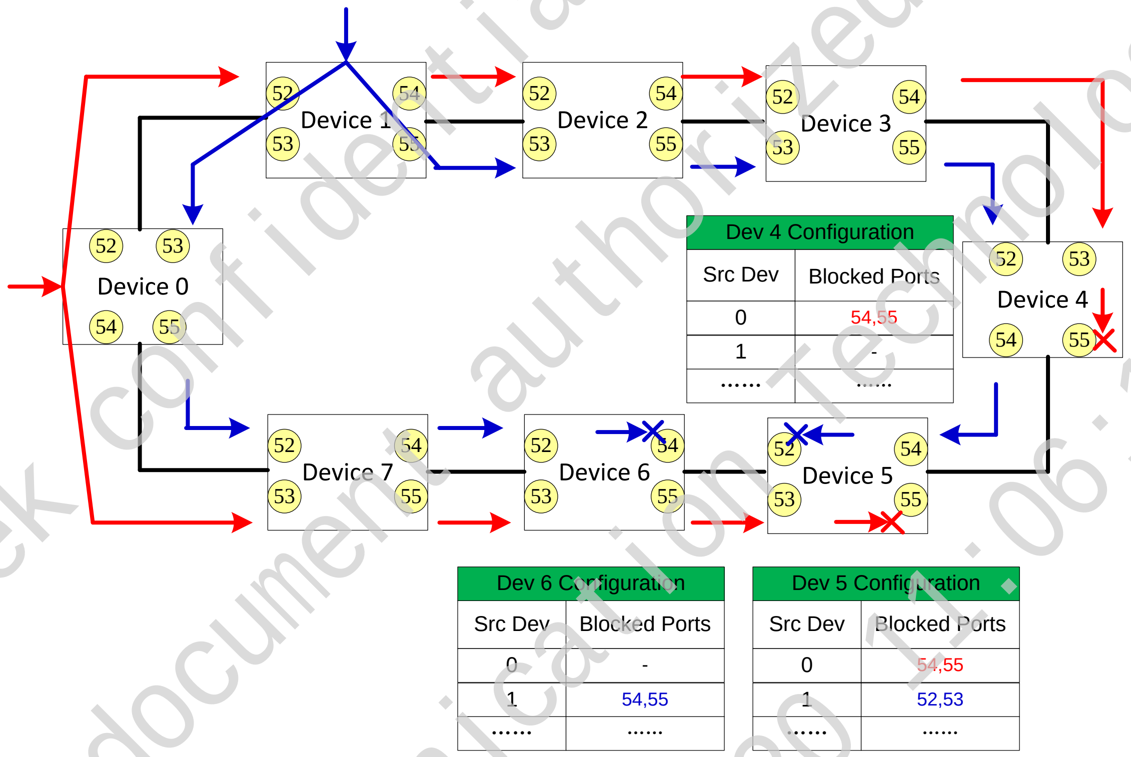
Table 15-6 STK_NON_UNICAST_BLOCK_CTRL Register

Bits	Field	Description
15:0	NON_UNICAST_BLOCK_PM	Blocking port mask of non-unicast packets. Value: 0b0: non-blocking port 0b1: blocked port

Take the broadcast traffic marked red in Figure 15-4 as an example, device 0 receives a broadcast packet and floods it to his left device and right device. Each device will do the same flooding operation, and finally the traffic will come back to device 0. Packet will be dropped by *DROP_MY_DEV* configuration to prevent traffic loop, but each device will receive the packet twice from his left and right device.

With *NON_UNICAST_BLOCK_PM* configuration in Figure 15-4, the red path between device 4 and device 5 and the blue path between device 5 and device 6 are cut off. The cut-off point should be in the middle of the ring, which can make packet reach each device in the shortest path.

Figure 15-4 Non-Unicast Traffic Block Diagram



Dev 4 Configuration	
Src Dev	Blocked Ports
0	54,55
1	-
……	……

Dev 6 Configuration	
Src Dev	Blocked Ports
0	-
1	54,55
……	……

Dev 5 Configuration	
Src Dev	Blocked Ports
0	54,55
1	52,53
……	……

15.4 Data Plane

In this chapter, we will introduce the data plane for different kind of traffic, including the pure L2 bridging packet and the L3 routing packet.

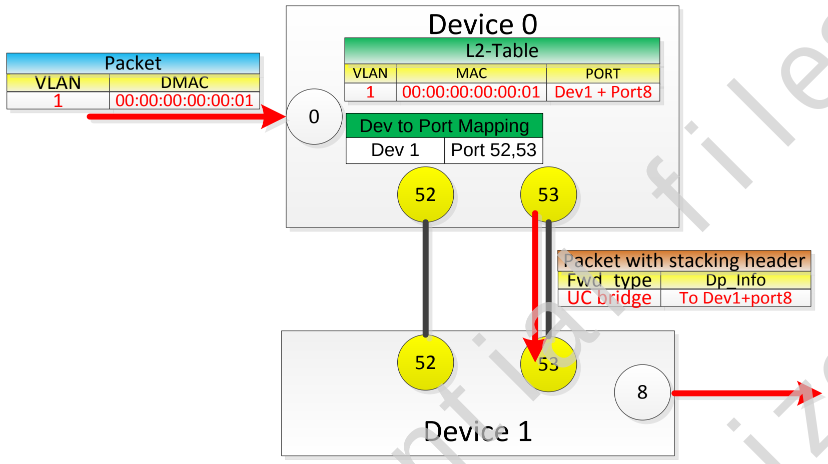
15.4.1 Unicast Bridging

15.4.1.1 Known Unicast

In the 1st device, unicast traffic will search the L2 Table with KEY <VID+DMAC> to determine destination <Device + Port>. If the destination <Device> is remote, the first device will convey the packet to the neighbor device through stacking ports, otherwise directly forwards to the local destination port.

If the neighbor device receives a known unicast packet from stacking port, this device will forward that packet straightly to its destination <Device + Port> without searching L2 Table again.

Figure 15-5 Known Unicast Data Plane Diagram



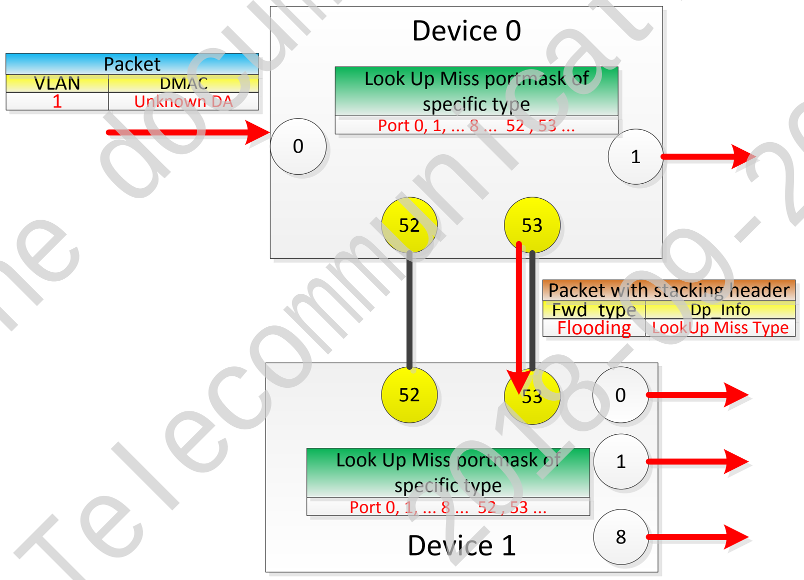
15.4.1.2 Unknown Unicast

In the 1st device, if L2 Table looks up missed, traffic is classified as unknown unicast and flooded into unicast look-up missed port-mask. If stacking port is included, traffic will go through the stacking port to the neighbor device and flooded into its unicast lookup miss port-mask if necessary.

If the neighbor device receives an unknown unicast packet from stacking port, this device will forward that packet straightly to the local unicast look-up missed port-mask.

By the way, the system provides a global unicast look-up missed port-mask. If you want to know more detailed information, please refer to L2 develop guide.

Figure 15-6 Unknown Unicast Data Plane Diagram


Device 0
Packet
VLAN
DMAC
1
Unknown DA
Look Up Miss portmask of specific type
Port 0, 1, ... 8 ... 52 , 53 ...
0
1
52
53
Packet with stacking header
Fwd_type
Dp_Info
Flooding
LookUp Miss Type
52
53
0
1
8
Look Up Miss portmask of specific type
Port 0, 1, ... 8 ... 52 , 53 ...
Device 1

Note

If you want to forward the unknown unicast packet to the neighbor device, the unknown unicast flooding port-mask must include the related stacking ports.

15.4.2 Multicast Bridging

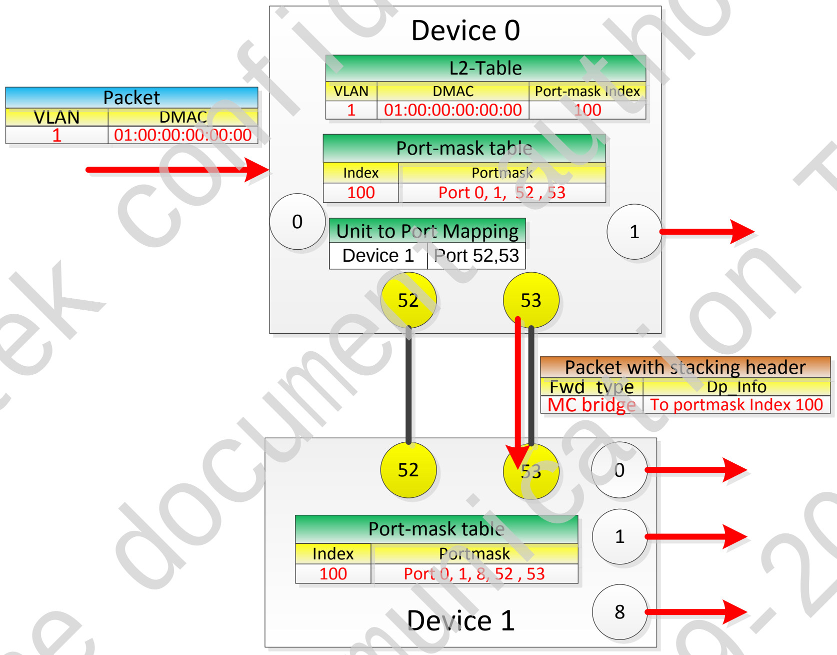
15.4.2.1 Known Multicast

In the 1st device, multicast traffic will search the Layer2 Address Table with KEY <VID+DMAC> or L3 Table with key<IP> to determine a <Index> pointing to an entry of port-mask Table , which contains a group of ports, and then the device will flood the multicast traffic to all the ports in these group.

If stacking port is included in the port group, traffic will go through the stacking port to the neighbor device.

And if the neighbor device receives a known multicast packet from stacking port, this device will search its port-mask Table with the same <index> and flood to all the ports in the group.

Figure 15-7 Known Multicast Data Plane Diagram



Note

If you want to forward the known multicast packet to the neighbor device, the multicast port-mask Table entry must include the related stacking ports.

15.4.2.2 Unknown Multicast

Unknown multicast data plane is similar with unknown unicast. If multicast lookup missed in the 1st device, traffic is classified as unknown multicast and flooded into L2/IPv4/IPv6 multicast lookup missed port-mask depending on the packet type (L2/IPv4/IPv6). If stacking port is included, traffic will go through the stacking port to the neighbor device and flooded into its L2/IPv4/IPv6 multicast lookup missed port-mask.

Note

If you want to forward the unknown multicast packet to the neighbor device, the unknown multicast flooding port-mask must include the related stacking ports depending on the packet type (L2/IPv4/IPv6).

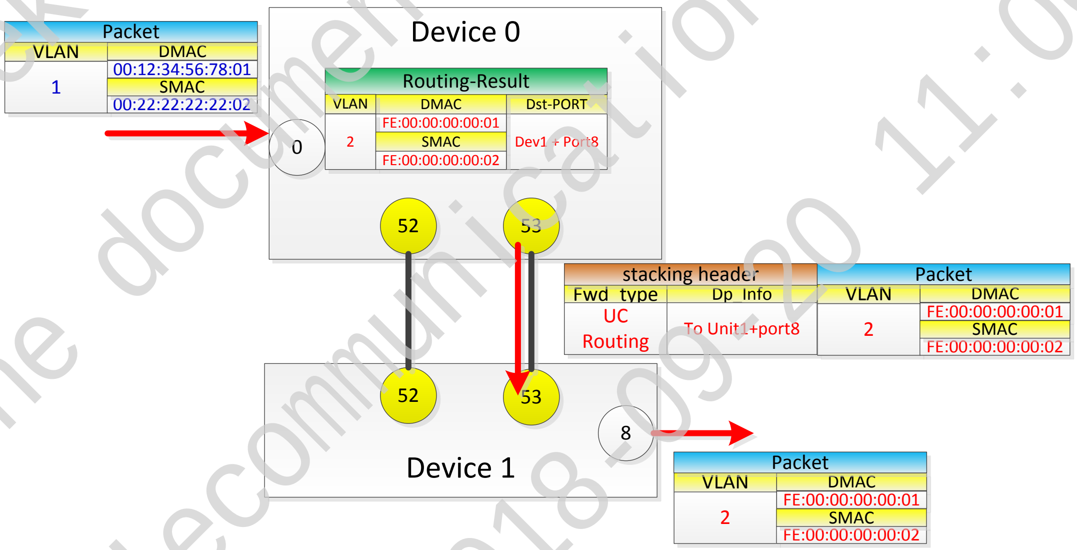
15.4.3 Broadcast Bridging

Broadcast traffic data plane is similar with unknown unicast/multicast, will be flooded in L2 broadcast flood port-mask instead. If stacking port is included, traffic will go through the stacking port to the neighbor device and flooded into its L2 broadcast flood port-mask.

15.4.4 Unicast Routing

Unicast routing process should be finished in the 1st device, include DMAC/SMAC replacement, interface ID change and TTL decrement. The other device will do nothing but forward the packet to its destination <Device + Port> after receiving the routed packet from the first device.

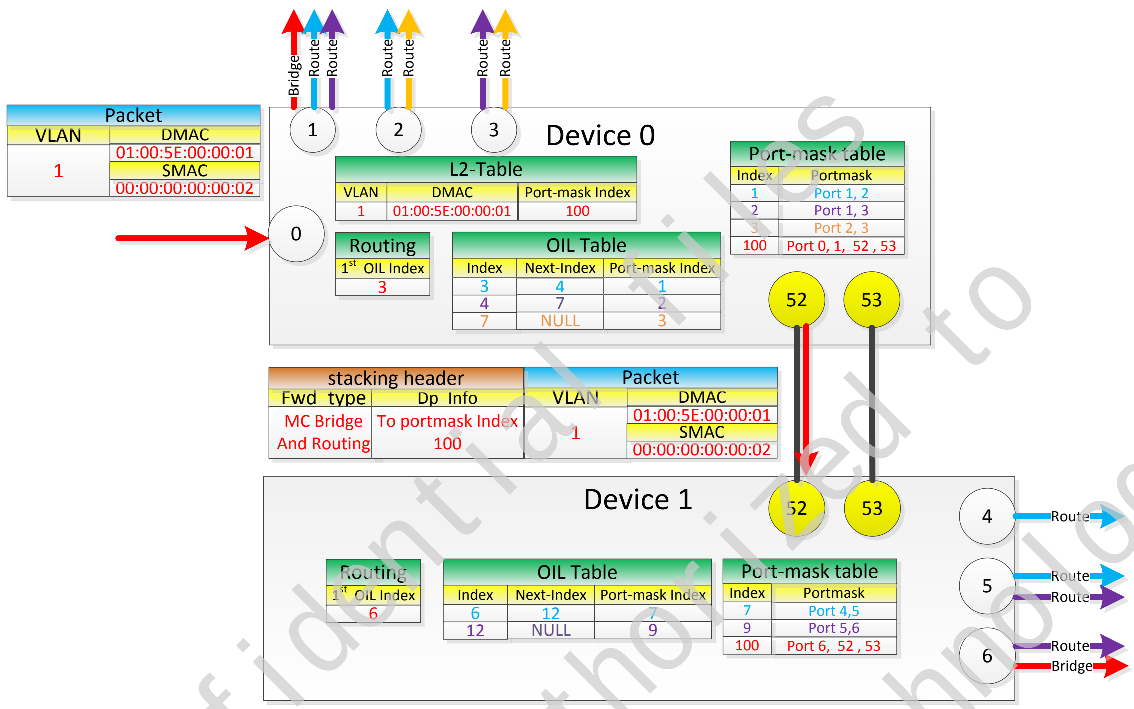
Figure 15-8 Unicast Routing Data Plane Diagram



15.4.5 Multicast Routing

Different from that unicast routing process is fulfilled in 1st device, multicast routing is processed distributing in each device. The first device does the multicast routing process and forwards a copy of original packet to the neighbor device at the same time if necessary. The neighbor device will do the same multicast routing process again and forwards a copy to the next device if necessary too.

Figure 15-9 Multicast Routing Data Plane Diagram



Note

If you want to route the multicast packet to the neighbor device, the port-mask field in routing entry should include stacking ports related. If you want to know more detailed information, please refer to L3 develop guide.

15.5 Control Plane

System provides several methods to allow different device communication and packet trap to CPU.

15.5.1 Device Talk

Each device in stacking system needs to communicate with others to exchange command and information, so they can cooperate as a single switch.

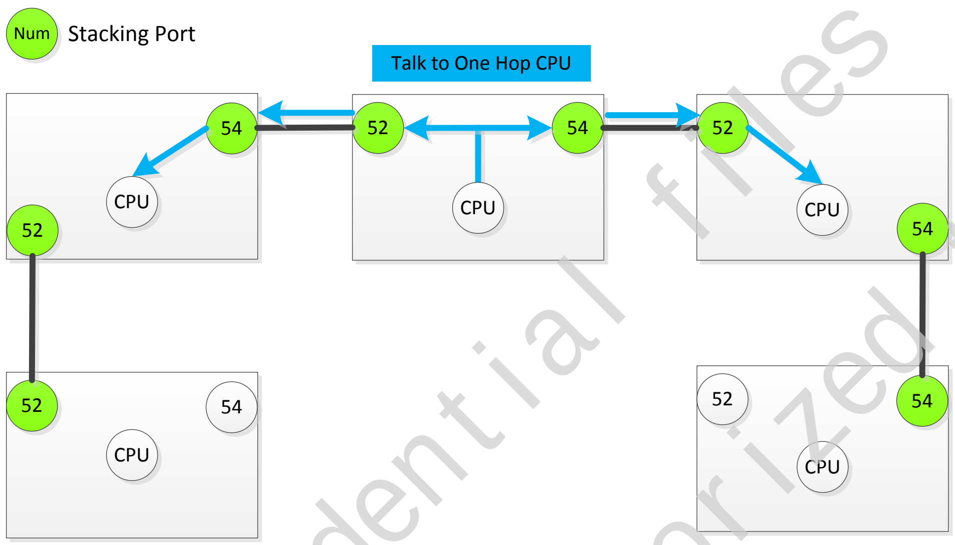
15.5.1.1 One-Hop to CPU

During the initial status, every device has the same device ID by default configuration, so device cannot be identified by device ID.

One-Hop packet is used in this initial status, because it travels only one hop and will not cause a traffic storm. With this packet, devices can communicate with its neighbors without any device ID. One-hop packet is very important, with its help, devices can communicate with its neighbors without any device

ID, exchange command and information to allocate unique device ID, select a Master role, make up the topology and finally build up a stacking system.

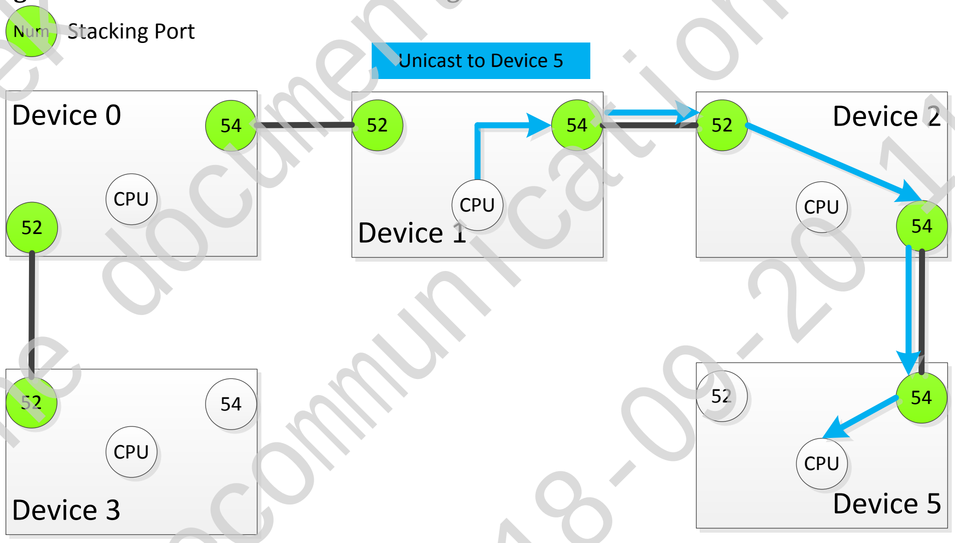
Figure 15-10 One-hop Packet Diagram



15.5.1.2 Unicast to CPU

After the unique device ID is allocated to each device, each CPU of device can communicate witch each other, by sending the corresponding packet with CPU Tag assigning the designated destination device (More detailed CPU Tag format information, please refer to CPU Tag develop guide).

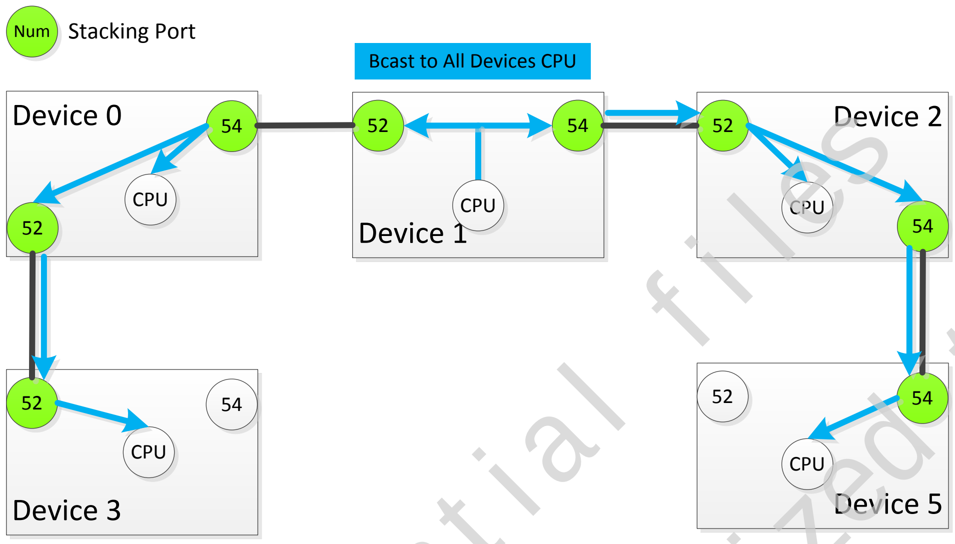
Figure 15-11 Unicast to CPU Packet Diagram



15.5.1.3 Broadcast to CPU

Like the mechanism of Unicast to CPU, if a message from one CPU must be sent to all device in the stacking system, Broadcast to CPU packet will be the most efficient way. What you need to do is just to set the right CPU Tag format of that message (More detailed CPU Tag format information, please refer to CPU Tag develop guide).

Figure 15-12 Broadcast to CPU Packet Diagram



15.5.2 Control Packet

Control packet can be sent to CPU by trap or copy to CPU method in standalone mode. But in a stacking mode, packets maybe need to be sent to the CPU of Master Device except trapped/copied to local CPU.

Each Trap/Copy configurations (RMA, ACL, etc.) will provide two extension actions for stacking system.

Trap to Master CPU	Trap packet to the CPU of Master Device
Copy to Master CPU	Copy packet to the CPU of Master Device

15.6 Stacking Port QoS

Each 10G stacking port provides 12 queues for more complex and flexible QoS application, instead of 8 queues for normal port. For other stacking port, 8 queues are provided as normal ports.

15.6.1 Data Plane

The 1st device will decide an internal priority for each packet, based on the packet's own characteristics like dot1p priority or DSCP, or even assigned directly by ACL.

The internal priority will be carried with packet to the next device. Each device uses the internal priority to decide the egress queue ID by a priority to queue mapping table.

In data plane, only queues 0 to 7 of stacking port are utilized.

Table 15-7 QM_INTPRI2QID_CTR Register

Bits	Field	Description
2:0	QID	Assign normal port queue ID(0~7) to specified internal-priority.

15.6.2 Control Plane

The packet transmitted by CPU software can be assigned a queue ID ranged from 0 to 31. If the packet is destined to normal port or non-10G stacking port, because normal port has only 8 queues, it needs a 32 queue to 8 queue mapping to decide the really egress queue ID before transmitting. Similarly, if the packet is destined to 10G stacking port, because 10G stacking port has only 12queues, it needs a 32 queue to 12 queues mapping to decide the real queue ID before transmitting too.

Besides that, packet trapped to CPU will also need to do the similar queue mapping operation. Based on its trap reason, a queue ID ranged from 0 to 31 will be assigned to this packet. If the packet is trapped to CPU of master device, it should go through the stacking port, which also needs the 31 queue to 12 queue mapping to get the really queue ID of stacking port.

The related mapping tables are as follows.

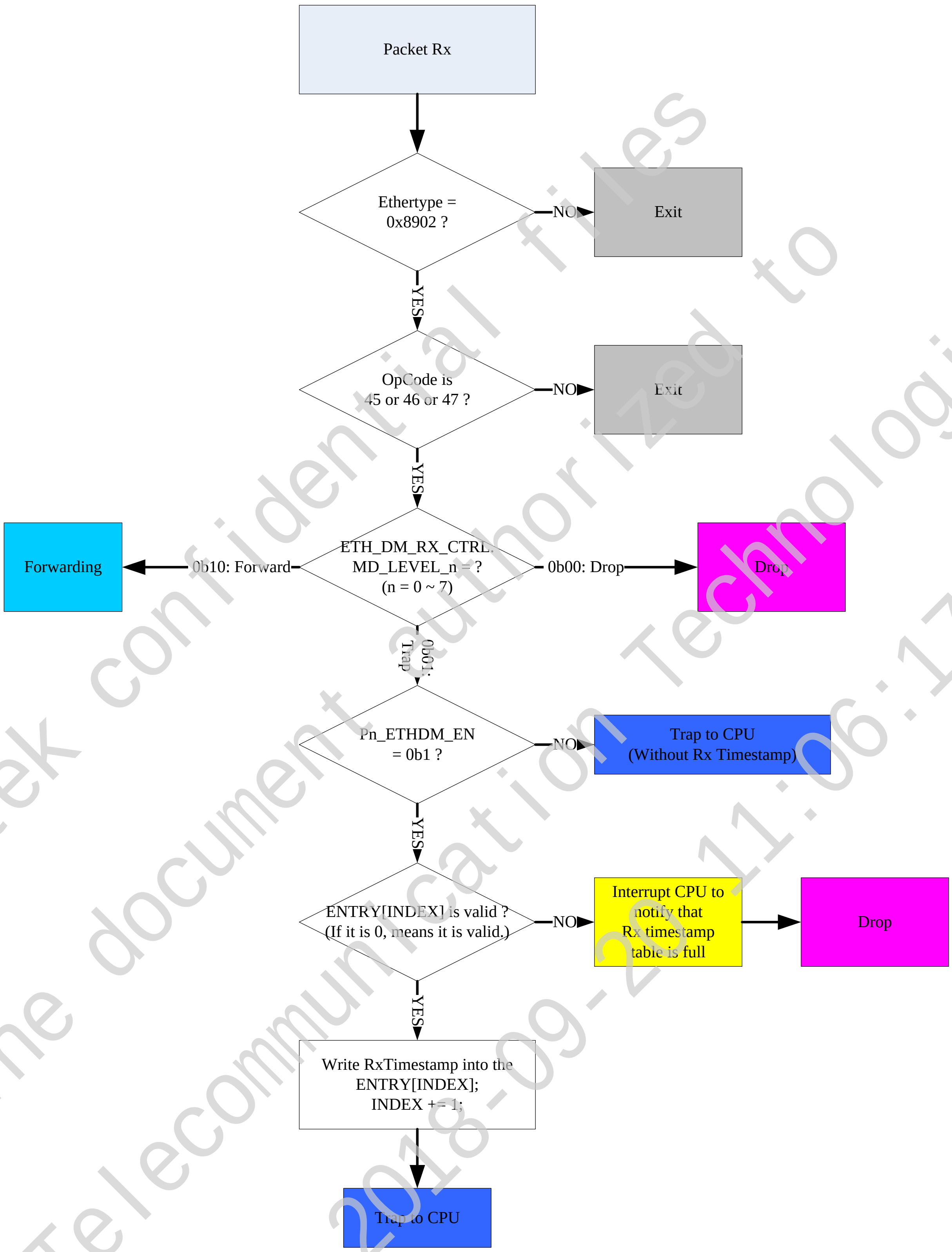
Table 15-8 QM_CPUQID2QID_CTRL Register

Bits	Field	Description
2:0	QID	Assign normal port queue ID(0~7) to specified CPU queue ID(0~31).

Table 15-9 QM_CPUQID2XGQID_CTRL Register

Bits	Field	Description
3:0	QID	Assign XG port queue ID(0~11) to specified CPU queue ID(0~31). 0x0~0xB: queue ID 0~11. 0xC~0xF: invalid value.

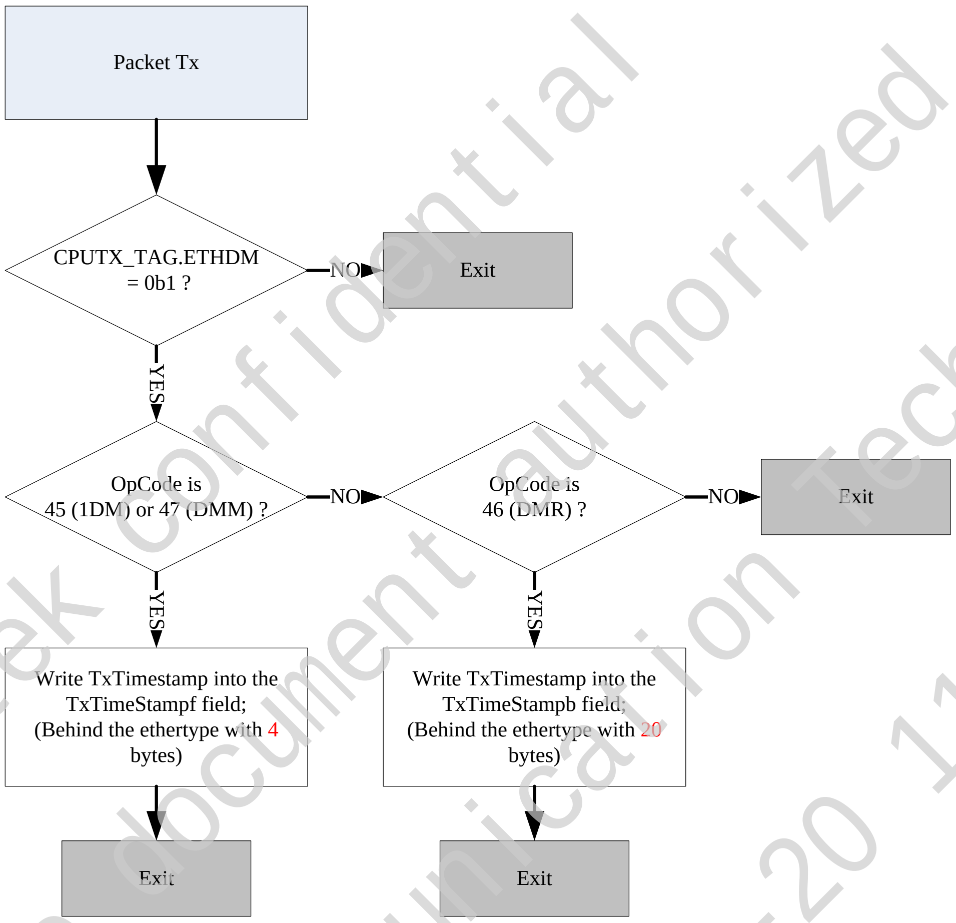
Figure 15-13 ETH-DM RX Timestamping Flow


Packet Rx
Ethertype = 0x8902 ?
NO
Exit
YES
OpCode is 45 or 46 or 47 ?
NO
Exit
YES
ETH_DM_RX_CTRL.MD_LEVEL_n = ? (n = 0 ~ 7)
0b10: Forward
Forwarding
0b00: Drop
Drop
0b01: Trap
Pn_ETHDM_EN = 0b1 ?
NO
Trap to CPU (Without Rx Timestamp)
YES
ENTRY[INDEX] is valid ? (If it is 0, means it is valid.)
NO
Interrupt CPU to notify that Rx timestamp table is full
Drop
YES
Write RxTimestamp into the ENTRY[INDEX]; INDEX += 1;
Trap to CPU

15.6.3 TX Timestamp Writing

For accurate TX timestamping, system supports writing TX timestamp to 1DM/DMM/ DMR by hardware on the fly. To trigger hardware TX timestamp writing, software should set the DM_PKT field of CPU TX tag of the frame to 1. Hardware would base on the opcode in frame to decide the TX timestamp offset. VLAN tagged and untag frames are supported. Figure 15-14 illustrates the TX timestamping workflow.

Figure 15-14 DMR PDU Format


Packet Tx
CPUTX_TAG.ETHDM = 0b1 ?
NO
Exit
YES
OpCode is 45 (1DM) or 47 (DMM) ?
NO
OpCode is 46 (DMR) ?
NO
Exit
YES
YES
Write TxTimestamp into the TxTimeStampf field; (Behind the ethertype with 4 bytes)
Write TxTimestamp into the TxTimeStampb field; (Behind the ethertype with 20 bytes)
Exit
Exit

16 OpenFlow

OpenFlow is a protocol that allows a controller to tell network switches how to forward packets. In a conventional switch, packet forwarding (the data path) and high-level routing (the control path) occur on the same device. An OpenFlow switch separates the data path from the control path. The data path portion resides on the switch itself; a separate controller makes high-level routing decisions. The switch and controller communicate through the OpenFlow protocol.

16.1 Feature Overview

The OpenFlow features supported by the system are listed below.

- OpenFlow v.1.5.1 Compliant
- OpenFlow Hybrid Switch
- Four Ingress Flow Tables and One Egress Flow Table
  - Ingress Full Match Flow Table 0 (shared 4K PIE entries)
  - Ingress L2 Optimized Flow Table 1 (16K entries)
  - Ingress L3 Optimized Flow Table 2 (12K entries)
  - Ingress Full Match Flow Table 3 (shared 4K PIE entries)
  - Egress Full Match Flow Table 0 (shared 4K PIE entries)
- Ingress Flow Table stage can be extended through loopback operation
- Group table support with 2048 group entries and 8192 action buckets
- Meter table support with 512 dual band meters

16.2 Flow Table Capability Summary

Below table summarizes the capability of each flow table for a quick reference. For the detail functionality description, please refer to the corresponding chapters.

Table 16-1 Flow Table Capability Comparison Table

Feature	Sub-Feature	Ingress Full Match Flow Table 0	Ingress L2 Flow Table 1	Ingress L3 Flow Table 2	Ingress Full Match Flow Table 3	Egress Full Match Flow Table 0
Table Size		4K(*)	16K	12K(Refer 16.6.6)	4K(*)	4K(*)
Match Field		L2~L4 header	(VID/0,SA,MD) (VID/0,DA,MD) (SA,DA,MD) (SIP,DIP,L4SPORT, L4DPORT,IPPRO/ VID)	(SIP,MD) (DIP,MD) (SIP,DIP,MD)	L2~L4 header	L2~L4 header
Priority		PIE block-based	N/A	N/A	PIE block-based	PIE block-based
Counter		packet-based byte-based	N/A	N/A	packet-based byte-based	packet-based byte-based

Instruction	Meter	512(*)	N/A	N/A	512(*)	512(*)
	Apply Action	N/A	N/A	N/A	N/A	N/A
	Clear Action	V	V	V	V	V
	Write Action	V	V	V	V	V
	Write Metadata	V(12-bit)	V(12-bit)	V(12-bit)	V(12-bit)	N/A
	Goto Table	V	V	V	V	N/A
Hit Indication		V	V	V	V	V
Action	Copy TLL Inward	V			V	
	Pop VLAN	V			V	V
	Pop MPLS	V			V	
	Push MPLS	V			V	
	Push VLAN	V	V	V	V	V
	Copy TTL Outward	V			V	
	Dec MPLS TTL	V			V	
	Dec IP TTL	V		V	V	V
	Num of Set Field	5	2	3	5	2
	Set Field – SMAC/DMAC	V		V	V	
	Set Field – VLAN ID/Pri.	V	V	V	V	V
	Set Field – IP DSCP	V	V	V	V	V
	Set Field – IP TTL	V			V	
	Set Field – IPv4 SIP/DIP	V			V	
	Set Field – L4 SPort/DPort	V			V	
	Set Field – MPLS label	V			V	
	Set Queue	V	V	V	V	V
	Group	V		V	V	V
	Output	V	V	V	V	V

Note

Symbol (*) means the resources are shared among flow tables.

16.3 Pipeline Processing

OpenFlow pipeline processing is divided into two stages: Ingress Processing and Egress Processing. The packet is processed by ingress pipeline firstly and is forwarded to the egress pipeline when the output port is determined by executing the Action Set.

16.3.1 Hybrid Switch Pipeline Processing

OpenFlow-hybrid switch supports OpenFlow operations and traditional Ethernet switching operations (i.e. traditional L2 Ethernet switching, VLAN isolation, L3 routing, and QoS processing) simultaneously.

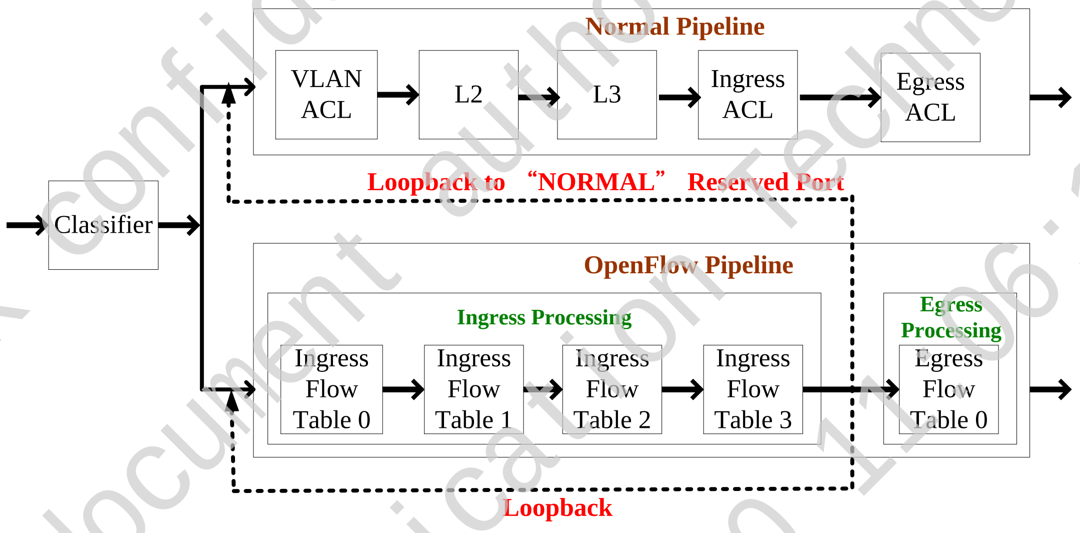
The system supports OpenFlow-hybrid switch which embeds two pipelines: OpenFlow pipeline and Normal pipeline. A classification mechanism is provided to dispatch the packet to the corresponding pipeline.

In addition to dispatch packet to one of the pipelines, system supports the packet first processed by OpenFlow pipeline and then loopback to normal pipeline ("NORMAL" reserved port in OpenFlow spec) or OpenFlow pipeline again.

Note

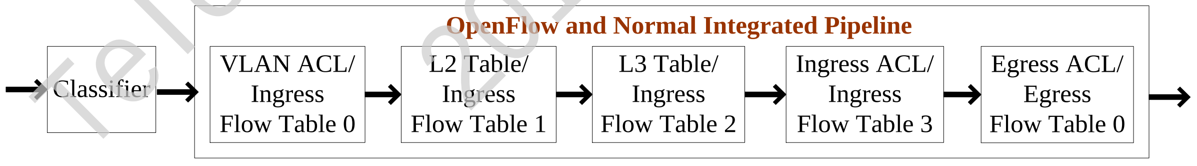
If packet is first processed by normal pipeline, it can be loopback to normal pipeline only. Extend ingress flow table stages can be done by loopback packet to OpenFlow pipeline itself.

Figure 16-1 OpenFlow-hybrid Switch Packet Processing Flow



OpenFlow and normal pipelines are logically separate. The physical resources are shared by two pipelines because a packet can only traverse a single pipeline for a given time. The resources sharing are VLAN ACL/Ingress Full Match Flow Table 0, L2 Table/Ingress L2 Optimized Flow Table 1, L3 Tables/Ingress L3 Optimized Flow Table 2, Ingress ACL/Ingress Full Match Flow Table 2, and Egress ACL/Egress Full Match Flow Table 0.

Figure 16-2 OpenFlow And Normal Integrated Pipeline


OpenFlow and Normal Integrated Pipeline
Classifier
VLAN ACL/ Ingress Flow Table 0
L2 Table/ Ingress Flow Table 1
L3 Table/ Ingress Flow Table 2
Ingress ACL/ Ingress Flow Table 3
Egress ACL/ Egress Flow Table 0

16.3.2 Classification

A packet is sent to either OpenFlow or Normal pipeline according to the classification mechanism. The system supports the classification mechanism by ingress port, VLAN, ingress port and VLAN.

16.3.2.1 Ingress Port

The system per ingress port provides below configuration.

Table 16-2 OF_PORT_CTRL Register

Bits	Field	Description
0:0	EN	Pipeline classification by port. Packet from OpenFlow enabled port is sent to OpenFlow Pipeline. 0b0: disabled 0b1: enabled.

16.3.2.2 VLAN

The system per VLAN provides below configuration. The VLAN is from the outermost VLAN for tagged packet and from port-based VLAN for untagged packet.

Table 16-3 OF_VLAN_EN Register

Bits	Field	Description
0:0	EN	Enable VLAN for OpenFlow processing. 0b0: disabled 0b1: enabled.

16.3.2.3 Ingress Port and VLAN

The system per VLAN provides below configuration. When the configuration is enabled, the packet is sent to OpenFlow pipeline when both *OF_PORT_CTRL.EN* and *OF_VLAN_EN.EN* corresponding to the packet are enabled. When the configuration is disabled, the packet is sent to OpenFlow pipeline when *OF_PORT_CTRL.EN* or *OF_VLAN_EN.EN* corresponding to the packet is enabled.

Table 16-4 OF_VLAN_AND_PORT_EN Register

Bits	Field	Description
0:0	EN	Enable qualifying VLAN and port for OpenFlow processing. 0b0: disabled 0b1: enabled

16.4 Full Match Flow Table

The system supports three stages Full Match Flow Tables which are Ingress Flow Table 0, Ingress Flow Table 3 and Egress Flow Table 0. These tables share the PIE resources with ACL. Per PIE block can be configured as Flow Table or ACL and can have its template mapping. Please refer to "PIE" chapter for the PIE configuration and description.

16.4.1 Full Match Flow Entry

A full match flow entry is composed of below fields.

Match Field	To match against packets including packet headers, ingress port, metadata…
Operation	REVERSE, AGG_IN_BLK, AGG_CROSS_BLK operations.
Counter	A 36-bit packet counter and a 42-bit byte counter.
Instruction	Instructions to be executed when the flow entry hit.
Hit Indication	Set when the flow entry hit.

16.4.2 Match Field

The match field is supported by PIE FCS (Field Content Select). For the PIE rule structure and Fixed Field definition, please refer to the "PIE" chapter. The Field Content Select supported by the OpenFlow are different from ACL and are listed below. We use IFT0, IFT3 and EFT0 as the abbreviation in below table for Ingress Flow Table 0, Ingress Flow Table 3 and Egress Flow Table 0 respectively.

Table 16-5 Field Content Select for the Full Match Flow Tables

Name	Bits of value	Bits Allocation	IFT0	IFT3	EFT0
SPM0	Source Physical Port Mask[15:0]	F10	O	O	
SPM1	Source Physical Port Mask[31:16]	F11	O	O	
SPM2	Source Physical Port Mask[47:32]	F12	O	O	
SPM3	[15] Loopback packet [14:9] Reserved [8:0] Source Physical Port Mask[56:48]	F13	O	O	
DPM0	Destnation Physical Port Mask[15:0]	F10			O
DPM1	Destnation Physical Port Mask[31:16]	F11			O
DPM2	Destnation Physical Port Mask[47:32]	F12			O
DPM3	[15:9] Reserved [8:0] Destination Physical Port Mask[56:48]	F13			O
DMAC0	DMAC[15:0]	F0/F3/F6/F9	O	O	O
DMAC1	DMAC[31:16]	F1/F4/F7/F10	O	O	O
DMAC2	DMAC[47:32]	F2/F5/F8/F11	O	O	O
SMAC0	SMAC[15:0]	F0/F3/F6/F9	O	O	O
SMAC1	SMAC[31:16]	F1/F4/F7/F10	O	O	O
SMAC2	SMAC[47:32]	F2/F5/F8/F11	O	O	O
ETHERTYPE	EtherType		O	O	O

OTAG	[15:13] OPRI [12] DEI [11:0] OVID		O	O	O
ITAG	[15:13] IPRI [12] CFI [11:0] IVID		O	O	O
SIP0	For IPv4/IPv6, SIP[15:0]. For ARP/RARP, sender protocol address.	F0/F2/F4/F6/F8/F10/F12	O	O	O
SIP1	For IPv4/IPv6, SIP[31:16]. For ARP/RARP, sender protocol address.	F1/F3/F5/F7/F9/F11/F13	O	O	O
DIP0	For IPv4/IPv6, DIP[15:0]. For ARP/RARP, target protocol address.	F0/F2/F4/F6/F8/F10/F12	O	O	O
DIP1	For IPv4/IPv6, DIP[31:16]. For ARP/RARP, target protocol address.	F1/F3/F5/F7/F9/F11/F13	O	O	O
IPTOSPROTO	For IPv4/IPv6, [15:8] IPv4 TOS/IPv6 Traffic Class [7:0] IPv4 Protocol/IPv6 Next Header For ARP/RARP, ARP/RARP Opcode. For Ethernet and LLC-SNAP packet other than IPv4/IPv6/ARP/RARP/OAM/PPPoE, Byte 0 and 1 after EtherType field.		O	O	O
L4SPORT	For ARP/RARP, sender hardware address [15:0]. For ICMP/IGMP, [15:8] ICMP Type/ICMPv6 Type/IGMP Type [7:0] ICMP Code/ICMPv6 Code/IGMP MAX Resp Code For TCP/UDP/SCTP, source port number. For IP packet exclude TCP/UDP/SCTP, L4 header byte 0 and 1.	F0/F2/F4/F6/F8/F10/F12	O	O	O
L4DPORT	ARP/RARP packet, sender hardware address [31:16]. For TCP/UDP/SCTP, destination port number. For IP packet exclude TCP/UDP/SCTP, L4 header byte 2 and 3.	F1/F3/F5/F7/F9/F11/F12	O	O	O

L34HEADER	[15] UNSEQ (Hop-by-Hop header position error) [14] UNREP (Unexpected repeats extesnsion header) [13] NONEXT (Ipv6 packet with No Next header) [12] IPv6 packet with Mobility header [11] IPv6 packet with ESP header [10] IPv6 packet with authentication header [9] IPv6 packet with destination option header [8] IPv6 packet with fragment header [7] IPv6 packet with routing header [6] IPv6 packet with hop-by-hop option header [5:3] IPv4 Flags/IPv6 Rsvd+M-Flag [2] IP fragment packet [1:0] IPv4 TTL/IPv6 Hop Limit 2b'00: TTL = 0 2b'01: TTL = 1 2b'10: 2<=TTL<255 2b'11: TTL = 255		O	O	O
	For ARP/RARP packet, target hardware address [15:0]				
TCPINFO	[15:13] Reserved [12:9] flow label[19:16] [8] TCP Non Zero Sequence [7:6] TCP ECN([7]: ECE, [6]: CWR) [5:0] TCP Flag([5]: URG, [0]: FIN)	F1/F3/F5/F7/F9	O	O	O
	For ARP/RARP packet, sender hardware address [47:32]				
FIELD_SELECTOR_VALID	[15:14] Reserved [13:0] Field Selector Valid Mask		O	O	
FIELD_SELECTOR0	User defined 16-bit field selector 0		O	O	
FIELD_SELECTOR1	User defined 16-bit field selector 1		O	O	
FIELD_SELECTOR2	User defined 16-bit field selector 2		O	O	
FIELD_SELECTOR3	User defined 16-bit field selector 3		O	O	
FIELD_SELECTOR4	User defined 16-bit field selector 4		O	O	
FIELD_SELECTOR5	User defined 16-bit field selector 5		O	O	
FIELD_SELECTOR6/ SIP2	User defined 16-bit field selector 6	F2	O	O	
	For IPv6, IPv6 SIP[47:32]		O	O	O
FIELD_SELECTOR7/ SIP3	User defined 16-bit field selector 7	F3	O	O	
	For IPv6, IPv6 SIP[63:48]		O	O	O
FIELD_SELECTOR8/ SIP4	User defined 16-bit field selector 8	F4	O	O	
	For IPv6, IPv6 SIP[79:64]		O	O	O
FIELD_SELECTOR9/	User defined 16-bit field selector 9	F5	O	O	

SIP5	For IPv6, IPv6 SIP[95:80]		O	O	O
FIELD_SELECTOR10/ SIP6	User defined 16-bit field selector 10	F6	O	O	
	For IPv6, IPv6 SIP[111:96]		O	O	O
FIELD_SELECTOR11/ SIP7	User defined 16-bit field selector 11	F7	O	O	
	For IPv6, IPv6 SIP[127:112]		O	O	O
FIELD_SELECTOR12	User defined 16-bit field selector 12	F8	O	O	
FIELD_SELECTOR13	User defined 16-bit field selector 13	F9	O	O	
DIP2	For IPv6, IPv6 DIP[47:32]	F2	O	O	O
	For ARP/RARP packet, target hardware address [47:32]				
DIP3	IPv6 DIP[63:48]	F3	O	O	O
DIP4	IPv6 DIP[79:64]	F4	O	O	O
DIP5	IPv6 DIP[95:80]	F5	O	O	O
DIP6	IPv6 DIP[111:96]	F6	O	O	O
DIP7	IPv6 DIP[127:112]	F7	O	O	O
PKT_INFO	[15:10] Source Physical Port [9:8]: DMAC Type 2b'00: Unicast 2b'01: Broadcast 2b'10: Reserved 2b'11: Multicast [7:0] Reserved		O	O	
FLOW LABEL	For IPv6, flow label[15:0]	F0/F2/F4/F6/F8	O	O	O
	For ARP/RARP packet, target hardware address [31:16]				
DSAP_SSAP	For LLC/SNAP packet, [15:8]: DSAP [7:0]: SSAP	F0/F2/F4/F6/F8	O	O	
	For GRE packet, GRE key[15:0]				
	For MPLS packet, second MPLS label [15:0]				
	For GTP packet, TEID[15:0]				
SNAP OUI	For LLC-SNAP, OUI[15:0]	F1/F3/F5/F7/F9	O	O	
	For GRE packet, GRE key [31:16]				
	For MPLS packet, [15:8] Reserved [7] second MPLS BoS bit [6:4] second MPLS TC [3:0] second MPLS label [19:16]				
	For GTP packet, TEID[31:16]				

RANGECHK	[15:0]: VID/L4 port/Packet Length/L3 Packet Length Range Check Mask *Note: ETF0 doesn't support packet length and L3 packet length*		O	O	O
SLP	[15:11] reserved [10] ingress port is a trunk port When ingress port is a trunk port, [9:7] Reserved [6:0] trunk ID Otherwise, [9:6] device ID [5:0] ingress port ID		O	O	O
DLP	[15:12] OUTPUT_TYPE in Action Set [11:0] OUTPUT_DATA in Action Set		O	O	O
META_DATA	[15:12] Target Loopback Time [11:0] META_DATA		O	O	O
TTID	[15] second MPLS label exist [14] first MPLS label exist [13:0] reserved	F0/F2/F4/F6/F8/F10/F12	O	O	
FIRST_MPLS1	For MPLS, first MPLS label[15:0]	F0/F2/F4/F6/F8/F10/F12	O	O	
	For VXLAN, VNI[15:0]				
FIRST_MPLS2	[15:8] IP range check	F1/F3/F5/F7/F9/F11/F13	O	O	
	For MPLS, [7] first MPLS BoS bit [6:4] first MPLS TC [3:0] first MPLS label[19:16]				
	For VXLAN, [7:0] VNI[23:16]				

16.4.3 Operation

Refer to the "PIE" chapter for the Rule Operation description.

16.4.4 Counter

Each full match flow entry supports a counter mode configuration, a 36-bit packet counter and a 42-bit byte counter.

CNTR_MODE	Counter mode which defines the counter usage. 0x0: packet counter and byte counter 0x1: packet counter and packet counter trigger threshold 0x2: byte counter and byte counter trigger threshold 0x3: disable
PKT_CNT/BYTE_CNTR_TH	Packet counter when CNTR_MODE=0x0 or 0x1. Byte counter trigger threshold when CNTR_MODE=0x2.
BYTE_CNT/PKT_CNTR_TH	Byte counter when CNTR_MODE=0x0 or 0x2. Packet counter trigger threshold when CNTR_MODE=0x1.

Counter mode 0x1 and 0x2 are used to support the STAT-TRIGGER instruction defined in OpenFlow v1.5.x.
When CNTR_MODE=0x1, PKT_CNT is a normal packet counter and PKT_CNT_TH is the interrupt trigger threshold for PKT_CNT. An interrupt is triggered when PKT_CNT == PKT_CNT_TH.
When CNTR_MODE=0x2, BYTE_CNT is a normal byte counter and BYTE_CNT_TH is the interrupt trigger threshold for BYTE_CNT. An interrupt is triggered when BYTE_CNT >= BYTE_CNT_TH for the first time.

Note

For the packet hit multiple flow entries concurrently, only the highest priority (selected) entry will update the counters.

16.4.5 Ingress Flow Table Instructions

Each flow entry can specify the instructions to be executed immediately when the flow entry hit. Instructions include packet modification, set metadata, goes to next flow table and so on.
Each flow entry of ingress flow table 0 and 3 supports below instruction masks to specify the instructions to be executed. The instructions are executed immediately by the listed order when the flow entry hit.

- Meter
- Clear Action Set
- Write Action Set
- Write Metadata
- Goto Table

Each instruction has specific parameters which are listed in below table. For the detail instruction and action description, please refer to

Note

For SET_FIELD action, all the modification is applied to the outermost possible header.

Instructions and Actions.

Table 16-6 Ingress Flow Table Instruction Parameters

Fields	Description
METER_IDX	Meter index.
METER_YELLOW_ACT	Take the action if packet is colored to Yellow. 0b0: drop 0b1: DSCP remark
METER_RED_ACT	Take the action if packet is colored to Red. 0b0: drop 0b1: DSCP remark

WA_COPY_TTL_INWARD	Apply copy TTL inward to the packet.
WA_POP_VLAN	Pop the outermost VLAN tag from the packet.
WA_POP_MPLS	Pop MPLS label from the packet. 0b0: disable 0b1: enable
WA_POP_MPLS_TYPE	Pop MPLS label type. 0b0: pop outermost label 0b1: pop double labels
WA_PUSH_MPLS	Push MPLS label to the packet. 0b0: disable 0b1: enable All the WA_PUSH_MPLS_XXX fields following are used only if WA_PUSH_MPLS=enable.
WA_PUSH_MPLS_MODE	0b0: push MPLS label to the packet 0b1: push/swap MPLS label according to WA_MPLS_LIB_IDX
WA_PUSH_MPLS_VPN_TYPE	0b0: L3 VPN 0b1: L2 VPN
WA_PUSH_MPLS_LIB_IDX	Index to MPLS encapsulation table for retrieving label related info. The field is used when WA_PUSH_MPLS_MODE=0b1.
WA_PUSH_MPLS_ETHTYPE	0b0: 0x8847 0b1: 0x8848
WA_PUSH_VLAN	Push VLAN tag to the packet.
WA_PUSH_VLAN_ETHTYPE	Index to outer TPID pool (*VLAN_TAG_TPID_CTRL.OTPID*).
WA_COPY_TTL_OUTWARD	Apply copy TTL outward to the packet.
WA_DEC_MPLS_TTL	Decrement the outermost MPLS TTL from the packet.
WA_DEC_IP_TTL	Decrement the outermost IPv4 TTL/IPv6 Hop Limit from the packet.
WA_SET_FIELD0_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Source MAC 0x2: VLAN Priority 0x3: MPLS Traffic Class 0x4: MPLS TTL 0x5: IPv4/IPv6 DSCP 0x6: IPv4 TTL/IPv6 Hop Limit 0x7: IP Flag Reserved Bit
WA_SET_FIELD0_DATA	Field Data 0. When WA _SET_FIELD0_TYPE = 0x1: index to Egress L3 INTF Table for retrieving SMAC 0x2: VLAN Priority 0x3: MPLS Traffic Class 0x4: MPLS TTL 0x5: IPv4/IPv6 DSCP 0x6: IPv4 TTL/IPv6 Hop Limit

WA_SET_FIELD1_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Destination MAC 0x2: VLAN Priority 0x3: MPLS Label 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
WA_SET_FIELD1_DATA	Field Data 1. When WA _SET_FIELD0_TYPE = 0x1: index to OpenFlow DMAC table 0x2: VLAN Priority 0x3: index to MPLS ENCAP. table 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
WA_SET_FIELD2_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: VLAN ID 0x2: VLAN Priority 0x3: MPLS Label 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
WA_SET_FIELD2_DATA	Field Data 2. When WA _SET_FIELD0_TYPE = 0x1: VLAN ID 0x2: VLAN Priority 0x3: index to MPLS ENCAP. table 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
WA_SET_FIELD3_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Source IPv4 Address 0x2: Destination IPv4 Address 0x3: VLAN Priority 0x4: MPLS Label 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP
WA_SET_FIELD3_DATA	Field Data 3. When WA _SET_FIELD3_TYPE = 0x1: Source IPv4 Address 0x2: Destination IPv4 Address 0x3: VLAN Priority 0x4: index to MPLS ENCAP. table 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP

WA_SET_FIELD4_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: L4 Source Port 0x2: L4 Destination Port 0x3: VLAN Priority 0x4: MPLS Label 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP
WA_SET_FIELD4_DATA	Field Data 4. When WA _SET_FIELD4_TYPE = 0x1: TCP/UDP Source Port 0x2: TCP/UDP Destination Port 0x3: VLAN Priority 0x4: index to MPLS ENCAP. table 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP
WA_SET_Q	Set Queue ID for the packet.
WA_SET_Q_DATA	Queue ID.
WA_GROUP	Apply group action to the packet.
WA_GROUP_IDX	Index to Group table for retrieving the group entry.
WA_OUTPUT_TYPE	Output type. 0x0: disable 0x1: physical port with source port filter 0x2: physical port without source port filter 0x3: trunk port with source port filter 0x4: trunk port without source port filter 0x5: multiple egress ports with source port and trunk filter 0x6: packet's ingress physical port 0x7: flood (all ports with source port, trunk, VLAN, Spanning Tree filtering) 0x8: loopback to normal pipeline 0x9: tunnel interface 0xA: failover (first link up port)
WA_OUTPUT_DATA	When WA_OUTPUT_TYPE = 0x1: device ID and physical egress port 0x2: device ID and physical egress port 0x3: trunk ID 0x4: trunk ID 0x5: index to port-mask table 0x9: index to tunnel start table 0xA: index to port-mask table
METADATA	12-bit Metadata.
METADATA_MASK	12-bit Metadata mask.
GOTO_TBL_ACT	0x0: general Goto 0x1: apply current Action Set and loopback packet to OpenFlow pipeline again 0x2: don't apply current Action Set and loopback packet to OpenFlow pipeline again 0x3: reserved

GOTO_TBL_ID	GOTO_TBL_ACT=0, the next table ID.
	GOTO_TBL_ACT=1, the next table ID after loopback.
	GOTO_TBL_ACT=2, the next table ID after loopback.

GOTO_TBL_LB_TIME	Target loopback time. Flow entry can filter "target loopback time" to emulate the flow table stage it resided.

Note

For SET_FIELD action, all the modification is applied to the outermost possible header.

16.4.6 Egress Flow Table Instructions

Each flow entry of egress flow table 0 supports below instruction masks to specify the instructions to be executed. The instructions are executed immediately by the listed order when the flow entry hit.

- Meter
- Write Action Set

Each instruction has specific parameters which are listed in below table. For the detail instruction and action description, please refer to Instructions and Action

Note

For SET_FIELD action, all the modification is applied to the outermost possible header.

Instructions and Actions

Table 16-7 Egress Flow Table Instruction Parameters

Fields	Description
METER_IDX	Meter index.
METER_YELLOW_ACT	Take the action if packet is colored to Yellow. 0b0: drop 0b1: DSCP remark
METER_RED_ACT	Take the action if packet is colored to Red. 0b0: drop 0b1: DSCP remark
WA_POP_VLAN	Pop the outermost VLAN tag from the packet.
WA_PUSH_VLAN	Push VLAN tag to the packet.
WA_PUSH_VLAN_ETHTYPE	Index to outer TPID pool (*VLAN_TAG_TPID_CTRL.OTPID*).

WA_SET_FIELD0_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: VLAN Priority 0x2: IPv4/IPv6 DSCP 0x3: reserved
WA_SET_FIELD0_DATA	Field Data 0. When WA _SET_FIELD0_TYPE = 0x1: VLAN Priority 0x2: IPv4/IPv6 DSCP
WA_SET_FIELD1_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: VLAN ID
WA_SET_FIELD1_DATA	Field Data 1. When WA _SET_FIELD0_TYPE = 0x1: VLAN ID
WA_OUTPUT_TYPE	Output type. 0x0: forward 0x1: drop

Note

For SET_FIELD action, all the modification is applied to the outermost possible header.

16.4.7 Instructions and Actions

16.4.7.1 Meter

Meter instruction comes with three parameters: METER_IDX, METER_YELLOW_ACT and METER_RED_ACT. METER_IDX points to a dual-band meter for bandwidth measurement. METER_YELLOW_ACT is executed when the packet is colored to be yellow while METER_RED_ACT is executed when the packet is colored to be red.

When METER_XXX_ACT is configured to DSCP remark, the IP DSCP filed is modified according to Table 16-8. For the packet's DSCP which is not in the assured forwarding behavior group, DSCP remark is not supported. When METER_XXX_ACT is configured to DROP, the packet gets dropped when the traffic rate is over than the specified rate.

Table 16-8 Assured Forwarding Behavior Group in RFC2475 (DiffServ)

	Class 1	Class 2	Class 3	Class 4
Low Drop(DP0, Green)	AF11(001010)	AF21(010010)	AF31(011010)	AF41(100010)
Med Drop(DP1, Yellow)	AF12(001100)	AF22(010100)	AF32(011100)	AF42(100100)
High Drop(DP2, Red)	AF13(001110)	AF23(010110)	AF33(011110)	AF43(100110)

Note

Once the DSCP is remarked by a certain flow table, the new DSCP can be matched by later flow tables.

Multiple flow entries can map to the same meter for rate aggregation. A packet can be measured by up to three meters which are meter in Ingress Flow Table 0, Ingress Flow Table 3 and Egress Flow Table 0 for hierarchical policing. The system supports 512 dual-band meters which are shared by OpenFlow and ACL function. Please refer to the Meter chapter for the meter mode, rate, and burst size configurations.

16.4.7.2 Clear Action Set

All the actions in the action set are cleared.

16.4.7.3 Write Action Set

Action Set is empty for the beginning. The instruction is for pushing actions to the Action Set. If the pushing action already existed in the Action Set, then overwrites it. Otherwise, pushes it.

Note

The action overwritten rule for SET FIELD action is different. The SET FIELD action is overwritten only if the SET FIELD TYPE is the same, different SET FIELD TYPE is treated as different action.
Example 1: Ingress Flow Table 0 entry with WA_SET_FIELD0_TYPE= VLAN Priority and Ingress Flow Table 3 entry with WA_SET_FIELD0_TYPE= IP DSCP, then the final Action Set contains SET FIELD action for both VLAN Priority and IP DSCP.
Example 2: Ingress Flow Table 0 entry with WA_SET_FIELD0_TYPE= IP DSCP and Ingress Flow Table 3 entry with WA_SET_FIELD1_TYPE= IP DSCP, then the final Action Set contains SET FIELD action for IP DSCP which specified by Ingress Flow Table 3 entry.

The system supports following actions and the actions are executed in the listed order.

COPY TTL INWARDS

Copy the TTL from outermost to next-to-outermost header with TTL. Copy can be MPLS-to-MPLS, or MPLS-to-IP.

Note

1. The action is not supported for IP-in-IP packet.
2. For L2MPLS-to-IP case, POP MPLS action is required for the action to take effect.

POP VLAN

Pop the outermost VLAN Tag from the packet. The action is ignored for a VLAN untag packet.

POP MPLS

Pop the outermost or double MPLS label(s) from the packet. The action is ignored for a MPLS untag packet.

PUSH MPLS

Push MPLS shim header onto the packet.
When WA_PUSH_MPLS_MODE=0b0(push MPLS label to the packet), it pushes a MPLS label with default content to the packet but doesn't specify the label content. User could use SET FILED action to specify the label content.
When WA_PUSH_MPLS_MODE=0b1(push/swap MPLS label according to WA_MPLS_LIB_IDX), it pushes label(s) according to the MPLS Encap entry which is indexed by WA_MPLS_LIB_IDX.
WA_PUSH_MPLS_VPN_TYPE is to indicate the MPLS encapsulation format and only takes effect for a MPLS untag packet. The system supports two MPLS labels processing at most and the action is ignored if the MPLS label number after pushing is over than two labels.

PUSH VLAN

Push VLAN Tag onto the packet. The system supports two VLAN tags processing at most and the action is ignored if the VLAN tag number after pushing is over than two tags. The action pushes a VLAN tag with default content to the packet but doesn't specify the tag content. User could use SET FILED action to specify the tag content.

COPY TTL OUTWARDS

Copy the TTL from next-to-outermost to outermost header with TTL. Copy can be MPLS-to-MPLS, or IP-to-MPLS.

Note

1. The action is not supported for IP-in-IP packet.
2. For IP-to-L2MPLS case, PUSH L2 MPLS action is required for the action to take effect.

DECREMENT MPLS TTL

Decrement the MPLS TTL from the outermost MPLS label. For the TTL exception, system provides below register for error handling.

Table 16-9 OF_EXCPT_CTRL Register

Bits	Field	Description
0:0	TTL_ACT	Action to take if MPLS/IP TTL exception (TTL=0 after decrement) is asserted. 0b0: drop 0b1: forward

DECREMENT IP TTL

Decrement the outermost IPv4 TTL or IPv6 Hop Limit field. For the TTL exception, Table 16-9 is used for error handling.

SET FILED

Apply packet modifications to the packet. SET FILED actions should always be applied to the outermost-possible header. The system can apply up to five SET FILED actions for a packet, thus, NAPT and routing applications can be supported.

SET QUEUE

Specify queue ID for the packet.

GROUP

If a group action is specified, apply the actions of the relevant group bucket(s).

OUTPUT

Forward the packet to the port(s) specified by the output action.
When WA_OUTPUT_TYPE=0x7(Flood), flood the packet to all ports and applies source port, trunk, VLAN and Spanning Tree filtering.
When WA_OUTPUT_TYPE=0x8(Loopback to normal pipeline), the packet is loopback to normal pipeline directly.
When WA_OUTPUT_TYPE=0x9(Tunnel Interface), forward the packet to the specified tunnel interface.
When WA_OUTPUT_TYPE=0xA(Failover), forward the packet to the first link up port by specified port list. The output type reduces the failover time efficiently and no CPU participation is required.

Note

1. If group action is specified in the Action Set, the actions in the Action Set and actions in the Action Bucket are merged. Action in the Action Bucket overwrites the Action Set for the same action. The actions are executed after merging.
2. The packet is dropped if neither output action nor group action are specified in the Action Set.

16.4.7.4 Write Metadata

Write the masked metadata value into the metadata field. The mask specifies which bits of the metadata field should be modified.

new_metadata = (old_metadata & ~mask) | (value & mask)

Metadata is the data passing through flow tables and usually used for classifying the packet.

16.4.7.5 Goto Table

Forward the packet to next ingress flow table for processing. The instruction doesn't support forwarding packet to egress flow table. A packet stops pipeline traversing if no Goto Table instruction is specified.

When GOTO_TBL_ACT=0x1(apply current Action Set and loopback packet to OpenFlow pipeline), means the pipeline traversing is finished and should loopback the packet to OpenFlow pipeline again after executing current Action Set .
When GOTO_TBL_ACT=0x2(don't apply current Action Set and loopback packet to OpenFlow pipeline), means the pipeline traversing is not finished yet and should loopback the packet to OpenFlow pipeline again with current Action Set.
GOTO_TBL_ACT=0x1 and 0x2 are both used to extend the ingress flow table stages.

The system provides a configuration for limiting the maximum loopback time. Loopback packet carries the GOTO_TBL_LB_TIME in the loopback header, flow entry can filter the match field "TARGET_LB_TIME" to emulate the flow table ID as wish. For example, a flow entry in Ingress Flow Table 3 can easily filter the TARGET_LB_TIME=2 to emulate the flow table ID 11(3+2*4) but the packet just loopback once. Another approach to emulate the flow table ID is through metadata.

Table 16-10 OF_ACT_CTRL Register

Bits	Field	Description
0:0	LB_TIME	Limit maximum loopback times. Note: value 0x0 means unlimited.

Note

The maximum loopback bandwidth is 10Gbps (including 40 byte loopback header).

16.4.7.6 Default Values For Push Action

For push MPLS and push VLAN actions, the MPLS label content and VLAN tag content are copied from existing fields. The content is filled with 0 if no existing fields (original packet is MPLS/VLAN untag) to inherit. User should use SET FILED action to modify the tag default values.

Table 16-11 Default Values for Push Action

Existing Fields	New Fields
VLAN ID	VLAN ID
VLAN priority	VLAN priority
MPLS label	MPLS label
MPLS traffic class	MPLS traffic class
MPLS TTL	MPLS TTL
IP TTL	

16.4.8 Hit Indication

Each full match flow entry supports a rule hit indication to indicate the hit status. Software can periodically polling the indication bit to implement the flow entry idle_timeout function. Refer to the "PIE" chapter for the Rule Hit Indication description.

16.5 L2 Optimized Flow Table

Many applications filter source and/or destination MAC address and consume lots of flow entries. The system supports L2 flow table for MAC-based applications, such as traditional L2 switching, MAC-based VLAN and so on.

L2 flow table is a hash-based flow table which shares the physical resources with L2 table. System also provides a L2 CAM-based flow table for resolving hash collision. There are 16K hash-based and 64 CAM-based L2 flow entries.

A packet lookups L2 flow table twice and the instructions of the hit entry by the first lookup are taken when both lookups are hit. The hash key can be selected by register *OF_L2_FLOW_TBL_CTRL*.

Table 16-12 OF_L2_FLOW_TBL_CTRL Register

Bits	Field	Description
3:0	LU_SEL	Select hash key (Match Fields) for the First and Second lookup. 0x0: (VID,SMAC) and (0, SMAC) 0x1: (VID, SMAC) and (VID,DMAC) 0x2: (VID, SMAC) and (0, DMAC) 0x3: (0, SMAC) and (VID, DMAC) 0x4: (0, SMAC) and (0, DMAC) 0x5: (VID, DMAC) and (0, DMAC) 0x6: (DMAC, SMAC) and (VID, SMAC) 0x7: (DMAC, SMAC) and (0, SMAC) 0x8: (DMAC, SMAC) and (VID, DMAC) 0x9: (DMAC, SMAC) and (0, DMAC) 0xA: (IP_PROTO,L4_DPORT,L4_SPORT,DIP,SIP) and (VID,L4_DPORT,L4_SPORT,DIP,SIP) 0xB~0xF: reserved

Note

1. The VID is from the outermost VLAN for tagged packet and from port-based VLAN for untagged packet.
2. When LU_SEL=0xA, only IPv4 is supported.

16.5.1 L2 Hash-based Flow Entry

Each L2 hash-based flow entry occupies two physical L2 entries (4-way index 0,1 or 2,3) and is composed of below fields.

Match Field	To match against packets.
Instruction	Instructions to be executed when the flow entry hit.
Hit Indication	Set when the flow entry hit.

Note

Counters are not supported in L2 flow entry.

16.5.2 L2 Flow Entry Match Field

L2 flow table supports four entry types: SMAC, DMAC, SMAC+DMAC and 5-TUPLE. The match fields according to different entry types are depicted in following sections.

16.5.2.1 L2 Flow Entry Match Field – SMAC/DMAC Type

Below table shows the match field of entry type SMAC/DMAC in logical view which is applied to hash key (VID, SMAC), (VID, DMAC), (0, DMAC) and (0, SMAC).

Table 16-13 L2 Flow Entry Match Field for (VID/0, MAC) Entry Type

Field	Description
VALID	Valid bit
FMT	Entry format 0b0: Ethernet 0b1: OpenFlow
TYPE	Entry type 0x0: SMAC 0x1: DMAC 0x2: SMAC+DMAC 0x3: 5-TUPLE
VID	VLAN ID
MAC	SMAC or DMAC address according to TYPE field
METADATA_CMP	Whether to compare Metadata
METADATA_KEY	6-bit Metadata

16.5.2.2 L2 Flow Entry Match Field – SMAC+DMAC Type

Below table shows the match field of entry type SMAC+DMAC in logical view which is applied to hash key (SMAC, DMAC).

Table 16-14 L2 Flow Entry Match Field for (SMAC, DMAC) Entry Type

Field	Description
VALID	Valid bit
FMT	Entry format 0b0: Ethernet 0b1: OpenFlow
TYPE	Entry type 0x0: SMAC 0x1: DMAC 0x2: SMAC+DMAC 0x3: 5-TUPLE

VID	VLAN ID
SMAC	Source MAC address
DMAC	Destination MAC address
METADATA_CMP	Whether to compare Metadata
METADATA_KEY	6-bit Metadata

16.5.2.3 L2 Flow Entry Match Field – 5-TUPLE Type

Below table shows the match field of entry type 5-TUPLE in logical view which is applied to hash key (IP_PROTO,L4_DPORT,L4_SPORT,DIP,SIP) and (VID,L4_DPORT,L4_SPORT,DIP,SIP).

Table 16-15 L2 Flow Entry Match Field for 5-TUPE Entry Type

Field	Description
VALID	Valid bit
FMT	Entry format 0b0: Ethernet 0b1: OpenFlow
TYPE	Entry type 0x0: SMAC 0x1: DMAC 0x2: SMAC+DMAC 0x3: 5-TUPLE
SIP	Source IPv4 address
DIP	Destination IPv4 address
L4SPORT	Layer 4 source port
L4DPORT	Layer 4 destination port
5-TUPLE_TYPE	5-Tuple Type 0b0: IP Protocol 0b1: VID
IP_PROTO/VID	When 5-TUPLE_TYPE=0b0, IP protocol.
	When 5-TUPLE_TYPE=0b1, VLAN ID.

Note

1. IPv6 is not supported.
2. L4SPORT/L4DPORT will be layer 4 header byte 0 and 1 if layer 4 protocols are not TCP/UDP/SCTP.
3. METADATA match field is not supported in 5-TUPLE entry type.

16.5.3 L2 Flow Entry Instructions

Each L2 flow entry regardless of hash key type has same instruction masks and instructions. Each L2 flow entry supports below instruction masks to specify the instructions to be executed. The instructions are executed immediately by the listed order when the flow entry hit.

- Clear Action Set
- Write Action Set
- Write Metadata
- Goto Table

Each instruction has specific parameters which are listed in below table.

Table 16-16 L2 Flow Entry Instruction Parameters

Fields	Description
WA_PUSH_VLAN	Push VLAN tag to the packet.
WA_PUSH_VLAN_ETHTYPE	Index to outer TPID pool (*VLAN_TAG_TPID_CTRL.OTPID*).
WA_SET_FIELD0_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: VLAN ID 0x2: IPv4/IPv6 DSCP 0x3: reserved
WA_SET_FIELD0_DATA	Field Data 0. When WA _SET_FIELD0_TYPE = 0x1: VLAN ID 0x2: IPv4/IPv6 DSCP
WA_SET_FIELD1_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: VLAN Priority
WA_SET_FIELD1_DATA	Field Data 1. When WA _SET_FIELD0_TYPE = 0x1: VLAN Priority
WA_SET_Q	Set Queue ID for the packet.
WA_SET_Q_DATA	Queue ID.
WA_OUTPUT_TYPE	Output type. 0x0: disable 0x1: physical port with source port filter 0x2: trunk port with source port filter 0x3: multiple egress ports with source port and trunk filter 0x4: packet's ingress physical port 0x5: flood (all ports with source port, trunk, VLAN, Spanning Tree filtering) 0x6: loopback to normal pipeline 0x7: tunnel interface

WA_OUTPUT_DATA	When WA_OUTPUT_TYPE = 0x1: device ID and physical egress port 0x2: trunk ID 0x3: index to port-mask table 0x7: index to tunnel start table
METADATA	12-bit Metadata.
METADATA_MASK	12-bit Metadata mask.
GOTO_TBL_ACT	0x0: general Goto 0x1: apply current Action Set and loopback packet to OpenFlow pipeline again 0x2: don't apply current Action Set and loopback packet to OpenFlow pipeline again 0x3: reserved

GOTO_TBL_ID	GOTO_TBL_ACT=0, the next table ID.
	GOTO_TBL_ACT=1, the next table ID after loopback.
	GOTO_TBL_ACT=2, the next table ID after loopback.

GOTO_TBL_LB_TIME	Target loopback time. Flow entry can filter "target loopback time" to emulate the flow table stage it resided.

Note

For SET_FIELD action, all the modification is applied to the outermost possible header.

16.5.4 L2 Flow Entry Hash Algorithm

The system supports the 2-left 2-way HASH algorithm, which can reduce the collision ratio. The 2-left means that the whole 16K L2 flow entries are divided into two blocks, per block can select its own HASH algorithm from global two HASH algorithms independently. The control register field is as below.

Table 16-17 OF_L2_FLOW_TBL_CTRL Register

Bits	Field	Description
5:5	BLK1_ALGO	Select the hash algorithm for L2 Hash-based Flow Entry block 1. 0b0: algorithm 0 0b1: algorithm 1
4:4	BLK0_ALGO	Select the hash algorithm for L2 Hash-based Flow Entry block 0. 0b0: algorithm 0 0b1: algorithm 1

16.6 L3 Optimized Flow Table

Application such as layer 3 routing filters destination IP address and could consume lots of flow entries due to huge number of network route and host route. The system supports L3 flow table to address IP-based applications, such as traditional L3 routing, IP-Subnet-based VLAN and so on.

L3 flow table is composed of two tables: L3 TCAM-based flow table and L3 hash-based flow table. L3 TCAM-based flow table shares the physical resources with L3 Network Route (LPM) table and L3 hash-based flow table shares the physical resources with L3 Host Route table respectively.

A packet lookups L3 TCAM-based flow table twice and the instructions of the hit entry by the first lookup are taken when both lookups are hit. The packet also lookups L3 hash-based flow table twice and the instructions of the hit entry by the first lookup are taken when both lookups are hit. The system provides a configuration to define the precedence when both TCAM-based and hash-based flow table are hit. The hash key for L3 TCAM-based and hash-based flow tables can be selected by below register fields.

Table 16-18 OF_L3_FLOW_TBL_CTRL Register

Bits	Field	Description
4:4	LU_PRI_SEL	Define the flow entry priority when both SRAM and TCAM are hit. 0b0: SRAM (hash-based) 0b1: TCAM
3:2	HASH_LU_SEL	Select hash-based flow table lookup key (Match Fields) for the First and Second lookup. 0x0: SIP and DIP 0x1: (SIP,DIP) and SIP 0x2: (SIP,DIP) and DIP 0x3: reserved
1:0	CAM_LU_SEL	Select TCAM-based flow table lookup key (Match Fields) for the First and Second lookup. 0x0: (Metadata,SIP) and (Metadata,DIP) 0x1: (Metadata,SIP,DIP) and (Metadata,SIP) 0x2: (Metadata,SIP,DIP) and (Metadata,DIP) 0x3: reserved

16.6.1 L3 TCAM-based Flow Entry

L3 TCAM-based flow entry is composed of below fields.

Match Field	To match against packets.
Instruction	Instructions to be executed when the flow entry hit.
Hit Indication	Set when the flow entry hit.

L3 TCAM-based flow table supports three entry types: SIP, DIP and SIP+DIP. The match fields according to different entry types are depicted in following sections.

Note

Counters are not supported in L3 TCAM-based flow entry.

16.6.1.1 L3 TCAM-based Flow Entry Match Field – SIP/DIP Type

Below table shows the match field of entry type SIP/DIP in logical view which is applied to hash key (Metadata,SIP) and (Metadata,DIP).

Table 16-19 L3 TCAM-based Flow Entry Match Field for SIP/DIP Entry Type

Field	Description
VALID	Valid bit
FMT	Entry format 0b0: L3 0b1: OpenFlow
IP_VER	IP version 0b0: IPv4 0b1: IPv6
TYPE	Entry type 0x0: SIP 0x1: DIP 0x2: SIP+DIP 0x3: reserved
IP_ADDR	Source/destination IPv4/IPv6 address
METADATA_KEY	12-bit Metadata
FMT_MASK	Entry format bitmask
IP_VER_MASK	IP version bitmask
TYPE_MASK	Entry type bitmask
IP_ADDR_MASK	Source/destination IPv4/IPv6 address bitmask
METADATA_KEY_MASK	12-bit Metadata bitmask

Note

1. Each IPv4 SIP or DIP entry occupies two physical L3 network route entries.
2. Each IPv6 SIP or DIP entry occupies three physical L3 network route entries.

16.6.1.2 L3 TCAM-based Flow Entry Match Field – SIP+DIP Type

Below table shows the match field of entry type SIP+DIP in logical view which is applied to hash key (Metadata,SIP,DIP).

Table 16-20 L3 TCAM-based Flow Entry Match Field for SIP+DIP Entry Type

Field	Description
VALID	Valid bit
FMT	Entry format 0b0: L3 0b1: OpenFlow
IP_VER	IP version 0b0: IPv4 0b1: IPv6
TYPE	Entry type 0x0: SIP 0x1: DIP

	0x2: SIP+DIP 0x3: reserved
SIP	Source IPv4/IPv6 address
DIP	Destination IPv4/IPv6 address
METADATA_KEY	12-bit Metadata
FMT_MASK	Entry format bitmask
IP_VER_MASK	IP version bitmask
TYPE_MASK	Entry type bitmask
SIP_MASK	Source IPv4/IPv6 address bitmask
DIP_MASK	Destination IPv4/IPv6 address bitmask
METADATA_KEY_MASK	12-bit Metadata bitmask

Note

1. Each IPv4 SIP+DIP entry occupies two physical L3 network route entries.
2. Each IPv6 SIP+DIP entry occupies six physical L3 network route entries.

16.6.2 L3 Hash-based Flow Entry

L3 hash-based flow entry has the same structure as TCAM-based flow entry which is composed of Match Filed, Instruction and Hit Indication.

L3 hash-based flow table supports three entry types: SIP, DIP and SIP+DIP. The match fields according to different entry types are depicted in following sections.

Note

Counters are not supported in L3 hash-based flow entry.

16.6.2.1 L3 Hash-based Flow Entry Match Field – SIP/DIP Type

Below table shows the match field of entry type SIP/DIP in logical view which is applied to hash key SIP and DIP.

Table 16-21 L3 Hash-based Flow Entry Match Field for SIP/DIP Entry Type

Field	Description
VALID	Valid bit
FMT	Entry format 0b0: L3 0b1: OpenFlow
IP_VER	IP version 0b0: IPv4

	0b1: IPv6
TYPE	Entry type 0x0: SIP 0x1: DIP 0x2: SIP+DIP 0x3: reserved
IP_ADDR	Source/destination IPv4/IPv6 address
METADATA_CMP	Whether to compare Metadata
METADATA_KEY	12-bit Metadata

Note

1. Each IPv4 SIP or DIP entry occupies two physical L3 host route entries.
2. Each IPv6 SIP or DIP entry occupies three physical L3 host route entries.

16.6.2.2 L3 Hash-based Flow Entry Match Field – SIP+DIP Type

Below table shows the match field of entry type SIP+DIP in logical view which is applied to hash key (SIP,DIP).

Table 16-22 L3 Hash-based Flow Entry Match Field for SIP+DIP Entry Type

Field	Description
VALID	Valid bit
FMT	Entry format 0b0: L3 0b1: OpenFlow
IP_VER	IP version 0b0: IPv4 0b1: IPv6
TYPE	Entry type 0x0: SIP 0x1: DIP 0x2: SIP+DIP 0x3: reserved
SIP	Source IPv4/IPv6 address
DIP	Destination IPv4/IPv6 address
METADATA_CMP	Whether to compare Metadata
METADATA_KEY	12-bit Metadata

Note

1. Each IPv4 SIP+DIP entry occupies two physical L3 host route entries.
2. Each IPv6 SIP+DIP entry occupies six physical L3 host route entries.

16.6.3 L3 Flow Entry Instructions

L3 TCAM-based and hash-based flow entry supports the same instruction masks and instructions. Each L3 flow entry supports below instruction masks to specify the instructions to be executed. The instructions are executed immediately by the listed order when the flow entry hit.

- Clear Action Set
- Write Action Set
- Write Metadata
- Goto Table

Each instruction has specific parameters which are listed in below table.

Table 16-23 L3 Flow Table Instruction Parameters

Fields	Description
WA_PUSH_VLAN	Push VLAN tag to the packet.
WA_PUSH_VLAN_ETHTYPE	Index to outer TPID pool (*VLAN_TAG_TPID_CTRL.OTPID*).
WA_DEC_IP_TTL	Decrement the outermost IPv4 TTL/IPv6 Hop Limit from the packet.
WA_SET_FIELD0_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Source MAC 0x2: VLAN Priority 0x3: IPv4/IPv6 DSCP
WA_SET_FIELD0_DATA	Field Data 0. When WA _SET_FIELD0_TYPE = 0x1: index to Egress L3 INTF Table for retrieving SMAC 0x2: VLAN Priority 0x3: IPv4/IPv6 DSCP
WA_SET_FIELD1_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Destination MAC 0x2: VLAN Priority 0x3: IPv4/IPv6 DSCP
WA_SET_FIELD1_DATA	Field Data 1. When WA _SET_FIELD0_TYPE = 0x1: index to OpenFlow DMAC table 0x2: VLAN Priority 0x3: IPv4/IPv6 DSCP

WA_SET_FIELD2_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: VLAN ID 0x2: VLAN Priority 0x3: IPv4/IPv6 DSCP
WA_SET_FIELD2_DATA	Field Data 2. When WA _SET_FIELD0_TYPE = 0x1: VLAN ID 0x2: VLAN Priority 0x3: IPv4/IPv6 DSCP
WA_SET_Q	Set Queue ID for the packet.
WA_SET_Q_DATA	Queue ID.
WA_GROUP	Apply group action to the packet.
WA_GROUP_IDX	Index to Group table for retrieving the group entry.
WA_OUTPUT_TYPE	Output type. 0x0: disable 0x1: physical port with source port filter 0x2: physical port without source port filter 0x3: trunk port with source port filter 0x4: trunk port without source port filter 0x5: multiple egress ports with source port and trunk filter 0x6: packet's ingress physical port 0x7: flood (all ports with source port, trunk, VLAN, Spanning Tree filtering) 0x8: loopback to normal pipeline 0x9: tunnel interface 0xA: failover (first link up port)
WA_OUTPUT_DATA	When WA_OUTPUT_TYPE = 0x1: device ID and physical egress port 0x2: device ID and physical egress port 0x3: trunk ID 0x4: trunk ID 0x5: index to port-mask table 0x9: index to tunnel start table 0xA: index to port-mask table
METADATA	12-bit Metadata.
METADATA_MASK	12-bit Metadata mask.
GOTO_TBL_ACT	0x0: general Goto 0x1: apply current Action Set and loopback packet to OpenFlow pipeline again 0x2: don't apply current Action Set and loopback packet to OpenFlow pipeline again 0x3: reserved

GOTO_TBL_ID	GOTO_TBL_ACT=0, the next table ID.
	GOTO_TBL_ACT=1, the next table ID after loopback.
	GOTO_TBL_ACT=2, the next table ID after loopback.

GOTO_TBL_LB_TIME	Target loopback time. Flow entry can filter "target loopback time" to emulate the flow table stage it resided.

Note

For SET_FIELD action, all the modification is applied to the outermost possible header.

16.6.4 L3 Flow Entry Hit Indication

Each L3 flow entry supports a rule hit indication to indicate the hit status. Software can periodically polling the indication bit to implement the flow entry idle_timeout function.

If both TCAM-based and hash-based L3 flow entries are hit concurrently, the precedence is defined by *OF_L3_FLOW_TBL_CTRL.LU_PRI_SEL* register field and the entry with higher precedence sets the hit indication state.

16.6.5 L3 Flow Entry Hash Algorithm

The system supports the 2-left N-way(N is 3 for IPv4, 2 for IPv6) HASH algorithm for L3 hash-based flow table, which can reduce the collision ratio. The 2-left means that the whole 6K IPv4/4K IPv6 L3 hash-based flow entries are divided into two blocks, per block can select its own HASH algorithm from global two HASH algorithms independently. The control register field is as below.

Table 16-24 OF_L3_FLOW_TBL_CTRL Register

Bits	Field	Description
6:6	BLK1_ALGO	Select the hash algorithm for L3 Hash-based Flow Entry block 1. 0b0: algorithm 0 0b1: algorithm 1
5:5	BLK0_ALGO	Select the hash algorithm for L3 Hash-based Flow Entry block 0. 0b0: algorithm 0 0b1: algorithm 1

16.6.6 L3 Flow Entry Capacity

Since different L3 flow entry type occupies different number of L3 physical entries. Here we summarize the maximum number of L3 flow entry for different entry type as below table.

Table 16-25 L3 Flow Entry Capacity

Entry Type	TCAM-based	Hash-based
IPv4 SIP or DIP	6K	6K
IPv4 SIP + DIP	6K	6K
IPv6 SIP or DIP	4K	4K
IPv6 SIP + DIP	2K	2K

16.7 Table Miss

The system per flow table supports a table miss configuration for handling lookup miss packets.

Table 16-26 OF_IGR_TBL_MISS Register

Bits	Field	Description
1:0	ACT	Ingress flow table miss action. 0x0: drop 0x1: trap 0x2: forward to next table 0x3: execute current action set

Table 16-27 OF_EGR_TBL_MISS Register

Bits	Field	Description
1:0	ACT	Egress flow table miss action. 0b0: forward 0b1: drop

When ACT is configured to Trap, system supports below register filed to specify the target CPU.

Table 16-28 OF_ACT_CTRL Register

Bits	Field	Description
6:6	TRAP_TARGET	Specify the target CPU. 0b0: local CPU 0b1: master CPU

Note

1. For ingress flow table 3, *OF_IGR_TBL_MISS.ACT*(forward to next table) is not supported.
2. When *OF_IGR_TBL_MISS.ACT*=execute current action set, the OpenFlow pipeline stops and the actions in Action Set are executed.

16.8 Group Table

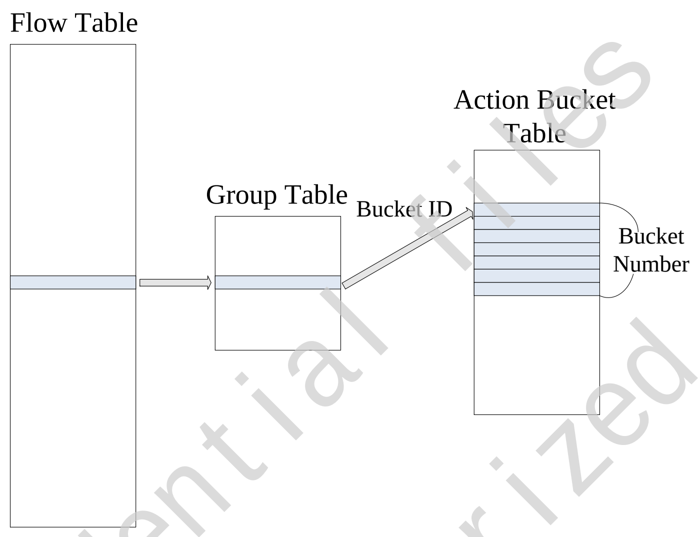
Group table is composed of group entries; the system supports 2048 group entries. Each group entry contains below fields.

Group Type	Group entry type.
Bucket Number	The number of consecutive action buckets attaches to the group entry. Maximum is 128.
Bucket ID	Index to the first action bucket.

The system supports two kinds of entry type: All and Select. Type "All" could be used for the applications such as multicast forwarding and routing, it executes all the action buckets indexed by Bucket ID and Bucket Number. Type "Select" could be used for the ECMP application, it doesn't execute the action bucket indexed by Bucket ID directly but the action bucket indexed by the hash value.

Figure 16-3 shows the relationship of Flow Table, Group Table, and Action Bucket Table.

Figure 16-3 Flow Table, Group Table and Action Bucket Table Relationship



Note

1. Group entry type "Indirect" can be supported by entry type "All" with Bucket Number=1.
2. Group entry type "Fast Failover" is not supported.

16.8.1 Group Entry Hash Algorithm

The group entry hash algorithm is used when the group entry type is "Select". The system provides below configurations to specify the parameters to participate the hash.

Table 16-29 OF_GRP_HASH_CTRL Register

Bits	Field	Description
6:6	L4_DPORT_INC	Include Layer 4(TCP/UDP/SCTP) destination port.
5:5	L4_SPORT_INC	Include Layer 4(TCP/UDP/SCTP) source port.
4:4	DIP_INC	Include destination IP address.
3:3	SIP_INC	Include source IP address.
2:2	DMAC_INC	Include destination MAC address.
1:1	SMAC_INC	Include source MAC address.
0:0	IGR_PORT_INC	Include physical ingress port. 0b0: exclude 0b1: include

Note

If the hash parameter is not available of a certain packet, then value 0 is used.

16.8.2 Action Bucket Table

Action Bucket table is composed of action buckets; the system supports 8192 action buckets. The actions supported by the action bucket table are listed in below.

Table 16-30 Actions for Action Bucket Table

Fields	Description
COPY_TTL_INWARD	Apply copy TTL inward to the packet.
POP_VLAN	Pop the outermost VLAN tag from the packet.
POP_MPLS	Pop MPLS label from the packet. 0b0: disable 0b1: enable
POP_MPLS_TYPE	Pop MPLS label type. 0b0: pop outermost label 0b1: pop double labels
PUSH_MPLS	Push MPLS label to the packet. 0b0: disable 0b1: enable All the WA_PUSH_MPLS_XXX fields following are used only if WA_PUSH_MPLS=enable.
PUSH_MPLS_MODE	0b0: push MPLS label to the packet 0b1: push/swap MPLS label according to WA_MPLS_LIB_IDX
PUSH_MPLS_VPN_TYPE	0b0: L3 VPN 0b1: L2 VPN
PUSH_MPLS_LIB_IDX	Index to MPLS encapsulation table for retrieving label related info. The field is used when WA_PUSH_MPLS_MODE=0b1.
PUSH_MPLS_ETHTYPE	0b0: 0x8847 0b1: 0x8848
PUSH_VLAN	Push VLAN tag to the packet.
PUSH_VLAN_ETHTYPE	Index to outer TPID pool (*VLAN_TAG_TPID_CTRL.OTPID*).
COPY_TTL_OUTWARD	Apply copy TTL outward to the packet.
DEC_MPLS_TTL	Decrement the outermost MPLS TTL from the packet.
DEC_IP_TTL	Decrement the outermost IPv4 TTL/IPv6 Hop Limit from the packet.
SET_FIELD0_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Source MAC 0x2: VLAN Priority 0x3: MPLS Traffic Class 0x4: MPLS TTL 0x5: IPv4/IPv6 DSCP 0x6: IPv4 TTL/IPv6 Hop Limit 0x7: IP Flag Reserved Bit

SET_FIELD0_DATA	Field Data 0. When WA _SET_FIELD0_TYPE = 0x1: index to Egress L3 INTF Table for retrieving SMAC 0x2: VLAN Priority 0x3: MPLS Traffic Class 0x4: MPLS TTL 0x5: IPv4/IPv6 DSCP 0x6: IPv4 TTL/IPv6 Hop Limit
SET_FIELD1_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Destination MAC 0x2: VLAN Priority 0x3: MPLS Label 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
SET_FIELD1_DATA	Field Data 1. When WA _SET_FIELD0_TYPE = 0x1: index to OpenFlow DMAC table 0x2: VLAN Priority 0x3: index to MPLS ENCAP. table 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
SET_FIELD2_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: VLAN ID 0x2: VLAN Priority 0x3: MPLS Label 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
SET_FIELD2_DATA	Field Data 2. When WA _SET_FIELD0_TYPE = 0x1: VLAN ID 0x2: VLAN Priority 0x3: index to MPLS ENCAP. table 0x4: MPLS Traffic Class 0x5: MPLS TTL 0x6: IPv4/IPv6 DSCP 0x7: IPv4 TTL/IPv6 Hop Limit
SET_FIELD3_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: Source IPv4 Address 0x2: Destination IPv4 Address 0x3: VLAN Priority 0x4: MPLS Label 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP

SET_FIELD3_DATA	Field Data 3. When WA _SET_FIELD3_TYPE = 0x1: Source IPv4 Address 0x2: Destination IPv4 Address 0x3: VLAN Priority 0x4: index to MPLS ENCAP. table 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP
SET_FIELD4_TYPE	Apply packet modification for specified Field Type. 0x0: disable 0x1: L4 Source Port 0x2: L4 Destination Port 0x3: VLAN Priority 0x4: MPLS Label 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP
SET_FIELD4_DATA	Field Data 4. When WA _SET_FIELD4_TYPE = 0x1: TCP/UDP Source Port 0x2: TCP/UDP Destination Port 0x3: VLAN Priority 0x4: index to MPLS ENCAP. table 0x5: MPLS Traffic Class 0x6: MPLS TTL 0x7: IPv4/IPv6 DSCP
SET_Q	Set Queue ID for the packet.
SET_Q_DATA	Queue ID.
OUTPUT_TYPE	Output type. 0x0: disable 0x1: physical port with source port filter 0x2: physical port without source port filter 0x3: trunk port with source port filter 0x4: trunk port without source port filter 0x5: multiple egress ports with source port and trunk filter 0x6: packet's ingress physical port 0x7: flood (all ports with source port, trunk, VLAN, Spanning Tree filtering) 0x8: loopback to normal pipeline 0x9: tunnel interface 0xA: failover (first link up port)
OUTPUT_DATA	When WA_OUTPUT_TYPE = 0x1: device ID and physical egress port 0x2: device ID and physical egress port 0x3: trunk ID 0x4: trunk ID 0x5: index to port-mask table 0x9: index to tunnel start table 0xA: index to port-mask table

Note

For SET_FIELD action, all the modification is applied to the outermost possible header.

16.9 Table Counters

In addition to per flow entry counters, the system per flow table supports below counters.

Table 16-31 OF_IGR_TBL_CNTR Register

Bits	Field	Description
63:32	LOOKUP	Packet lookup counter for specific ingress flow table. Read to clear.
31:0	MATCH	Packet match counter for specific ingress flow table. Read to clear.

Table 16-32 OF_EGR_TBL_CNTR Register

Bits	Field	Description
63:32	LOOKUP	Packet lookup counter for egress flow table. Read to clear.
31:0	MATCH	Packet match counter for egress flow table. Read to clear.