Packet Architects AB

Ethernet Switch/Router
L2/L3/MPLS 12x10G
User Guide

Core Revision unknown

Datasheet Revision unknown

Contents

List of Figures

List of Tables

Chapter 1

Overview

This L2/L3 Ethernet Switching/Routing Core offers full wire-speed on all 12 ports. Each port has 8 egress queues which are controlled by a strict priority scheduler.

The core is built around a shared buffer memory architecture capable of simultaneous wire-speed switching on all ports without head of line blocking. Packets are stored in the shared buffer memory as fixed size cells of 160 bytes. In total the buffer memory has a capacity of 4096 cells.

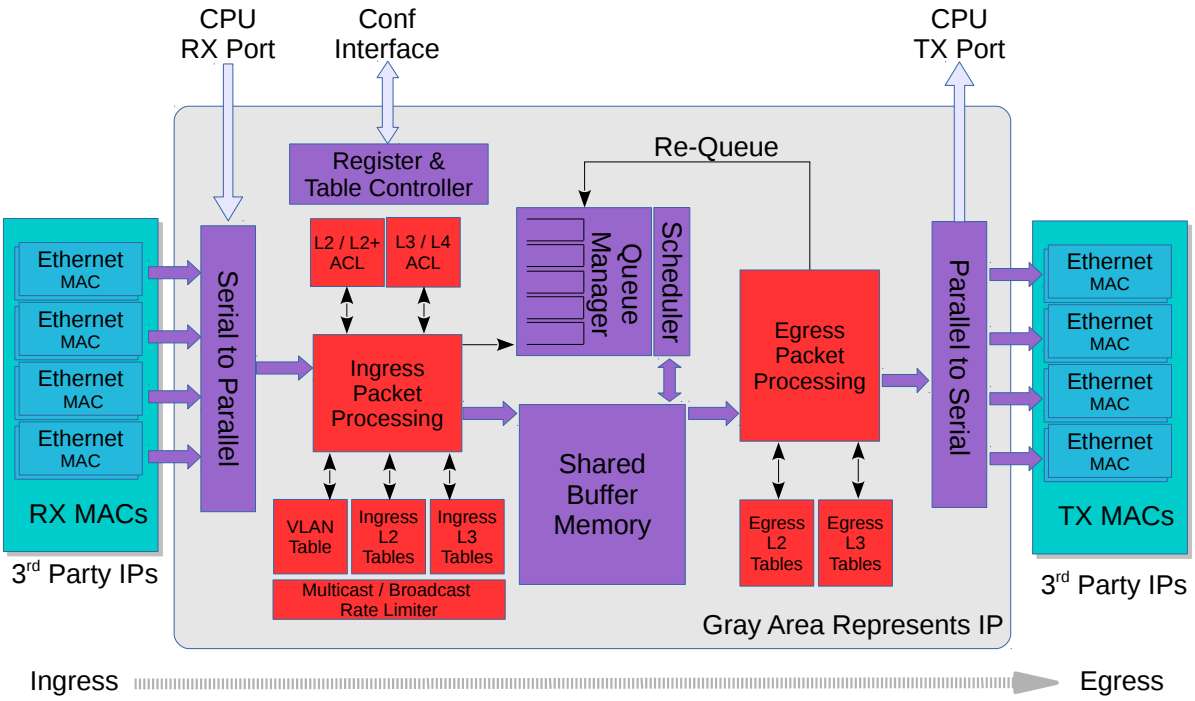


Figure 1.1: Switch Core Overview

Configuring tables and registers are done through a Configuration interface. However it is not required to perform any configuration. The core is ready to receive and forward Ethernet frames once the reset sequence has been completed.

1.1 Feature Overview

- 12 ports of 10 Gigabit Ethernet.
- Full wire-speed on all ports and all Ethernet frame sizes.
- Store and forward shared memory architecture.
- Support for jumbo packets up to 16367 bytes.
- Passes maximum overlap mesh test (RFC2899) excluding the CPU port, for all packet sizes up to 1601 bytes.
- Queue management operations:
  - Disable scheduling of packets on a port.
  - Disable queuing new packets to a port.
  - Allow a port to be drained without sending out packets.
  - Allow checking if a port is empty or not.
- Input and output mirroring.
- 4 source MAC address ranges with a number of different actions.
- 4 destination MAC address ranges with a number of different actions.
- 4,096 entry L2 MAC table, hash based 4-way.
- 4,096 entry VLAN table.
- 16 entry synthesized CAM to solve hash collisions.
- 4 entries of the synthesized CAM are fully maskable.
- 64 entry L2 multicast table.
- Automatic aging and wire-speed learning of L2 addresses. Does not require any CPU/software intervention.
- Spanning tree support, ingress and egress checks.
- 16 multiple spanning trees, ingress and egress checks.
- VLAN priority tag can bypass VLAN processing and be popped on egress.
- MPLS forwarding with support for swap,push,pop and penultimate pop operations.
- 4 entry VRF table.
- 1,024 * 4 hash based L3 routing table.
- 16 entry L3 routing TCAM.
- 1,024 entry next hop table. Pointed to from the routing entries.
- 1,024 entry packet modification table used by the next hop table to determine how build l2 fields in a packet to find the next hop.
- Configurable ECMP support based on L3 protocol field,L3 Tos, and L4 SP/DP.
- ECMP supports with up to 256 paths.
- L2 classification rules. Consists of Source Port, DA MAC, SA MAC, the packets VLAN VID field, the packets VLAN PCP field, the packets VLAN CFI field, Ethernet type.
- 32 entry of classification / ACL consist of source port, routed flag, VRF IPv4 packet type, IPv6 packet type, MPLS packet type, source IP address, destination IP address, TOS, L4 type, L4 source port, L4 destination port, TCP flags.

- Support for allowing L2 and L3 classification rules to be combined to larger than 5 tuple lookups.
- 5242880 bits shared packet buffer memory for all ports divided into 4096 cells each of 160 bytes size
- 8 priority queues per egress port.
- Configurable mapping of egress queue from IP TOS, MPLS exp/tc or VLAN PCP bits.
- 16 ingress admission control entries.
- Strict Priority Scheduler.
- Egress queue resource limiter with four sets of configurations.
- Configuration interface for accessing configuration and status registers/tables.
- Multicast/Broadcast storm control with separate token buckets for flooding, broadcast and multicast packets.
- Multicast/Broadcast storm control is either packet or byte-based, configurable per egress port.
- LLDP frames can optionally be sent to the CPU.

A Packets Way Through The Core

This section describes the path of a packet through the core from reception to transmission, i.e from the RX MAC bus to the TX MAC bus. See Figure 1.1.

1. A packet is received on the RX MAC bus with a *start of packet* signal.
2. Ingress port counters are updated.
3. The asynchronous ingress FIFO synchronizes the incoming data from the data rate of the MAC clock to the data rate of the core clock.
4. The serial to parallel converter accumulates 160 bytes to build a cell, and the cell is sent to ingress processing, if a packet consists of more than 160 bytes then a new cell is built. This is repeated until the *end of packet* signal is asserted.
5. Ingress processing (see chapter 3.1) determines the destination port (or ports) and egress queue of the packet. It then decides whether the packet shall be queued or dropped. Many different tables and registers are used in the process to determine the final portmask and final egress queue for the packet.
6. If the packet matches a certain traffic type whose bandwidth is monitored by the core, it will be pointed to one of the 16 meter-marker-droppers to do the rate measurement. The result may drop the packet or change the packet color.
7. Packets are never modified before they are written into the buffer memory. Rather an ingress to egress header (I2E header) is appended to the packet. Any modifications are done in the egress packet processing pipeline, based on the I2E header.
8. Unless the packet is dropped, the packet is written cell-by-cell into the buffer memory with the I2E header appended.
9. The buffer memory has enough read and write performance for any traffic scenario and will never cause head of line blocking due to read / write conflicts.
10. Once the entire packet is written to buffer memory, it is placed in one or more egress queues and made available to the egress scheduler.
11. Each queue is a linked list of pointers to the first cell in each packet linked to the queue. Each egress queue can link all the packets in the buffer memory even if the buffer memory is filled with only minimum size packets.
12. Counters of the number of cells per ingress port, per ingress port priority, per egress port and egress port queue are updated according to where the packet is sent.
13. When an instance of the packet is selected for output by the egress scheduler, the queue manager will read the packet from the buffer memory and send it, cell-by-cell to the egress packet processing.
14. Egress processing (see chapter 3.2) determines how and if the packet shall be sent out and does the final modifications of the packet. A packet can be re-queued again if it shall be sent out multiple times, which could be the case if input/output mirroring is used. L3 multicast may also re-queue a packet multiple times to the same port.
15. Once the packet is no longer part of any egress queue, the cells it occupied in the buffer memory are deallocated so they can be used by other packets.
16. The parallel to serial converter divides the cell into MAC-bus sized chunks.
17. One asynchronous FIFO per egress port synchronizes the outgoing data from the core clock to the MAC clock.
18. Data is transmitted on the output port.
19. Egress port counters are updated.

1.2 Port Numbering Table

Table 1.1 shows the port numbering. Port 11 can serve as a CPU port.

Interface Number	BW	Clock	Clock Frequency	Sync With Core Clock	Port Number & Multicast Table Bit	CPU Port
0	10.0Gbit/s	clk_mac_rx/tx_0	156.25MHz	No	0	No
1	10.0Gbit/s	clk_mac_rx/tx_1	156.25MHz	No	1	No
2	10.0Gbit/s	clk_mac_rx/tx_2	156.25MHz	No	2	No
3	10.0Gbit/s	clk_mac_rx/tx_3	156.25MHz	No	3	No
4	10.0Gbit/s	clk_mac_rx/tx_4	156.25MHz	No	4	No
5	10.0Gbit/s	clk_mac_rx/tx_5	156.25MHz	No	5	No
6	10.0Gbit/s	clk_mac_rx/tx_6	156.25MHz	No	6	No
7	10.0Gbit/s	clk_mac_rx/tx_7	156.25MHz	No	7	No
8	10.0Gbit/s	clk_mac_rx/tx_8	156.25MHz	No	8	No
9	10.0Gbit/s	clk_mac_rx/tx_9	156.25MHz	No	9	No
10	10.0Gbit/s	clk_mac_rx/tx_10	156.25MHz	No	10	No
11	10.0Gbit/s	clk_mac_rx/tx_11	156.25MHz	No	11	Yes

Table 1.1: Port Numbering Table

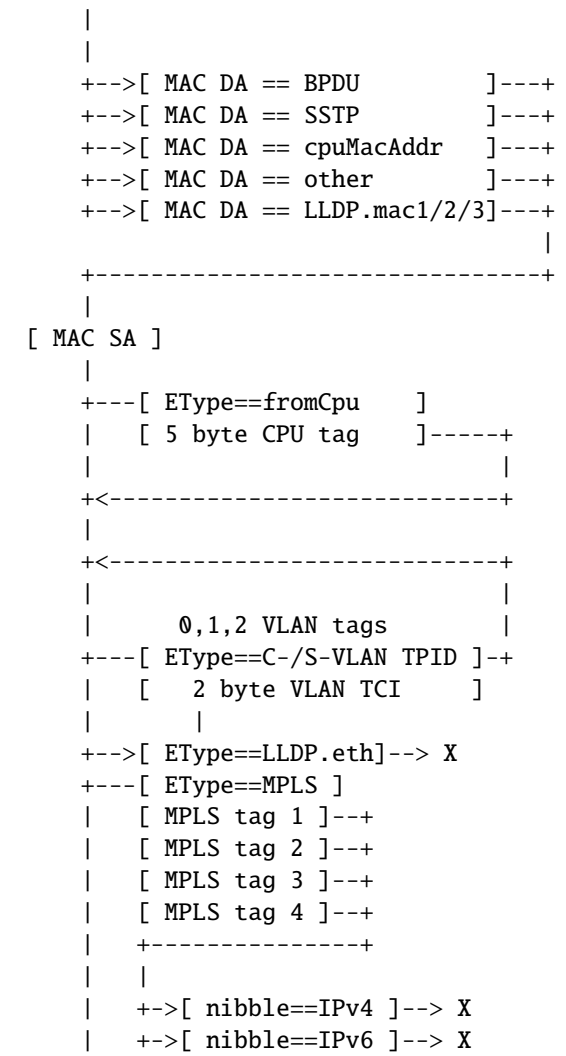
Chapter 2

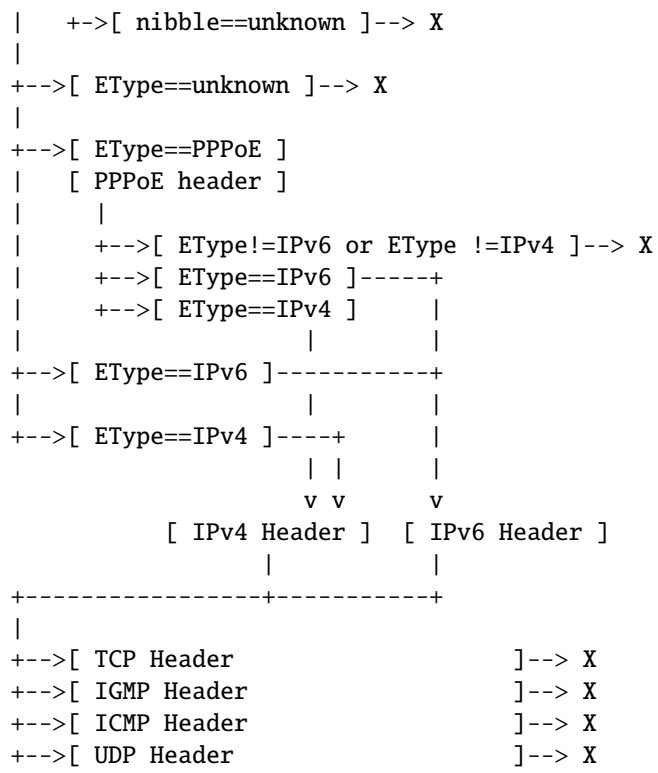
Packet Decoder

The packet decoder identifies protocols and extracts information to be used in the packet processing.

2.1 Decoding Sequence

In the following diagram the decoding of the incoming packet header is described. The comparison used to determine protocol types are described as well as the order they are decoded. The end of decoding process is denote by an *X*.





The packet decoding is done according to the figure above. The packet decoding steps are described below.

1. A packet arrives at the ingress packet processing pipeline.
2. The destination MAC address is extracted and compared.
   (a) If the address matches the BPDU multicast address (01:80:C2:00:00:00) the packet can be sent to the CPU if enabled in **Send to CPU**. There is no decoding done apart from the MAC address comparison. BPDU frames are usually 802.3 encapsulated with a 802.2 LLC header. This decoding is not done by the switch. Note that packets that match the LLDP criteria described below will not be considered BPDU packets.
   (b) If the address matches the SSTP (Shared Spanning Tree Protocol) multicast address (01:00:0C:CC:CC:CD) the packet can be sent to the CPU if enabled in **Send to CPU**. There is no decoding done apart from the MAC address comparison.
   (c) If the address matches the configurable **cpuMacAddr** and this feature is enabled then the packet will be sent to the CPU port.
   (d) If the address matches one of the mac1/mac2/mac3 addresses in the **LLDP Configuration** the packet will subject to further LLDP decoding.
3. The source MAC address is extracted from the packet.
4. The Ethernet type is extracted from the packet and is then compared to known types.
   (a) LLDP
   If the MAC DA address is equal to any of the **LLDP Configuration** mac1/mac2/mac3 addresses and the Ethernet Type is equal to the register **LLDP Configuration** field **eth** then the field **portmask** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. Default is to forward LLDP frames to the CPU port. A packet that matches the LLDP critera will not be considered a BPDU packet even if it matches the BPDU multicast address.

(b) VLAN Tags
There are a number of fixed VLAN types that are identified as well as configurable types. The VLAN processing will use the VLAN tags that decoding has identified and ignore intermediate tags of other types.

i. Customer VLAN Type - 0x8100

ii. Service VLAN Tag - 0x88A8

iii. Configurable VLAN Type setup **Ingress Ethernet Type for VLAN tag**.

When using the Configurable Customer/Service VLAN Type the egress pipeline needs to be setup with the same values if there are actions configured that pushes new VLAN tags to the packet. This is setup in register **Egress Ethernet Type for VLAN tag**.

(c) MPLS.
One MPLS tag is decoded. No other L3 decoding will be done after this.

(d) From CPU Tags
Packets from CPU will use a Ethernet type value of 0x9988.The From CPU Tag is further described in Chapter 24.

(e) IPv4 or IPv6.
If the type identifies these protocols (potentially also after a PPPoE header) the following IPv4 or IPv6 headers are decoded. IPv4 packet with wrong header checksum can be accepted or dropped according to the **Check IPv4 Header Checksum** register. If the L4 protocol is TCP or UDP these headers are also decoded.

(f) L4 Protocol.
If the packet is either a IPv4 or IPv6 and if the L4 protocol is either UDP or TCP then the source port and destination port fields will be extracted.

i. ICMP header
The ICMP type along with the code extracted.

ii. IGMP header
The IGMP type along with the code and IPv4 group address is extracted.

(g) Unknown.
After an unknown Ethernet type no further decoding is done.

Chapter 3

Packet Processing

3.1 Ingress Packet Processing

The ingress packet processing is done as soon as the packet enters the switch. The packet is not sent to the buffer memory until the ingress packet processing is done.

1. Source Port to Link Aggregate
   Source port is mapped to a link aggregate through the **Link Aggregation Membership** table. From this point all references to source ports are actually link aggregate numbers. For details see the **Link Aggregation** chapter.
2. Packet Decoding
   The packet headers are decoded and data extracted. For details see the **Packet Decoding** chapter.
3. Destination MAC Address Range Classification
   The destination MAC address is compared with **Reserved Destination MAC Address Range** table to determine if it should be dropped, sent to CPU or if priority should be forced.
4. Source MAC Address Range Classification
   The destination MAC address is compared with **Reserved Source MAC Address Range** table to determine if it should be dropped, sent to CPU or if priority should be forced.
5. SMON
   If the packets source port and the VID for the outermost VLAN matches an SMON counter then that counter will be updated (see the **Statistics** chapter).
6. Ingress Port Packet Type Filter
   The ingress packet type filter, setup through **Ingress Port Packet Type Filter** per source port, determines if the packet will be dropped or be processed further. This is based on protocol type and type of VLAN. See the **VLAN and Packet Type Filtering** chapter.
7. Ingress Spanning Tree
   The ingress spanning tree state of the source port (from the **Source Port Table**) is checked to determine if packet processing should continue. STP is further described in the **Spanning Tree** chapter.
8. Ingress VLAN Processing
   VLAN processing consists of two parts. Determining the VLAN membership and performing VLAN header modifications.

   The VLAN membership is determined from the assigned ingress VID. See the **Assignment of Ingress VID** section. This will then be used to index into the **VLAN Table** to determine, among other things, VLAN port membership , MSTP and Global ID used in L2 lookups.
9. Ingress MSTP
   The VLAN membership determines which MSTP the packet belongs to by pointing into the **Ingress Multiple Spanning Tree State** table. The state of the source port within this MSTP is checked to

determine if packet processing should continue. MSTP is further described in the **Spanning Tree** chapter.

10. IP Routing
    The routing function figures out where to forward the packet by determining the Next Hop. For details on the routing function see the **Routing** chapter.

    (a) Determine Next Hop
        The routing function is entered if an IP packet matches the router ports MAC address (**Router Port MAC Address**) and routing is allowed on the packets VLAN. L2 lookup, learning and aging will not be performed on routed packets. The router will search for the IP destination address in the routing tables to determine the packets Next Hop, i.e. which port to send the packet to.

    (b) VLAN Operations
        The Next Hop will also determine up to two VLAN operations to perform on the routed packet.

11. IPv4 checksum check and drop.
    For IPv4 packets calculate the checksum value and optionally drop the packet with wrong checksum value. For a routed IPv4 packet the check and drop is always performed.

12. L3 ACL
    The packet is classified on L3/L4 level by matching selected headers with the ACL rules setup in **Ingress L3/L4 ACL Match Data Entries**. There are numerous actions that can be applied when a packet matches an ACL entry. These are configured in **Ingress L3/L4 ACL Result Operation Entries**. When a packet matches an ACL rule the L3 ACL statistics is update. For details one L3/L4 ACLs see **L3 and L4 Classifiction** section.

13. L2 Switching
    If the packet is not routed the destination MAC address is searched for in the **L2 DA Hash Lookup Table**. If the address is found the corresponding entry in the **L2 Destination Table** will return a single destination port or multiple egress ports (if the destination address points to a multicast entry). The status in the **L2 Aging Table** is also updated. If the destination address is not found then the packet will be flooded to all ports that are members of the packets VLAN. See chapter **L2 Switching** for details.

14. Egress Spanning Tree
    When the destination port(s) are known, the spanning tree state for the destination ports are checked in **Egress Spanning Tree State** register.

15. Egress MSTP
    The MSPT state for the destination ports are checked in the **Egress Multiple Spanning Tree State** register. The MSTP id, determined above, is used to index the table.

16. Learning Lookup
    If the packet is not routed the source MAC address is searched in the **L2 SA Hash Lookup Table**. If the address is not found or it has moved to a different port then the Learning Engine will update the tables unless the packet was marked to be dropped. See the **Learning and Aging** chapter for details.

17. IP Statistics
    Statistics of IP unicast, multicast and routed packets are updated.

18. Ingress/Egress Port Packet Type Filter
    As the packet is ready to be queued, the **Ingress Egress Port Packet Type Filter** is applied for each egress port where the the packet is to be queued. See chapter **VLAN and Packet Type Filtering**.

19. Link Aggregation
    The destination ports are now mapped to physical ports using a hash function on the packet headers. The hash index selects which of the physical member ports of this link aggregate that the packet should be sent to. See the **Link Aggregation** chapter.

20. Multicast Broadcast Storm Control
    Multicast packets that are destined for physical ports that have exceeded the MBSC limits will be dropped at this point. See chapter **Multicast Broadcast Storm Control**.

21. Input Mirroring
    If the source port is setup to be input mirrored the mirror port is now added to the list of destination ports. A copy of the input packet, without modifications, will be transmitted on the selected mirror port.

22. Determine Egress Queue Priority
    Egress queues are assigned to packets based on their L2/L3 protocols or classification results. See the **Determine Egress Queue Priority** section.

23. Packet Initial Coloring
    Initial colors are assigned to packets based on their L2/L3 protocols or classification results to represent the drop precedence. See the **Ingress Packet Initial Coloring** section.

24. Queue Management
    If queue management has turned off queuing to a port the packet will be dropped at this point. See section **Queue Management** for details.

25. Drop Statistics
    If the preceding processing has not set any destination ports then the packet is dropped and the **Empty Mask Drop** counter is incremented.

26. Ingress Admission Control
    Packets are grouped into traffic groups based on source port numbers and packet headers, and the bandwidth of each traffic group is measured. If a traffic group exceeds the configured bandwidth or burst size, the initial packet color can be remarked or the packet can be dropped. See the **Ingress Admission Control** section. While the groupping process is through sequence of ingress packet processing steps, the metering process is after all other ingress packet processing are done and before the enqueuing of the packet.

3.2 Egress Packet Processing

After ingress packet processing the packet is stored in the packet buffer memory. The egress packet processing is done when the packet is scheduled for transmission. A single packet can be sent out in multiple copies, for example due to broadcast or mirroring. If the copies are not identical, or multiple copies should be transmitted on the same port, then the packet will be re-queued. This means that it will be re-inserted into the queue engine, where it will again be selected for output and passed once more through the egress packet processing.

1. Output Mirroring
   If output mirroring is enabled for the egress port then the packet is re-queued, so that a copy of the outgoing packet will be transmitted on the output mirror destination port. See the **Mirroring** chapter.

2. IP Header Update
   For routed packets the IP checksum is updated after TTL update, as setup in **Egress Router Table**.

3. Routed DA/SA MAC Update
   For routed packets update the MAC addresses based on the Next Hop.

4. Egress Port VLAN
   A VLAN header operation can be performed based on the physical output port. See the **VLAN Processing** chapter.

5. Egress Port Packet Type Filter
   The egress packet type filter, setup through **Egress Port Configuration** per egress port, determines if the packet will be dropped or be allowed to be transmitted. See the **VLAN and Packet Type Filtering** chapter.

6. VRF Statistics
   If the packet is routed it will be counted in **Transmitted Packets on Egress VRF** counter for the VRF it belongs to.
7. Reassemble Packet Headers
   The final step in the egress processing is to reassembly the outgoing packet header.

Chapter 4

Latency and Jitter

This chapter is meant as an introduction to the causes of latency and jitter in the core. It gives some numbers, but mostly points out the general principles.

The switch has a fixed minimal latency, the bulk of which comes from the ingress and egress packet processing, the store-and-forward operation, and the dataflow registers between design units.

4.1 Latency

The major contributors to latency:

1. The Serial to Parallel converter (SP) gathers the data chunks from the MAC into wider cells.
2. The IPP has a fixed latency of 23 core clock cycles.
3. The queue engine stores the entire packet in buffer memory before adding it to the queues.
4. The EPP has a fixed latency of 6 core clock cycles.
5. Packet modifications that decrease the packet size (for example removing a VLAN) will cause a packet to be delayed one scheduling slot for certain packet sizes.

4.2 Jitter

There are tree places (t1-t3) in the core where latency jitter can be introduced. See Figure 4.1 on page 26.

t1 In the SP the ports are visited in a fixed order, thus introducing a jitter the size of the port visitation period. There is also an asyncronous FIFO between the port and the core clock regions, adding one clock period (of the slowest clock) of jitter.

t2 The egress scheduler visits the ports in a fixed order, introducing a jitter the size of the port visitation period.

t3 The asyncronous FIFO between the core and port clock regions adds one core clock period (of the slowest clock) of jitter.

Note, though, that the core is dimensioned to handle even the worst case jitter without causing packet drops or increased IFG.

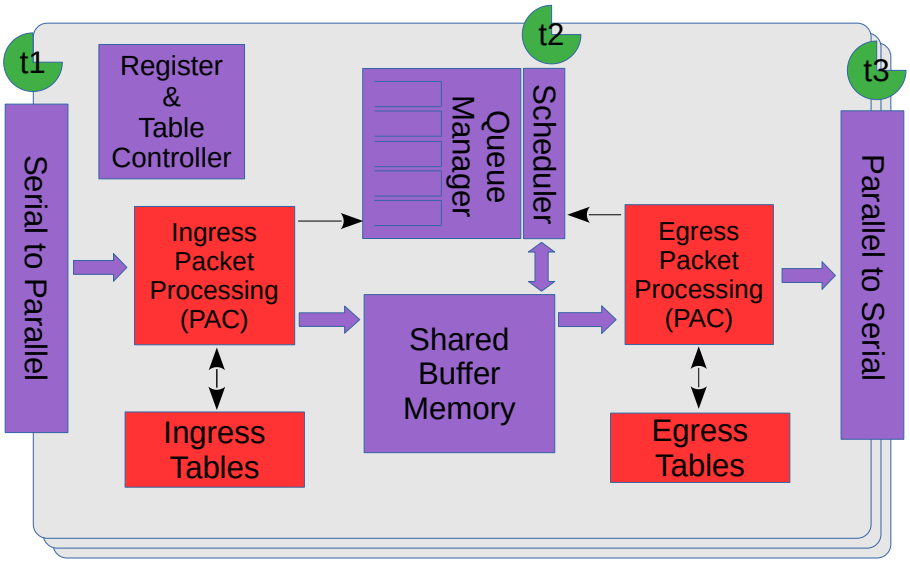


Figure 4.1: Jitter Overview

Chapter 5

VLAN Processing

5.1 Assignment of Ingress VID

All packets entering the switch will be assigned an ingress VID even if the incoming packet doesn't have a VLAN header. This is the VID used to lookup in the **VLAN Table**.

The ingress VID assignment is processed in several steps. The initial assignment is controlled per source port by the **vlanAssignment** in the **Source Port Table** and then it can be updated in a number of ways ranging from L2 to L4 protocols.

5.1.1 VID Assignment from Packet Fields

Ingress VID can be assigned from certain packet fields, other than the packets incoming VID.

There exists a number of these field tables listed below:

- On the L2 MAC layer in **Ingress VID MAC Range Search Data** and its result table **Ingress VID MAC Range Assignment Answer**, the search data can be either on source MAC or destination MAC ranges.
- On the Outer VID in **Ingress VID Outer VID Range Search Data** and its result table **Ingress VID Outer VID Range Assignment Answer**. If the packet has no outer VID then this is skipped. There exists options if the packets VID shall be matched depending on if this is a S-tag or C-tag.
- On the Inner VID in **Ingress VID Inner VID Range Search Data** and its result table **Ingress VID Inner VID Range Assignment Answer**. If the packet has no inner VID then this is skipped. There exists options if the packets VID shall be matched depending on if this is a S-tag or C-tag.
- On the Ethernet Type which is following the innermost VLAN tag. The setup is in **Ingress VID Ethernet Type Range Search Data** and its result table **Ingress VID Ethernet Type Range Assignment Answer**.

VID Assignment Search Order

If there are matches in multiple tables then the "order" field determines which result to use. The result with the highest order value will be used. The search order within a table is not affected by the order field.

The search is carried out as follows:

1. The MAC ranges, defined in **Ingress VID MAC Range Search Data**
2. The Outer VID ranges, defined in **Ingress VID Outer VID Range Search Data**
3. The Inner VID ranges, defined in **Ingress VID Inner VID Range Search Data**

4. The Ethernet Type ranges, defined in **Ingress VID Ethernet Type Range Search Data**

5.1.2 Force Ingress VID from L2 ACL

The L2 ACL engine has an option to override the ingress VID assigned above. If the **forceVidValid** field in the **Ingress L2 ACL Result Operation Entries** is set to 1, the corresponding **forceVid** field will be used as the new ingress VID value. The detailed L2 ACL match and action are described in the **L2 Classification** section.

5.2 VLAN membership

All packets entering the switch will be member of a VLAN, either assigned from the incoming VLAN headers or through a default configuration described below.

The VLAN membership defines which ports that are part of a VLAN. Packets belonging to a VLAN can only enter on the ports that are member of the VLAN.

The L2 switching can only send out packet on the ports that are members of the VLAN, including broadcast, multicast and flooding. This limitation does not apply to routed packets.

The VLAN membership also assigns a global identifier (GID) to a packet which is used during L2 lookup to allow multiple VLANs to share the same L2 tables.

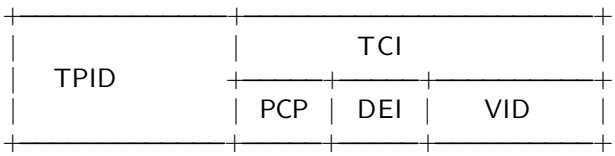
The VLAN membership also determines which multiple spanning tree (MSTP) a packet is part.

The egress queue priority can also be assigned from the VLAN membership (see chapter 17.1).

5.3 VLAN operations

There are a number of operations that can be performed on the packet's VLAN headers such as push/pop etc. Multiple operations can be performed in sequence such that the resulting VLAN header stack from one operation becomes the input to the following operation. However the content of the VLAN headers do not come from previous VLAN operations, they are always created from the original incoming packet or from tables.

For reference here is the 802.1Q VLAN header:



When referring to outermost and innermost VLAN header, outermost means the first VLAN header that the packet decoding has identified as a VLAN header. Innermost means the second VLAN header as identified by the packet decoder.

The VLAN operations that can be performed are:

- Pop - The outermost VLAN header in the packet is removed.
- Push - A new VLAN header is added to the packet before any previous VLANs. It will become the new outer VLAN. The selection of each of the VLAN fields such as TPID, VID, PCP and DEI/CFI are configurable. These fields can either come from existing VLAN headers in the original incoming packet or from tables.
- Swap/Replace - The outermost VLAN header in the packet is replaced. The selection of each of the VLAN fields such as TPID, VID, PCP and DEI/CFI are configurable. These fields can either come from existing VLAN headers in the original incoming packet or from tables.
- Penultimate Pop - All VLAN headers (up to as many as supported by the packet decoder) are removed from the packet.

Figure 5.1 shows the effect of one of these operations on a packet.

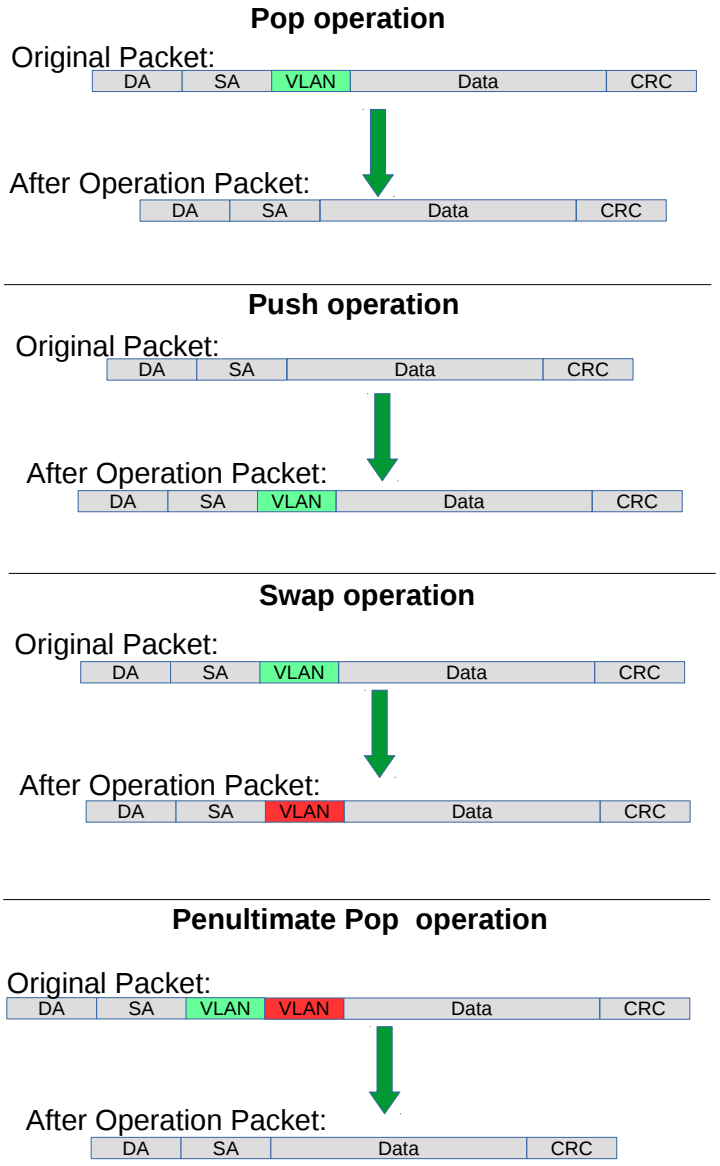


Figure 5.1: VLAN Packet Operations

5.3.1 Default VLAN Header

When a packet enters without a VLAN header an internal default VLAN header will be created. The internal header will have VID, CFI and PCP from **Source Port Table** fields **defaultVid**, **defaultCfiDei**, **defaultPcp**.

The default VLAN header is only used in VLAN operations that selects data from the VLAN packet header.

5.3.2 Source Port VLAN Operation

A VLAN operation to be performed (e.g. push, pop, swap) can be selected by the **vlanSingleOp** field in **Source Port Table**.

If the packet is routed this VLAN operation will not be performed.

5.3.3 L2 ACL VLAN Swap Operation

Ingress L2 ACL Result Operation Entries provide three fields **updateVid**, **updatePcp** and **updateCfiDei** to perform a VLAN swap operation. VLAN push and pop operations are not supported in this ACL.

If the packet is routed then the VLAN swap operation in the L2 ACL will not be performed.

5.3.4 VLAN Table Operation

The **VLAN Table** defines the VLAN port membership, which GID (Global Identifier) to use in L2 lookups, the MSPT to use , if routing is allowed and a VLAN operation to be performed (e.g. push, pop or swap).

If the packet is routed then the VLAN operation from **VLAN Table** will not be performed.

5.3.5 Egress Port VLAN Operation

A VLAN operation to be performed (e.g. push, pop, swap) can be selected by the **vlanSingleOp** field in **Egress Port Configuration**.

A pop operation is done on packets that match a specific VID if **enablePriorityTag** is set in **Source Port Table**.

5.3.6 Priority Tagged Packets

Priority tagged packets are packets that have a VLAN tag with VLAN ID equal to 0. The purpose of these are to extract the PCP bits and use as priority.

The priority extraction can be done as described in 17.1 **Determine Egress Queue** section.

The priority tag can be ignored in all VLAN processing and finally removed on the egress if **enablePriorityTag** is set in **Source Port Table**. Which VLAN ID that triggers this is configured in **priorityVid**

The priority extraction is not dependent on the **enablePriorityTag** setting.

5.3.7 Router VLAN Operations

- If a packet is routed then any VLAN headers in the incoming packet detected by the packet decoder will be removed on the egress.
- All other VLAN operations during ingress packet processing will not be done on routed packets.
- The routers next hop will point to the **Next Hop Packet Modifications** table which can specify up to two push VLAN operations to perform.
- The **Egress Port Configuration** VLAN operation is performed on routed packets after the VLAN operations specified in **Next Hop Packet Modifications**.

5.3.8 VLAN Operation Order

All VLAN operations are performed in sequence on a packet. They follow the order as:

1. One of the four VLAN operations from:
    - **Source Port Table** VLAN operation.
    - Inner VLAN push operation from routers **Next Hop Packet Modifications**.
2. One VLAN swap operation from:
    - **updateVid**, **updatePcp** or **updateCfiDei** in Ingress L2 ACL Result Operation Entries.
3. One of the four VLAN operations from:
    - **VLAN Table** VLAN operation.

- Outer VLAN push operation from routers **Next Hop Packet Modifications**.

4. One of the four VLAN operations from:
   - **Egress Port Configuration** VLAN operation.

The input to the first VLAN operation is the incoming packet. The packet decoder identifies the position of the VLAN headers in the packet and this information is used for the subsequent VLAN operations.

The output from one VLAN operation is input to the next VLAN operation. For example if the first VLAN operation is a push and the second is a swap then the effect will be that the pushed header is replaced by the swap.

If a VLAN operation needs a VLAN header in the packet, i.e. a swap or a pop, and there is no VLAN header in the packet then the operation will not be performed.

5.3.9 VLAN Operation Examples

This process is first described informally with a few examples but to fully specify the behavior it is also described as pseudo code.

Here are examples of sequences of VLAN operations performed on packets with mixed VLANs and custom tags. The incoming packet headers, sequence of VLAN operations and outgoing packet header are briefly described.

```
'V1'..'V2' are VLAN tags in original packet
'new V1'..'new V2' are VLAN tags that have been created by the VLAN operations


Example 1)
incoming packet:
[DA][SA][V1]

VLAN operations: 1. swap new V1
outgoing packet:
[DA/SA][new V1]

Example 2)
incoming packet:
[DA][SA][V1]

VLAN operations: 1. push new V1

outgoing packet:
[DA/SA][new V1][V1]

Example 3)
incoming packet:
[DA][SA][V1][V2]

VLAN operations: 1. push new V1

outgoing packet:
[DA/SA][new V1][V1][V2]

Example 4)
incoming packet:
[DA][SA][V1][V2]
```

```
VLAN operations: 1. pop

outgoing packet:
[DA/SA][V2]

Example 5)
incoming packet:
[DA][SA][V1][V2]

VLAN operations: 1. pop
VLAN operations: 2. swap new V1
VLAN operations: 3. push new V2

outgoing packet:
[DA/SA][new V2][new V1]
```

5.3.10 VLAN Reassembly

The reassembly of the VLAN headers uses data from the packet decoding together with data from the VLAN operations to create the new packet headers.

The following is Python code that exactly models the reassembly operation. The process starts when the L3 and payload in the outgoing packet has been reassembled but before any VLAN or other L2 tags have been added.

The code uses the same incoming packet and VLAN operations as **Example 5)** in the previous section to illustrate the data structure.

```
# The design supports this number of VLAN tags in the ingress packet.
nr_of_ingress_vlans = 2

# Packet decoding results in a list of all VLAN tags from the ingress packet.
pkt_vlan_tags = [ 'V2', 'V1' ]

# Number of VLAN tags that will be used from the original packet.  Before any
# VLAN operations this equals number of incoming VLANs, it could be decreased by
# swap or pop but can't be increased. When nr_of_new_vlans==0, pop or swap will
# decrement it. At any time popAll will set it to 0.
nr_of_pkt_vlans = 2

# Number of new VLAN tags to be used in the reassembly. Push and swap operations
# will increment this and at the same time the new VLAN to the end of new_vlans.
# popAll will set it to 0.
nr_of_new_vlans = 0

# New VLAN tags to be used in the reassembly.
new_vlans = []

# After all VLAN operation sequences: pop, swap new V1, push new V2, VLAN
reassembly collects needed information to get started.
nr_of_pkt_vlans = 0
nr_of_new_vlans = 2
pkt_vlan_tags = [ 'V2', 'V1' ]
new_vlan_tags = [ 'new V1', 'new V2' ]

# At the starting point of re-assembling the VLAN tags the egress packet contains the
```

```
# updated packet after the original tags, i.e. L3/L4/payload.
egress_pkt = ['payload']

# Reassemble the tags with updated VLANs.
while nr_of_pkt_vlans > 0: # Egress packet has VLAN tags from ingress
    # Pop inner most tag from pkt_vlan_tags and insert it first in the egress_pkt
    egress_pkt.insert(0,pkt_vlan_tags[0])
    pkt_vlan_tags = pkt_vlan_tags[1:]
    nr_of_pkt_vlans -= 1

while nr_of_new_vlans > 0: # Egress packet has new VLAN tags
    # Insert a new VLAN first in the egress_pkt from internal VLAN stack.
    egress_pkt.insert(0,new_vlan_tags[0])
    new_vlan_tags = new_vlan_tags[1:]
    nr_of_new_vlans -= 1


# Now egress_pkt contains all updated VLAN headers and tags.  After this new DA/SA
# and other new tags like to_cpu_tag is added to get the final egress packet.
```

Chapter 6

Switching

Most packets will be subjected to a L2 MAC destination address lookup to determine the destination egress port (or ports). These are the exceptions:

- Packet decoder determines that this protocol should be send to the CPU. See Packet Decoder chapter.
- A classification unit action dropped the packet, sent the packet to the CPU, or sent the packet to a specific egress port. See Classification chapter.
- The packet has a From CPU tag which allows the normal packet forwarding process to be bypassed. See Packet From CPU Port section.
- The packet is routed. See the Routing chapter.
- The packet is dropped earlier in the packet processing chain. See chapter Ingress Packet Processing for details.

6.1 L2 Destination Lookup

If none of the above applies a L2 MAC address destination lookup will be performed in the following manner:

- The GID is given by the **gid** field from the **VLAN Table** lookup. See the VLAN Processing chapter.
- The hash is calculated with {GID,DA MAC} as key (see MAC Table Hashing).
- The hash is used as index into the **L2 DA Hash Lookup Table**. 4 entries are read out in parallel, each corresponding to a hash bucket.
- The bucket entries are all compared with the {GID,DA MAC} key and if one entry is equal to the key that entry is considered a match.
- The {GID, DA MAC} key is also compared with all the entries in the **L2 Lookup Collision Table** CAM. The CAM is searched starting from entry 0 and the first matching entry is treated as a match. Any following matching entries are ignored.
- Some entries in **L2 Lookup Collision Table** has per-bit masks. These are set up in the **L2 Lookup Collision Table Masks** registers. Using the mask an entry can define with single-bit granularity what shall be included in the comparison. A zero in the mask means that the corresponding bit shall be ignored, while a one means that the bit shall be compared.
- An entry in the **L2 DA Hash Lookup Table** is only compared if the corresponding valid bits are set. The valid bits are located in the **L2 Aging Table** , the **L2 Aging Status Shadow Table** and the **L2 Aging Status Shadow Table - Replica** . If all the valid bits are not set then this will result in a non-match even if the {destination MAC , GID} in the **L2 DA Hash Lookup Table** entry matches. For the collision CAM the valid bits are located in the **L2 Aging Collision Table** and **L2 Aging Collision Shadow Table**. See figure 6.1.

- If both CAM and L2 hash tables return a match, the result from the CAM table will take precedence.
- Once the final entry has been determined, the result is read out from the **L2 Destination Table**. It has enough entries to fit the destinations for both the L2 hash table and the L2 CAM table. The L2 CAM table entries are located after the L2 hash table entries.
- If the **pktDrop** field in the **L2 Destination Table** is set the packet will be dropped.
- If the destination shall be a single port (i.e. it is not to be multicasted) then the **uc** field shall be set to one and the **destPort or mcAddr** field shall contain the egress port number.
- If a packet shall be sent to multiple output ports then the **uc** field shall be set to zero and the **destPort or mcAddr** field shall contain a pointer to a entry in the **L2 Multicast Table**. The entry in the **L2 Multicast Table** contains a portmask where bit 0 represents port 0, bit 1 port 1, and so on. A bit set to one results in the corresponding port receiving a packet.
- The DA MAC address ff:ff:ff:ff:ff:ff is the broadcast address, meaning that all the member ports in the VLAN (configured in the **VLAN Table vlanPortMask** field) will receive a packet.

  A packet can be sent to its source port only when it hits the corresponding unicast entry in the **L2 Destination Table**. Broadcast, flooded, **L2 Multicast Table** hit packet will have its source port excluded from the destination portmask.
- Ports that are not members of the VLAN will be removed from the portmask. If there are no ports left in the port mask then the packet is dropped and counted in the **L2 Lookup Drop** register.
- If there is no hit in either the **L2 DA Hash Lookup Table** or the **L2 Lookup Collision Table**, then the packet will be flooded, i.e. sent out to all ports in the VLAN. This means that the port mask for the outgoing packet will be taken from the **vlanPortMask** field in the **VLAN Table**.
- If there is a hit then the hit bit in the **L2 Aging Table** is set to one.
- The final physical port is determined by the link aggregation. See chapter Link Aggregation for more information.
- Learning new unknown SA MAC addresses is described in chapter Learning and Aging.

6.2 Software Interaction

Observe that L2 tables can not be directly written by software if learning engine is turned on. Doing so can cause packets to be dropped and/or flooded and the learning engine may stop working. See chapter Learning and Aging for information how to safely update the L2 tables.

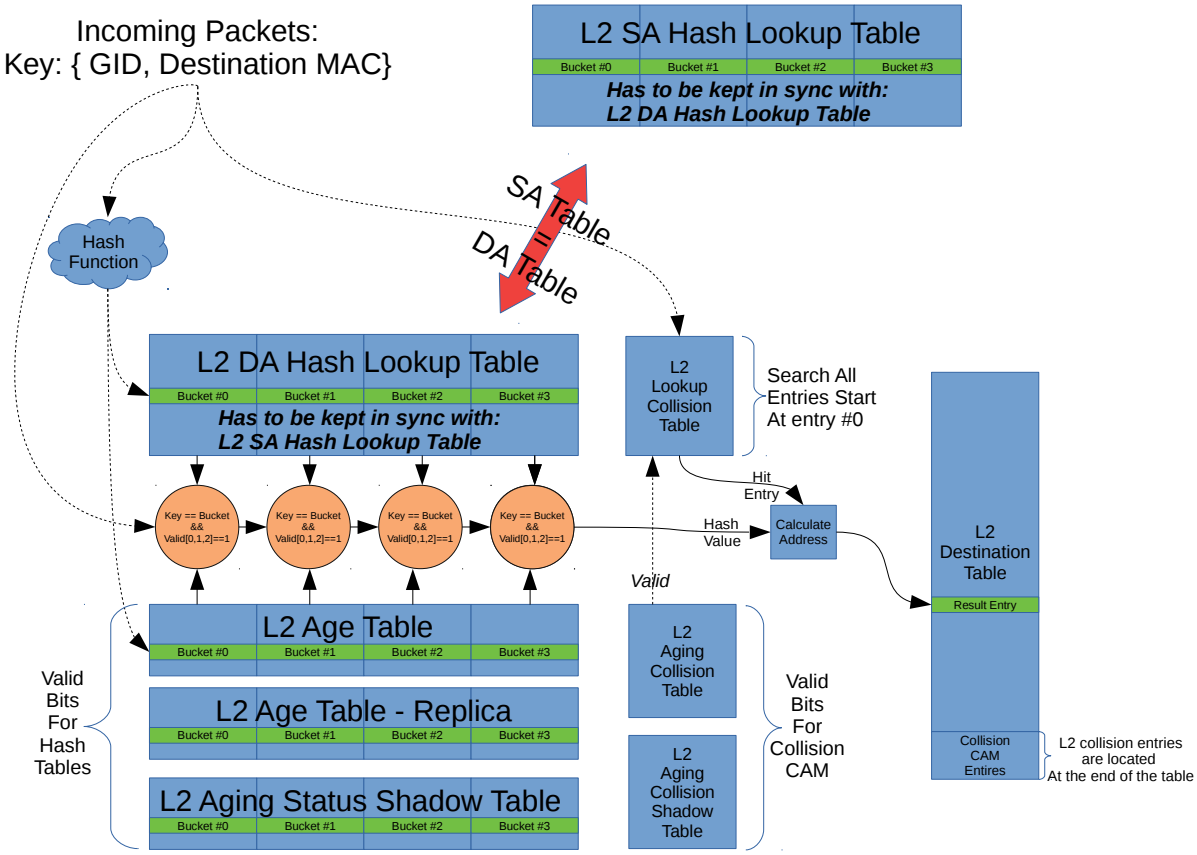


Figure 6.1: L2 Lookup Overview

Chapter 7

Routing

This core supports IPv4 and IPv6 routing as well as MPLS switching.

The routing is disabled by default and needs to be setup from the configuration interface before it can be used. This core supports virtual router ports/functions (VRFs). The VRFs allow the core to handle multiple virtual routers sharing the same set of tables and register. A VRF identifier is used to determine which virtual router each table entry belongs to.

The routing is done separately from the L2 switching. There is no switching done before or after the router. The router is entered when a packets destination MAC address equals the routers MAC address. The packet exits the router directly to an egress port.

MPLS follows the same order of operations as IP routing and uses the same tables. The MPLS processing is therefore described here.

7.1 Order of Operation

Routing function is done after the L2 ACLs. The routing engine performs the following steps:

1. Check if the VLAN allows packets to be routed. If this is not the case normal L2 lookups will be done. This is specified by the **allowRouting** field in **VLAN Table**.
2. Compare the incoming packets MAC destination address with all the entries in the **Router Port MAC Address**. If no match then the routing function is skipped. If the router port search found a match then the packet enters the router with an assigned VRF from the table.
3. The carried packet type (IPv4, IPv6 or MPLS) is checked against the allowed type that are setup in **Ingress Router Table**. If the type is not allowed the packet will be dropped. There is a alternative to dropping the packets and instead send them to the CPU. This can be archived by setting the **sendToCpuOrDrop** bit to one.
4. If the incoming packets TTL is below the allowed TTL, as specified in **Ingress Router Table** then the packet is dropped.
5. To determine the packets destination/next hop the destination address combined with the assigned VRF is searched for in the **Hash Based L3 Routing Table** and in the **L3 Routing TCAM**. If there is a match in both the TCAM and the hash table then the hash entry is selected since the hash table always contains the longest prefix. For the hash based search the next hop result is setup in the **Hash Based L3 Routing Table** and for the LPM search it is setup in **L3 LPM Result**.

   The difference between MPLS and IP search is that in MPLS the 20-bit MPLS label from the outermost MPLS header is used as destination address.
6. If there is a match in the routing tables and the ECMP is enabled in the matched entry (either the **useECMP** in the **Hash Based L3 Routing Table** or **useECMP** in the **L3 LPM Result** table) then ECMP next hop calculation is performed.

ECMP calculates a hash based on the IP source and destination addresses, the IP proto field, IP TOS and the TCP/UDP source port and destination port.

For MPLS the ECMP hash key consists of the outermost header and does not include embedded IP headers. The hash value is added as an offset to the **nextHopPointer** after masking (**ecmpMask**) and shifting (**ecmpShift**).

7. If there is no hit in the destination address search then the default next hop is used. The default is defined in **L3 Routing Default** per VRF. There are also options to drop the packet or send to CPU port.
8. IP statistics is updated in the **Received Packets on Ingress VRF** registers. MPLS forwarded packets are only counted in **Received Packets on Ingress VRF**
9. The next hop from the previous steps is used as index into the **Next Hop Table**. The entries determine where to route the packet, which is either a single destination port or a pointer to a L2 multicast entry. There are also options to drop the packet or send to CPU port.

   Each entry also contains a packet modification pointer which points to several tables that determines what header modification that should be done when the packet exits the router.

   - The **Next Hop Packet Modifications** table determines what VLAN operations to perform when exiting the router. If the entry's valid bit is not set the packet will be send to the CPU.
   - The **Next Hop DA MAC** which determines the destination MAC address to use in the outgoing packet.
   - For MPLS the **Next Hop MPLS Table** determines what MPLS header modifications that should be done on the outgoing packets. These are described in detail in the register description and in the **MPLS** chapter.
10. An MTU check, as specified in the **Router MTU Table**, is performed on incoming routed packets. This check is executed by comparing the IPv4 Total Length field with the limit configured in field **maxIPv4MTU**, separately for each destination port and VRF. Similarly, the IPv6 Payload Length field is compared with field **maxIPv6MTU**. If either length field exceeds its respective limit, the packet will be forwarded to the CPU for further processing. Notably, the MTU check is not applied to MPLS packets.
11. When next hop hit status updates are enabled in the **Ingress Router Table** then each time a packet is routed using a **Next Hop Table** entry the corresponding status bit is set in the **Next Hop Hit Status**.
12. The ingress part of routing is now completed. This is followed by other ingress functions such as L3 ACL etc. Finally the packet is queued to one or multiple egress ports.
13. The egress processing of the routed packet performs the packet header modifications. First step is update of the TTL field which is controlled by the **Egress Router Table**.
14. There exists an option called **Next Hop Packet Insert MPLS Header** which enables a outgoing routed packet to add MPLS labels after the L2 / VLAN headers. This allows the router to enter a MPLS tunnel in order to reach the next hop though a MPLS network. If a packet is already a MPLS packet this option offers a way to insert extra MPLS headers on top of the MPLS label stack. NOTE: It is not possible to insert MPLS headers if the packet has a PPPoE header, If the packet is a PPPoE then no MPLS insertion is then carried out.
15. A new L2 header is constructed with a DA MAC from the **Next Hop DA MAC** table. The SA MAC will be the incoming DA MAC.
16. The routers VLAN operations are performed. See the **VLAN Processing** chapter.
17. The IPv4 header checksum is recalculated.
18. Egress router statistics is updated in **Transmitted Packets on Egress VRF**.
19. The egress ports VLAN operations are performed. See the **VLAN Processing** chapter.

Chapter 8

MPLS

This core is equipped with MPLS forwarding. The processing of MPLS packets follows the same pattern as IP routing, with the major difference that an MPLS header operation (such as push, pop, swap and penultimate pop) can be carried out. Since the order of operation for MPLS is almost identical to IP routing it is described in the **Routing** chapter.

8.1 MPLS Header Operations

In addition to the processing that is done for IP routed packets the MPLS router can perform operations on the MPLS header stack.

The **Next Hop MPLS Table** determines which operation to perform.

- Pop - The outermost MPLS header in the packet is removed.
- Push - A new MPLS header is added to the packet before any previous MPLS headers. The label for the new header and the source for the EXP bits are specified in the table entry.
- Swap/Replace - The outermost MPLS header in the packet is replaced. The label for the new header and the source for the EXP bits are specified in the table entry.
- Penultimate Pop - All MPLS headers (up to as many as supported by the packet decoder, see **Packet Decoding** chapter) are removed from the packet. In addition the Ethernet Type is set to IPv4 or IPv6, see the following section.
- Remapping of EXP bits in the outermost MPLS header. Either use the existing value, use form the table or use a remapping table **Egress Queue To MPLS EXP Mapping Table**.

The **Egress MPLS TTL Table** determines which operation on the TTL field to perform when exiting the VRP, either decrement the TTL or set a new TTL. Each VRP can have their own setting.

8.2 MPLS Penultimate Pop

A normal Pop operation removes one MPLS header but leaves the Ethernet Type unmodified (identifying the packet as still being a MPLS packet).

The Penultimate Pop operation removes all MPLS headers and also updates the packets Ethernet Type. This assumes that the payload in the MPLS packet is an IP packet. The first nibble in the payload is then decoded (see **Packet Decoding** chapter) to determine if the packet is IPv4 or IPv6 and then the Ethernet Type is updated accordingly.

8.3 MPLS Header Insertion To Reach Next Hop

There exists an option called **Next Hop Packet Insert MPLS Header** which enables a outgoing routed packet to add up to MPLS labels after the L2 / VLAN headers. The operation is pointed out by the field **nextHopPacketMod** in table **Next Hop Table**. If a packet is already a MPLS packet this option offers a way to insert extra MPLS headers on top of the MPLS label stack.

NOTE: It is not possible to insert MPLS headers if the packet has a PPPoE header. If the packet is a PPPoE then no MPLS insertion is then carried out.

Chapter 9

Mirroring

This core supports both input and output mirroring.

9.1 Input Mirroring

Input mirroring allows all packets received by an ingress port to be copied to an egress port without packet modifications.

- For each port, one input mirroring port can be configured through the **Source Port Table**. The **inputMirrorEnabled** field enables a input mirror copy and send it to the port configured in the **destInputMirror** field.

By default the input mirror copy will bypass any packet modification or drop decisions during the ingress or egress packet processing. Extra options are given in the **Source Port Table** to limit the range of the mirroring destination. **imUnderVlanMembership** only allows the input mirror copy to be sent to the members of the VLAN. **imUnderPortIsolation** only allows the input mirror copy to be sent to the destination that does not block the current source port from the **Ingress Egress Port Packet Type Filter**.

9.2 Output Mirroring

Output mirroring allows the user to select an egress port to be mirrored so that packet that is transmitted to that egress port can have a copy sent to an egress port. For each port, one output mirroring port can be configured through the **Output Mirroring Table**:

1. The output mirroring functionality can be enabled per port using the **outputMirrorEnabled** field from the **Output Mirroring Table**.
2. The port to which the mirror copy is sent is setup by the **outputMirrorPort** field in the **Output Mirroring Table**. Multiple input ports can use the same output mirroring destination port.

With input mirroring, a port can be used to observe the traffic received by any port. With output mirroring, a port can be used to observe the traffic transmitted from any port. When there are multiple mirror copies requested or the CPU port is involved, the switch works as follows:

- An input mirrored packet can be output mirrored again.
- An output mirrored packet will not be mirrored again even if the destination port has output mirroring turned on.
- When a packet is mirrored to the CPU port, it will not carry an extra to-CPU tag since it is the copy of another packet.

It is possible that a packet is sent out in multiple copies on the same port when mirroring is turned on. In this case at most four instances of the same received packet can appear on an egress port. The order of the packet instances will be:

1. Normal switched/routed packet
2. Input mirror copy
3. Output mirror copy of the switched/routed packet
4. Output mirror copy of the input mirror copy

9.2.1 Requeueing FIFO

Output mirroring (and input mirroring to oneself) is accomplished by requeuing the packets in separate requeueing FIFOs after External Packet Processing. There is one requeue FIFO per egress port.

The egress scheduling will only see the packet at the head of each FIFO, but this packet will be selected before the packets belonging to the same queue in the normal egress queues.

This method of output mirroring means that:

1. The requeuing FIFOs are truly FIFOs per port, so there will be head-of-line blocking between packets of different egress queues mirrored to the same port.
2. The (up to three) mirroring copies for a single input packet are created in series. The first one is not created until the original packet has been scheduled and gone through Egress Packet Processing, the second one not until the first copy has been scheduled and gone through Egress Packet Processing and so on...
3. When several ports output mirror to the same port, or a higher speed port mirrors to a lower speed port (physical or shaped port speed) the requeueing FIFO for the mirroring destination port may fill up and cause packet drops.

The depth of the requeueing FIFOs is ten packets per egress port.

Drops due to the requeueing FIFOs overflowing are counted in the **Re-queue Overflow Drop** register.

Chapter 10

Link Aggregation

Link aggregation is a solution to bundle multiple ports into a higher bandwidth link. Each link aggregate is setup using the **Link Aggregation Membership** and **Link Aggregation To Physical Ports Members**.

The **Link Aggregation Membership** register maps the incoming packets source port number to a link aggregate number. The link aggregate number is then used during ingress packet processing instead of source port/destination port numbers.

When a destination port (destination link aggregate number) has been determined by ingress packet processing the **Link Aggregation To Physical Ports Members** table maps the link aggregate number to which physical ports that are part of the link aggregate, i.e. the physical ports the packet shall be transmitted to.

Note that once link aggregation is enabled all ports needs to be setup as link aggregates, even if a port only has a single port part of its link aggregate. These ports are usually setup as having a one-to-one mapping, i.e. source port number, link aggregate number and physical port number are all the same.

The **Link Aggregation Membership** register and the **Link Aggregation To Physical Ports Members** register must be kept in sync by software.

To distribute the packets over the ports that are part of a link aggregate, a hash is calculated over some of the packets fields which is configured by register **Link Aggregation Ctrl**. The hash value calculated is used to index the **Link Aggregate Weight** table which results in a port mask of the ports that will be used for this specific hash.

The ratio that each port in a link aggregate is used is determined by the number of times the port is set in the **Link Aggregate Weight** table divided by the number of entries in the table.

It is important to setup all entries in the **Link Aggregate Weight** table with one port set for each link aggregate, otherwise a certain hash value will have no port set thereby causing the packet to be dropped.

10.0.1 One-to-one Port Mapping

To setup a one-to-one mapping, then the bit which corresponds to the port number shall be set in the **members**. This maps each link aggregate number to a physical port with the same number.

The **la** should then be set so that each source port number maps to the link aggregate with the same number, i.e. table entry 0 should hold a value of 0, table address 1 should hold a value 1, etc.

10.1 Example

Lets say that a link aggregate shall use physical ports 0,1,2 and each port shall have equal amount of traffic. Another link aggregate will use ports 6,7 also with equal load between the ports. The remaining ports are setup to be one-to-one. In this example these are ports 3,4 and 5, on a switch with 8 ports.

To setup the **Link Aggregation Membership** register we associate the source port with the link aggregate number that it belongs to. Ports 0,1,2 are part of link aggregate 0 and port 6,7 are part or link aggregate 1. The remaining ports are setup to use the same link aggregate number as the port number.

```
for port in [0,1,2]:
  rg_sp2la[port] = 0

for port in [6,7]:
  rg_sp2la[port] = 1

for port in [3,4,5]:
  rg_sp2la[port] = port
```

In **Link Aggregation To Physical Ports Members** we need to setup the relation from link aggregate number to physical port members.

```
  rg_la2Phy[0] = 0b00000111  # la #0 = ports 0,1,2
  rg_la2Phy[1] = 0b11000000  # la #1 = ports 6,7
  rg_la2Phy[3] = 0b00001000  # la #3 = port 3
  rg_la2Phy[4] = 0b00010000  # la #4 = port 4
  rg_la2Phy[5] = 0b00100000  # la #5 = port 5
```

To setup how the traffic is distributed between the link aggregate member ports we first select which packet headers that will be used in the hash calculation. In this example we chose to select source MAC, destination MAC, IP addres, L4, TOS value and vlan header as calculation base for the hash value.

```
  rg_linkAggCtrl.useSaMacInHash = 1
  rg_linkAggCtrl.useDaMacInHash = 1
  rg_linkAggCtrl.useIpInHash = 1
  rg_linkAggCtrl.useL4InHash = 1
  rg_linkAggCtrl.useTosInHash = 1
  rg_linkAggCtrl.useVlanInHash = 1
```

The table **Link Aggregate Weight** shall then be setup so that ports 0,1,2 have equal weight. This is accomplished by configuring so that the number of bits set for port 0 in all hash entries are equal to number of bits for port 1 and port 2. Which bits are set are not important as long as only one bit per entry are set and the total number of bits per port are equal.

If the hash of the packets fields are distributed evenly then $1/3$ of the packets will be distributed to each of the three ports part of the link aggregate.

Similarly to setup a link aggregate on ports 6,7 with equal load between the ports then each entry in the **Link Aggregate Weight** table must have bit 6 or 7 set and with equal number of bits for the two ports.

The ratio for link aggregation 0, is 34% on port 0, 33% on port 1 and 33% on port 2. For link aggregation 1, it is 50% on each port.

```
  for hash_index in range(0,85):        # 34%
    r_hash2LA[hash_index] = 0b00000001  # port 0
  for hash_index in range(86,170):      # 33%
    r_hash2LA[hash_index] = 0b00000010  # port 1
  for hash_index in range(171,256):     # 33%
    f_hash2LA[hash_index] = 0b00000100  # port 2
```

```
for hash_index in range(128):          # 50%
  r_hash2LA[hash_index] |= 0b01000000 # port 6
for hash_index in range(128,256):      # 50%
  r_hash2LA[hash_index] |= 0b10000000 # port 7

for hash_index in range(256):          # 100%
  r_hash2LA[hash_index] |= 0b00001000 # port 3
  r_hash2LA[hash_index] |= 0b00010000 # port 4
  r_hash2LA[hash_index] |= 0b00100000 # port 5
```

Finally when all the registers have been configured the link aggregation function is enabled in the **Link Aggregation Ctrl** register.

```
rg_linkAggCtrl.enable = 1
```

10.2 Hash Calculation

The hash key consists of the following fields in the order listed starting with the msb.

- MAC DA, 48 bits
- MAC SA, 48 bits
- VLAN ID, 12 bits
- IP TOS, 8 bits
- TCP/UDP Source Port, 16 bits
- TCP/UDP Destination Port, 16 bits
- IP Proto, 8 bits
- IPv4/IPv6 Source Address, 128 bits
- IPv4/IPv6 Destination Address, 128 bits
- Source Port, 4 bits

If a field is disabled in the **Link Aggregation Ctrl** register then the field in the hash key will be 0.

If a packet is routed then the MAC DA field will contain the next hop pointer instead of the MAC address and the VLAN ID will be 0.

The hashing is done in two steps, first the key is build, and the fields used in the key depends on the **Link Aggregation Ctrl** register, once the key is build then hash function is used to determine the address used ot lookup the **Link Aggregation To Physical Ports Members**.

```
def build_key(daMac, useDaMacInHash,
              saMac, useSaMacInHash,
              vlanId, useVlanIdInHash,
              tos, useTosInHash,
              sp, useL4InHash,
              dp,
              proto,
              salp, useIpInHash,
              dalp,
              nextHop, useNextHopInHash,
              srcPort, routed):
    # This function builds the key to be
    #   used for calculating the hash.
```

```
    final_data = 0
    if useDaMacInHash==0:
        daMac = 0
    if useNextHopInHash==0:
        nextHop = 0
    if routed==1:
        daMac = nextHop
        vlanId = 0

    final_data = final_data <<48
    final_data = final_data | daMac
    final_data = final_data <<48
    if useSaMacInHash==1:
        final_data = final_data | saMac
    final_data = final_data <<12
    if useVlanIdInHash==1:
        final_data = final_data | vlanId
    final_data = final_data <<8
    if useTosInHash==1:
        final_data = final_data | tos
    final_data = final_data <<16
    if useL4InHash==1:
        final_data = final_data | sp
    final_data = final_data <<16
    if useL4InHash==1:
        final_data = final_data | dp
    final_data = final_data <<8
    if useL4InHash==1:
        final_data = final_data | proto
    final_data = final_data <<128
    if useIpInHash==1:
        final_data = final_data | saIp
    final_data = final_data <<128
    if useIpInHash==1:
        final_data = final_data | daIp
    final_data = final_data <<4
    final_data = final_data | srcPort
    return final_data


def calcLaHash( key ):
    mask = (1 << 8) - 1
    _hash = 0
    for j in range(52):
        _hash     = _hash ^ (key & mask)
        key = key >> 8
    return _hash & mask
```

Chapter 11

Classification

There are a number of classification engines avaliable.

11.1 L2 Classification

- L2 Destination MAC range classification is setup in table **Reserved Destination MAC Address Range**.
  - The table is searched starting from entry 0.
  - When a range is matched the corresponding actions (drop, send to cpu, force egress queue) will be activated.
  - If multiple ranges are matched, any matching range that sets drop will cause a drop.
  - Any match that sets sendToCpu will cause send to CPU (this has priority over drop).
  - When multiple ranges that match has set the forceQueue then the highest numbered entry will determine the value.
- L2 Source MAC range classification is setup in table **Reserved Source MAC Address Range**.
  - The table is searched starting from entry 0.
  - When a range is matched the corresponding actions (drop, send to cpu, force egress queue) will be activated.
  - If multiple ranges are matched, any matching range that sets drop will cause a drop.
  - Any match that sets sendToCpu will cause send to CPU (this has priority over drop).
  - When multiple ranges that match has set the forceQueue then the highest numbered entry will determine the value.
- If the destination MAC address bits [47:8] matches the **L2 Reserved Multicast Address Base** then bits [7:0] of the destination MAC address is used as a index in the table **L2 Reserved Multicast Address Action** which determines what action to take on the packet. Actions are set per source port and can either be to drop the packet or to send it to the CPU.
- L2 ACL engine search data fields are setup in table **Ingress L2 ACL Match Data Entries** and result actions are setup in register **Ingress L2 ACL Result Operation Entries**.
  - The entries in the table are searched starting with entry 0.
  - The statistics counter which can be updated are located in the **Ingress L2 ACL Match Counter**
  - When multiple entries match (are hit) the associated actions from all matching entries will be executed.

  - If two or more entries which match contain the same action then data from the highest (last) entry will be choosen. For example if two entries has the action *force to queue priority* and the lowest hit has a destination queue of 2 while the highest hit has a destination queue of 4 then the packet will have a destination queue of 4.

11.2 L3 and L4 Classification

- L3 and L4 classification engine search data is setup in table **Ingress L3/L4 ACL Match Data Entries** and result actions are setup in table **Ingress L3/L4 ACL Result Operation Entries**.
  - The entries in the table are searched starting with entry 0.
  - When multiple entries match (are hit) the associated actions from all matching entries will be executed.
  - If two or more entries which match contain the same action then action from the highest (last) will be choosen. For example if two entries has the action force to queue priority and the lowest hit has a destination queue of 2 while the highest hit has a destiantion queue of 4 then the packet will have a destiantion queue of 4.
  - The statistics counter which can be updated are located in the **Ingress L3 ACL Match Counter**.

11.3 Chaining

Chaining is a way to connect a L2 ACL entry to a L3 ACL Entry forming a multiple tuple lookup. The chain ID from will be used as search data in **Ingress L3/L4 ACL Match Data Entries** table to be compared with the **chainTag** field.

Chapter 12

VLAN and Packet Type Filtering

This chapter gives an overview of the filtering options available on ingress and egress. Filtering allows different types of packets to be accepted or dropped.

A filter is applied at the source port as packets enter the switch core. This is set up in the **Ingress Port Packet Type Filter** register.

When the packet is ready to be queued, the **Ingress Egress Port Packet Type Filter** is applied for each egress port the packet is to be queued onto. If the packet is dropped then a drop counter is updated for each packet which is dropped.

Before a packet is to be sent out, the egress port it is checked in the **Egress Port Configuration** to see if the packet is allowed to be sent out.

The settings are unique for each port.

A packet of a certain type may be allowed to enter on a certain ingress port. But this does not mean the frame is ultimately allowed to be transmit, since ingress and egress port filters are setup independently.

In addition to the egress port packet type filter, there is also a source port filter on the egress port. This is found in **srcPortFilter**. The source port filter on the egress port allows a user to decide whether packets from a certain source port are allowed to be sent out on an egress port. The outcome of the filtering options are either to drop a packet, or to allow it.

Since the source port table, vlan table and egress port configuration can all have VLAN operations which changes the packet, it is important to understand on which packet the filtering is actually done.

- The source port filtering is done on the packet as it enters the switch without any packet modifications.
- The ingress egress port filtering is done on the packet after the source port and VLAN table VLAN operations. The L2 Multicast is calculated in the same way as MBSC register **L2 Multicast Handling**.
- The egress port filtering is done after all the VLAN operations has been carried out including the egress ports own VLAN operations.

Note that if a user defined VLAN tag is pushed, it will always be regarded as a C-VLAN tag by the filtering.

Chapter 13

Hashing

Hashing is used to enable the use of SRAM memories instead of using CAMs for lookups.

13.1 Hashing Functions

This section describes the hash functions used in this core.

13.1.1 MAC Table Hashing

The hash function receives the destination MAC address and GID as an input and it returns a hash with the same bit width as the address for the **L2 DA Hash Lookup Table** divided by number of buckets (4). The table is divided into equal sized parts/buckets which are readout in parallel.

Hash Function for MAC Table

The XOR hash function splits the key into 6 parts, each with the width of the hash value. To obtain the hash value a bitwise XOR is performed on all the parts.

When learning random MAC addresses the hash function results in an average utilization of the L2 table of 34% (including/excluding multicast addresses does not change this). When learning sequential MAC addresses (such as in the RFC2889) the utilization is 100%.

Python code for the hashing function is shown below as well as a test case to clarify how the key is calculated.

```
def calc_l2_hash( key ):
  """ key: 60 bits hash key
          key[59:48] = GID
          key[47:0] = MAC
      fold count = 6
      returns: 10 bits hash value
  """
  hashval = key & 0b1111111111
  hashval = hashval ^ (key>>10)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>20)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>30)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>40)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>50)
  hashval = hashval & 0b1111111111
  return hashval
```

```
def mac_str2int( mac_adr ):
    """ Convert Ethernet MAC address from string format, e.g.'46:61:62:bc:84:dd'
    to integer. """
    hx = ''.join(mac_adr.split(':'))
    return int(hx,16)

def l2_hash( gid, mac ):
  """ Calculate index into L2 hash table from GID and MAC address.
      Both parameters must be integers """
  key = (gid & 0xfff) << 48
  key |= mac & 0xffffffffffff
  return calc_l2_hash( key )



def l2_hash_test():
    # Simple test of the hash function to clarify how the key is calculated.
    # MAC: 46:61:62:bc:84:dd (leftmost byte is first byte received)
    # GID:478
    key = (478)<< 48 | 0x466162bc84dd
    hashval = calc_l2_hash(key) # the hash value is used as index into the L2 DA Hash T
    assert hashval == 611
```

13.1.2 IP Table Hashing

The hash function receives the destination IP address and VRF as key and returns a hash with the same number of bits as the address for the **Hash Based L3 Routing Table** .

Hash Function for IPv4

The XOR hash function splits the key into parts, each with the width of the hash value. To obtain the hash value a bitwise XOR is performed on all the parts.

When learning random IPv4 addresses the hash function results in an average utilization of the hash table of 17% .

Python code for the IPv4 hashing function is shown below as well as a test case to clarify how the key is calculated.

```
def calc_l3_ipv4_hash( key ):
  """ key: 34 bits hash key
          key[33:32] = VRF
          key[33:0] = IP address
      fold count = 4
      returns: 10 bits hash value
  """
  hashval = key & 0b1111111111
  hashval = hashval ^ (key>>10)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>20)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>30)
  hashval = hashval & 0b1111111111
  return hashval

def ipv4_str2int( ip_addr ):
    """ Convert IPv4 address from string format, e.g. 192.168.0.123,
```

```
        to integer """
    parts = ip_addr.split('.')
    res = 0
    for p in parts:
        res <<= 8
        res |= int(p)
    return res

def l3_ipv4_hash( vrf, ip_addr ):
  """ Calculate index into L3 hash table from VRF and IP address.
      Both parameters must be integers. """
  key = (vrf & 0x3) << 32
  key |= ip_addr
  return calc_l3_ipv4_hash( key )



def ipv4_hash_test():
    # Simple test of the hash function to clarify how the key is calculated.
    # IP: 70.119.98.188 (leftmost byte is first byte received)
    # VRF:3
    vrf = 3
    ip = 0x467762bc
    key = ( vrf << 32 ) | ip
    # the hash value is used as index into the Hash Based L3 Routing Table
    hashval = calc_l3_ipv4_hash(key)
    assert hashval == 782
```

Hash Function for IPv6

The XOR hash function splits the key into parts, each with the width of the hash value. To obtain the hash value a bitwise XOR is performed on all the parts.

When learning random IPv6 addresses the hash function results in an average utilization of the hash table of 16% .

Python code for the IPv6 hashing function is shown below as well as a test case to clarify how the key is calculated.

```
def calc_l3_ipv6_hash( key ):
  """ key: 130 bits hash key
          key[129:128] = VRF
          key[129:0] = IP address
      fold count = 13
      returns: 10 bits hash value
  """
  hashval = key & 0b1111111111
  hashval = hashval ^ (key>>10)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>20)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>30)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>40)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>50)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>60)
```

```
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>70)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>80)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>90)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>100)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>110)
  hashval = hashval & 0b1111111111
  hashval = hashval ^ (key>>120)
  hashval = hashval & 0b1111111111
  return hashval

def l3_ipv6_hash( vrf, ip_addr ):
  """ Calculate index into L3 hash table from VRF and IP address.
      Both parameters must be integers. """
  key = (vrf & 0x3) << 128
  key |= ip_addr
  return calc_l3_ipv6_hash( key )



def ipv6_hash_test():
    # Simple test of the hash function to clarify how the key is calculated.
    # IP: d8a7:da8b:: (leftmost byte is first byte received)
    # VRF:3
    vrf = 3
    ip = 0xd8a7da8b00000000000000000000000
    key = ( vrf << 128 ) | ip
    hashval = calc_l3_ipv6_hash(key)
    # the hash value is used as index into the Hash Based L3 Routing Table
    assert hashval == 559
```

13.1.3 MPLS Table Hashing

The hash function receives the outermost MPLS label, source port number and VRF as key and returns a hash with the same number of bits as the address for the **Hash Based L3 Routing Table**

Hash Function for MPLS

The XOR hash function splits the key into parts , each with the width of the hash value. To obtain the hash value a bitwise XOR is performed on all the parts.

When storing random MPLS labels the hash function results in an average utilization of the hash table of 17% .

Python code for the MPLS hashing function is shown below as well as a test case to clarify how the key is calculated.

```
def calc_l3_mpls_hash( key ):
  """ key: 26 bits hash key
          key[25:24] = VRF
          key[23:4] = MPLS label
          key[3:0] = source port
      fold count = 3
      returns: 10 bits hash value
```

```
    """
    hashval = key & 0b1111111111
    hashval = hashval ^ (key>>10)
    hashval = hashval & 0b1111111111
    hashval = hashval ^ (key>>20)
    hashval = hashval & 0b1111111111
    return hashval

def l3_mpls_hash( vrf, source_port, label ):
    key = (vrf & 0xfff) << 24
    key |= label & 0xfffff << 4
    key |= ( source_port & 0xf )
    return calc_l3_mpls_hash( key )



def mpls_hash_test():
    # Simple test of the hash function to clarify how the key is calculated.
    # MPLS label: 889984 (leftmost byte is first byte received)
    # VRF:3
    # source port:8
    mpls_label = 889984
    vrf = 3
    srcport = 8
    key = (vrf << (4 + 20) |
           srcport << 20 |
           mpls_label)
    hashval = calc_l3_mpls_hash(key)
    # the hash value is used as index into the Hash Based L3 Routing Table
    assert hashval == 989
```

Chapter 14

Learning and Aging

The switch supports automatic hardware learning and aging as well as software controlled learning and aging.

- With hardware learning the switch can be functional after reset without any software setup. The hardware learning engine saves the source port number, the source MAC address with a Global Identifier (GID) from the **VLAN Table** in the forwarding information base.
- If the destination MAC address and the GID of a packet is in the L2 forwarding information base, the L2 forwarding process will know the destination port of this packet.
- If a learned {GID, MAC} has not been hit by a source or destination MAC address for a while, the hardware aging engine will remove this entry from the table.
- When a learned MAC address is received as MAC SA on a different port than it was setup in the **L2 Destination Table**, it is considered a port move.
- When the hardware aging is enabled, all non-static entries will be aged out after a certain silent period. **Hardware Learning Configuration** configures the initial status of the newly learned entries.
- The software learning and aging feature allows users to fully control the L2 forwarding information base.
- The hardware learning and aging functions are by default turned on and can be turned off through the **Learning And Aging Enable** register.
- When the hardware learning is enabled, all source ports are allowed to get their unknown source MAC address learned. By setting **learningEn** field in the **Source Port Table** to 0 the learning process can be disabled on the corresponding source port.
- For an unknown MAC DA, **dropUnknownDa** field in the **Source Port Table** determines either to drop the packet or allow it to be flooded.

14.1 L2 Forwarding Information Base (FIB)

Multiple tables in groups are involved in the learning and aging functions when making L2 forwarding decisions:

14.1.1 Tables for MAC DA lookup

1. L2 Hash tables.
   (a) **L2 DA Hash Lookup Table**
   (b) **L2 Aging Status Shadow Table**
2. L2 Collision tables.

   (a) **L2 Lookup Collision Table**

   (b) **L2 Aging Collision Shadow Table**

3. **L2 Destination Table**.

4. **L2 Multicast Table**.

MAC DA lookups are used to find L2 forwarding destinations and the related tables are written as results from learning or aging functions. The forwarding function relies on a hash algorithm described in Section MAC Table Hashing and a search algorithm described in Section L2 Destination Lookup. In this core, destination MAC addresses and GIDs are combined together to create a 60-bit hash key and the hash function returns a 10-bit hash value.

14.1.2 Tables for MAC SA lookup

1. **L2 SA Hash Lookup Table**. Holding the same contents as **L2 DA Hash Lookup Table**.

2. **L2 Aging Status Shadow Table - Replica**. Holding the same contents as **L2 Aging Status Shadow Table**.

3. **L2 Destination Table - Replica**. Holding the same contents as **L2 Destination Table**.

The MAC SA lookups are used to create new learning requests and requiring the same tables as MAC DA lookups. Due to the fact that the core mostly uses tables with single read port towards the ingress processing pipeline, there are three MAC DA tables duplicated to MAC SA tables listed above to support one read per cycle from the ingress processing pipeline (one MAC DA lookup and one MAC SA lookup at every clock cycle). No matter when the MAC DA/MAC SA lookup tables are updated, the corresponding SA/DA lookup tables need to be filled with the same updates. The L2 collision tables are built to support parallel read by both DA and MAC SA lookups and therefore are not duplicated.

The MAC SA lookups form a key-hash pair by {GID,MAC SA} and do a two step check:

1. Hit or not. Hit is given in two cases:

   (a) The key-hash pair is found in the **L2 SA Hash Lookup Table** and the related entry in **L2 Aging Status Shadow Table - Replica** is valid.

   (b) The key is found in the **L2 Lookup Collision Table** and the related entry in **L2 Aging Collision Table** is valid.

2. The source port number matches the port number in the L2 destination table.

Based on the lookup result there are three possible learning decisions:

1. Learn a new entry: Not hit.

2. Port move request: Hit with port number mismatching.

3. SA hit update operation: Hit with port number matching.

Figure 6.1 demonstrates how the FIB addressing looks like.

14.1.3 Status Tables

1. **L2 Aging Table**

2. **L2 Aging Collision Table**

The status tables are located inside the learning and aging engine to monitor and maintain the status of all entries in the FIB. An FIB entry has three status bits:

1. **`valid`**: Indicate if a hit in the FIB is valid.

2. **`stat`**: Indicate if an entry is static. Static entries cannot be modified by hardware.

3. **`hit`**: Indicate either MAC SA or DA has successfully hit this entry since the last aging scan.

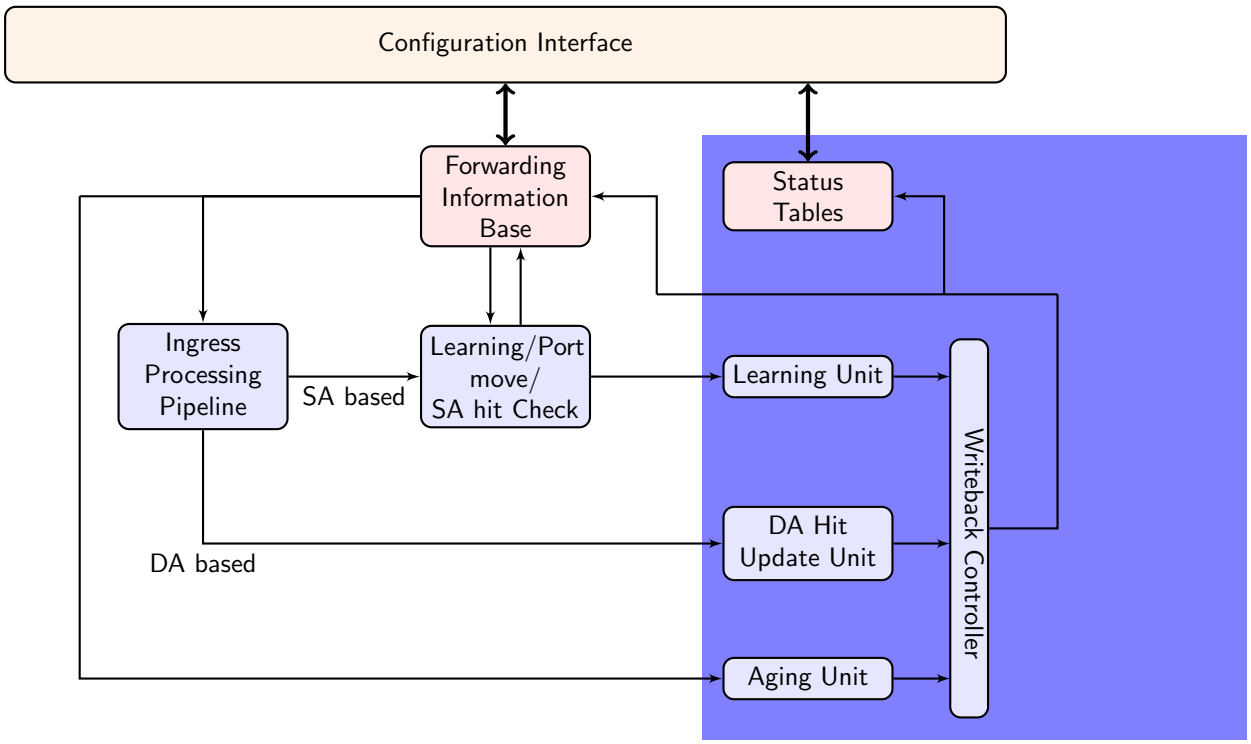


Figure 14.1: Learning and Aging Engine

When the hardware learning or aging updates the status table, the **valid** bit will be copied to the shadow tables in the ingress processing pipeline.

As in Figure 14.1 the FIB can be accessed from three units:

1. From software through the configuration interface: read and write.
2. Learning and aging unit: read and write.
3. Ingress processing pipeline: read only.

Notice that shadow tables in the FIB have to be updated simultaneously with status tables. MAC SA lookup tables have to be updated simultaneously with MAC DA lookup tables. Unexpected behavior will occur if the tables do not have the same content.

14.1.4 Hash Collision Accommodation

In order to solve hash collisions, the **L2 DA Hash Lookup Table** has 4 buckets each with 1,024 entries. A given key-hash pair can search in the 4 buckets in parallel by reading from the address that equals the hash value. The 4 buckets entries are all compared with the {GID,MAC DA} key and if one entry is equal to the key that entry is considered a match.

Besides the **L2 DA Hash Lookup Table**, there is an extra **L2 Lookup Collision Table** in case the number of hash collisions is more than the **L2 DA Hash Lookup Table** can handle. For instance, if the hash function calculated the same hash value for more than 4 keys, the first 4 keys can be accommondated in the 4 buckets of **L2 DA Hash Lookup Table** while the rest are stored in the **L2 Lookup Collision Table**. Searching in the **L2 Lookup Collision Table** will return the first entry index that holds the corresponding key.

Addressing into the **L2 Destination Table** is based on the hit index from either the **L2 DA Hash Lookup Table** or the **L2 Lookup Collision Table**.

- Hit in the **L2 DA Hash Lookup Table**: get a 12-bit hit index with the hash value in the lower 10

bits and the bucket number in the higher 2 bits. The corresponding **L2 Destination Table** address equals the hit index.

- Hit in the **L2 Lookup Collision Table**: get a 4-bit hit index from the hit entry address. The corresponding **L2 Destination Table** address is (hit index + 4,096).

14.2 Hardware Learning and Aging

14.2.1 Learning Unit

The core has a dedicated learning unit in hardware, which is tasked with learning L2 MAC addresses combined with GIDs as entries to do L2 destination port lookups. A new learning request is created and processed in several steps:

1. For every packet a learning check is performed based on its MAC SA and GID and issues learning requests to the learning unit.
2. If it is a known entry but the **hit** bit in the status table is 0, the **hit** bit will be refreshed to 1.
3. If the learning request is to learn a new entry, **Hardware Learning Counter** will be checked against the **learnLimit** in **Hardware Learning Configuration**. **learnLimit** limits the maximum number of entries can be learned on a port.
4. If the maximum learning limit is not reached on a port, the status table lookup will try to provide an available entry in a certain order:
   (a) Find a free entry.
      i. Select a free bucket for this hash value.
      ii. If all hash buckets are used, select a free collision table entry.
   (b) If there is no free entry and **lru** in the **Learning And Aging Enable** register is 0, the learning unit will search in the collision table and overwrite the non-static entries in a round robin order.
   (c) If there is no free entry and **lru** in the **Learning And Aging Enable** register is 1, the learning unit will overwrite a least recently used non-static entry as follows:
      i. Search in hash buckets for a bucket with **hit**=0 and **stat**=0. Return the last match.
      ii. If all buckets have **hit**=1 or **stat**=1, search in the collision table for an entry with **hit**=0 and **stat**=0. Return the first match.
   (d) If all entries are static or have been hit since the last aging scan, overwrite a non-static entry.
      i. Search in hash buckets for a bucket with **stat**=0. Return the last match.
      ii. If all buckets are static, search in the collision table for an entry with **stat**=0 in a round robin order.
5. If the learning unit failed to accomondate the unknown MAC SA and GID combination, or the learning limit on a port is reached, the learning request will be ignored and the corresponding MAC SA, GID and port number will be updated to the **Learning Overflow** register.
6. If a valid entry is found, the learning unit will link it to the port number from the learning request as a L2 unicast entry.
7. If the learning request is for a port move, the process will operate on existing non-static entries directly. For static entries, the **Port Move Options** register gives optional operations for each previously learned port.
8. If the learning unit failed to execute port move due to immutable static entry or the learning limit is reached, the learning request will be ignored and the corresponding MAC SA, GID and port number will be updated to the **Learning Conflict** register.

9. A valid learning decision is sent to a writeback bus which manages all decisions from different learning and aging units. The learning decisions have the highest priority to use the writeback bus.
10. The writeback bus sends decisions to the FIB.

14.2.2 Hardware Learning Exceptions

The switch support fine granular control to allow certain packets with unknown MAC SA address to not be learned. These settings described below enables a varity of different ways to turn it off on a per packet basis.

- Source port exceptions.
  - If **uniqueCpuMac** is set to 1, the CPU port cannot be learned.
  - If the packet from the CPU port has a from CPU tag, it will bypass L2 lookup hence bypass the learning process.
  - For any source port if its **learningEn** is set to 0 the learning process is disabled.
- To CPU packet. If the packet is sent to the CPU port with a non-zero reason code. [1]
- Classification.
  - If the packet hit in a classification rule that override L2 lookup (i.e. force the destination port), it will not be learned.
- Routed. A routed packet will not be learned.
- Dropped. If the ingress processing drops the packet (post-ingress processing is not counted), the packet will not be learned unless it is due to the ingress spanning tree drop and the state says **`Learning`**. [2]
- Multicast MAC SA. In the switch core a MAC address with the least-significant bit of the first octet equals 1 (e.g. 01:80:c2:00:00:00) but not equals to ff:ff:ff:ff:ff:ff is marked as Ethernet multicast address. By default a MAC SA that matches an Ethernet multicast address will not be learned. This can be configured per port through the **learnMulticastSaMac** field in the **Source Port Table**.

14.2.3 Aging Unit

When a new L2 entry is learned by the hardware learning unit, the initial entry status is from the **Hardware Learning Configuration** register. A valid non-static entry will be aged out if no L2 MAC SA/DA lookup hit it within a certain time and static entries must have software interactions to get aged/changed. By default a non-static entry will be learned with both **hit** and **valid** set to 1 to prevent it from being aged out immediately. Static entries can be established on a per source port basis by setting the **stat** field in **Hardware Learning Configuration** to 1.

The hardware aging function does a periodic check of the L2 entry status in the **L2 Aging Table** and the **L2 Aging Collision Table**. The waiting period between two checks is tick based [3] and configurable via the **Time to Age** register. During an aging check period, the aging unit loops through all entries in the **L2 Aging Table** and **L2 Aging Collision Table** to get the current status. The possible updates are listed in Table 14.1. If the **`valid`** bit (bit 0) is turned to 0 the entry is aged out. An aged out entry can be learned again.

If the **Time to Age** register is reconfigured during runtime, the updated **tickCnt** will not be available to aging unit until the current aging period is complete. In order to load new values immediately, the aging unit needs to be restarted via the **agingEnable** field in the **Learning And Aging Enable** register. However, changes to the **tick** selection are always applied immediately.

[1] Check all reason codes in Table 24.2
[2] See more in Chapter Spanning Tree.
[3] The system ticks are described in Chapter Tick.

Current Status	Update Status
0b101	0b001
0b001	0b000(entry cleared)
Other values	No update

Table 14.1: Hardware Aging Operations

14.2.4 MAC DA Hit Update Unit

The learning unit has a built-in MAC SA hit update unit to refresh the **hit** bit while another MAC DA hit update unit can operate in parallel. The MAC DA hit update unit can be turned on or off by the **daHitEnable** field in the **Learning And Aging Enable** register and works as such:

1. A packet with L2 MAC DA lookup returns a valid and non-static entry issues a hit update request for the corresponding MAC DA.
2. A hit update FIFO is prepared to buffer the update requests.
3. A hit update request is popped from the FIFO when the writeback bus is free.
4. If the writeback bus keeps busy with learning decisions and causes a buildup in the hit update FIFO, new hit update requests will be ignored when the FIFO is full.
5. The writeback bus forwards the hit update request to the FIB.

According to Table 14.1, the automatic **hit** bit update for an non-static L2 entry will keep the hardware aging unit away from setting the **valid** bit to 0, hence avoid aging out the entry.

14.3 Software Learning and Aging

Instead of automatic learning and aging, the switch provides an option for software to manipulate learning and aging behaviors.

14.3.1 Direct Access to FIB

All tables in the FIB allow direct software writes through a configuration interface. However, the learning and aging engine may constantly update the FIB. Before updating the FIB from the configuration interface the learning and aging engine needs to be turned off through the **Learning And Aging Enable** register to avoid hazards. An alternative approach is to use reserved static entries as described in Section Software Reserved Entry.

If the hardware learning unit needs to be turned on again after software setups, it is important to write to both L2 aging tables and the corresponding shadow tables while setting valid entries. Partial validation will cause inconsistencies between the L2 forwarding process and the learning and aging engine. Since the FIB consists of multiple tables it is recommended that the shadow tables are updated in the last step, to ensure the data consistency.

14.3.2 Software Reserved Entry

If the **stat** field in the **L2 Aging Table** is set to 1 and the **valid** field is set to 0, the corresponding entry in the FIB is considered as a reserved static entry and can be used for future software configuration. A reserved static entry is not used for L2 forwarding and is not available as a hardware learning entry.

A typical use case is to pre-allocate entries for L2 multicast. The hardware learning unit can automatically learn L2 unicast but not L2 multicast. One way to reserve entries for L2 multicast is to create a reserved static bucket, i.e. choose one bucket from the L2 hash table and make all entries reserved static. This approach allows the software to update entries in the reserved bucket during traffic without checking hash collisions, and without turning off the hardware learning and aging engine.

Chapter 15

Spanning Tree

Spanning-Tree Protocol (STP) and Multiple Spanning-Tree Protocol (MSTP) support is provided in order to create loop-free logical topology when several ethernet switches are connected. Through registers the STP state of the ports can be controlled by the host SW. The default behavior at power up is that spanning tree is not enabled and spanning tree functionality must therefore be configured by SW before it can be used. A switch running the spanning-tree protocols utilizes BPDU (Bridge Protocol Data Unit) frames to exchange information with other switches in order to decide how to configure it's ports to get a loop-free (tree) logical network topology.

BPDUs are forwarded to the CPU based on the used destination address. By default the MAC multicast addresses 01:80:C2:00:00:00 and 01:00:0C:CC:CC:CD are forwarded to the CPU. Modifications of this is possible through the register **Send to CPU**.

In order to be able to forward BPDU frames from the CPU to other switches on egress ports where general forwarding is currently not allowed, the bit **enable** in register **Forward From CPU** shall be set.

More information on the forwarding features to and from the CPU port is available in Chapter 24

15.1 Spanning Tree

The Spanning-Tree Protocol (STP) state for a port can be independently configured for source and egress behaviors to allow precise management. For ingress in the **spt** field of **Source Port Table**. Similarly for egress, the STP state can be configured in the **sptState** in the **Egress Spanning Tree State**. When STP is used on a port, all the port's associated MSTP instance states (ingress and egress) shall be **Forwarding**, i.e. MSTP is not enabled for this port.The behavior of the different STP states. The difference between Ingress and Egress STP state is only that learning is not affected by the Egress state.

- Blocking and Listening
  Learning is disabled and no frames are forwarded except BPDU which will be forwarded to the CPU. Frames that are not forwarded is counted in a drop counter.
- Learning
  Learning is enabled but no frames are forwarded except BPDU which will be forwarded to the CPU. Frames that are not forwarded is counted in a drop counter.
- Forwarding and Disabled
  Normal operation, learning is enabled and normal switching. BPDU frames will be forwarded to the CPU.

15.2 Multiple Spanning Tree

When VLANs are used in a network there is a need for the Multiple Spanning Tree Protocol (MSTP) to manage the individual spanning-tree instances for the different VLANs. If an incoming frame doesn't have an assigned VLAN membership it will get a default VLAN membership automatically as described

in Chapter 5. VLAN membership decides which MSTP instance (MSTI) the frame belongs to. Hence, all frames will belong to an MSTI. The **msptPtr** in the register **VLAN Table** is an index to the MSTI tables which the packet shall be assigned to. The port's states of this MSTI are available in the tables **Ingress Multiple Spanning Tree State** and **Egress Multiple Spanning Tree State** for ingress and egress respectively. When a port uses MSTP it's STP states (source and egress) shall be set to **Disabled**, i.e. STP is not enabled for this port.

15.3 Spanning Tree Drop Counters

When a port's ingress or egress spanning tree states causes a frame to be dropped, the frames direction and spanning-tree state are used to select which drop counter to increase with one. The available drop counter registers are:

- **Ingress Spanning Tree Drop: Listen**
- **Ingress Spanning Tree Drop: Learning**
- **Ingress Spanning Tree Drop: Blocking**
- **Egress Spanning Tree Drop**

The ingress registers are common for all ports. There is one egress register per port.

The registers above are also used to count MSTI-state caused frame drops. A port's ingress-MSTI drop-causing state is mapped as follows: The state **Learning** is mapped to the register **Ingress Spanning Tree Drop: Learning** and **Discarding** to **Ingress Spanning Tree Drop: Blocking**. For a port's egress MSTI, both the states **Learning** and **Discarding** are mapped to the port's generic egress drop counter **Egress Spanning Tree Drop**.

Chapter 16

Token Bucket

This core provides a rich set of QoS functions, and when a function needs to compare the internal packet or byte rate to a configurable rate, we use token bucket as the basic measurement component. A token bucket is usually combined with packet classifications, packet colorings or the shared buffer memory to achieve metering, marking, policing or shaping with different granularities.

A token bucket has four key parameters:

- bucket capacity
- bucket threshold
- initial tokens in the bucket
- token fill in rate

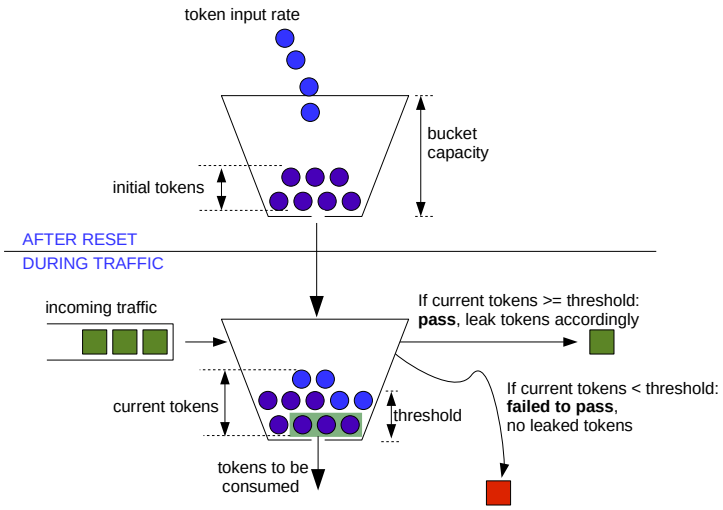


Figure 16.1: General Token Bucket Illustration

Figure 16.1 shows a token bucket with adjustable bucket threshold, the remaining tokens below the threshold can be used to handle the burst. This type of token bucket is used by:

- multicast broadcast storm control

In different QoS functions, tokens are represented as packets or bytes. The token fill in rate is achieved by periodically adding a certain number of tokens to the bucket and the fill in frequency is determined by one of the five core ticks.

Chapter 17

Egress Queues and Scheduling

The order of packet output on each egress port is decided by a complex interaction of back-pressure and different QoS functions, but at the heart of the matter is the the egress queue. The egress queues are the lists of packet pointers created by the queue manager when packets have been written to the packet buffer. Each egress port has eight such queues.

When a packet has been written in full to the packet buffer, the queue manager will add pointers to the packet to the end of at least one egress queue[1].

More than one egress port may get the packet linked (due to multicast), but on any single port the same packet may only be linked once. You cannot have the same packet in more than one egress queue on any single egress port.

The order in each egress queue is fixed. Once the packets are linked, the order cannot be changed. What QoS functions and back-pressure can affect is the order in which the packets in different queues are output.

The egress queue is determined by the ingress packet processing. If a packet is forwarded to multiple egress ports, each packet instance will have the same egress queue assigned.

17.1 Determine Egress Queue

Figure 17.1 describes how the egress queue is determined. If a configuration in the diagram includes a reference number in the end, the related field or register to setup can be found in the list below:

1. **forceQueue** in **Ingress L3/L4 ACL Result Operation Entries**
2. **forceQueue** in **Ingress L2 ACL Result Operation Entries**
3. **forceQueue** in **Reserved Source MAC Address Range**
4. **forceQueue** in **Reserved Destination MAC Address Range**
5. **prioFromL3** in **Source Port Table**
6. **IPv4 TOS Field To Egress Queue Mapping Table**
7. **IPv6 Class of Service Field To Egress Queue Mapping Table**
8. **MPLS EXP Field To Egress Queue Mapping Table**
9. **eQueue** in **Force Unknown L3 Packet To Specific Egress Queue**
10. **forceQueue** in **Force Non VLAN Packet To Specific Queue**

This process is completed only once per packet, and the result is applied to all destination ports for the packet. The input to the process can come from:

[1] That is unless the packet is to be dropped, because then the pointer is instead added to the end of the throw queue.

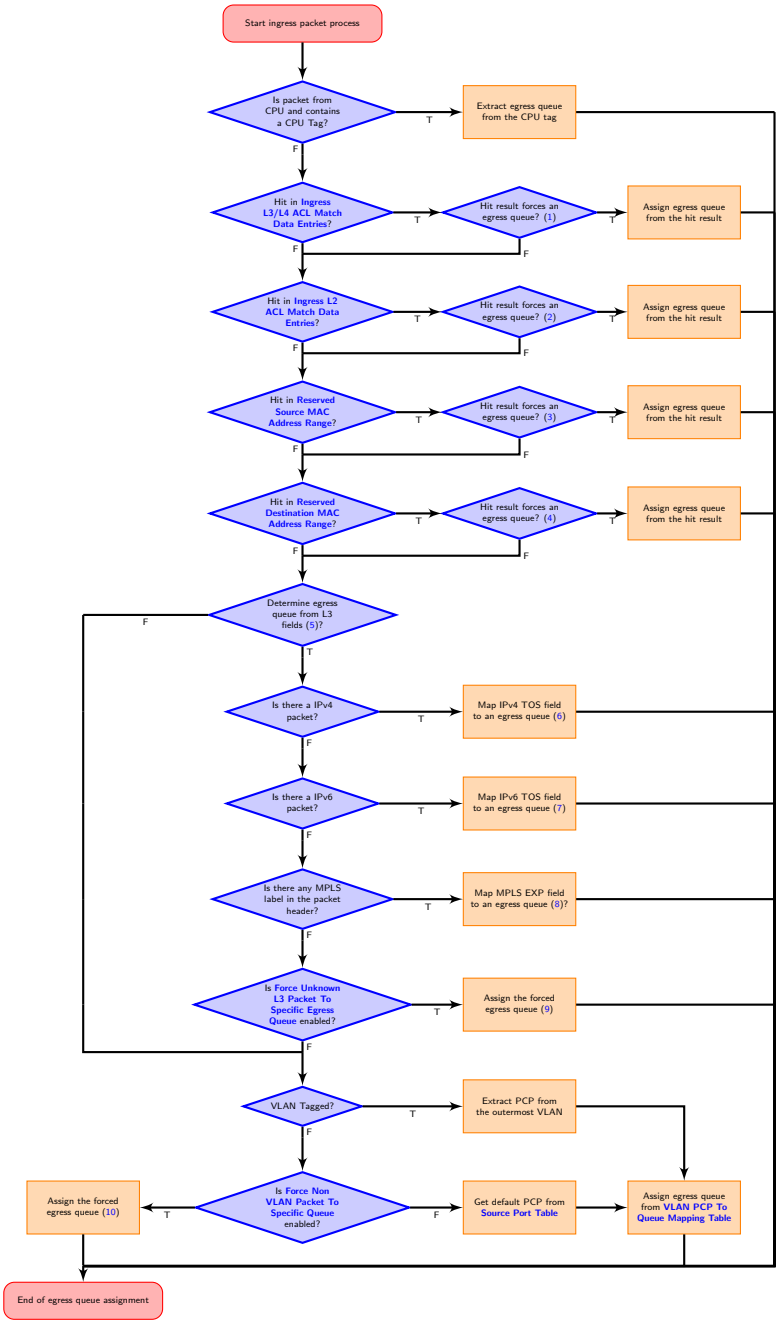


Figure 17.1: Egress Queue Selection Diagram

- Packet L2 headers
- Packet L3 headers
- Packet L4 ports
- Classification results

The available classification engines are described in the Classification chapter.

Egress queue from packet headers is operated under either trust L2 mode, to map egress queues from L2 headers, or trust L3 mode, to map egress queues from both L2 and L3 headers. In trust L2 mode, the egress queue can be mapped from:

- Priority code point(PCP) field from the outermost VLAN tag
- Source port default PCP when packet is non-VLAN tagged
- Optionally force non-VLAN tagged packets to the same egress queue, ignores source port based default mapping.

In trust L3 mode, a packet first tries to get its egress queue by mapping from:

- Type of Service (TOS)/DiffServ field from IPv4
- Traffic Class(TC) field from IPv6
- Traffic Class(TC)/EXP field from MPLS
- When none of the above are executed, the egress queue mapping under trust L3 mode will fall back on the trust L2 mode and get the egress queue from L2 headers of the packet.

17.2 Determine a packets outgoing QoS headers PCP, DEI and TOS fields

17.2.1 Remap Egress Queue to Packet Headers

This core supports remapping determined egress queues to outgoing packets' headers. These remappings are done first then if field **useEgressQueueRemapping** is set to one the remapping described in 17.2.2.

- Egress queue to next hop router VLAN PCP remapping:
  For routed packets, packets' original VLAN tags are removed and at most two next hop router VLANs are added. Egress queue can be mapped to the PCP field in these VLAN tags through the **Router Egress Queue To VLAN Data** table when:
  1. **innerVlanAppend** is set and its PCP field selection(**innerPcpSel**) chooses mapping from egress queue.
  2. **outerVlanAppend** is set and its PCP field selection(**outerPcpSel**) chooses mapping from egress queue.
- Egress queue to next hop router VLAN CFI/DEI remapping:
  Similar with next hop router VLAN PCP mapping, egress queue can be mapped to the CFI/DEI field in next hop router VLANs through the **Router Egress Queue To VLAN Data** table when:
  1. **innerVlanAppend** is set and its CFI/DEI field selection(**innerCfiDeiSel**) chooses mapping from egress queue.
  2. **outerVlanAppend** is set and its CFI/DEI field selection(**outerCfiDeiSel**) chooses mapping from egress queue.
- Egress queue to outgoing outermost VLAN PCP remapping:
  Egress port VLAN push or swap operation provides an option to map egress queue to the outgoing outermost VLAN PCP field. The mapping table is **Egress Queue To PCP And CFI/DEI Mapping Table** and the required configurations are:
  1. **vlanSingleOp** in **Egress Port Configuration** is *push* or *swap*.
  2. **pcpSel** in **Egress Port Configuration** selects mapping from egress queue.
- Egress queue to outgoing outermost VLAN CFI/DEI remapping:
  Similar with outgoing outermost VLAN PCP mapping, egress port VLAN push or swap operation provides an option to map egress queue to the outgoing outermost VLAN CEI/DEI field. The mapping table is **Egress Queue To PCP And CFI/DEI Mapping Table** and the required configurations are:
  1. **vlanSingleOp** in **Egress Port Configuration** is *push* or *swap*.

2. **cfiDeiSel** in **Egress Port Configuration** selects mapping from egress queue.

- Egress queue to MPLS TC/EXP remapping:
  Packets with MPLS labels have an option to map their egress queues to MPLS TC/EXP field when egressing the core. The mapping table is **Egress Queue To MPLS EXP Mapping Table** and the required configurations are:

  1. **mplsOperation** is *push* or *swap*.
  2. **expSel** in **Next Hop MPLS Table** selects mapping from egress queue.

17.2.2 Using Packet Type, Destination Port and Switching/Routing to do QoS Mappings

This core supports remapping determined by egress queues to outgoing packets' headers using the information if the packet was switched, routed, forwarded by classification rules, if the packet type was IP or MPLS and packets outgoing PCP, DEI, TOS and EXP fields. The steps to remap the packet are described below. The input values for PCP, DEI comes from the remapping tables described in 17.2.1. The TOS values comes from the **Color Remap From Ingress Admission Control** or **Color Remap From Egress Port**.

1. Determine Which Mapping Table To Use
   The mapping table to use to map the internal state to a the outgoing packet is determined by the table **Select Which Egress QoS Mapping Table To Use**. The packets destination port, packet type and packet forwarding type is used to calculate which entry to read out from the table. This table then points to the one of the QoS remapping tables which remapps the internal state to the outgoing packets PCP,DEI and potentially L3 fields such as TOS field . Since the address takes egress port, forwarding type and packet type into consideration there can be seperate rules setup for how to remap the fields in the outgoing packet.
2. Mapping Tables
   Use the Mapping tables to map into outgoing packets PCP,DEI , TOS and EXP values.
   (a) **L2 QoS Mapping Table**
   This table can be used for all packets being sent out. There exists 2which the field **whichTablePtr** points to which to use.
   (b) **IP QoS Mapping Table**
   This table can be used for IPv6 and IPv4 packets. There exists 2L3 mapping tables. This remaps part of the TOS byte which has to do with ECN and uses the higher TOS bits [7:2] from the coloring tables (**Color Remap From Ingress Admission Control** or **Color Remap From Egress Port**).
   (c) **TOS QoS Mapping Table**
   This table can be used for IPv6 and IPv4 packets. There exists 2TOS mapping tables. This remaps the whole of the TOS byte from **Color Remap From Ingress Admission Control** or **Color Remap From Egress Port** to a new TOS bytes along with PCP and DEI information. There is a support to remap to EXP values which can be used if the packet enters a MPLS tunnel in the Next Hop Tables
   (d) **MPLS QoS Mapping Table**
   This table can be used for MPLS packets. This remaps the outgoing packets PCP, DEI and EXP values. There exists 2TOS mapping tables.

17.3 Priority Mapping

The queues are prioritized in decending order, queue zero having the higest priority and queue seven the lowest.

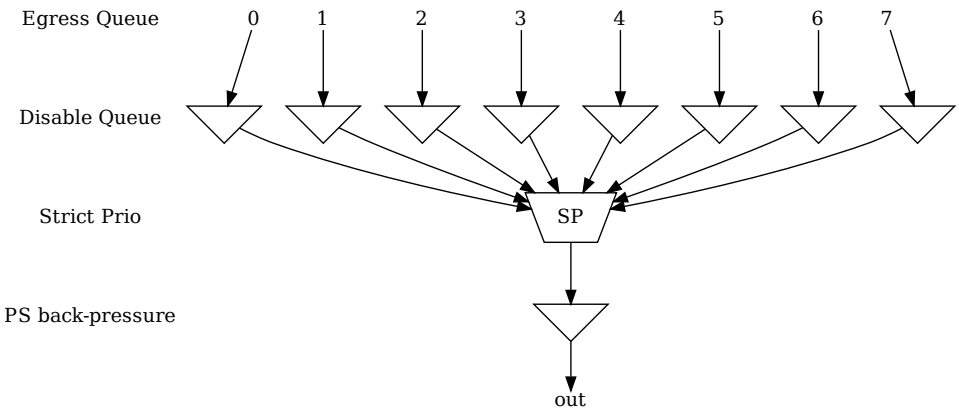


Figure 17.2: Egress Queue Scheduling Graph.

The core uses a strict priority scheduler which always serves the highest priority queue first. If the highest priority queue is empty it will move on to the second highest and so on. This will cause starvation of the lower priority queues if there is enough packets of in the higher priority queues to fill the port.

17.4 Queue Management

This core features a set of queue management operations which can be used by the CPU to monitor, redirect and disable queues and ports. The current size of the queues can be readout by using the **Egress Port Depth** and **Egress Queue Depth** registers, while the current total number of cells left available can be seen in the **Buffer Free** register. The minimum level reached since core was initialized is available in **Minimum Buffer Free**. From this status the CPU can take active actions to determine what the core shall do with the packets on the ports. The optional operations are listed below.

- Disable scheduling to port: Disable the core from scheduling a new packet for transmission on a specific port and queue. This is setup in the **Output Disable** register. This allows per-queue granularity of what packets gets scheduled on a specific port. The packets are still kept in the queues until the port or queue is enabled again.
- Disable queueing to port: Disable the enqueueing of packets to a specific port and queue. Once the corresponding bit in the **Enable Enqueue To Ports And Queues** register is cleared, no new packets will be queued to that egress queue. New packets destined to that specific queue will be dropped and the **Queue Off Drop** counter for the egress port will be incremented.
- Drain port: Drop all packets in all queues on one specific port. This allows the user to clear all packets which have been queued on a port. The register **Drain Port** is used to control this functionality. Statistics for this operation is collected in the **Drain Port Drop** counter.

17.5 How To Make Sure A Port Is Empty

First, so that no new packets are queued to the port, use the **Enable Enqueue To Ports And Queues** to disable all the queues on the port. If the already queued packets should not be sent out, then use the **Drain Port** functionality. Once this is done start to read out the **Packet Buffer Status** and check the bit which corresponds to the port. When the port bit is high, this means that all the queues on this port are empty.

Now, there may still be a few cells of data being processed in the egress packet processing pipeline, or stored in the parallel-to-serial memories. This data will be drained at the speed of the port, so the time from the port-bit going high in the **Packet Buffer Status** register to the port being truly empty will depend on the port speed.

Chapter 18

Packet Coloring

18.1 Ingress Packet Initial Coloring

This core marks packets with 3 colors internally to represent packet drop precedences. The three colors are coded as in Table 18.1.

Color	Code
Green	0
Yellow	1
Red	2

Table 18.1: Code for Colors

A packet's initial color is assigned according to L2/L3 protocols or classification results. It follows similar process steps as the egress queue assignment described in Section 17.1.

1. **forceColor** in **Ingress L3/L4 ACL Result Operation Entries**
2. **forceColor** in **Ingress L2 ACL Result Operation Entries**
3. **forceColor** in **Reserved Source MAC Address Range**
4. **forceColor** in **Reserved Destination MAC Address Range**
5. **colorFromL3** in **Source Port Table**
6. **IPv4 TOS Field To Packet Color Mapping Table**
7. **IPv6 Class of Service Field To Packet Color Mapping Table**
8. **MPLS EXP Field To Packet Color Mapping Table**
9. **forceColor** in **Force Unknown L3 Packet To Specific Color**
10. **forceColor** in **Force Non VLAN Packet To Specific Color**

A diagram in Figure 18.1 describes how initial colors are determined. All classification engines which can force egress queues also have an option to force packet initial colors. If none of the engines force the color and the initial color marking is operating under trust L2 mode, the color is mapped from:

- Priority Code Point(PCP) field with Drop Eligible Indicator(DEI) field from the ingress outermost VLAN tag.
- Source port default PCP with default DEI when packet is non-VLAN tagged.
- Optionally force non-VLAN tagged packets to the same specific initial color, ignores source port based default marking.

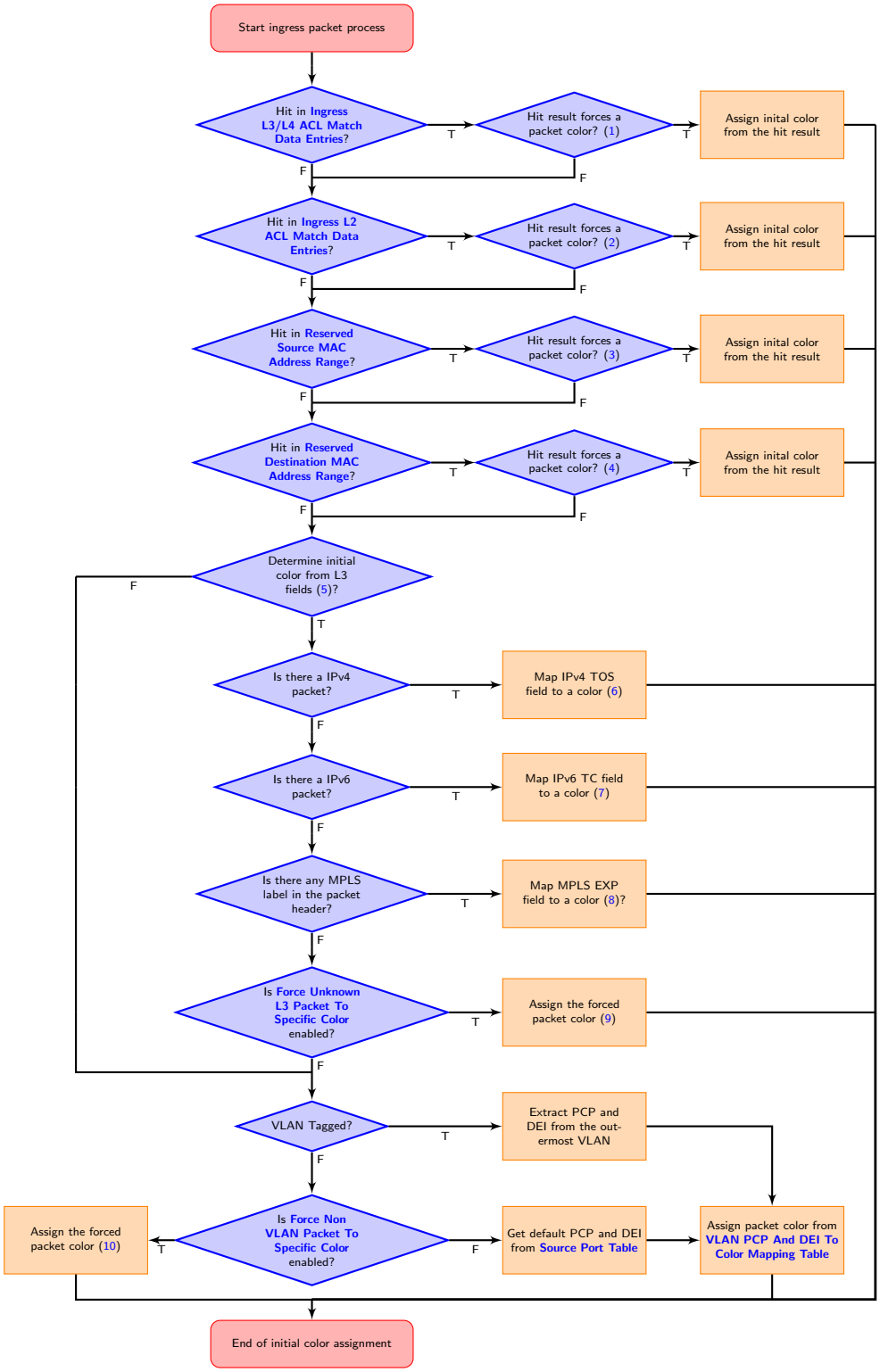


Figure 18.1: Packet Initial Color Selection Diagram

Otherwise, the initial color marking will be working under trust L3 mode and the color is mapped from:

- Type of Service(TOS)/DiffServ field from IPv4
- Traffic Class(TC) field from IPv6
- Optionally force non-IP packets to the same initial color.
- When none of the above markings are executed, the initial color marking under trust L3 mode falls into processes in trust L2 mode.

By default, green marked packets have low drop probability, yellow marked packets have medium drop probability and red marked packets have high drop probability. But the remarking process has its own configurable settings to decide if packets with a certain remarked color shall be dropped.

18.2 Remap Packet Color to Packet Headers

During egress packet processing, each egress port can be set as color aware or color blind through the **colorRemap** field in the **Egress Port Configuration** table. If an egress port is color blind, packets to that port will not have its color represented in packet headers. If an egress port is color aware, a color remap process is executed to optionally remap the egress packet color to outgoing packet headers.

When an egress port is color aware, the default remap options for that port are configured in the **Color Remap From Egress Port** table. If a packet to a color aware egress port has ingress admission control applied, its meter-marker-policer pointer can also provide color remap options from the **Color Remap From Ingress Admission Control** table. The **enable** field in the table determines whether to perform a color remap operation for each pointer.

The color remap has four modes:

- Skip/Disable:
  Color is not remapped to packet headers. This includes overriding previous color remap decisions.
- Remap to L3 only:
  Color is remapped to IPv4 TOS field or IPv6 TC field with an AND mask (tosMask). For each bit in the TOS/TC field, the update requires the corresponding bit in the mask set to one. i.e.

```
tos[i] = ( color2Tos[i] & tosMask[i] ) | ( tos[i] & (~tosMask[i]) )
```

- Remap to L2 only:
  A valid color remap updates the DEI bit in the VLAN tag of the outgoing packet. The updated DEI bit will not be changed during further egress packet processes. If there are more than one VLAN tag in the transmitted packet, the color to DEI mapping will be operated on the outermost VLAN.
- Remap to L2 and L3:
  Color is remapped to both L2 and L3 fields as listed above.

Chapter 19

Admission Control

19.1 Ingress Admission Control

This core features an ingress admission control unit to control the bandwidth of certain traffic types. If the traffic flow in a group exceeds the configured bandwidth it may get the packet color changed or get denied to be enqueued in the buffer memory.

Ingress admission control includes two main functions. The first function creates admission control groups to classify packets based on source information in packet headers or ACL matches. The second function measures the classified traffic rate against a certain policy to make permit/deny decisions. The decision may take the given packet color into account.

19.1.1 Traffic Groups

The traffic group is classified based on source port number and L2 or L3 packet headers. Initially packets are grouped by their source port numbers and L2 priorities, but during the subsequent admission control processes they may fall into other traffic groups. For each potential traffic group, three configurations are given to validate a policy:

1. mmpValid: Determine if there is a valid Meter-Marker-Policer(MMP) pointer. If there is no valid pointer through the entire process, the packet will not be classified to any traffic group.
2. mmpOrder: Order of the pointer. If a valid pointer exists, its order needs to be higher than the order of previously assigned pointers to override them.
3. mmpPtr: MMP pointer for this traffic group.

The process to set the MMP pointer is illustrated in Figure 19.1. A packet can only belong to one traffic group so hierarchical traffic groups are not possible.

The order of the classification sequence is:

1. Source port number and L2 priority:
   First assignment for traffic groups and MMP pointers. For VLAN tagged packet, L2 priority is from its outermost VLAN PCP field. For non-VLAN tagged packet, L2 priority is the default PCP based on the source port number (**defaultPcp** in the **Source Port Table**). Lookup in the **Ingress Admission Control Initial Pointer** table gives a base pointer and its order, also indicates if it is a valid pointer.
2. Source MAC:
   Source MAC hit an entry in the **Reserved Source MAC Address Range**.
3. Destination MAC:
   Destination MAC hit an entry in the **Reserved Destination MAC Address Range**.
4. Ingress VID:
   Lookup in **VLAN Table** based on the **ingress VID**.

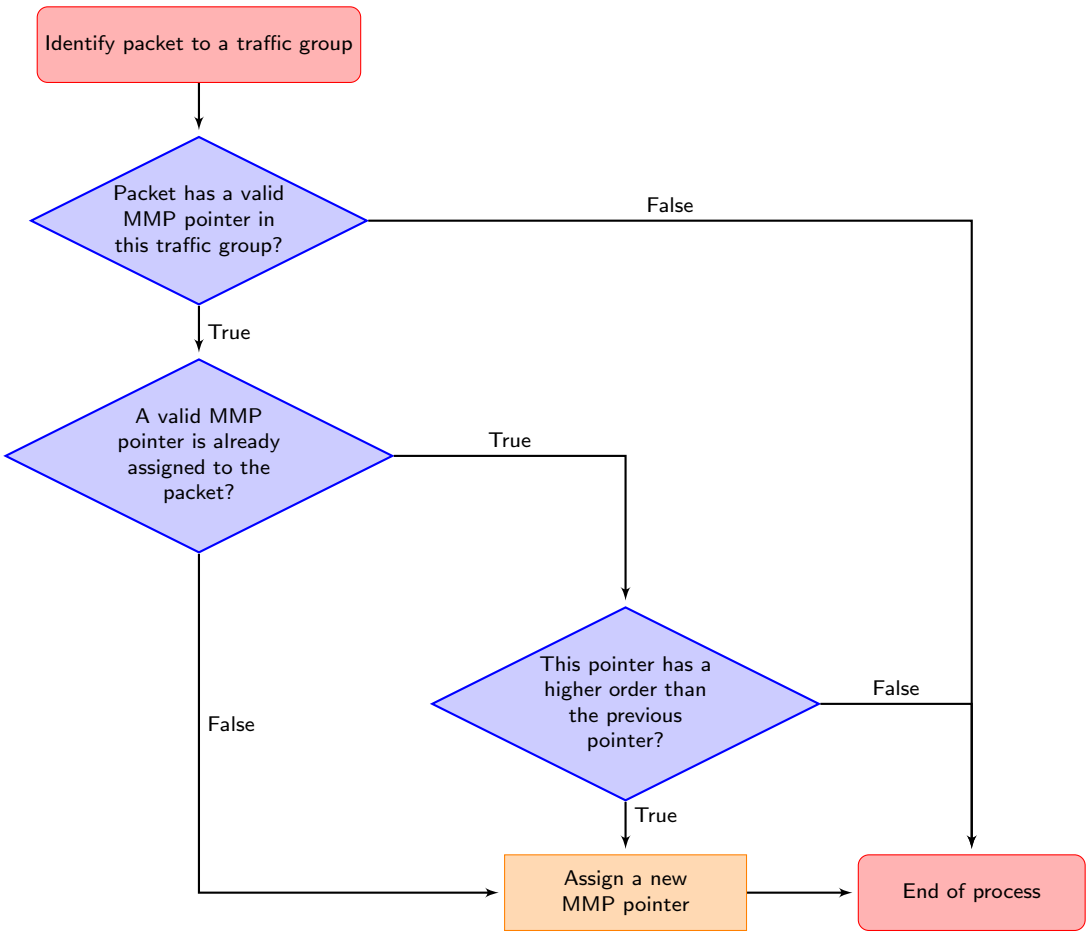


Figure 19.1: MMP pointer Selection Diagram

5. VRF:
   For a routed packet, lookup in **Ingress Router Table** based on its VRF.
6. L3 and L4 ACL:
   Hit in **Ingress L3/L4 ACL Match Data Entries**.'

When a packet arrives to ingress packet processing, it walks through ingress admission control classifications in the order above. A hit in one of the above groups will result in a pointer and a matching order. The pointer is linked to a policy/entry in a meter-marker-policer engine, which will measure the byte rate belonging to this entry. Although a packet can have multiple hits in traffic groups, it will finally fall into one pointer according to the order of the pointers. Later matches only win when they have a higher order than the previous ones.

19.2 Meter-Marker-Policer

An admission control unit contains a meter-marker-policer (MMP) bank where each MMP refers to one admission control policy. An MMP is based on token buckets, and each entry includes two configurable buckets.

The MMP bank used by ingress admission control consists of 16 policies/entries with three related tables.

1. **Ingress Admission Control Token Bucket Configuration**
2. **Ingress Admission Control Reset**

3. **Ingress Admission Control Current Status**

While only one ingress admission control policy is applied to any single packet, the same policy/entry can be pointed to from several different traffic types.

In the Ingress Admission Control , an MMP entry is configured through the **Ingress Admission Control Token Bucket Configuration** register to perform either a single rate three color marker (RFC2697: srTCM) or a two rate three color marker (RFC2698: trTCM). The selected marker is operated in either color-aware or color-blind mode, and the packet is marked with a new color when the rate exceeds a certain bandwidth. Based on the updated packet color, **dropMask** from register **Ingress Admission Control Token Bucket Configuration** decides whether the packet is allowed to be enqueued in the buffer memory.

An MMP entry has a **Ingress Admission Control Mark All Red Enable** option to permanently block the metering process and drop all packets with the corresponding MMP pointer. When **Ingress Admission Control Mark All Red Enable** is set to one, a packet drop on this entry will raise the **Ingress Admission Control Mark All Red** to one, then further packets to that entry will be dropped before metering. The blocking status can be cleared by writing zero to one of the two registers.

When an MMP is selected to be either srTCM or trTCM, it still requires configurations of the two token buckets to make it work properly.

- srTCM: Only the length, not the peak rate of the burst determines service eligibility.
  - Committed Information Rate (CIR): Combining **tokens 0** and **tick 0** to achieve the target rate. Details for tick is described in the **Tick** chapter. Configuration examples are shown in Table 19.1. Under srTCM mode, rate settings for the second token bucket (**tokens 1** and **tick 1**) will not take effect.
  - Committed Burst Size (CBS): **bucketCapacity 0**.
  - Excess Burst Size (EBS): **bucketCapacity 1**.
- trTCM: Enforce peak rate separately from the committed rate.
  - Committed Information Rate (CIR): **tokens 0** and **tick 0**.
  - Committed Burst Size (CBS): **bucketCapacity 0**.
  - Peak Information Rate (PIR): **tokens 1** and **tick 1**.
  - Peak Burst Size (PBS): **bucketCapacity 1**.
- Runtime configuration update:
  Any update to register **Ingress Admission Control Token Bucket Configuration** requires writing 1 to register **Ingress Admission Control Reset**. This will reset the buckets to the initial state.
- Status update from hardware:
  Besides **Ingress Admission Control Reset**, MMP has a another status register: **Ingress Admission Control Current Status**. It shows the number of tokens in each bucket. Hardware updates these two registers only when a metering process is done, hence **Ingress Admission Control Current Status** shows the number of tokens left in the bucket since the last token consumption in this bucket. **Ingress Admission Control Reset** is always changed back to 0 again after token consumptions.

Bandwidth	Token Bucket Update Frequency	Tick Index	Added Tokens Per Tick (bytes)
8000 bit/s	1KHz	3	1
16000 bit/s	1KHz	3	2
N*64000 bit/s	1KHz	3	N*8
N*1544000 bit/s	1KHz	3	N*193
N*56000 bit/s	1KHz	3	N*7
10M bit/s	10KHz	2	125
250M bit/s	10KHz	2	3125
N*1G bit/s	1Mhz	0	N*125

Table 19.1: Rate Configuration Example (Assume tickFreqList = [1MHz, 100KHz, 10KHz, 1KHz, 100Hz])

Chapter 20

Tick

All token buckets - and all other functions dependent on measuring time - in the core are basing their time measurements on the system ticks.

Tick number zero is the master tick. It is created by dividing the core clock by the number configured in the clkDivider field of the **Core Tick Configuration** register. The following tick signals (five in total) are created by dividing the previuous tick by a factor set up in the stepDivider field of the **Core Tick Configuration** register, so `tick1` is clkDivider slower than `tick0`, `tick2` is clkDivider slower than `tick1`, and so on.

If the **Core Tick Configuration** is updated during runtime, all features relying on the core tick need to be updated accordingly. Meanwhile, inaccurate time measurement will be performed until the first tick after the reconfiguration is generated.

By default the input to the Core Tick divider is the core clock, but using the **Core Tick Select** register the input to the tick divider can be disabled, or chosen to be driven from *debug_write_data* pin 0.

Chapter 21

Multicast Broadcast Storm Control

The multicast/broadcast storm control (MBSC) unit is used to make sure that a switch does not flood the network with too much multicast/broadcast traffic. The MBSC unit prevents several traffic types from transmitting to an egress port if the corresponding traffic rate on that egress port has exceeded a certain limit.

The basic component of the MBSC unit is a token bucket (illustrated in Figure 16.1). For each egress port there is one token bucket per inspected traffic type. In principle a token bucket controls the traffic rate (packet rate or byte rate) on an egress port. A token bucket operates as follows:

1. A configurable number of tokens are periodically added to the token bucket. The bucket level will saturate at the configured capacity.
2. When a packet of the traffic type is received a configurable number of tokens are consumed, i.e. the bucket level is decreased. The number of tokens consumed per packet is either packet length plus IFG adjustment or one per packet.
3. As long as the bucket level is at or above the threshold the bucket will accept all given traffic.
4. When the bucket level drops below the threshold all packets of the inspected traffic type, destined for the corresponding egress port, are dropped. Note that instances of the same packet destined for other egress ports are not affected and have their own token buckets to check the traffic rate.
5. The **MBSC Drop** counter will be incremented once for each egress port where the packet is dropped.

In this core three kinds of traffic are checked by the MBSC unit:

- L2 Broadcast
- L2 Flooding
- L2 Multicast

For each type of traffic there is an individual control unit, consisting of one token bucket per egress port. Every token bucket can be turned on or off separately through a control register (listed in the next section).

21.1 Inspected Traffic

- L2 Broadcast: A Packet with DA = ff:ff:ff:ff:ff:ff.
  - Token bucket configurations:
    - **L2 Broadcast Storm Control Enable**
    - **L2 Broadcast Storm Control Bucket Capacity Configuration**
    - **L2 Broadcast Storm Control Bucket Threshold Configuration**

        * **L2 Broadcast Storm Control Rate Configuration**

- L2 Flooding: A packet that will be L2 switched but the DA is unknown. In this case the packet is flooded to all VLAN member ports.
  - Token bucket configurations:
    * **L2 Flooding Storm Control Enable**
    * **L2 Flooding Storm Control Bucket Capacity Configuration**
    * **L2 Flooding Storm Control Bucket Threshold Configuration**
    * **L2 Flooding Storm Control Rate Configuration**
- L2 Multicast: A packet that will be L2 switched and has a known multicast DA MAC in the L2 tables. (The DA MAC has Ethernet multicast bit set to 1). The core can optionally include or exclude certain packets as L2 multicast traffic. The configuration is through the **L2 Multicast Handling** register.
  - Token bucket configurations:
    * **L2 Multicast Storm Control Enable**
    * **L2 Multicast Storm Control Bucket Capacity Configuration**
    * **L2 Multicast Storm Control Bucket Threshold Configuration**
    * **L2 Multicast Storm Control Rate Configuration**

21.2 Rate Configuration

From the configuration registers a token bucket can be shaped with its capacity, threshold and token settings. The L2 broadcast storm control is here used as an example to demonstrate the operations.

From the **L2 Broadcast Storm Control Rate Configuration** register a user can configure how tokens are consumed by a packet, and how new tokens are supplemented to the bucket.

- Token consumption
  1. The token bucket can be set to count either packets or bytes by the **packetsNotBytes** field. This setting puts a token bucket in either packet or byte mode to control the maximum packet rate or byte rate on an egress port respectively.
  2. – In packet mode, every L2 broadcast packet instance to an egress port will consume one token and the bucket value will be decreased by one.
     – In byte mode, every L2 broadcast packet instance to an egress port will consume as many tokens as there are bytes in the packet plus the specified IFG correction in the **ifgCorrection** field.
- Token Injection
  1. The token injection frequency is tick [1] based. The tick timer determines the time period between token injections. The **tick** field from the **L2 Broadcast Storm Control Rate Configuration** register selects which tick timer to use.
  2. When it is time to inject new tokens, the number of tokens that will be added is configured in the **tokens** field.
- Token bucket capacity and threshold. The two configuration registers **L2 Broadcast Storm Control Bucket Capacity Configuration** and **L2 Broadcast Storm Control Bucket Threshold Configuration** are used to setup how the token bucket handles traffic bursts.

By default the MBSC unit is operating in packet mode, and all token buckets are set to allow the inspected traffic to have at most 5% of the full packet rate for 64-byte packets. Python example code to configure the maximum packet rate to 5% follows:

[1] The system ticks are described in Chapter 20.

```
#!/usr/bin/python

rate    = 0.05

minLen  = 64 # bytes
slice   = 1 # switch slices
ifg     = 20 # bytes
pnb     = 1 # = packet mode
portBW  = 10000 # Mbits/s
tickFreqList = [1.0,
                0.1,
                0.01,
                0.001,
                0.0001] # Mhz

fullByteRate           = portBW/8.0
fullPktRate            = fullByteRate/(minLen+ifg)

pktRate  = fullPktRate*rate
pktTokenIn         = 10*slice

tick = len(tickFreqList)-1
for i in range(len(tickFreqList)):
    if tickFreqList[i] * pktTokenIn <= pktRate:
       tick = i
       break

pktTokenIn = int(1.0*pktRate   / tickFreqList[tick])

pktCap  = pktTokenIn * 20
pktThr  = pktTokenIn * 10

# Field settings for the rate configuration register
settings = {
    'packetsNotBytes' : pnb,
    'tokens'          : pktTokenIn,
    'tick'            : tick,
    'ifgCorrection'   : ifg,
    'capacity'        : pktCap,
    'threshold'       : pktThr}
```

Chapter 22

Egress Resource Manager

The core includes an Egress Resource Manager (ERM) unit for controlling the shared buffer memory occupancy of egress ports and queues. The primary objective of the egress resource manager is to avoid persistent buildup of queue length in the buffer memory and prevent the blockage of enqueuing at other ports and queues. Additionally, during buffer memory congestion, ERM facilitates prioritized enqueuing of egress queues with higher priorities.

The resource management granularity is cells and there are 4096 cells, each 160 byte wide, available in the buffer memory. A packet is written to the buffer memory with the original packet data plus a 16 byte ingress to egress header, thus a 1600 byte packet will have 1616 bytes and occupy ten cells. A packet plus the ingress to egress header longer than *n* cells but shorter than *(n+1)* cells will require *(n+1)* cells for storage. For example, a 145 byte packet will use two cells. ERM traces the buffer memory occupancy and decides if a cell is allowed to be written to the buffer memory.

The ERM determines the congestion of the buffer memory based on the amount of free space (number of free cells) available. The ERM classifies the congestion levels into Green (no congestion), Yellow (slightly congested) or Red (heavily congested). When the buffer memory is in the yellow or red zone, **Resource Limiter Set** gives four sets of limits to check the queue length for different egress ports and queues. An egress port chooses limit sets for each of its queues from the **Egress Resource Manager Pointer** lookup.

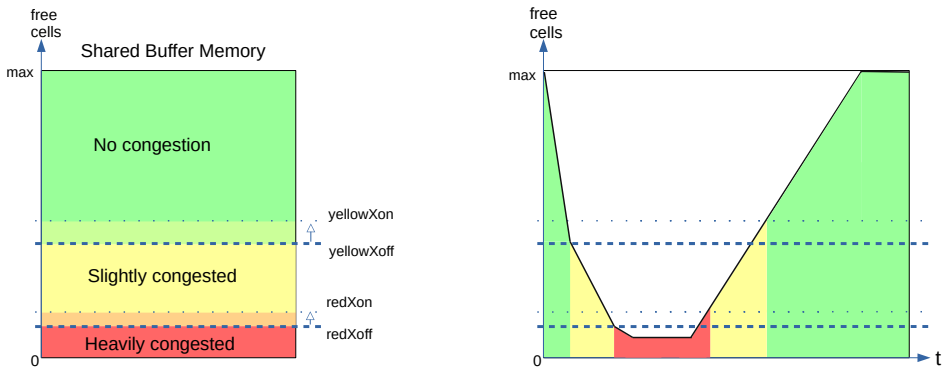


Figure 22.1: Buffer memory congestion zones

22.1 Yellow Zone

ERM Yellow Configuration defines how to enter and exit the yellow zone. The yellow zone is entered when the number of free cells goes below **yellowXoff**. To leave the yellow zone, the number of free cells need to go above **yellowXon**.

ERM checks

The buffer memory is considered partially congested when it is in the yellow zone. The ERM allows moderate buildups in all queues to a certain limit. An incoming cell of a packet is not allowed to be enqueued under two conditions:

1. The number of enqueued cells in the assigned egress queue is more than **yellowLimit**, while the total number of enqueued cells in the same queue and higher priority queues is more than **yellowAccumulated**.
2. **ERM Yellow Configuration** offers an optional check on a per egress port basis. A port can be considered as a red port in the yellow zone if the enqueued cells on that port are above **redPortXoff**. An incoming cell to a red port is not allowed if the length of the assigned queue is larger than **redLimit**.

22.2 Red Zone

ERM Red Configuration defines how to enter and exit the red zone. The red zone is entered when the number of free cells goes below **redXoff**. To leave the red zone, the number of free cells need to go above **redXon**.

ERM checks

The buffer memory is considered severely congested when it is in the red zone and the ERM shall only accept enqueuing to nearly empty queues. An incoming cell of a packet is not allowed to be enqueued in two cases:

1. The number of enqueued cells in the assigned egress queue is more than **redLimit**.
2. The ongoing packet length in cells has exceeded **redMaxCells**.

22.3 Green Zone

When the buffer memory is neither in the yellow zone nor in the red zone, the ERM considers the buffer memory to be uncongested and all incoming cells are accepted and stored in their assigned queues.

22.4 Configuration Example

A commonly used non-default ERM configuration involves allowing a queue to grow up to length **G** without packet drops (guarantees), and preventing new packets from being enqueued when the queue length is beyond **L** (limits). Between queue length **G** and **L** the enqueuing decision is made based on the overall free space in the buffer memory. This configuration imposes the following requirements:

1. $\texttt{redXon} \geq \texttt{redXoff} \geq sum(\texttt{redLimit})$
   The red zone is used as guarantees, its configuration needs to ensure that **redXon** is large enough so that the buffer memory does not get full before all queues reach their **redLimit**. Set **redLimit** a few cells more than the desired guarantee size to have a margin for the latency.
2. Set **yellowAccumulated** to 0, ensuring that **yellowLimit** is always checked in the yellow zone.

3. **yellowXon** $\geq$ **yellowXoff** $\geq$ **maxBufferFree**
   Put the ERM in the yellow zone even when the buffer memory is empty hence keep **yellowLimit** check under an always on state.

Chapter 23

Statistics

Short Name	Register Name
3. macBrokenPkt	MAC RX Broken Packets
4. macRxMin	MAC RX Short Packet Drop
5. spOverflow	SP Overflow Drop
11. ipppDrop	Unknown Ingress Drop Empty Mask Drop Ingress Spanning Tree Drop: Listen Ingress Spanning Tree Drop: Learning Ingress Spanning Tree Drop: Blocking L2 Lookup Drop Ingress Packet Filtering Drop Ingress L2 ACL Drop Reserved MAC DA Drop Reserved MAC SA Drop VLAN Member Drop Minimum Allowed VLAN Drop Maximum Allowed VLAN Drop Invalid Routing Protocol Drop Expired TTL Drop L3 Lookup Drop IP Checksum Drop L3 ACL Drop L2 Reserved Multicast Address Drop
11. smon	SMON Set 0 Packet Counter SMON Set 1 Packet Counter SMON Set 0 Byte Counter SMON Set 1 Byte Counter
11. ippAcl	Ingress L2 ACL Match Counter Ingress L3 ACL Match Counter
11. vrfIn	Received Packets on Ingress VRF
11. nextHop	Next Hop Hit Status
11. preEppDrop	Queue Off Drop Egress Spanning Tree Drop MBSC Drop Ingress-Egress Packet Filtering Drop
12. ipmOverflow	IPP PM Drop
13. ippTxPkt	IPP Packet Head Counter IPP Packet Tail Counter
14. eopDrop	IPP Empty Destination Drop
14. mmp	Flow Classification And Metering Drop

Short Name	Register Name
15. erm	**Egress Resource Manager Drop**
16. bmOverflow	**Buffer Overflow Drop**
18. pbTxPkt	**PB Packet Head Counter** **PB Packet Tail Counter**
19. epppDrop	**Unknown Egress Drop** **Egress Port Disabled Drop** **Egress Port Filtering Drop**
19. vrfOut	**Transmitted Packets on Egress VRF**
21. drain	**Drain Port Drop**
22. epmOverflow	**EPP PM Drop**
24. rqOverflow	**Re-queue Overflow Drop**
24. eppTxPkt	**EPP Packet Head Counter** **EPP Packet Tail Counter**
25. psTxPkt	**PS Packet Head Counter** **PS Packet Tail Counter**
25. psError	**PS Error Counter**

Table 23.1: Sequence of Statistics Counters

This core supports full statistics with 32-bit wrap around counters. The statistics is divided into groups depending on the type of statistics and location in the switch. Figure 23.1 gives the location of the counters from ingress to egress, with a sequence number to show their process orders. The counters which are green are for packet drops based on forwarding decisions while the red counters are related to system errors. The details of the counters in Figure 23.1 can be found through Table 23.1.

23.1 Packet Processing Pipeline Drops

During the ingress/egress packet processing, the forwarding algorithm can drop a packet for various reasons. For each type of drop reason at least one drop counter is attached. The counter update is either based on received packets or to-be-transmitted packets.

- Statistics: IPP Ingress Port Drop.

  Each drop reason has a unique drop identifier (drop ID). The IPP ingress port drop statistics has a counter for each drop ID. In two cases a corresponding drop ID counter can be updated:

  1. When a received packet is dropped before any destination port is assigned.
  2. When all targeting destination ports are filtered out the **Empty Mask Drop** counter is updated.

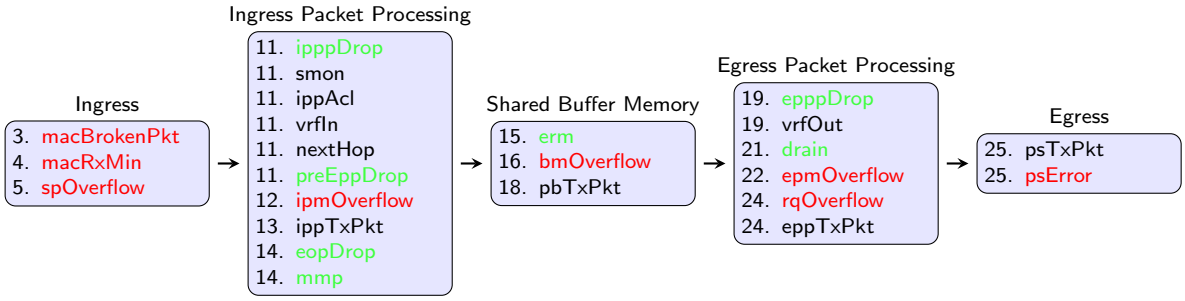


Figure 23.1: Location of Statistics Counters

- Statistics: IPP Egress Port Drop.

  This is a per drop ID and per egress port counter located in the ingress processing pipeline. When a packet has obtained one or more destination ports but the following ingress packet process filters out one of the obtained destination ports, a counter is updated for the corresponding egress port with the related drop ID. The **Empty Mask Drop** counter might be updated at the same time if no more destination port is set after the filtering.

- Statistics: EPP Egress Port Drop.

  This is similar to IPP egress port drop statistics but located in the egress packet processing pipeline. Drops that occur in EPP will cause bubbles on the transmit interface.

23.2 ACL Statistics

When a packet matches an ACL rule as described in Chapter Classification, the result operation can be configured to update a counter. In this case the result operation has a pointer to which counter to update. All the related counters are in Section Statistics: ACL.

23.3 SMON Statistics

There are 2 sets of SMON counters located in the ingress packet processing pipeline, each equipped with one counter per PCP value. The combination of the ingress port number and packet VLAN ID will provide the target SMON set to update through the **SMON Set Search** register. Each SMON set counts both the number of packets and number of bytes as shown in Section Statistics: SMON.

23.4 Routing Statistics

Section Statistics: Routing has three routing related statistics:

- **Received Packets on Ingress VRF**. Update when a packet enters a VRF in the ingress processing pipeline.
- **Transmitted Packets on Egress VRF**. Update when a packet leaves a VRF in the egress processing pipeline.
- **Next Hop Hit Status**. Update when IPv4/IPv6/MPLS packets hit a next hop entry.

23.5 Packet Datapath Statistics

Section Statistics: Packet Datapath gives a list of start of packet and end of packet counters in the main blocks of the core. They act as datapath checkpoints and can be helpful in tracing unexpected packet drops or corruptions.

A packet will cross three clock domains on its way through the core:

- RX MAC clock domain.

  There are no packet statistics in the RX MAC clock domain.

- TX MAC clock domain.

  Packet datapath statistics in the TX MAC clock domain are on the transmit edge of the switch, counting transmitted packets as well as protocol errors on the TX interface of the switch. Clock crossing synchronizations are applied to these counters in order to share the same configuration bus in the core clock domain.

- Core clock domain.

Packet datapath statistics in the core clock domain are counting in different internal blocks. Each block has a pair of counters for packet heads and tails to identify the pass through of a complete packet. The datapath counting follows the order in Figure 1.1:

1. **IPP Packet Head Counter** and **IPP Packet Tail Counter**.
2. **PB Packet Head Counter** and **PB Packet Tail Counter**.
3. **EPP Packet Head Counter** and **EPP Packet Tail Counter**.
4. **PS Packet Head Counter** and **PS Packet Tail Counter**.

If a stage has unequal packet head and tail counters while the counters in the previous stages are identical, packets are corrupted in this stage.

23.6 Miscellaneous Statistics

The core is designed to have no silent packet drops and all missing packets on the transmit interface can be found in a dedicated drop counter. Besides the drop counters mentioned above, there are more counters located in all other places where a packet drop might occur. Detailed drop counter list is in Section Statistics: Misc.

23.7 Debug Statistics

Section Statistics: Debug lists a group of statistics prepared for debug purposes. These counters indicate possible locations when fatal errors occurred inside the core. Typical error events include inaccurate clock frequencies, unacceptable configurations, etc. The switch will try to remain functional after an error state, but a correct behaviour cannot be guaranteed.

23.7.1 Debug Statistics Accuracy

Some of the statistics counters are located in a different clock domain than the configuration bus. The values are therefore transferred through synchronization registers. In order to reduce the hardware cost of these debug counters the synchronization can result in reading incorrect values if readout is done while the counters are incrementing. The counter itself will always have the correct value. It's only the readout that, with a very low probability, can have incorrect value on bits that are toggling.

Chapter 24

Packets To And From The CPU

The CPU port (number 11) has support for two special CPU tags in the packet header. In packets received by the switch on the CPU port, the tag can determine which port the packet shall be sent to. A tag can also be added to packets transmitted by the switch on the CPU port. This allows the software stack to determine where the packet came from and the reason why it was sent to the CPU port.

24.1 Packets From the CPU

Packets sent from the CPU are normally processed as any other packet that enters the switch, so the destination port is determined by the L2 lookup. When the CPU needs to direct a packet to a specific port, bypassing the normal L2 lookup, it is accomplished by adding a protocol header.

Byte Number	Contents of Byte
0-1	[11:0] port bit mask. Bit 0 is port number 0, bit 1 is port number 1 etc. Port 0 is located in bit 0 of byte number 1. The port numbers are physical ports, not link aggregation port numbers. The link aggregation will always be bypassed when sending packets with a From CPU Tag.
2	Bits [2:0] specifies which egress queue the packet shall use.

Table 24.1: From CPU tag format

The header consists of a specific Ethernet Type (39065) followed by a CPU Tag. The CPU tag has a 2 byte(s) destination port mask field[1] and 1 byte egress queue field (encoded as specified in table 24.1). The switch core will remove the extra protocol header and send out the packet on the ports requested by the destination port mask in the protocol header. This is shown in the figure 24.1.

The port mask in the CPU Tag field determines which ports the packet shall be sent to. If multiple bits are set in the port mask, the packet is treated as a multicast packet in the resource limiters. The packet will be sent out on all ports with the corresponding bit set.

24.1.1 From CPU Header and Packet Modification and Operations

There are a number of operations which are not carried out when a packet is sent in with the From CPU header. The following lists details this in greater detail what is done and what is not done.

- Link Aggregation is done.
- None of the VLAN operations are carried out.
- Mirroring is done. However with regards to ACL mirroring see below.

[1] The ordering described in 24.1 is the receive/transmit order.

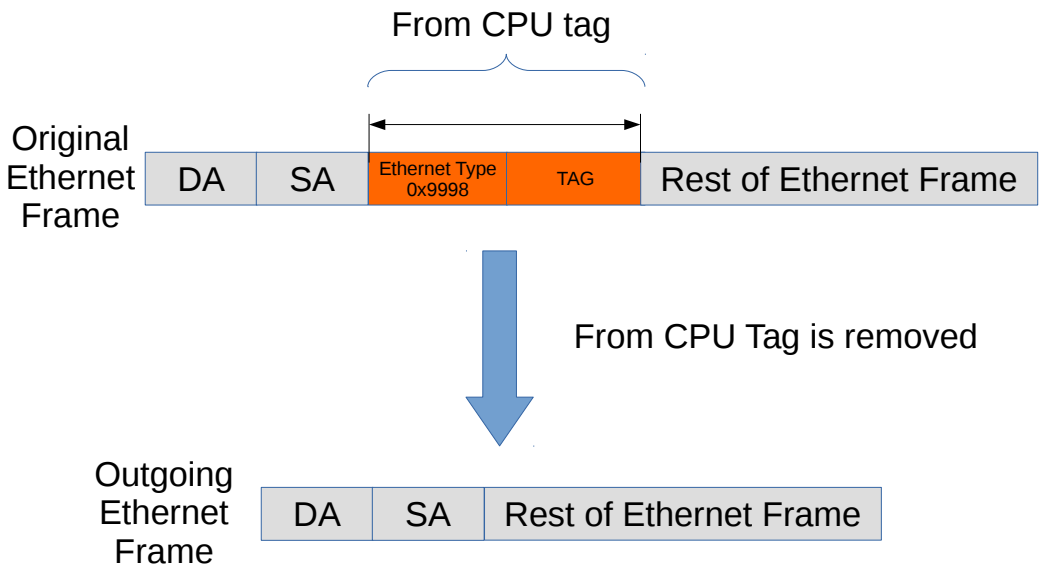


Figure 24.1: Packet from CPU with CPU tag

- Drops are ignored, example VLAN table , spanning tree / multiple spanning tree drops.
- L2 Lookup result is ignored.
- If the packet hits decoding rules for BPDU, Rapid Stanning Tree, Multiple Spanning tree, or other protocols such as then the packet will still send a extra copy to the CPU port. This can be disabled by setting the cpu port to zero in the send-to-cpu bitmask in each function.
- Routing is not carried out.
- SMON statistics is performed.
- Basic assignment of MMP is done.
- Meter-Marker-Policer check is done.
- MBSC is bypassed.
- All spanning tree and multiple spanning treeperations are bypassed.
- No learning operation.
- Check Reserved DMAC is done.
- Check Reserved SMAC is done.
- ACL operations are done.
- ACL statistics are done.
- SMON statistics is done.

24.2 Packets To the CPU

Packets can also be sent to the CPU port bypassing the normal L2 lookup. By default all packets to the CPU port have an extra protocol header (as shown in Figure 24.2). The header indicates the reason that the packet was sent to the CPU, and the port on which it was received. Packets which arrives on the CPU Port are modified according to what actions the packet was subjected to one example is VLAN header modifications.

When packets are sent to the CPU port (number 11 in this core), the packets are tagged with a specific Ethernet Type (type 39321). Figure 24.2 shows the Ethernet type field followed by a tag, and together

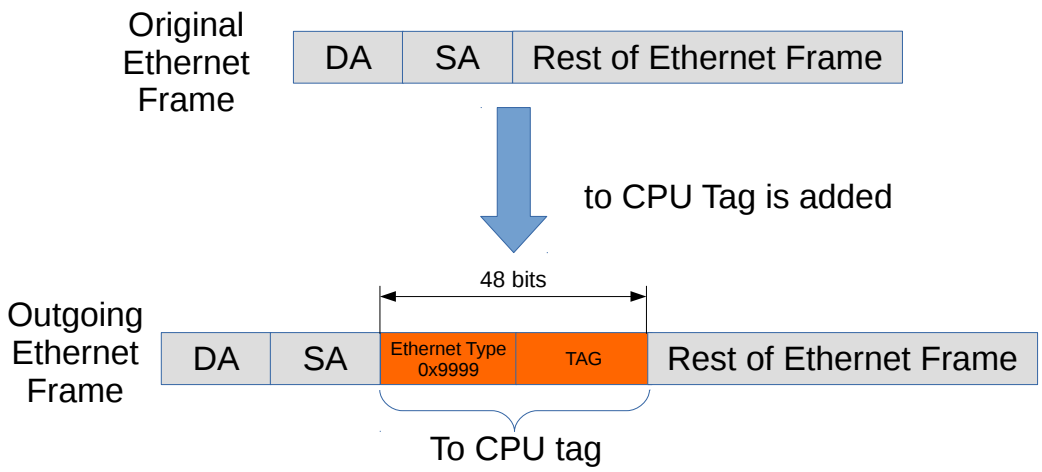


Figure 24.2: Packet to CPU with CPU tag

these constitute the extra protocol header mentioned above. The unmodified incoming packet follows just after this header.

The insertion of the extra protocol header can be disabled by setting the register **Disable CPU tag on CPU Port** to 1.

Byte Number	Contents of Byte
0	Bits [3:0] contains the source port where the packet entered the switch.
1	Reason for packet sent to CPU. See table 24.3.
2	Reserved
3	Reserved

Table 24.2: To CPU tag format

24.2.1 Reason Table

The reason codes why a packet was sent to the CPU. Reason code 0 means that the packet was switches or routed and the CPU port was part of the normal forwardings destination ports.If a packet can be directed to the CPU port with multiple reasons, the first hit in the check list below will give the reason code to the egress packet header.

Reason	Description
0	The MAC table, L2 MC table, ACL send to port action, MPLS table, the from-CPU-TAG contained the CPU port or routing tables sent the packet to the CPU port.
1	The packet decoder requires more than one cell.
2	This is a BPDU / RSTP frame.
3	The Unique MAC address to the CPU was hit.
4 + HitIndex	The Source MAC range sent the packet to the CPU..Index to rule.
8 + HitIndex	The Destination MAC range sent the packet to the CPU..Index to rule.
12 + HitIndex	The first L2 classification sent the packet to the CPU..Index to rule.
44 + HitIndex	The L3 / L4 classification sent the packet to the CPU..Index to rule.
76	This is an LLDP frame.
77	The IP TTL field was expired in the packet.

Reason	Description
78	The router ports check about which IPv4/IPv6/MPLS packets was allowed in the router failed.
79	The default routes send2cpu bit was set.
80	The IP length exceeded the MTU setup.
81	The entry in the **Next Hop Table** is invalid.
82	The entry in **Next Hop Packet Modifications** pointed to from the **Next Hop Table** is invalid.
83	The next hop entry had a send2cpu bit set.
84	The IPv4 header size field was not equal to five.
85	IPv4/IPv6 multicast was detected and redirected to CPU.
86	The maxium number of MPLS tags was detected in a packet.
87	Packet matched an L2 Multicast Reserved Address

Table 24.3: Reason for packet sent to CPU

The possible reasons are listed in Table 24.3.

1. Hit in the **Reserved Source MAC Address Range** with a **sendToCpu** action.
2. Hit in the **Reserved Destination MAC Address Range** with a **sendToCpu** action.
3. Hit in the **L2 Reserved Multicast Address Base** with **sendToCpuMask** enabled for the corresponding source port.
4. Hit in the **LLDP Configuration**.
5. Hit in the **Send to CPU** register.
   - Notice that when **uniqueCpuMac** is enabled then unicast packet will not be switched to the CPU port. Instead packets from any source port with MAC DA equal to **cpuMacAddr** will be sent to the CPU. Other mechanism for sending to the CPU port are not affected (e.g. ACL's).
6. Hit in **Ingress L2 ACL Match Data Entries** with a **sendToCpu** action.

Chapter 25

Core Interface Description

This chapter describes the interfaces to the core. An *input* is an input to the core, and an *output* is a signal driven by the core. In analogy *reception* refers to packets to the core and *transmission* means packets from the core.

25.1 Clock, Reset and Initialization interface

There is a core clock, mac clock signals for the packet interfaces, a global reset signal, mac reset signals for the packet interfaces, and a *doing_init* output (indicating when the core is in initialization and thus not ready to receive packets).

When the global reset, *rstn*, is asserted all packets buffered in the switch will be dropped, the learning and aging engines and all statistics counters will be reset to the initial status. Reset can be pulled at any time, but any ongoing transmit packets will be immidiately interrupted and no end of packet signal will be given.

The packet interface resets cannot be used independently. If one reset is asserted, all resets (including the core reset) have to be asserted before any reset can be released.[1]

[1] Thus the packet interface resets cannot be used to empty a specific packet interface. To do that, follow the procedure in Section 17.5, while making sure that the packet interface halt is kept low.

Signal Name	Size	In Out	Description
clk	1	In	Core clock. For 120 Gbit/s wire-speed throughput use a core clock frequency of 180 MHz
rstn	1	In	Global asynchronous reset (active low)
clk_mac_rx_N	1	In	Clock for the RX packet interface for port **N**.
rstn_mac_rx_N	1	In	Asynchronous reset (active low) for the RX packet interface for port **N**
clk_mac_tx_N	1	In	Clock for the TX packet interface for port **N**.
rstn_mac_tx_N	1	In	Asynchronous reset (active low) for the TX packet interface for port **N**
assert_reset	1	Out	Signal indicating that the core has experienced an unrecoverable error, and should be reset.
consistency_check	1	In	When pulled high internal checks will be made. This is a simulation-only port, it shall be tied low in hardware.
idle	1	Out	Indicates when the packet processing pipelines are empty.
doing_init	1	Out	Indicates that the core is in initialization. The operation of the core is undefined if packets are injected on the rx-interfaces when the core is in initialization

Table 25.1: Clock and Reset interfaces

Core Initialization

Before packets are sent to the core it needs to be initialized. The initialization is initiated when reset is released. Reset activation is asynchronous to any clock. The reset should be kept low at least one cycle of the slowest clock. Releasing reset must be done synchronously with respect to all clocks. During initialization *doing_init* is kept high. See Figure 25.1. The length of the initialization is dependent on the depth of the deepest initialized memory.

During initialization no activity is expected on the configuration interface or on the packet RX interfaces, and the operation of the core is undefined if any such activity occurs.

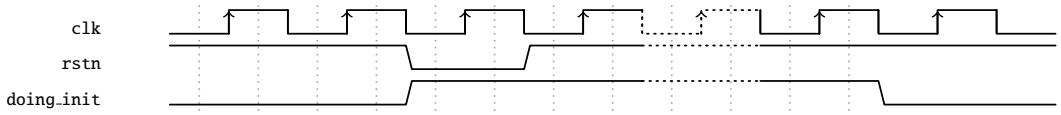


Figure 25.1: Core Initialization

25.1.1 Assert Reset

The *assert_reset* signal will go high, and stay high, if the core experiences an unrecoverable error. The behaviour of the core when *assert_reset* is high is undefined, and the only way to get back to normal operation is to reset the core.

The configuration bus will most likely still work when *assert_reset* is high.

25.2 Packet Interface

There are 12 packet interfaces, or ports for short, each divided into a reception part and a transmission part. The ports are numbered from 0 to 11.

Pin	Size	Direction	Description
rx_axis_tvalid_**N**	1	In	Set high to indicate that the current bus cycle is valid. The core must accept the data, there is no backpressure mechanism.
rx_axis_tlast_**N**	1	In	End-of-packet flag. Indicates that the current bus cycle contains the last data transfer for the packet. This is the only time a partially-filled data word is permitted.
rx_axis_tdata_**N**	64	In	Packet data.
rx_axis_tkeep_**N**	8	In	A per-byte data valid indication for the last word. Only valid when tlast is high. If tkeep[0] is high, tdata[7:0] is valid; if tkeep[1] is high, tdata[15:8] is also valid; and so on and so forth. The axis_tkeep port shall be connected to the LSBś of axis_tkeep_user.
rx_axis_tuser_**N**	1	In	Error indication for the packet. Valid only when tlast is high. The axis_tuser port shall be connected to the MSB of axis_tkeep_user.

Table 25.2: Packet RX interface for ports 0-11. **N** is the ingress interface number.

Each direction of a packet interface consists of *tvalid*, *tlast*, *tkeep*, *tdata* and *tuser* fields. The transmit direction has an additional *tready* signal to allow the receiving end to moderate the data rate transmitted from the core.

Packet data is presented in order, i.e. the most recent byte is the, so far, highest numbered byte in the packet. The first valid byte on the bus is byte 0, and all bytes are valid up to the last byte indicated by *tkeep*. Unless the *tlast* flag is set all bytes or no bytes must be valid.

Sending and Receiving packets

Data transmission, either to or from the core, begins with a transaction where the *tvalid* field is high and the *valid_bytes* field is non-zero, and ends with a data transmission where the *tlast* field is high. Idle transactions—where tkeep, *tvalid* and *tlast* are all zero—are allowed at any time, but unless halted there will be no idle transactions on the transmission interfaces other than between packets.

By default, the core has a short packet size limit of 60 bytes. All shorter packets will be dropped. This assumes that the receiving MAC removes the FCS before sending the packet to the core.

Jumbo packets

The maximum packet length that this core can cope with is 16367 bytes. If this length is exceeded either on the ingress or the egress it may corrupt the internal counters.

It should be noted that it is not guaranteed that a packet of that length will always be able to pass through the switch, even if the destination queue is not congested. Depending on the Egress Resource Management settings, and/or the congestion status of other ports, there may not be enough free cells in the packet buffer to store such a large packet. But the switch core will, when properly configured and reasonably uncongested, be able to switch 16367-byte packets.

Longest Packet for No-Overlap Mesh

The longest packet that can pass a no-overlap mesh test is highly dependent on the ERM settings. But with the default settings you can expect to pass a no-overlap mesh test with 1601-byte packets.

Pin	Size	Direction	Description
tx_axis_tvalid_**N**	1	Out	Set high to indicate that the current bus cycle is valid.
tx_axis_tlast_**N**	1	Out	End-of-packet flag. Indicates that the current bus cycle contains the last data transfer for the packet. This is the only time a partially-filled data word is permitted.
tx_axis_tdata_**N**	64	Out	Packet data.
tx_axis_tkeep_**N**	8	Out	A per-byte data valid indication for the last word. Only valid when tlast is high. If tkeep[0] is high, tdata[7:0] is valid; if tkeep[1] is high, tdata[15:8] is also valid; and so on and so forth. The axis_tkeep_user signal is created by concatenating {axis_tuser,axis_tkeep}.
tx_axis_tuser_**N**	1	Out	Error indication for the packet. Valid only when tlast is high.
tx_axis_tready_**N**	1	In	Driven by the MAC to indicate that the interface is able to accept the data currently present on the bus. If the tready signal deasserts during a transfer, the current data on the bus must be held until tready is asserted again.

Table 25.3: Packet TX interface for ports 0-11. **N** is the egress interface number.

Inter-frame gap

For small packets it is possible to feed the switch with more packets than it can handle. This will cause the SP to overflow, and packets to be dropped. To avoid packet drops an inter-frame gap (IFG) of at least 192 bits is needed between each packet. There is a small fifo in the SP, so a single smaller IFG is fine, but it needs to average at or above the minimum IFG over a window of a few packets.

On the output from the switch packets will be sent back to back, without IFG, and it is up to the receiver to halt the transmission using the *tready* interface to prevent overflows.

Broken packets

A packet ending with *tuser* set high is considered a broken packet. Broken packets received by the core will never be output on the egress ports, but will be dropped at the earliest convenience. So any broken packets output from the switch are packet that were somehow corrupted in the core. There are no benign cases where this happens. Depending on the packet length a broken packet input to the core will be dropped either before or after ingress packet processing. Broken packets larger than a cell will pass through the packet processing pipeline and then been dropped, while packets shorter than a cell will be filtered out before the packet processing pipeline.

All broken packets are counted in the **MAC RX Broken Packets**.

Byte Order

We define the packet byte order by the first transmitted/received byte on the wire labeled byte 0, as in IEEE 802.3. On a packet interface wider than 8 bits the packets byte 0 is placed on the bits data[7:0] followed by byte 1 on bits data[15:8] and so on.

The *tkeep* indicates how many of the bytes of the data field that holds valid packet data. From the start of a packet this must always be all bytes on the bus up till the last transfer. At the end of the packet on the last bus transfer the *tkeep* can indicate less than the full bus width. In this case the byte order is still

the same as previous transfers. For example when *tkeep* is 1 the last byte of the packet is placed on bits [7:0] and with *tkeep* of 3 the last byte of the packet is placed on bits [15:8] and the second to last is on bits [7:0].

25.3 Configuration Interface

The CPU-accessible registers and tables in the core are accessed using the configuration interface.

Each transaction on the configuration interface consists of a request to the core and a resulting reply from the core.

The pins for the configuration interface are listed in Table 25.4 below.

Pin	Size	Direction	Description
apb_paddr	20	In	Address. This is the APB address bus. The highest address bit (19) on the APB bus is not a normal address bit and is referred to as the Accumulator Bit. This is described further in section 26.
apb_psel	1	In	Select.
apb_penable	1	In	Enable.
apb_pwrite	1	In	Direction. This signal indicates an APB write access when HIGH and an APB read access when LOW.
apb_pwdata	32	In	Write data.
apb_pready	1	Out	Ready. The slave uses this signal to extend an APB transfer.
apb_prdata	32	Out	Read Data.
apb_pslverr	1	Out	Error. This signal indicates a transfer failure.

Table 25.4: The APB interface signals

The *paddr* is a byte address, however the core only supports accessing complete 32-bit words. The lowest address bits, which addresses the byte within a bus word, will always be discarded. The register addresses described in this document always refer to word addresses, not byte addresses.

The core has a varying access latency and therefore an APB master should use *pready*.

The *pslverr* signal is set when a transaction is aborted due to an internal timeout. This can occur if the core clock is lower than required and there is a high traffic rate. It will also occur if the address is outside of any defined register.

For a detailed description of the APB interface see the AMBA APB Protocol Specification Version 2.0, available at developer.arm.com

25.4 Debug Write Interface

The debug write interface is an input port to the Switch Core that can be used for debugging purposes. In normal operation the *debug_write_data* pins must be tied low. The function of the debug write interface is controlled by registers in the individual blocks. In this core only the tick dividers use the debug write interface. See **Core Tick Select**.

Pin	Direction	Size	Description
debug_write_data	In	1	The debug write input data. Must be tied low for normal switch operation.

Table 25.5: The Debug Write interface

Chapter 26

Configuration Interface

The configuration interface is an AMBA APB interface used for monitoring the core and for configuration of internal registers and tables. The pins are described in Table 25.4 on page 105, but for a detailed description of the APB interface see the AMBA APB Protocol Specification Version 2.0, available at developer.arm.com

26.1 Response time

The response time may vary between registers, and even vary for the same register depending on how busy the core is switching packets. The response time is in the order of tens of core clock cycles.

26.2 Out of range accesses

There is no range check on the configuration interface, so an access to an address that is not mapped to any register will result in a internal timeout and raise the *pslverr* on the bus.

26.3 Atomic Wide Access

There are a few recommendations how to access wide registers (registers that are wider than the APB data bus). The interface does allow a more flexible access pattern than what is described here. If that is needed then see the next section.

The highest address bit (19) on the APB bus is not a normal address bit. It is used to control wide register access. It will be referred to as the Accumulator Bit in the following description.

- Wide Reads
  - always read wide register starting with the lowest address and ending with the highest address.
  - when reading the lowest address of the register the Accumulator Bit should be 0.
  - when reading the other addresses of the register the Accumulator Bit should be 1.
- Wide Writes
  - always write wide register starting with the lowest address and ending with the highest address.
  - when writing the highest address of the register the Accumulator Bit should be 0.
  - when writing the other addresses of the register the Accumulator Bit should be 1.
- Narrow reads and writes
  If the register fits within the APB data bus width then the Accumulator Bit should be 0.

Note that if there are bridges between the CPU and the APB bus then they need to be set up to guarantee the order of accesses.

The software API implementation provided with the switch handles the Accumulator Bit thereby hiding it completely for the software that use the API.

26.4 Accumulator Accesses

Each table or register bank where the data is wider than the configuration data bus, will be equipped with a shadow-register called an accumulator. The accumulator allows the full data width to be updated atomically even though the bus width is narrower than the data. The accumulator is accessed by setting bit 19 of the address high during a normal register access. An access with bit 19 of the address low we call a `DEFAULT` access, while an access with bit 19 of the address high is called an `ACCUMULATOR` access. The register section of the datasheet will only list the addresses for `DEFAULT` access to the registers. Address bit 19 is considered an accumulator flag, and not a part of the address.

A `DEFAULT` read will return the requested data in the reply, and at the same time load the full data width into the accumulator. Thus following up the `DEFAULT` read with `ACCUMULATOR` reads will allow reading the state of the register at the time of the original `DEFAULT` read. If data consistency is not important, all the reads can be of the `DEFAULT` type, but there is no point because the read performance is the same. In fact reading a table will potentially be faster using the accumulator, because only the first access will have to wait for access to the physical memory.

Writes work similarly, but the other way around. The accumulator will first be loaded using `ACCUMULATOR` writes and then the contents of the accumulator is written to the register. The final `DEFAULT` write will use the data given as *wdata*, and fill it out with the data in the accumulator. Writing data wider than the bus cannot be done without taking the accumulator into account.

If only a part of a very wide register is to be written, the most efficient approach may be to do a `DEFAULT` read (loading the accumulator) followed by a `DEFAULT` write. But note that there is no way to do a truly atomic read-modify-write. Any write that the core slips in while the accumulator is loaded will be over-written by the following `DEFAULT` write.

When the data is wider than the bus the address is stepped by 2^n between table indexes or registers. For instance a 32-bit bus and a 65 bit wide table will result in index 1 starting at address 4, with address 3 unused and address 2 only containing a single valid bit.

Chapter 27

Implementation

27.1 Floorplanning

The top of the core is the *pa_top* level, it wraps the switch core, *pa_top_switch*, and may also contain interface bridges.

The switch hierarchy is divided into six major blocks that we call floorplan blocks. These are: SP, IPP, BM, PB, EPP, and PS. There is also two smaller blocks: ingress_common, interface_common. In some configurations these are very small, but in some the ingress_common can be quite substantial.

Besides the configuration bus, which spreads it's tentacles to every corner of the core, the dataflow through the floorplan blocks is basically that of the path of a packet. The flow from ingress to egress is SP, IPP, BM/PB, EPP, and PS. The PB/BM are lumped together in the list because the packet data goes through the BM, and the control data through the PB. The ingress_common contains auxillary functions for the ingress packet processing and thus mainly talks to the IPP. The other small block, interface_common, is mostly comprised of shim logic for the external interfaces.

27.1.1 Pipelining

The number of pipeline stages in the data paths between the floorplan blocks can be set freely when the RTL is generated. The same goes for the number of input flops and output flops on each floorplan block. If you need to change the number of pipeline stages it is a trivial task, but the RTL has to be re-generated. It cannot be adjusted in the existing verilog files.

Connection	Pipeline stages
SP $\leftrightarrow$ IPP	0
IPP $\leftrightarrow$ PB/BM	0
PB $\leftrightarrow$ BM	0
BM $\leftrightarrow$ EPP	0
EPP $\leftrightarrow$ PS	0

Table 27.1: The settings for pipeline flops between floorplan blocks

Floorplan block	Input flops	Output flops
SP	0	0
IPP	0	0
PB	0	1
BM	0	0
EPP	0	0
PS	1	1

Table 27.2: The settings for input and output flops for the floorplan blocks

The pipeline settings used when generating this core are shown in Table 27.1, and the input/output flops are listed in Table 27.2[1].

27.1.2 Configuration and debug

The configuration and debug busses are in principle extremely flexible in how they can be pipelined. Flops can be added and removed anywhere so long as each bus is still in sync. This, as the other changes in pipelining, can only be done by generating new RTL.

27.2 Clock crossings

The bulk of the core is in a single clock domain, the core domain, driven by the *clk* clock. Each packet interface has separate clock domains for TX and RX. All paths between these domains are synchronized by either two synchronization flops, or by an asynchronous memory. The synchronization flops are always instantiations of the *verilog_sync_flops* verilog module, and the asynchronous memories are always instantiations of *verilog_memory_2c*.

27.2.1 IPP and EPP Structure

The IPP and EPP modules are both pipelines with a main dataflow from input to output. The floorplan is recommended to follow the pipeline dataflow. The logic input to a memory comes from the preceding pipeline stage and the output goes to the following pipeline stage. Which pipeline stage a specific memory belongs to is documented in the delivered files eppp0_raw_opt.ramstat and ippp0_raw_opt.ramstat.

In addition to the memory instances, the pipeline flipflops belonging to each pipeline stage is documented in ippp0_raw_opt.fflist and eppp0_raw_opt.fflist.

The exact Verilog instance names are not listed in these files but the names in the lists are part of the instance names and uniquely identify them.

In addition to the main dataflow there is also a configuration bus that has access to all memory instances and to the configuration registers. These paths are normally not in the critical path.

The configuration registers as opposed to the configuration memories can be accessed in multiple pipeline stages and therefore does not have a simple placement strategy.

27.3 Memory timing

All memories in the design can be selected to have either:

- One cycle latency
- Two cycles latency, with the flop added on the input to the memory
- Two cycles latency, with the flop added on the output from the memory
- Three cycles latency, with flops added on both the input and the output

27.4 Lint set up

For spyglass linting the following settings are assumed:

- set_parameter ignore_local_variables yes
- set_parameter handle_zero_padding "W362"

[1]It should be noted that the input/output flops for the PS is not as clear cut as for the other blocks, due to the slightly more complex interface to the MAC.

27.4.1 Waivers

Besides the inline waivers in the code these blanket waivers shall be applied:

- waive -rule STARC05-2.11.3.1 -comment "Case statements are used in the sequential blocks of state-machines. This is not an issue"
- waive -rule STARC05-2.2.3.3 -comment "Flip-flops may be written several times in the same sequential block. This is not an issue"
- waive -regexp -du "consistency_check.*" -rule "W240" -comment "consistency_check is guarded by SYNTHESIS, and is not used in hardware."
- waive -rule W415a -comment "Assigning multiple times in the same always block is a code style we use. This is not an issue"
- waive -rule W528 -comment "The way we pipeline will leave a lot of unread signals. This is not an issue"

Chapter 28

Registers and Tables

Contents

All registers and tables that are accessible from a configuration interface are listed in this chapter. A user guide for the configuration interface is found in Chapter 26, and the pins for the configuration interfaces are described in Section 25.3.

28.1 Address Space For Tables and Registers

All tables in the address space are linear. The size of a table entry is always rounded up to nearest power of two of the bus width. For example if the bus is 32 bits and a entry in a table is 33 bits wide, it will then use two addresses per entry. Second example, the bus is still 32 bits, but the entry is 181 bits wide, the entry will then use a address space of 8 addresses per table entry (181 bits fits within 6 bus words but is rounded up to nearest power of two). This is shown in figure 28.1. The total address space used by this core is 95263 addresses.

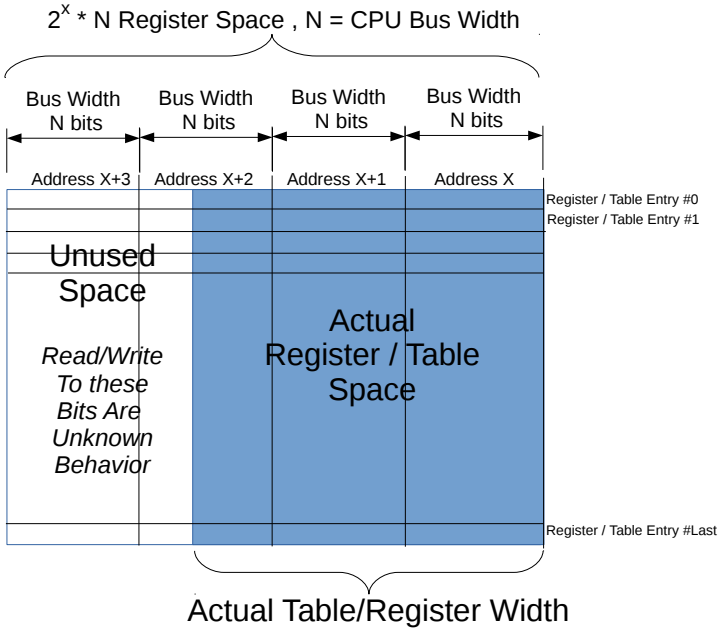


Figure 28.1: Address space usage by tables

28.2 Byte Order

When a register field is wider than a byte and the field represents an integer value or the field is related to a packet header field, the order of the bytes needs to be defined.

Integer fields in the registers have a little endian byte order so that the lowest bits in a field will be at lowest bits on the configuration bus. When a field spans multiple configuration bus addresses the lowest address will hold the lowest bits of the field. If this is memory mapped and accessed by a host CPU it will be in the correct byte order for a little endian CPU.

In network byte order the first transmitted or received byte has byte number 0. One example is the Ethernet MAC address with the printed representation *a1-b2-c3-d4-e5-f6* where *a1* would be sent first and would be byte 0). When used in a register field the highest bits in the register field corresponds to the lowest network byte. Therefore the MAC address above would be the value *0xa1b2c3d4e5f6* and as seen by a little endian host CPU the byte *0xf6* would be at the lowest address.

A special case are IPv6 addresses. In the standard printed representation *0102:0304:0506:...* the leftmost byte *01* is byte 0 in the network order followed by byte *02* as network byte 1. When configuring this in a register field the lowest bytes are from the lowest network byte numbers. However each pair of bytes are also swapped. The address above would therefore be the value *0x....050603040102*.

28.3 Register Banks

A bank is a hardware unit which holds a number of registers or a single table. In a bank containing data wider than 32 bits, registers (or table entries) must be accessed one at a time, or the accesses will interfere with each other.

Bank Name	Connected Registers or Tables
switch_info_regbank	Core Version
top_regs	Buffer Free Core Tick Configuration Core Tick Select Scratch
rx_length_drop	MAC RX Broken Packets[0..11] MAC RX Short Packet Drop[0..11]
l2_broadcast_storm_control_rate_settings	L2 Broadcast Storm Control Rate Configuration
l2_broadcast_storm_control_bucket_settings	L2 Broadcast Storm Control Bucket Capacity Configuration L2 Broadcast Storm Control Bucket Threshold Configuration
l2_broadcast_storm_control_misc	L2 Broadcast Storm Control Enable
l2_multicast_storm_control_rate_settings	L2 Multicast Storm Control Rate Configuration
l2_multicast_storm_control_bucket_settings	L2 Multicast Storm Control Bucket Capacity Configuration L2 Multicast Storm Control Bucket Threshold Configuration
l2_multicast_storm_control_misc	L2 Multicast Storm Control Enable
l2_flooding_storm_control_rate_settings	L2 Flooding Storm Control Rate Configuration
l2_flooding_storm_control_bucket_settings	L2 Flooding Storm Control Bucket Capacity Configuration L2 Flooding Storm Control Bucket Threshold Configuration
l2_flooding_storm_control_misc	L2 Flooding Storm Control Enable
le_ae_status	Learning Conflict Learning Overflow
le_ae_control	Learning And Aging Enable Hardware Learning Configuration[0..11] Time to Age
age_cam_register_bank	L2 Aging Collision Table[0..15]
mac_cnt_register_bank	Hardware Learning Counter[0..11]
L2 Aging Table	L2 Aging Table
count_sp_ss0	SP Overflow Drop
count_broken_pkt_ss0	IPP PM Drop IPP Empty Destination Drop
count_pa top switch ipp0 conf	Unknown Ingress Drop Empty Mask Drop Ingress Spanning Tree Drop: Listen Ingress Spanning Tree Drop: Learning

Bank Name	Connected Registers or Tables
	Ingress Spanning Tree Drop: Blocking L2 Lookup Drop Ingress Packet Filtering Drop Ingress L2 ACL Drop Reserved MAC DA Drop Reserved MAC SA Drop VLAN Member Drop Minimum Allowed VLAN Drop Maximum Allowed VLAN Drop Invalid Routing Protocol Drop Expired TTL Drop L3 Lookup Drop IP Checksum Drop L3 ACL Drop L2 Reserved Multicast Address Drop
count_opkt_pa top switch ipp0 conf	IPP Packet Head Counter IPP Packet Tail Counter
L2 Reserved Multicast Address Action	L2 Reserved Multicast Address Action
Ingress Admission Control Initial Pointer	Ingress Admission Control Initial Pointer
Ingress VID MAC Range Assignment Answer	Ingress VID MAC Range Assignment Answer
Ingress VID Outer VID Range Assignment Answer	Ingress VID Outer VID Range Assignment Answer
Ingress VID Inner VID Range Assignment Answer	Ingress VID Inner VID Range Assignment Answer
VLAN Table	VLAN Table
Ingress Multiple Spanning Tree State	Ingress Multiple Spanning Tree State
Ingress Router Table	Ingress Router Table
L3 LPM Result	L3 LPM Result
Hash Based L3 Routing Table	Hash Based L3 Routing Table
Next Hop Table	Next Hop Table
Next Hop Packet Modifications	Next Hop Packet Modifications
IPv4 TOS Field To Egress Queue Mapping Table	IPv4 TOS Field To Egress Queue Mapping Table
IPv6 Class of Service Field To Egress Queue Mapping Table	IPv6 Class of Service Field To Egress Queue Mapping Table
VLAN PCP And DEI To Color Mapping Table	VLAN PCP And DEI To Color Mapping Table
IPv4 TOS Field To Packet Color Mapping Table	IPv4 TOS Field To Packet Color Mapping Table
IPv6 Class of Service Field To Packet Color Mapping Table	IPv6 Class of Service Field To Packet Color Mapping Table
MPLS EXP Field To Packet Color Mapping Table	MPLS EXP Field To Packet Color Mapping Table
Router Egress Queue To VLAN Data	Router Egress Queue To VLAN Data
L2 Aging Status Shadow Table	L2 Aging Status Shadow Table
L2 DA Hash Lookup Table	L2 DA Hash Lookup Table
L2 Destination Table	L2 Destination Table
Egress Multiple Spanning Tree State	Egress Multiple Spanning Tree State
L2 SA Hash Lookup Table	L2 SA Hash Lookup Table
L2 Aging Status Shadow Table - Replica	L2 Aging Status Shadow Table - Replica
L2 Destination Table - Replica	L2 Destination Table - Replica
ipp_register_bank_ss0	Link Aggregation Ctrl Ingress Ethernet Type for VLAN tag Check IPv4 Header Checksum

Bank Name	Connected Registers or Tables
	Force Non VLAN Packet To Specific Queue Force Unknown L3 Packet To Specific Egress Queue Force Non VLAN Packet To Specific Color Force Unknown L3 Packet To Specific Color Forward From CPU Port Move Options L2 Multicast Handling Ingress MMP Drop Mask Debug srcPort Debug dstPortmask Enable Enqueue To Ports And Queues Link Aggregation To Physical Ports Members Link Aggregate Weight Ingress Egress Port Packet Type Filter Router MTU Table L2 Multicast Table L2 Aging Collision Shadow Table MPLS EXP Field To Egress Queue Mapping Table VLAN PCP To Queue Mapping Table Ingress L3/L4 ACL Result Operation Entries L3 Routing Default Ingress VID Ethernet Type Range Assignment Answer Ingress Port Packet Type Filter SMON Set Search Link Aggregation Membership Source Port Table Router Port MAC Address Ingress VID MAC Range Search Data Ingress L2 ACL Result Operation Entries Reserved Source MAC Address Range Reserved Destination MAC Address Range Send to CPU LLDP Configuration Ingress L2 ACL Match Data Entries L2 Reserved Multicast Address Base Egress Spanning Tree State L2 Lookup Collision Table Masks L2 Lookup Collision Table Ingress VID Ethernet Type Range Search Data Ingress VID Inner VID Range Search Data Ingress VID Outer VID Range Search Data Ingress L3/L4 ACL Match Data Entries
ipp_register_bank_misc_ss0	Ingress Drop Options
L3 Routing TCAM	L3 Routing TCAM
count_packets ipp0_smonStatisticsBlock	SMON Set 0 Packet Counter[0..7] SMON Set 1 Packet Counter[0..7]
count_bytes ipp0_smonStatisticsBlock	SMON Set 0 Byte Counter[0..7] SMON Set 1 Byte Counter[0..7]
count_ipp0_aclL2StatisticsBlock	Ingress L2 ACL Match Counter[0..31]
count_ipp0_vrfInStatisticsBlock	Received Packets on Ingress VRF[0..3]
Next Hop Hit Status	Next Hop Hit Status
count_ipp0_l3AclStatisticsBlock	Ingress L3 ACL Match Counter[0..31]
count_ipp0_egressDropStatisticsBlock	Queue Off Drop[0..11] Egress Spanning Tree Drop[0..11] MBSC Drop[0..11]

Bank Name	Connected Registers or Tables
	Ingress-Egress Packet Filtering Drop[0..11]
bk_mmp_stat_0	Flow Classification And Metering Drop
bk_ingress_admission_control_all_red_en_0	Ingress Admission Control Mark All Red Enable
bk_ingress_admission_control_all_red_0	Ingress Admission Control Mark All Red
Ingress Admission Control Token Bucket Configuration	Ingress Admission Control Token Bucket Configuration
Ingress Admission Control Reset	Ingress Admission Control Reset
Ingress Admission Control Current Status	Ingress Admission Control Current Status
bk_erm_ss0	ERM Yellow Configuration ERM Red Configuration Resource Limiter Set[0..3] Egress Resource Manager Pointer[0..11]
count_erm_ss0	Egress Resource Manager Drop[0..11]
pb_info_regbank_ss0	Packet Buffer Status
count_drop_pa top switch pb0	Buffer Overflow Drop
count_drop_pa top switch pb0 iRequeue	Re-queue Overflow Drop
qe_register_bank_ss0_sp0	Egress Port Depth[0..11] Egress Queue Depth[0..95]
pb_r_register_bank_ss0	Minimum Buffer Free
disable_queue_output_register_bank_ss0	Output Disable[0..11]
count_opkt_pa top switch pb0	PB Packet Head Counter PB Packet Tail Counter
drain_port_ss0	Drain Port
drain_drop_ss0	Drain Port Drop[0..11]
count_pa top switch epp0 conf	Unknown Egress Drop[0..11] Egress Port Disabled Drop[0..11] Egress Port Filtering Drop[0..11] EPP PM Drop
count_opkt_pa top switch epp0 conf	EPP Packet Head Counter EPP Packet Tail Counter
Egress Port Configuration	Egress Port Configuration
Color Remap From Egress Port	Color Remap From Egress Port
Color Remap From Ingress Admission Control	Color Remap From Ingress Admission Control
Egress Router Table	Egress Router Table
Next Hop DA MAC	Next Hop DA MAC
Next Hop MPLS Table	Next Hop MPLS Table
Egress MPLS TTL Table	Egress MPLS TTL Table
Next Hop Packet Insert MPLS Header	Next Hop Packet Insert MPLS Header
Egress Queue To PCP And CFI/DEI Mapping Table	Egress Queue To PCP And CFI/DEI Mapping Table
Select Which Egress QoS Mapping Table To Use	Select Which Egress QoS Mapping Table To Use
L2 QoS Mapping Table	L2 QoS Mapping Table
IP QoS Mapping Table	IP QoS Mapping Table
TOS QoS Mapping Table	TOS QoS Mapping Table
MPLS QoS Mapping Table	MPLS QoS Mapping Table
epp_register_bank_ss0	Output Mirroring Table Egress Queue To MPLS EXP Mapping Table Egress MPLS Decoding Options Egress Ethernet Type for VLAN tag Disable CPU tag on CPU Port
count_epp0_vrfOutStatisticsBlock	Transmitted Packets on Egress VRF[0..3]
count_opkt_pa top switch ps0 ps_wrap_bridge	PS Packet Head Counter

Bank Name	Connected Registers or Tables
	PS Packet Tail Counter
count_error_pa top switch ps0 ps_wrap_bridge	PS Error Counter

28.4 Registers and Tables in Alphabetical Order

Name	Address Range
Buffer Free	1
Buffer Overflow Drop	86101
Check IPv4 Header Checksum	82825
Color Remap From Egress Port	86334 - 86357
Color Remap From Ingress Admission Control	86358 - 86389
Core Tick Configuration	2
Core Tick Select	3
Core Version	0
Debug dstPortmask	82835
Debug srcPort	82834
Disable CPU tag on CPU Port	95196
Drain Port	86258
Drain Port Drop	86259 - 86270
EPP PM Drop	86307
EPP Packet Head Counter	86308
EPP Packet Tail Counter	86309
ERM Red Configuration	86066
ERM Yellow Configuration	86064
Egress Ethernet Type for VLAN tag	95195
Egress MPLS Decoding Options	95194
Egress MPLS TTL Table	89466 - 89469
Egress Multiple Spanning Tree State	66407 - 66422
Egress Port Configuration	86310 - 86333
Egress Port Depth	86103 - 86114
Egress Port Disabled Drop	86283 - 86294
Egress Port Filtering Drop	86295 - 86306
Egress Queue Depth	86115 - 86210
Egress Queue To MPLS EXP Mapping Table	95186 - 95193
Egress Queue To PCP And CFI/DEI Mapping Table	93566 - 93573
Egress Resource Manager Drop	86088 - 86099
Egress Resource Manager Pointer	86076 - 86087
Egress Router Table	86390 - 86393
Egress Spanning Tree Drop	85867 - 85878
Egress Spanning Tree State	83896
Empty Mask Drop	4467
Enable Enqueue To Ports And Queues	82836 - 82847
Expired TTL Drop	4480
Flow Classification And Metering Drop	85903
Force Non VLAN Packet To Specific Color	82828
Force Non VLAN Packet To Specific Queue	82826
Force Unknown L3 Packet To Specific Color	82829
Force Unknown L3 Packet To Specific Egress Queue	82827

Name	Address Range
Forward From CPU	82830
Hardware Learning Configuration	232 - 243
Hardware Learning Counter	262 - 273
Hash Based L3 Routing Table	13111 - 45878
IP Checksum Drop	4482
IP QoS Mapping Table	93894 - 94149
IPP Empty Destination Drop	4465
IPP PM Drop	4464
IPP Packet Head Counter	4485
IPP Packet Tail Counter	4486
IPv4 TOS Field To Egress Queue Mapping Table	48951 - 49206
IPv4 TOS Field To Packet Color Mapping Table	49479 - 49734
IPv6 Class of Service Field To Egress Queue Mapping Table	49207 - 49462
IPv6 Class of Service Field To Packet Color Mapping Table	49735 - 49990
Ingress Admission Control Current Status	86016 - 86031
Ingress Admission Control Initial Pointer	4743 - 4870
Ingress Admission Control Mark All Red	85920 - 85935
Ingress Admission Control Mark All Red Enable	85904 - 85919
Ingress Admission Control Reset	86000 - 86015
Ingress Admission Control Token Bucket Configuration	85936 - 85999
Ingress Drop Options	84474
Ingress Egress Port Packet Type Filter	83116 - 83127
Ingress Ethernet Type for VLAN tag	82824
Ingress L2 ACL Drop	4473
Ingress L2 ACL Match Counter	84763 - 84794
Ingress L2 ACL Match Data Entries	83638 - 83893
Ingress L2 ACL Result Operation Entries	83466 - 83593
Ingress L3 ACL Match Counter	85823 - 85854
Ingress L3/L4 ACL Match Data Entries	83962 - 84473
Ingress L3/L4 ACL Result Operation Entries	83272 - 83303
Ingress MMP Drop Mask	82833
Ingress Multiple Spanning Tree State	13075 - 13090
Ingress Packet Filtering Drop	4472
Ingress Port Packet Type Filter	83312 - 83323
Ingress Router Table	13091 - 13094
Ingress Spanning Tree Drop: Blocking	4470
Ingress Spanning Tree Drop: Learning	4469
Ingress Spanning Tree Drop: Listen	4468
Ingress VID Ethernet Type Range Assignment Answer	83308 - 83311
Ingress VID Ethernet Type Range Search Data	83938 - 83945
Ingress VID Inner VID Range Assignment Answer	4879 - 4882
Ingress VID Inner VID Range Search Data	83946 - 83953
Ingress VID MAC Range Assignment Answer	4871 - 4874
Ingress VID MAC Range Search Data	83450 - 83465
Ingress VID Outer VID Range Assignment Answer	4875 - 4878
Ingress VID Outer VID Range Search Data	83954 - 83961
Ingress-Egress Packet Filtering Drop	85891 - 85902
Invalid Routing Protocol Drop	4479
L2 Aging Collision Shadow Table	83240 - 83255
L2 Aging Collision Table	246 - 261
L2 Aging Status Shadow Table	50007 - 54102
L2 Aging Status Shadow Table - Replica	74615 - 78710
L2 Aging Table	274 - 4369

Name	Address Range
L2 Broadcast Storm Control Bucket Capacity Configuration	124 - 135
L2 Broadcast Storm Control Bucket Threshold Configuration	136 - 147
L2 Broadcast Storm Control Enable	148
L2 Broadcast Storm Control Rate Configuration	112 - 123
L2 DA Hash Lookup Table	54103 - 62294
L2 Destination Table	62295 - 66406
L2 Destination Table - Replica	78711 - 82822
L2 Flooding Storm Control Bucket Capacity Configuration	198 - 209
L2 Flooding Storm Control Bucket Threshold Configuration	210 - 221
L2 Flooding Storm Control Enable	222
L2 Flooding Storm Control Rate Configuration	186 - 197
L2 Lookup Collision Table	83906 - 83937
L2 Lookup Collision Table Masks	83898 - 83905
L2 Lookup Drop	4471
L2 Multicast Handling	82832
L2 Multicast Storm Control Bucket Capacity Configuration	161 - 172
L2 Multicast Storm Control Bucket Threshold Configuration	173 - 184
L2 Multicast Storm Control Enable	185
L2 Multicast Storm Control Rate Configuration	149 - 160
L2 Multicast Table	83176 - 83239
L2 QoS Mapping Table	93830 - 93893
L2 Reserved Multicast Address Action	4487 - 4742
L2 Reserved Multicast Address Base	83894
L2 Reserved Multicast Address Drop	4484
L2 SA Hash Lookup Table	66423 - 74614
L3 ACL Drop	4483
L3 LPM Result	13095 - 13110
L3 Lookup Drop	4481
L3 Routing Default	83304 - 83307
L3 Routing TCAM	84475 - 84730
LLDP Configuration	83630
Learning And Aging Enable	231
Learning Conflict	223
Learning Overflow	227
Link Aggregate Weight	82860 - 83115
Link Aggregation Ctrl	82823
Link Aggregation Membership	83326 - 83337
Link Aggregation To Physical Ports Members	82848 - 82859
MAC RX Broken Packets	48 - 59
MAC RX Short Packet Drop	60 - 71
MBSC Drop	85879 - 85890
MPLS EXP Field To Egress Queue Mapping Table	83256 - 83263
MPLS EXP Field To Packet Color Mapping Table	49991 - 49998
MPLS QoS Mapping Table	94662 - 95173
Maximum Allowed VLAN Drop	4478
Minimum Allowed VLAN Drop	4477
Minimum Buffer Free	86211
Next Hop DA MAC	86394 - 88441
Next Hop Hit Status	84799 - 85822
Next Hop MPLS Table	88442 - 89465
Next Hop Packet Insert MPLS Header	89470 - 93565
Next Hop Packet Modifications	46903 - 48950
Next Hop Table	45879 - 46902

Name	Address Range
Output Disable	86212 - 86223
Output Mirroring Table	95174 - 95185
PB Packet Head Counter	86256
PB Packet Tail Counter	86257
PS Error Counter	95250 - 95261
PS Packet Head Counter	95248
PS Packet Tail Counter	95249
Packet Buffer Status	86100
Port Move Options	82831
Queue Off Drop	85855 - 85866
Re-queue Overflow Drop	86102
Received Packets on Ingress VRF	84795 - 84798
Reserved Destination MAC Address Range	83610 - 83625
Reserved MAC DA Drop	4474
Reserved MAC SA Drop	4475
Reserved Source MAC Address Range	83594 - 83609
Resource Limiter Set	86068 - 86075
Router Egress Queue To VLAN Data	49999 - 50006
Router MTU Table	83128 - 83175
Router Port MAC Address	83386 - 83449
SMON Set 0 Byte Counter	84747 - 84754
SMON Set 0 Packet Counter	84731 - 84738
SMON Set 1 Byte Counter	84755 - 84762
SMON Set 1 Packet Counter	84739 - 84746
SMON Set Search	83324 - 83325
SP Overflow Drop	4416 - 4427
Scratch	4
Select Which Egress QoS Mapping Table To Use	93574 - 93829
Send to CPU	83626
Source Port Table	83338 - 83385
TOS QoS Mapping Table	94150 - 94661
Time to Age	244
Transmitted Packets on Egress VRF	95197 - 95200
Unknown Egress Drop	86271 - 86282
Unknown Ingress Drop	4466
VLAN Member Drop	4476
VLAN PCP And DEI To Color Mapping Table	49463 - 49478
VLAN PCP To Queue Mapping Table	83264 - 83271
VLAN Table	4883 - 13074

28.5 Active Queue Manager

28.5.1 ERM Red Configuration

Configurations to mark the buffer memory congestion status as Red (heavily congested).

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	86066

Field Description

Bits	Field Name	Description	Default Value
12:0	redXoff	Number of free cells below this value will invoke the red congestion check for the incoming cells. The checks include the number of enqueued cells in the current queue and the packet length. The incoming packet might be terminated and dropped based on the check result.	0x199
25:13	redXon	Once the red congestion check is applied, number of free cells need to go above this value to disable the check again. The value needs to be larger than **redXoff** to provide an effective hysteresis.	0x400
32:26	redMaxCells	Maximum allowed packet length in cells when the buffer memory congestion status is red.	0xb

28.5.2 ERM Yellow Configuration

Configurations to mark the buffer memory congestion status as Yellow (slightly congested).

Number of Entries : 1
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Address Space : 86064

Field Description

Bits	Field Name	Description	Default Value
12:0	yellowXoff	Number of free cells below this value will invoke yellow congestion checks for the incoming cells. The checks include the number of enqueued cells in the current queue, higher priority queues and optionally the total number of enqueued cells for the current egress port. Incoming packets might be terminated and dropped based on the check result.	0x69f
25:13	yellowXon	Once the yellow congestion check is applied, number of free cells need to go above this value to disable the check again. The value needs to be larger than **yellowXoff** to provide an effective hysteresis.	0x9e4
37:26	redPortEn	When the buffer memory congestion status is yellow and a single port consumes more than **redPortXoff** cells, this field can apply the **redLimit** check on a per port basis.	0xfff
50:38	redPortXoff	When the buffer memory congestion status is yellow and the total number of cells enqueued on an egress port is larger than this value, **redLimit** check for that port will be invoked. Only valid when **redPortEn** is turned on.	0x2ab

28.5.3 Egress Resource Manager Pointer

This table provides each egress port a set of limiters. Different egress queues can have different pointers to the **Resource Limiter Set**.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress port
Address Space : 86076 to 86087

Field Description

Bits	Field Name	Description	Default Value
1:0	q0	Pointer to the **Resource Limiter Set** for egress queue 0.	0x0
3:2	q1	Pointer to the **Resource Limiter Set** for egress queue 1.	0x0
5:4	q2	Pointer to the **Resource Limiter Set** for egress queue 2.	0x0
7:6	q3	Pointer to the **Resource Limiter Set** for egress queue 3.	0x0
9:8	q4	Pointer to the **Resource Limiter Set** for egress queue 4.	0x0
11:10	q5	Pointer to the **Resource Limiter Set** for egress queue 5.	0x0
13:12	q6	Pointer to the **Resource Limiter Set** for egress queue 6.	0x0
15:14	q7	Pointer to the **Resource Limiter Set** for egress queue 7.	0x0

28.5.4 Resource Limiter Set

This resource limiter is for comparing how many cells are ahead of the incoming cell for scheduling, that includes cells are enqueued in the same egress queue and all cells with a higher scheduling priority.

Number of Entries : 4
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Addressing : Pointer from the **Egress Resource Manager Pointer**
Address Space : 86068 to 86075

Field Description

Bits	Field Name	Description	Default Value
12:0	yellowAccumulated	When the buffer memory is slightly congested (yellow), the ERM allows accumulation of cells with the same queue or higher scheduling priorities to the limit in this field before appling the **yellowLimit**.	0x72
25:13	yellowLimit	When the buffer memory is slightly congested (yellow)and **yellowAccumulated** is reached, the packet will be terminated and dropped if the enqueued cells in the corresponding queue is more than this value.	0x20
38:26	redLimit	When the buffer memory is heavily congested (red), the incoming packet will be terminated and dropped if the enqueued cells in the corresponding egress queue is more than this value.	0x1a

Bits	Field Name	Description	Default Value
45:39	maxCells	Maximum allowed packet length in cells for this limiter. Packet with cells more than this value will be dropped.	0x7f

28.6 Core Information

28.6.1 Core Version

Adress 0 is reserved for the core version. Make sure the register value is the same as the revision number in the front page of the datasheet.

Number of Entries : 1
Type of Operation : Read Only
Address Space : 0

Field Description

Bits	Field Name	Description	Default Value
31:0	version	Version of the core.	0xcda53817

28.7 Egress Packet Processing

28.7.1 Color Remap From Egress Port

Options for remapping internal packet color to outgoing packet headers. Each egress port has a separate color to field mapping.

Number of Entries : 12
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Addressing : Egress Port
Address Space : 86334 to 86357

Field Description

Bits	Field Name	Description	Default Value
1:0	colorMode	0 = Skip remap 1 = Remap to L3 only 2 = Remap to L2 only 3 = Remap to L2 and L3	0x1
25:2	color2Tos	New TOS/TC value based on packet color. bits [0:7] : TOS/TC value for green bits [8:15] : TOS/TC value for yellow bits [16:23] : TOS/TC value for red	0x0

Bits	Field Name	Description	Default Value
33:26	tosMask	Mask for updating the TOS/TC field. For each bit in the mask, 0 means keep original value, 1 means update new value to that bit.	0x0
36:34	color2Dei	New DEI value based on packet color. This is located in the outermost VLAN of the transmitted packet. bit 0 : DEI value for green bit 1 : DEI value for yellow bit 2 : DEI value for red	0x0

28.7.2 Color Remap From Ingress Admission Control

Options from ingress admission control to remap internal packet color to outgoing packet headers.

Number of Entries : 16
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Addressing : Meter Pointer
Address Space : 86358 to 86389

Field Description

Bits	Field Name	Description	Default Value
0	enable	If set, the **colorMode** field determines the remap process. Otherwise color remapping based on the ingress admission control is skipped.	0x0
2:1	colorMode	0 = Remap disabled 1 = Remap to L3 only 2 = Remap to L2 only 3 = Remap to L2 and L3	0x0
26:3	color2Tos	New TOS/TC value based on packet color. bits [0:7] : TOS/TC value for green bits [8:15] : TOS/TC value for yellow bits [16:23] : TOS/TC value for red	0x0
34:27	tosMask	Mask for updating the TOS/TC field. For each bit in the mask, 0 means keep original value, 1 means update new value to that bit.	0x0
37:35	color2Dei	New DEI value based on packet color. This is located in the outermost VLAN of the transmitted packet. bit 0 : DEI value for green bit 1 : DEI value for yellow bit 2 : DEI value for red	0x0

28.7.3 Disable CPU tag on CPU Port

When a packet is sent to the CPU port normally a To CPU Tag will be added to the packet. This register provides a option to disable the CPU tag

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 95196

Field Description

Bits	Field Name	Description	Default Value
0	disable	When set, the CPU port will no longer add a CPU Tag to packets going to the CPU port. 0 = To CPU Tag enabled 1 = To CPU Tag disabled	0x0
1	disableReason0	When set, the CPU port will no longer add a CPU Tag to packets going to the CPU port with reason code 0(default reason). 0 = To CPU Tag enabled 1 = To CPU Tag disabled	0x0

28.7.4 Drain Port

Drop all packets on all queues to egress ports. The dropped packets are counted in the **Drain Port Drop** counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 86258

Field Description

Bits	Field Name	Description	Default Value
11:0	drainMask	Egress ports to be drained. One bit for each port in the current switch slice where bit 0 corresponds to local port 0.	0x0

28.7.5 Egress Ethernet Type for VLAN tag

Ethernet type used in VLAN operations when typeSel selects User Defined VLAN type. This Ethernet type is only used in VLAN push operations. In VLAN filtering a pushed user defined VLAN will be considered to be a C-VLAN.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 95195

Field Description

Bits	Field Name	Description	Default Value
15:0	typeValue	Ethernet Type value.	0xffff

28.7.6 Egress MPLS Decoding Options

When doing a Penultimate Pop then compare the first nibble after the innermost MPLS tag with this registers field nibbleForIpv4 to determine if the outgoing packet should have an IPv4 or IPv6 Ethernet Type.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 95194

Field Description

Bits	Field Name	Description	Default Value
3:0	nibbleForIpv4	The nibble value which is used to identify a IPv4 packet after a MPLS header. If the nibble does not match this value it is assumed to be an IPv6 packet.	0x4

28.7.7 Egress MPLS TTL Table

Configuration of what modification shall be done on the TTL field in MPLS routed packets.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : Packets VRF
Address Space : 89466 to 89469

Field Description

Bits	Field Name	Description	Default Value
0	addNewTTL	Select if the router should decremented TTL in the outgoing packet or if it should be set to a fixed value. 0 = Decrement TTL 1 = Set the TTL to **newTTL**	0x0
8:1	newTTL	New TTL for the packet. Only used when addNewTTL is set to 1	0x0

28.7.8 Egress Multiple Spanning Tree State

Table of egress Multiple Spanning Tree Protocol Instances. Depends on routed or not, the pointer used to address the instance/entry in this table can from **msptPtr** in the **Next Hop Packet Modifications** table or **msptPtr** in the **VLAN Table**. Each entry contains the ingress spanning tree states for all ports in this MSTI.

Number of Entries : 16
Type of Operation : Read/Write
Addressing : msptPtr from VLAN Table or Next Hop Packet Modifications Table
Address Space : 66407 to 66422

Field Description

Bits	Field Name	Description	Default Value
23:0	portSptState	The egress spanning tree state for this MSTI. Bit[1:0] is the state for port #0, bit[3:2] is the state for port #1, etc. 0 = Forwarding 1 = Discarding 2 = Learning	0x0

28.7.9 Egress Port Configuration

This table configures various functions that are dependent on which port the packet leaves the switch. A VLAN operation (e.g. push, pop, swap) to be performed can be selected by the **vlanSingleOp** field. For the push and swap operations the information used to create the new VLAN header is controlled by the fields **vidSel**, **cfiDeiSel**, **pcpSel** and **typeSel**. Other configurations are VLAN LUT index, port disable and different filtering rules based on packet VLAN fields when the egress processing is done.

Number of Entries :	12
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	86310 to 86333

Field Description

Bits	Field Name	Description	Default Value
0	colorRemap	If set, color remapping to outgoing packet headers is allowed. The default color remapping options are based on the egress port number from the **Color Remap From Egress Port** table. If a packet is subjected to ingress admission control, its ingress admission control pointer can provide remap options from the **Color Remap From Ingress Admission Control** table to override default options.	0x0
3:1	vlanSingleOp	The egress port VLAN operation to perform on the packet. 0 = No operation. 1 = Swap. 2 = Push. 3 = Pop. 4 = Penultimate pop(remove all VLAN headers).	0x0
5:4	typeSel	Selects which TPID to use when building a new VLAN header in a push or swap operation. 0 = C-VLAN - 0x8100. 1 = S-VLAN - 0x88A8. 2 = User defined VLAN type from register **Egress Ethernet Type for VLAN tag** field **typeValue**.	0x0

Bits	Field Name	Description	Default Value
7:6	vidSel	Selects which VID to use when building a new VLAN header in a egress port push or swap operation. If the selected outermost VLAN header doesn't exist in the packet then this table entry's **vid** will be used. 0 = From outermost VLAN in the packet (if any). 1 = From this table entry's **vid**. 2 = From the Ingress VID as selected in the **Source Port Table**.	0x0
9:8	cfiDeiSel	Selects which CFI/DEI to use when building a new VLAN header in a egress port push or swap operation. If the selected outermost VLAN header doesn't exist in the packet then this table entry's **cfiDei** will be used. 0 = From outermost VLAN in the packet (if any). 1 = From this table entry's **cfiDei**. 2 = From **Egress Queue To PCP And CFI/DEI Mapping Table**.	0x0
11:10	pcpSel	Selects which PCP to use when building a new VLAN header in a egress port push or swap operation. If the selected outermost VLAN header doesn't exist in the packet then this table entry's **cfiDei** will be used. 0 = From outermost VLAN in the packet (if any). 1 = From this table entry's **pcp**. 2 = From **Egress Queue To PCP And CFI/DEI Mapping Table**.	0x0
23:12	vid	The VID used in egress port VLAN push or swap operation if selected by **vidSel**.	0x0
24	cfiDei	The CFI/DEI used in egress port VLAN push or swap operation if selected by **cfiDeiSel**.	0x0
27:25	pcp	The PCP used in egress port VLAN push or swap operation if selected by **pcpSel**.	0x0
28	disabled	Disabling this port. All packets to this port is dropped and **Egress Port Disabled Drop** is incremented. 0 = All packets will be sent out. 1 = All packets will be dropped.	0x0
29	dropCtaggedVlans	Drop or allow customer VLANs tagged packets on this egress port. Will only drop packets that has exactly one VLAN tag. Must set moreThanOneVlans when this is used. 0 = Allow C-VLANs. 1 = Drop C-VLANs.	0x0
30	dropStaggedVlans	Drop or allow service VLANs tagged packets on this egress port. Will only drop packets that has exactly one VLAN tag. Must set moreThanOneVlans when this is used. 0 = Allow S-VLANs. 1 = Drop S-VLANs.	0x0

Bits	Field Name	Description	Default Value
31	moreThanOneVlans	When filtering with dropCtaggedVlans or dropStaggedVlans then this field must be set to 1.	0x0
32	dropUntaggedVlans	Drop or Allow packets that are VLAN untagged on this egress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
33	dropSingleTaggedVlans	Drop or Allow packets that has one VLAN tag on this egress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
34	dropDualTaggedVlans	Drop or allow packets which has more than one VLAN tag on this egress port. 0 = Allow packets which has more than one VLAN tag. 1 = Drop packets which has more than one VLAN tag.	0x0
35	dropCStaggedVlans	Drop or allow packets which has a C-VLAN followed by a S-VLAN tagged on this egress port. 0 = Allow packets which has a C-VLAN tag followed by a S-VLAN tag. 1 = Drop packets which has a C-VLAN tag followed by a S-VLAN tag.	0x0
36	dropSCtaggedVlans	Drop or allow packets which has a S-VLAN followed by a C-VLAN tagged on this egress port. 0 = Allow packets which has a S-VLAN followed by a C-VLAN tag. 1 = Drop packets which has a S-VLAN tag followed by a C-VLAN tag.	0x0
37	dropCCtaggedVlans	Drop or allow packets which has a C-VLAN followed by a C-VLAN tagged on this egress port. 0 = Allow packets which has a C-VLAN tag followed by a C-VLAN tag. 1 = Drop packets which has a C-VLAN tag followed by a C-VLAN tag.	0x0
38	dropSStaggedVlans	Drop or allow packets which has a S-VLAN followed by a S-VLAN tagged on this egress port. 0 = Allow packets which has a S-VLAN tag followed by a S-VLAN tag. 1 = Drop packets which has a S-VLAN tag followed by a S-VLAN tag.	0x0
39	useEgressQueueRemapping	Which remapping to final PCP, DEI, EXP and TOS fields shall be used for this port. 0 = Only use Egress Queue Remapping Tables 1 = First use the Egress Queue Remapping Tables then use the **Select Which Egress QoS Mapping Table To Use** to determine the final DEI,CFI,TOS and EXP fields.	0x0

28.7.10 Egress Queue To MPLS EXP Mapping Table

Map from egress queue number to MPLS EXP value to be used in MPLS operations selected by **Next Hop MPLS Table** and by **Next Hop Packet Insert MPLS Header** .

Number of Entries : 8
Type of Operation : Read/Write
Addressing : Egress Queue
Address Space : 95186 to 95193

Field Description

Bits	Field Name	Description	Default Value
2:0	exp	The outgoing Exp value for this queue.	0x0

28.7.11 Egress Queue To PCP And CFI/DEI Mapping Table

Get PCP and CFI/DEI from egress queues if selected by egress port VLAN operations push or swap.

Number of Entries : 8
Type of Operation : Read/Write
Addressing : Egress Queue
Address Space : 93566 to 93573

Field Description

Bits	Field Name	Description	Default Value
0	cfiDei	Map from egress queue to CFI/DEI.	0x0
3:1	pcp	Map from egress queue to PCP.	0x0

28.7.12 Egress Router Table

Configuration of what modification shall be done on the TTL field in routed packets.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : Packets VRF
Address Space : 86390 to 86393

Field Description

Bits	Field Name	Description	Default Value
0	addNewTTL	Select if the router should decremented TTL in the outgoing packet or if it should be set to a fixed value. 0 = Decrement TTL 1 = Set the TTL to **newTTL**	0x0
8:1	newTTL	New TTL for the packet. Only used when addNewTTL is set to 1	0x0

28.7.13 IP QoS Mapping Table

Set the outgoing packets PCP and CFI values for the outermost VLAN ID and ECN bits in the TOS Byte if selected from **Select Which Egress QoS Mapping Table To Use**. The rest of the TOS bits comes from the coloring mapping or MMP mapping tables.

Number of Entries : 256
Type of Operation : Read/Write

Addressing :

Address [2:0] :	The egress queue which the packet was queued on.
Address [4:3]:	The color of the packet.
Address [6:5] :	The ECN ToS bits TOS[1:0] after coloring operation.
Address [7] :	The Pointer from the **Select Which Egress QoS Mapping Table To Use whichTablePtr**.

Address Space : 93894 to 94149

Field Description

Bits	Field Name	Description	Default Value
0	updateCfiDei	Update CfiDei field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
1	cfiDei	Packets new CFI/DEI	0x0
2	updatePcp	Update Pcp field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
5:3	pcp	Packets new PCP	0x0
7:6	ecnTos	The outgoing TOS [1:0] ECN bits	0x0
8	updateExp	If the packet enterns a new MPLS tunnel using the **Next Hop Packet Insert MPLS Header** then use this Exp for the outermost MPLS label. 0 = No. Dont Remap. 1 = Yes. Remap to this new value	0x0
11:9	newExp	New Exp value to be used.	0x0

28.7.14 L2 QoS Mapping Table

Set the outgoing packets PCP and CFI values for the outermost VLAN ID if selected from **Select Which Egress QoS Mapping Table To Use**.

Number of Entries : 64
Type of Operation : Read/Write

Addressing :

Address [2:0] :	The egress queue which the packet was queued on.
Address [4:3]:	The color of the packet.
Address [5] :	The Pointer from the **Select Which Egress QoS Mapping Table To Use whichTablePtr**.

Address Space : 93830 to 93893

Field Description

Bits	Field Name	Description	Default Value
0	updateCfiDei	Update CfiDei field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
1	cfiDei	Packets new CFI/DEI.	0x0
2	updatePcp	Update Pcp field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
5:3	pcp	Packets new PCP.	0x0

28.7.15 MPLS QoS Mapping Table

Set the outgoing packets PCP and CFI values for the outermost VLAN ID and outermost EXP MPLS label if selected from **Select Which Egress QoS Mapping Table To Use**.

Number of Entries : 512
Type of Operation : Read/Write

Addressing :

Address [2:0] :	The egress queue which the packet was queued on.
Address [4:3]:	The color of the packet.
Address [7:5] :	The outermost label EXP bits.
Address [8] :	The Pointer from the **Select Which Egress QoS Mapping Table To Use whichTablePtr**.

Address Space : 94662 to 95173

Field Description

Bits	Field Name	Description	Default Value
0	updateCfiDei	Update CfiDei field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
1	cfiDei	Packets new CFI/DEI.	0x0
2	updatePcp	Update Pcp field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
5:3	pcp	Packets new PCP.	0x0
8:6	exp	The outgoing Exp value for this queue in the outermost MPLS label.	0x0

28.7.16 Next Hop DA MAC

Determines the destination MAC address to use in the packet exiting the router.

Number of Entries : 1024
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Addressing : **nextHopPacketMod**
Address Space : 86394 to 88441

Field Description

Bits	Field Name	Description	Default Value
47:0	daMac	The destination MAC address for the next hop.	0x0

28.7.17 Next Hop MPLS Table

Determines the MPLS tag operation to perform.

Number of Entries : 1024
Type of Operation : Read/Write
Addressing : **nextHopPacketMod**
Address Space : 88442 to 89465

Field Description

Bits	Field Name	Description	Default Value
2:0	mplsOperation	The egress MPLS tag operation to perform on the packet. 0 = No operation. 1 = Swap. 2 = Push. 3 = Pop. 4 = Penultimate Pop(remove all MPLS tags).	0x0
4:3	expSel	Select which EXP bits to use when building a new MPLS tag in Push or Swap operation. 0 = From this entries EXP field. 1 = From egress queue remapping in **Egress Queue To MPLS EXP Mapping Table** 2 = From the MPLS label (outermost MPLS tag if a swap and innermost if a push.	0x0
7:5	exp	Value to use for the EXP field when building a new MPLS tag in a swap or push operation.	0x0
27:8	label	MPLS label to use when building a new MPLS tag in a swap or push operation.	0x0

28.7.18 Next Hop Packet Insert MPLS Header

Shall MPLS lables (up tp 2) be inserted on the packet before it is sent out. This enables a IP packet to go into a MPLS tunnel. Header is placed after L2 and VLANs before the IP packet header. MPLS EXP field comes from destination queue to EXP mapping table defined in **Egress Queue To MPLS EXP Mapping Table**.

Number of Entries : 1024
Number of Addresses per Entry : 4
Type of Operation : Read/Write
Addressing : **nextHopPacketMod**
Address Space : 89470 to 93565

Field Description

Bits	Field Name	Description	Default Value
1:0	howManyLabelsToInsert	How many labels shall be inserted. Setting a zero here means no labels will be added.	0x0
2	whichEthernetType	Which Ethernet Type shall be used for these MPLS labels. 0 = 0x8847 1 = 0x8848	0x0
22:3	mplsLabel0	First/Outermost MPLS label to be inserter.	0x0
23	copyTtl0	Where shall the TTL come from in the MPLS label 0. 0 = From this table, field ttl0. 1 = From the inner packet.	0x0
31:24	ttl0	TTL table value for MPLS label 0.	0x0
32	expFromQueue0	Where shall the EXP come from in the MPLS label 0. 0 = From this table, field exp0. 1 = From the **Egress Queue To MPLS EXP Mapping Table**.	0x0
35:33	exp0	EXP table value for MPLS label 0.	0x0
55:36	mplsLabel1	MPLS label 1 to be inserter.	0x0
56	copyTtl1	Where shall the TTL come from in the MPLS label 1. 0 = From this table, field ttl1. 1 = From the inner packet.	0x0
64:57	ttl1	TTL table value for MPLS label 1.	0x0
65	expFromQueue1	Where shall the EXP come from in the MPLS label 1. 0 = From this table, field exp1. 1 = From the **Egress Queue To MPLS EXP Mapping Table**.	0x0
68:66	exp1	EXP table value for MPLS label 1.	0x0

28.7.19 Output Mirroring Table

Output mirroring configuration. An egress port can be set to have a mirrored port, but output mirroring cannot link more than one port. i.e. If Port A has an output mirroring Port B, Port B has an output mirroring Port C, packets sent to port A will not be mirrored to Port C.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress port
Address Space : 95174 to 95185

Field Description

Bits	Field Name	Description	Default Value
0	outputMirrorEnabled	If set to one, output mirroring is enabled for this port.	0x0
4:1	outputMirrorPort	Destination of output mirroring. Only valid if outputMirrorEnabled is set. Notice if the design contains more than one switch slice, packets egressed on one slice cannot be mirrored to another slice.	0x0

28.7.20 Select Which Egress QoS Mapping Table To Use

This is the initial table which is looked up by all packets in order to determine how the mapping from internal QoS to packets final PCP, DEI, TOS/EXP field shall look like. In order for this table to be executed the field **useEgressQueueRemapping** must be set to one.

Number of Entries : 256

Type of Operation : Read/Write

Addressing :

Address Bit [1:0]:	Forwarding type to this port. 0 = Switched Packet 1 = Routed Packet 2 = Classification Rule Forwarded Packet 3 = Others - Send-to-CPU and packet from CPU
Address Bit [3:2]:	Packet type 0 = L2 - Not IPv4/IPv6/MPLS 1 = IPv4 2 = IPv6 3 = MPLS
Address Bit [8:4]:	Egress Port

Address Space : 93574 to 93829

Field Description

Bits	Field Name	Description	Default Value
2:0	whichTableToUse	Select which table type to use. 0 = None. No remapping 1 = **L2 QoS Mapping Table** 2 = **IP QoS Mapping Table** 3 = **TOS QoS Mapping Table** 4 = **MPLS QoS Mapping Table** 5 = Use this tables remapping of DEI and PCP bits.	0x0
3	whichTablePtr	Which index of the tables to use. For most QoS tables there exists multiple tables to choose from.	0x0
4	updateCfiDei	Update CfiDei field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
5	cfiDei	Packets new CFI/DEI.	0x0
6	updatePcp	Update Pcp field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
9:7	pcp	Packets new PCP.	0x0

28.7.21 TOS QoS Mapping Table

Set the outgoing packets PCP and CFI values for the outermost VLAN ID and TOS Byte if selected from **Select Which Egress QoS Mapping Table To Use**. The input TOS byte to this mapping table comes from the coloring or MMP mapping tables.

Number of Entries : 512
Type of Operation : Read/Write

Addressing :	Address [7:0] : The TOS byte.
	Address [8] : The Pointer from the **Select Which Egress QoS Mapping Table To Use whichTablePtr**.

Address Space : 94150 to 94661

Field Description

Bits	Field Name	Description	Default Value
0	updateCfiDei	Update CfiDei field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
1	cfiDei	Packets new CFI/DEI	0x0
2	updatePcp	Update Pcp field in outgoing packet. 0 = Do not update. 1 = Update.	0x0
5:3	pcp	Packets new PCP	0x0
13:6	newTos	The outgoing new TOS bits	0x0
14	updateExp	If the packet enterns a new MPLS tunnel using the **Next Hop Packet Insert MPLS Header** then use this Exp for the outermost MPLS label. 0 = No. Dont Remap. 1 = Yes. Remap to this new value	0x0
17:15	newExp	New Exp value to be used.	0x0

28.8 Global Configuration

28.8.1 Core Tick Configuration

Global register for setting the frequency of the core tick

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 2

Field Description

Bits	Field Name	Description	Default Value
17:0	clkDivider	The master Core Tick will be issued once every $rg\_tick\_div.clkDivider$ core clock cycles. If set to zero, there will be no tick.	0xb4
21:18	stepDivider	The four ticks derived from the master core tick are issued once every $rg\_tick\_div.stepDivider^{tick\_number+1}$ master ticks. The master tick is tick number 0. If stepDivider is set to zero, there will be no ticks except possibly the master tick.	0xa

28.8.2 Core Tick Select

Global register for setting clock input to the core tick divider

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 3

Field Description

Bits	Field Name	Description	Default Value
1:0	clkSelect	Select the source clock for the Core Tick divider. 0: disabled, 1: core clock, 2: debug_write_data[0], 3: reserved	0x1

28.8.3 Scratch

Scratch Register

Number of Entries : 1
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Address Space : 4

Field Description

Bits	Field Name	Description	Default Value
63:0	scratch	scratch field.	0x0

28.9 Ingress Packet Processing

28.9.1 Check IPv4 Header Checksum

This register provides an option to drop the IPv4 packet if its header checksum field has an incorrect value.The option is only for not routed IPv4 packet. For a routed IPv4 packet, the checksum check is always performed.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82825

Field Description

Bits	Field Name	Description	Default Value
0	dropErrorChkSum	If set, always calculate the checksum of the received IPv4 packet. If the calculated value does not match the IPv4 checksum field, the packet is dropped.	0x0

28.9.2 Debug dstPortmask

Packet processing pipeline status for dstPortmask.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82835

Field Description

Bits	Field Name	Description	Default Value
11:0	value	Status from last processed packet.	0x0

28.9.3 Debug srcPort

Packet processing pipeline status for srcPort.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82834

Field Description

Bits	Field Name	Description	Default Value
31:0	value	Status from last processed packet.	0x0

28.9.4 Egress Spanning Tree State

Spanning tree state for each egress port. The state Disabled implies that spanning tree protocol is not enabled and hence frames will be forwarded on this egress port.

Number of Entries : 1
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Address Space : 83896

Field Description

Bits	Field Name	Description	Default Value
35:0	sptState	State of the spanning tree protocol. Bit[2:0] is port #0, bit[5:3] is port #1 etc. 0 = Disabled 1 = Blocking 2 = Listening 3 = Learning 4 = Forwarding	0x0

28.9.5 Enable Enqueue To Ports And Queues

This register is used to control if a particular port and queue shall be able to enqueue new packets. One queue mask exists for each port, setting a bit in the queue mask means packet is allowed to be queued on the respective queue. Packets that are directed to a queue that is turned off will be dropped and counted in **Queue Off Drop**.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Port
Address Space : 82836 to 82847

Field Description

Bits	Field Name	Description	Default Value
7:0	q_on	If a bit is set, the corresponding queue is on.	0xff

28.9.6 Force Non VLAN Packet To Specific Color

If a packet is non-VLAN tagged, there is an option to force these packets to a certain initial color.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82828

Field Description

Bits	Field Name	Description	Default Value
0	forceColor	When set, packets which are non-VLAN tagged are forced to a color.	0x0
2:1	color	Initial color of the packet	0x0

28.9.7 Force Non VLAN Packet To Specific Queue

If a packet is non-VLAN tagged, there is an option to force these packets to a certain ingress/egress queue.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82826

Field Description

Bits	Field Name	Description	Default Value
0	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 17.1	0x0

Bits	Field Name	Description	Default Value
3:1	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0

28.9.8 Force Unknown L3 Packet To Specific Color

If a packet does not contain IPv4, IPv6, MPLS or PPPoE carrying IPv4/IPv6 field there is an option to force the packet to a certain initial color.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82829

Field Description

Bits	Field Name	Description	Default Value
0	forceColor	When set, unknown L3 packet types are forced to a color.	0x0
2:1	color	Initial color of the packet	0x0

28.9.9 Force Unknown L3 Packet To Specific Egress Queue

If a packet does not contain IPv4, IPv6, MPLS or PPPoE carrying IPv4/IPv6 field there is an option to force the packet to a certain egress queue.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82827

Field Description

Bits	Field Name	Description	Default Value
0	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 17.1	0x0
3:1	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0

28.9.10 Forward From CPU

Indicates if all frames received on the CPU port shall be forwarded while ignoring the egress port's spanning tree status.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82830

Field Description

Bits	Field Name	Description	Default Value
0	enable	If set, any frame received on the CPU port is forwarded without consideration of the egress port's spanning tree state.	0x0

28.9.11 Hardware Learning Configuration

Configure default status for a newly learned entry, learning limits and learning exceptions.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Ingress Port
Address Space : 232 to 243

Field Description

Bits	Field Name	Description	Default Value
0	valid	For a new packet which is to be learned what value shall the valid bit have?	0x1
1	stat	For a new packet which is to be learned what value shall the static bit have?	0x0
2	hit	For a new packet which is to be learned what value shall the hit bit have?	0x1
15:3	learnLimit	Maximum number of entries can be learned on this port. 0 means no limit.	0x0
16	portMoveException	When the hardware learning unit is turned on and the ingress packet processing determines to bypass the hardware learning check, set this field to one to still perform the port move action.	0x0
17	saHitException	When the hardware learning unit is turned on and the ingress packet processing determines to bypass the hardware learning check, set this field to one to still perform the SA hit update action.	0x0

28.9.12 Hardware Learning Counter

Number of MAC addresses learned by the hardware learning unit. Write 0 to clear.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Ingress Port
Address Space : 262 to 273

Field Description

Bits	Field Name	Description	Default Value
12:0	cnt	Number of learned L2 entries.	0x0

28.9.13 Hash Based L3 Routing Table

This is the routing table used to determine the next hop. The IP lookup is done by hashing the VRF and the destination address extracted from the incoming packet. The hash is used to index this table. For each hash value the table has 4 buckets. The incoming IP address is compared with the destIPAddr field in all the buckets for the selected hash value. The packets assigned VRF is compared with the vrf fields and the protocol type is compares against the entries protocol. If there is a match in any bucket then the other fields in the matched bucket will be used for next hop processing. If ECMP is enabled for this entry an offset is added to the **nextHopPointer** and used when indexing the **Next Hop Table**.

Number of Entries :	4096
Number of Addresses per Entry :	8
Type of Operation :	Read/Write
Addressing :	address[0:9] : hash of {VRF, IP destination address} or {Source port and outermost MPLS label} address[10:11] : bucket number
Address Space :	13111 to 45878

Field Description

Bits	Field Name	Description	Default Value
0	ipVersion	Select if this is an IPv4 or IPv6 entry. 0 = IPv4 entry. 1 = IPv6 entry.	0x0
1	mpls	This is an MPLS entry, 0 = IP entry. 1 = MPLS entry.	0x0
3:2	vrf	This entries VRF. The packets assigned VRF will be compared with this field.	0x0
131:4	destIPAddr	The IP or MPLS address to be matched. If the entry is an IPv4 entry then only bits [31:0] is used. If the entry is a MPLS entry then bits [4-1:0] contains the source port while bits [4+19:4] contains the MPLS label to match.	0x0
141:132	nextHopPointer	Index into the **Next Hop Table** for this destination.	0x0
142	useECMP	Enables the use of ECMP hash to calculate the next hop pointer. 0 = Use ECMP hash. 1 = Do not use ECMP hash.	0x0
150:143	ecmpMask	How many bits of the ECMP hash will be used when calculating the ECMP offset. This byte is AND:ed with the ECMP hash to determine which bits shall be used as offset.	0x0
153:151	ecmpShift	How many bits the masked ECMP hash will be right shifted.	0x0

28.9.14 IPv4 TOS Field To Egress Queue Mapping Table

Mapping table from TOS in the IPv4 header to an egress queue.

Number of Entries : 256
Type of Operation : Read/Write
Addressing : Incoming IPv4 packets TOS
Address Space : 48951 to 49206

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueue	Egress queue.	0x1

28.9.15 IPv4 TOS Field To Packet Color Mapping Table

Mapping table from TOS in the IPv4 header to a packet inital color.

Number of Entries : 256
Type of Operation : Read/Write
Addressing : Incoming IPv4 packets TOS pointer
Address Space : 49479 to 49734

Field Description

Bits	Field Name	Description	Default Value
1:0	color	Packet initial color.	0x0

28.9.16 IPv6 Class of Service Field To Egress Queue Mapping Table

Mapping table from Class of Service in the IPv6 header to an egress queue.

Number of Entries : 256
Type of Operation : Read/Write
Addressing : Incoming IPv6 packets Class of Service
Address Space : 49207 to 49462

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueue	Egress queue.	0x1

28.9.17 IPv6 Class of Service Field To Packet Color Mapping Table

Mapping table from Class of service in the IPv6 header to a packet inital color.

Number of Entries : 256
Type of Operation : Read/Write
Addressing : Incoming IPv6 packets Class os Service pointer
Address Space : 49735 to 49990

Field Description

Bits	Field Name	Description	Default Value
1:0	color	Packet initial color.	0x0

28.9.18 Ingress Admission Control Current Status

Number of tokens currently in the token bucket.

Number of Entries : 16
Type of Operation : Read/Write
Addressing : Meter Pointer
Address Space : 86016 to 86031

Field Description

Bits	Field Name	Description	Default Value
15:0	tokens_0	Number of tokens after the last visit for token bucket 0.	0x0
31:16	tokens_1	Number of tokens after the last visit for token bucket 1.	0x0

28.9.19 Ingress Admission Control Initial Pointer

Initial ingress admission control pointer based on source port number and L2 priority. L2 priority is from either the outermost VLAN PCP field or **defaultPcp**. Further processes may overwrite the initial pointer by comparing the order of the pointer.

Number of Entries : 128
Type of Operation : Read/Write
Addressing :

address[3:0] :	Ingress Port
address[6:4] :	L2 Priority

Address Space : 4743 to 4870

Field Description

Bits	Field Name	Description	Default Value
0	mmpValid	If set, this entry contains a valid MMP pointer	0x0
4:1	mmpPtr	Initial pointer to the ingress MMP.	0x0
6:5	mmpOrder	Order of the initial ingress MMP pointer.	0x0

28.9.20 Ingress Admission Control Mark All Red

Blocking status of the MMP entry due to packet drops in the MMP.

Number of Entries : 16
Type of Operation : Read/Write
Addressing : Meter Pointer
Address Space : 85920 to 85935

Field Description

Bits	Field Name	Description	Default Value
0	markAllRed	When this field is set to 1 by the core, the corresponding MMP entry is under the blocking status. As a consequence, all packets with this MMP pointer will be dropped. Clear this field to allow packets enter the MMP entry again.	0x0

28.9.21 Ingress Admission Control Mark All Red Enable

Option to block metering after MMP packet drops.

Number of Entries : 16
Type of Operation : Read/Write
Addressing : Meter Pointer
Address Space : 85904 to 85919

Field Description

Bits	Field Name	Description	Default Value
0	markAllRedEn	After setting this field to 1, if a packet is dropped by a MMP entry, this MMP entry will stop metering and drop all packets with the corresponding MMP pointer.	0x0

28.9.22 Ingress Admission Control Reset

Reset token buckets so that it is back to the inital status. The reset will be kept high till new traffic arrives, then the traffic is metered with a bucket full of tokens and the reset is deactivated. It is helpful when the token bucket configuration is changed during runtime.

Number of Entries : 16
Type of Operation : Read/Write
Addressing : Meter Pointer
Address Space : 86000 to 86015

Field Description

Bits	Field Name	Description	Default Value
0	bucketReset	if set, reload with full tokens for token buckets in this entry.	0x1

28.9.23 Ingress Admission Control Token Bucket Configuration

Configuration options for token buckets used by Ingress Admission Control. Each entry refers to either a single rate three color marker (srTCM) or a two rate three color marker (trTCM) with two token buckets. For each token bucket the rate is configured by filling in a certain number of tokens at one of the available frequencies. Token bucket 0 shall always use the committed information rate (CIR). Runtime configuration update requires writting 1 to the **Ingress Admission Control Reset** first.

Number of Entries : 16
Number of Addresses per Entry : 4
Type of Operation : Read/Write
Addressing : Meter Pointer
Address Space : 85936 to 85999

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity_0	Capacity for token bucket 0.	0x0
27:16	tokens_0	Number of tokens added each tick for token bucket 0.	0x0
30:28	tick_0	Select one of the 5 available ticks for token bucket 0. The tick frequencies are configured globaly in the Core Tick Configuration register.	0x0
46:31	bucketCapacity_1	Capacity for token bucket 1.	0x0
58:47	tokens_1	Number of tokens added each tick for token bucket 1.	0x0
61:59	tick_1	Select one of the 5 available ticks for token bucket 1. The tick frequencies are configured globaly in the Core Tick Configuration register.	0x0
62	bucketMode	0 = srTCM 1 = trTCM	0x0
63	colorBlind	0 = color-aware: The metering result is based on the initial coloring from the ingress process pipeline. 1 = color-blind: The metering ignores any pre-coloring.	0x0
66:64	dropMask	Drop mask for the three colors obtained from the metering result. For each bit set to 1 the corresponding color shall drop the packet. Bit 0, 1, 2 represents drop or not for green, yellow and red respectively	0x4
80:67	maxLength	Maximum allowed packet length in bytes. Packets with bytes larger than this value will be dropped before metering.	0x3fff

Bits	Field Name	Description	Default Value
82:81	tokenMode	0 = Count in bytes and add extra bytes for metering. 1 = Count in bytes and substract extra bytes for metering. 2 = Count in packets. 3 = No tokens are counted.	0x0
90:83	byteCorrection	Extra bytes per packet for IFG correction, only valid under byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

28.9.24 Ingress Drop Options

Options to enable or disable learning when the the L2 forwarding process drops the packet.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 84474

Field Description

Bits	Field Name	Description	Default Value
0	learnL2DestDrop	Allow learning when L2 Destination Table drops the packet.	0x0
1	learnL2FloodDrop	Allow learning when the packet is dropped due to unknown DA.	0x0
2	learnL2DestVlanMemberDrop	Allow learning when the packt is dropped due to destination VLAN membership check.	0x1

28.9.25 Ingress Egress Port Packet Type Filter

This sets up which packets are to be dropped or allowed to be transmitted on each of the egress ports. This filtering is done after the source port tables VLAN operation and the VLAN tables VLAN operation. Notice this filter applies to L2 L3 forwarding result only, any other special rules could bypass it (traffic to/from CPU port, classifications, etc). Packets dropped due to this filter will be counted in **Ingress-Egress Packet Filtering Drop**.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress port
Address Space : 83116 to 83127

Field Description

Bits	Field Name	Description	Default Value
0	dropCtaggedVlans	Drop or allow customer VLAN tagged packets on this egress port. Will only drop packets that has exactly one VLAN tag. Must set moreThanOneVlans when this is used. Note that after a VLAN push operation the pushed VLAN will be regarded as a C-VLAN. 0 = Allow C-VLANs. 1 = Drop C-VLANs.	0x0
1	dropStaggedVlans	Drop or allow service VLAN tagged packets on this egress port. Must set moreThanOneVlans when this is used. Note that after a VLAN push operation the pushed VLAN will be regarded as a C-VLAN. 0 = Allow S-VLANs. 1 = Drop S-VLANs.	0x0
2	moreThanOneVlans	When filtering with dropCtaggedVlans or dropStaggedVlans then this field must be set to 1.	0x0
3	dropSingleTaggedVlans	Drop or Allow packets that are VLAN untagged on this egress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
4	dropUntaggedVlans	Drop or Allow packets that are VLAN untagged on this egress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
5	dropIPv4Packets	Drop or allow IPv4 packets on this egress port. 0 = Allow IPv4 packets. 1 = Drop IPv4 packets.	0x0
6	dropIPv6Packets	Drop or allow IPv6 packets on this egress port. 0 = Allow IPv6 packets. 1 = Drop IPv6 packets.	0x0
7	dropMPLSPackets	Drop or allow MPLS packets on this source port. 0 = Allow MPLS packets. 1 = Drop MPLS packets.	0x0
8	dropIPv4MulticastPackets	Drop or allow IPv4 Multicast packets on this egress port. 0 = Allow IPv4 MC packets. 1 = 1 = Drop IPv4 MC packets.	0x0
9	dropIPv6MulticastPackets	Drop or allow IPv6 Multicast packets on this egress port. 0 = Allow IPv6 MC packets. 1 = Drop IPv6 MC packets.	0x0
10	dropL2BroadcastFrames	Drop or allow L2 broadcast packets on this egress port. 0 = Allow L2 broadcast packets. 1 = Drop L2 broadcast packets.	0x0
11	dropL2FloodingFrames	Drop or allow L2 flooding packets on this egress port. Observe that this rule takes the **unknownL2McFilterRule** into account. 0 = Allow L2 flooding packets. 1 = Drop L2 flooding packets.	0x0

Bits	Field Name	Description	Default Value
12	dropL2MulticastFrames	Drop or allow L2 multicast packets on this egress port. Observe that this L2 multicast bit takes the register **L2 Multicast Handling** into account to determine if this packet is a L2 multicast packet or not. 0 = Allow L2 multicast packets 1 = Drop L2 multicast packets.	0x0
13	dropDualTaggedVlans	Drop or allow packets with has more than one VLAN tag on this egress port. 0 = Allow packets which has more than one VLAN tag. 1 = Drop packets which has more than one VLAN tag.	0x0
14	dropCStaggedVlans	Drop or allow packets with has a C-VLAN followed by a S-VLAN tagged on this egress port. Note that after a VLAN push operation the pushed VLAN will be regarded as a C-VLAN. 0 = Allow packets which has a C-VLAN tag followed by a S-VLAN tag. 1 = Drop packets which has a C-VLAN tag followed by a S-VLAN tag.	0x0
15	dropSCtaggedVlans	Drop or allow packets with has a S-VLAN followed by a C-VLAN tagged on this egress port. Note that after a VLAN push operation the pushed VLAN will be regarded as a C-VLAN. 0 = Allow packets which has a S-VLAN followed by a C-VLAN tag. 1 = Drop packets which has a S-VLAN tag followed by a C-VLAN tag.	0x0
16	dropCCtaggedVlans	Drop or allow packets with has a C-VLAN followed by a C-VLAN tagged on this egress port. Note that after a VLAN push operation the pushed VLAN will be regarded as a C-VLAN. 0 = Allow packets which has a C-VLAN tag followed by a C-VLAN tag. 1 = Drop packets which has a C-VLAN tag followed by a C-VLAN tag.	0x0
17	dropSStaggedVlans	Drop or allow packets with has a S-VLAN followed by a S-VLAN tagged on this egress port. Note that after a VLAN push operation the pushed VLAN will be regarded as a C-VLAN. 0 = Allow packets which has a S-VLAN tag followed by a S-VLAN tag. 1 = Drop packets which has a S-VLAN tag followed by a S-VLAN tag.	0x0
18	dropRouted	Drop or allow packets which has been routed on this egress port. 0 = Allow packets which has been routed. 1 = Drop packets which has been routed.	0x0
30:19	srcPortFilter	Each egress port has an optional way of ensuring that a specific source port does not send out a packet on a specific egress port. By setting a bit in this port mask, the packets originating from that source port will be dropped and not be allowed to reach this egress port.	0x0

28.9.26 Ingress Ethernet Type for VLAN tag

When decoding VLAN tags, if the Ethernet Type matches the **typeValue** it will be considered to be a VLAN tag in addition to the standard values of 0x8100 and 0x88A8. The **type** field determines if the VLAN should be regarded as a Service VLAN or Customer VLAN.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82824

Field Description

Bits	Field Name	Description	Default Value
15:0	typeValue	Ethernet Type value.	0xffff
16	type	User defined VLAN type. 0 = Customer VLAN. 1 = Service VLAN.	0x0
17	valid	User defined VLAN is valid. 0 = Not Valid. 1 = Valid.	0x0

28.9.27 Ingress L2 ACL Match Data Entries

All packets entering the switch will be subjected to ACL filtering. This allows custom packet processing to be done for certain selected packets. For each entry it can be selected which fields that shall be part of the comparison. The entries in the table are searched starting with entry 0.

Number of Entries : 32
Number of Addresses per Entry : 8
Type of Operation : Read/Write
Addressing : All entries are read out in parallel
Address Space : 83638 to 83893

Field Description

Bits	Field Name	Description	Default Value
0	compareEthType	Determines if the EthType field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
1	typeOfComparisonEthType	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
17:2	ethType	Ethernet Type after VLAN.	0x0
18	compareDaMac	Determines if the DaMac field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0

Bits	Field Name	Description	Default Value
19	typeOfComparisonDaMac	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
67:20	daMac	Destination MAC address.	0x0
68	compareSaMac	Determines if the SaMac field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
69	typeOfComparisonSaMac	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
117:70	saMac	Source MAC address.	0x0
118	compareVid	Determines if the Vid field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
119	typeOfComparisonVid	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
131:120	vid	Compared with the packets VLAN VID after Ingress VID assignment and Source Port Table VLAN operation.	0x0
132	comparePcp	Determines if the Pcp field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
133	typeOfComparisonPcp	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
136:134	pcp	Compared with the packets VLAN PCP after Source Port Table VLAN operation.	0x0
137	compareDei	Determines if the Dei field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
138	typeOfComparisonDei	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
139	dei	Compared with the packets VLAN CFI/DEI after Source Port Table VLAN operation.	0x0

Bits	Field Name	Description	Default Value
140	compareHasVlans	Determines if the HasVlans field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
141	typeOfComparisonHasVlans	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
142	hasVlans	Is there at least one VLAN in the packet. 0 = No VLAN 1 = One or More VLAN	0x0
143	compareCstag	Determines if the Cstag field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
144	typeOfComparisonCstag	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
145	cstag	Is the outermost VLAN tag a C-tag or S-Tag. If a packet does not have a VLAN or the VLAN was removed due to a pop operation in the source port vlan operation then the value will be set to zero(0). 0 = C-tag 1 = S-tag	0x0
157:146	ports	Ports that this filter rule applies to.	0x0

28.9.28 Ingress L2 ACL Result Operation Entries

The highest entry rule that is matched by the ACL comparison will determine what operations to perform from the corresponding entry number in this table. The VLAN swap operations, updateCfiDei, updatePcp and updateVid are performed after any **VLAN Table** operation.

Number of Entries : 32
Number of Addresses per Entry : 4
Type of Operation : Read/Write
Addressing : **Ingress L2 ACL Match Data Entries** hit index
Address Space : 83466 to 83593

Field Description

Bits	Field Name	Description	Default Value
0	dropEnable	If set, the packet shall be dropped and the **Ingress L2 ACL Drop** counter is incremented.	0x0
1	sendToCpu	If set, the packet shall be sent to the CPU port.	0x0
2	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 17.1	0x0

Bits	Field Name	Description	Default Value
5:3	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
7:6	color	Inital color of the packet	0x0
8	forceColor	If set, the packet shall have a forced color.	0x0
9	mmpValid	If set, this entry contains a valid MMP pointer	0x0
13:10	mmpPtr	Ingress MMP pointer.	0x0
15:14	mmpOrder	Ingress MMP pointer order.	0x0
16	sendToPort	Send the packet to a specific port. 0 = Do not sent to a port. 1 = Send to port.	0x0
20:17	destPort	The port which the packet shall be sent to.	0x0
21	forceVidValid	A new ingress VID shall be used when doing the VLAN table lookup. This is the VID which is used for the VLAN lookup overriding the source port tables VID assignment.	0x0
34:22	forceVid	The new ingress VID which shall be used in the VLAN lookup.	0x0
35	updateCounter	When set the selected statistics counter will be updated.	0x0
40:36	counter	Which counter in **Ingress L2 ACL Match Counter** to update.	0x0
41	updateCfiDei	The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
42	newCfiDeiValue	CFIDEI The value to update to.	0x0
43	updatePcp	The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
46:44	newPcpValue	The PCP value to update to.	0x0
47	updateVid	The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
59:48	newVidValue	The VID value to update to.	0x0
60	updateEType	The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
62:61	newEthType	Selects which TPID to use in the outer VLAN header. 0 = C-VLAN - 0x8100. 1 = S-VLAN - 0x88A8. 2 = User defined VLAN type from register **Egress Ethernet Type for VLAN tag** .	0x0
63	useChain	A resulting chain ID shall be used when doing the L3 ACL lookups	0x0
69:64	chainId	The chain Identifier value to use.	0x0

28.9.29 Ingress L3/L4 ACL Match Data Entries

All packets entering the switch will be subjected to the L3/L4 ACL filering. Each of the fields has a valid bit saying if the field shall be included in the comparison. Each field has a type of comparison allowing direct match or not equal. For each entry which results in a hit the corresponding result entry in **Ingress**

L3/L4 ACL Result Operation Entries is read out and acted on. When multiple entries match (are hit) the associated actions from all matching entries will be executed. The entries in the table are searched starting with entry 0.

Number of Entries :	32
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	83962 to 84473

Field Description

Bits	Field Name	Description	Default Value
127:0	sip	The Source IPv4/IPv6 address value to be compared. If IPv4 then only bits [31:0] are compared.	0x0
255:128	dip	The Destination IPv4/IPv6 address or the outermost MPLS label value to be compared. If IPv4 then only bits [31:0] are compared. If MPLS then only bits [19:0] are compared.	0x0
264:256	compareTcpFlagMask	Which of the TCP bits shall be compared. For each bit: 1 = The bit has to match, 0 = The bit is ignored.	0x0
273:265	tcpFlags	The TCP flags to compare. Bit [8] = ns, Bit [7] = cwr, Bit[6] = ece, Bit[5] = urg, Bit[4] = ack, Bit[3] = psh, Bit [2] = rst, Bit[1] = syn , Bit[0] = fin	0x0
281:274	compareIPv4OptionsByteMask	Which bits of the IPv4 options byte shall be compared. Setting a bit to 1 means the bit has to match, setting the bit to 0 means it is ignored.	0x0
289:282	IPv4OptionsByte	The first IPv4 options byte to compare to.	0x0
290	compareSip	In this ACL entry shall the Source IPv4/IPv6 Address be compared. 0 = Do not compare. 1 = Include the comparison in the entry hit decision.	0x0
291	typeOfComparisonSip	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
292	compareDip	In this ACL entry shall the Destination IPv4/IPv6 Address or the outermost MPLS label be compared. 0 = Do not compare. 1 = Include the comparison in the entry hit decision.	0x0
293	typeOfComparisonDip	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
294	compareHasChain	Determines if the HasChain field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0

Bits	Field Name	Description	Default Value
295	typeOfComparisonHasChain	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
296	hasChain	A chain tag from a first or second ACL shall be used.	0x0
297	compareL4sp	Determines if the L4sp field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
298	typeOfComparisonL4sp	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
314:299	l4sp	L4 Source Port.	0x0
315	compareL4dp	Determines if the L4dp field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
316	typeOfComparisonL4dp	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
332:317	l4dp	L4 Destination Port.	0x0
333	compareTos	Determines if the Tos field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
334	typeOfComparisonTos	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
342:335	tos	TOS Byte.	0x0
343	compareL4Type	Determines if the L4Type field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
344	typeOfComparisonL4Type	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
352:345	l4Type	L4 Payload Type.	0x0
353	compareIsIPv4	Determines if the IsIPv4 field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0

Bits	Field Name	Description	Default Value
354	typeOfComparisonIsIPv4	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
355	isIPv4	IPv4 Packet Type.	0x0
356	compareIsIPv6	Determines if the IsIPv6 field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
357	typeOfComparisonIsIPv6	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
358	isIPv6	IPv6 Packet Type.	0x0
359	compareIsMPLS	Determines if the IsMPLS field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
360	typeOfComparisonIsMPLS	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
361	isMPLS	MPLS Packet Type.	0x0
362	compareRouted	Determines if the Routed field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
363	typeOfComparisonRouted	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
364	routed	Routed Result.	0x0
365	compareVrf	Determines if the Vrf field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0
366	typeOfComparisonVrf	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
368:367	vrf	Packet VRF	0x0
369	compareChainTag	Determines if the ChainTag field in this entry shall be compared. 0 = Do not compare. 1 = Include the comparison in the entry comparison.	0x0

Bits	Field Name	Description	Default Value
370	typeOfComparisonChainTag	What type of comparison shall be considered as hit. 0 = Not equal shall be considered a hit. 1 = Equal as hit.	0x1
376:371	chainTag	The chain tag from either ingress l2 classification lookups.	0x0
388:377	ports	For which source ports should this filter rule apply.	0x0

28.9.30 Ingress L3/L4 ACL Result Operation Entries

The highest entry rule that is matched by the L3/L4 ACL comparison will determine what operations to perform by looking up corresponding entry number in this table.

Number of Entries : 32
Type of Operation : Read/Write
Addressing : **Ingress L3/L4 ACL Match Data Entries** hit index
Address Space : 83272 to 83303

Field Description

Bits	Field Name	Description	Default Value
0	dropEnable	If set, the packet shall be dropped and the **L3 ACL Drop** counter is incremented.	0x0
1	sendToCpu	If set, the packet shall be sent to the CPU port.	0x0
2	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 17.1	0x0
5:3	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
7:6	color	Inital color of the packet.	0x0
8	forceColor	If set, the packet shall have a forced color.	0x0
9	mmpValid	If set, this entry contains a valid MMP pointer	0x0
13:10	mmpPtr	Ingress MMP pointer.	0x0
15:14	mmpOrder	Ingress MMP pointer order.	0x0
16	sendToPort	Send the packet to a specific port. 0 = Dont send. 1 = Send to port. Observe that if other ACL units also has sendToPort turned on then the packet will be sent to both ports.	0x0
20:17	destPort	The port which the packet shall be sent to.	0x0
21	updateCounter	Update a counter.	0x0
26:22	counter	Which counter to update.	0x0

28.9.31 Ingress MMP Drop Mask

This register provides an option to let ingress MMP not drop packets on certain ports after metering.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82833

Field Description

Bits	Field Name	Description	Default Value
11:0	dropMask	Each bit in this mask refers to if ingress MMP drop is allowed on the corresponding egress port.	0xfff

28.9.32 Ingress Multiple Spanning Tree State

Table of ingress Multiple Spanning Tree Protocol Instances. For routed packets the pointer used to address this table is from the **msptPtr** field in the **Next Hop Packet Modifications** table. For switched packets is is from the **msptPtr** field in the **VLAN Table**. Each entry contains the ingress spanning tree states for all ports in this MSTI.

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	msptPtr from VLAN Table or Next Hop Packet Modifications Table
Address Space :	13075 to 13090

Field Description

Bits	Field Name	Description	Default Value
23:0	portSptState	The ingress spanning tree state for this MSTI. Bit[1:0] is the state for port #0, bit[3:2] is the state for port #1, etc. 0 = Forwarding 1 = Discarding 2 = Learning	0x0

28.9.33 Ingress Port Packet Type Filter

This configures which packet types that are to be dropped or allowed on each source port. Each entry corresponds to one ingress port. Packets dropped due to the filter are counted in **Ingress Packet Filtering Drop**.

Number of Entries :	12
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	83312 to 83323

Field Description

Bits	Field Name	Description	Default Value
0	dropCtaggedVlans	Drop or allow customer VLAN tagged packet on this ingress port. Will only drop packets that has exactly one VLAN tag. Must set moreThanOneVlans when this is used. 0 = Allow C-VLANs. 1 = Drop C-VLANs.	0x0

Bits	Field Name	Description	Default Value
1	dropStaggedVlans	Drop or allow service VLANs tagged packets on this ingress port. Will only drop packets that has exactly one VLAN tag. Must set moreThanOneVlans when this is used. 0 = Allow S-VLANs. 1 = Drop S-VLANs.	0x0
2	moreThanOneVlans	When filtering with dropCtaggedVlans or dropStaggedVlans then this field must be set to 1.	0x0
3	dropUntaggedVlans	Drop or Allow packets that are VLAN untagged on this ingress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
4	dropSingleTaggedVlans	Drop or Allow packets that are VLAN untagged on this ingress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
5	dropIPv4Packets	Drop or allow IPv4 packets on this ingress port. 0 = Allow IPv4 packets. 1 = Drop IPv4 packets.	0x0
6	dropIPv6Packets	Drop or allow IPv6 packets on this ingress port. 0 = Allow IPv6 packets. 1 = Drop IPv6 packets.	0x0
7	dropMPLSPackets	Drop or allow MPLS packets on this ingress port. 0 = Allow MPLS packets. 1 = Drop MPLS packets.	0x0
8	dropIPv4MulticastPackets	Drop or allow IPv4 multicast packets on this ingress port. 0 = Allow IPv4 MC packets. 1 = Drop IPv4 MC packets.	0x0
9	dropIPv6MulticastPackets	Drop or allow IPv6 multicast packets on this ingress port. 0 = Allow IPv6 MC packets. 1 = Drop IPv6 MC packets.	0x0
10	dropL2BroadcastFrames	Drop or allow L2 broadcast packets on this ingress port. 0 = Drop L2 broadcast packets. 1 = Allow L2 broadcast packets.	0x0
11	dropL2MulticastFrames	Drop or allow L2 multicast packets on this ingress port. Observe that this L2 multicast bit takes the register **L2 Multicast Handling** into account to determine if this packet is a L2 multicast packet or not. 0 = Allow L2 multicast packets 1 = Drop L2 multicast packets.	0x0
12	dropDualTaggedVlans	Drop or allow packets which has more than one VLAN tag on this ingress port. 0 = Allow packets which has dual tags. 1 = Drop packets which has dual tags.	0x0

Bits	Field Name	Description	Default Value
13	dropCStaggedVlans	Drop or allow packets which has a C-VLAN followed by a S-VLAN tagged on this ingress port. 0 = Allow packets which has a C-VLAN tag followed by a S-VLAN tag. 1 = Drop packets which has a C-VLAN tag followed by a S-VLAN tag.	0x0
14	dropSCtaggedVlans	Drop or allow packets which has a S-VLAN followed by a C-VLAN tagged on this ingress port. 0 = Allow packets which has a S-VLAN followed by a C-VLAN tag. 1 = Drop packets which has a S-VLAN tag followed by a C-VLAN tag.	0x0
15	dropCCtaggedVlans	Drop or allow packets which has a C-VLAN followed by a C-VLAN tagged on this ingress port. 0 = Allow packets which has a C-VLANs tag followed by a C-VLAN tag. 1 = Drop packets which has a C-VLAN tag followed by a C-VLAN tag.	0x0
16	dropSStaggedVlans	Drop or allow packets which has a S-VLAN followed by a S-VLAN tagged on this source port. 0 = Allow packets which has a S-VLAN tag followed by a S-VLAN tag. 1 = Drop packets which has a S-VLAN tag followed by a S-VLAN tag.	0x0

28.9.34 Ingress Router Table

The ingress router table or the Virtual Router Function (VRF), controls which packets are allowed to get access to this router. If a packet is dropped due to the settings of **Ingress Router Table** accept fields then the **Invalid Routing Protocol Drop** will be incremented. Updates for the **Next Hop Hit Status** is also controlled in this table.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : **vrf**
Address Space : 13091 to 13094

Field Description

Bits	Field Name	Description	Default Value
0	acceptIPv4	Accept IPv4 packets. If disabled and an IPv4 packet reaches the router the packet will be dropped and the **Invalid Routing Protocol Drop** incremented. 0 = Deny 1 = Accept	0x0
1	acceptIPv6	Accept IPv6 packets. If disabled and an IPv6 packet reaches the router the packet will be dropped and the **Invalid Routing Protocol Drop** incremented. 0 = Deny 1 = Accept	0x0

Bits	Field Name	Description	Default Value
2	acceptMPLS	Accept MPLS packets. If disabled and an MPLS packet reaches the router the packet will be dropped and the **Invalid Routing Protocol Drop** incremented. 0 = Deny 1 = Accept	0x0
10:3	minTTL	Minimum TTL. Packets with a TTL below this value will not be accepted. The packet will be dropped and the **Expired TTL Drop** counter incremented. If the **minTtlToCpu** is set the packet will be sent to CPU instead of being dropped. The TTL check is done for IPv4, IPv6 and MPLS routed packets.	0x0
11	minTtlToCpu	If this is set then packets below minimum TTL will be send to CPU instead of dropped.	0x0
12	ipv4HitUpdates	Enable updates of the **Next Hop Hit Status** for routed IPv4 packets. 0 = Disable 1 = Enable	0x0
13	ipv6HitUpdates	Enable updates of the **Next Hop Hit Status** for routed IPv6 packets. 0 = Disable 1 = Enable	0x0
14	mplsHitUpdates	Enable updates of the **Next Hop Hit Status** for routed MPLS packets. 0 = Disable 1 = Enable	0x0
15	ecmpUseIpDa	Use IP destination address as part of ECMP hash key.	0x1
16	ecmpUseIpSa	Use IP source address as part of ECMP hash key.	0x1
17	ecmpUseIpTos	Use IP TOS/Traffic Class as part of ECMP hash key.	0x0
18	ecmpUseIpProto	Use IP Protocol/Next Header as part of ECMP hash key.	0x1
19	ecmpUseIpL4Sp	Use TCP/UDP source port as part of ECMP hash key.	0x1
20	ecmpUseIpL4Dp	Use TCP/UDP destination port as part of ECMP hash key.	0x1
21	mmpValid	If set, this entry contains a valid MMP pointer. Only valid when packets get routed	0x0
25:22	mmpPtr	Ingress MMP pointer.	0x0
27:26	mmpOrder	Ingress MMP pointer order.	0x0
28	sendToCpuOrDrop	When a check if the packet protocols are allowed on this Ingress Router Table shall the packets be dropped or sent-to-CPU? 0 = Dropped. 1 = Sent-To-CPU	0x0

28.9.35 Ingress VID Ethernet Type Range Assignment Answer

The ingress VID to be assigned when the corresponding range matched.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : **Ingress VID Ethernet Type Range Search Data** hit index
Address Space : 83308 to 83311

Field Description

Bits	Field Name	Description	Default Value
11:0	ingressVid	Ingress VID.	0x0
13:12	order	Order for this assignment. If the ingress VID can be assigned from other packet field ranges, the one with the highest order wins.	0x0

28.9.36 Ingress VID Ethernet Type Range Search Data

This Ethernet type range can be used to assign the ingress VID. The search starts from entry 0 and returns the first match to lookup in the **Ingress VID Ethernet Type Range Assignment Answer** table.

Number of Entries : 4
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Addressing : All entries are read out in parallel
Address Space : 83938 to 83945

Field Description

Bits	Field Name	Description	Default Value
11:0	ports	Ports that this range search is activated on.	0x0
27:12	start	Start of Ethernet type range.	0x0
43:28	end	End of Ethernet type range.	0x0

28.9.37 Ingress VID Inner VID Range Assignment Answer

The ingress VID to be assigned when the corresponding range matched.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : **Ingress VID Inner VID Range Search Data** hit index
Address Space : 4879 to 4882

Field Description

Bits	Field Name	Description	Default Value
11:0	ingressVid	Ingress VID.	0x0
13:12	order	Order for this assignment. If the ingress VID can be assigned from other packet field ranges, the one with the highest order wins.	0x0

28.9.38 Ingress VID Inner VID Range Search Data

If a packet has an inner VLAN tag, this inner VID range can be used to assign the ingress VID. The search starts from entry 0 and returns the first match to lookup in the **Ingress VID Inner VID Range Assignment Answer** table.

Number of Entries :	4
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	83946 to 83953

Field Description

Bits	Field Name	Description	Default Value
11:0	ports	Ports that this range search is activated on.	0x0
12	vtype	Shall this entry match S-Type or C-Type VLAN. 0 = C-Type 1 = S-Type	0x0
24:13	start	Start of VID range.	0x0
36:25	end	End of VID range.	0x0

28.9.39 Ingress VID MAC Range Assignment Answer

The ingress VID to be assigned when the corresponding range matched.

Number of Entries :	4
Type of Operation :	Read/Write
Addressing :	**Ingress VID MAC Range Search Data** hit index
Address Space :	4871 to 4874

Field Description

Bits	Field Name	Description	Default Value
11:0	ingressVid	Ingress VID.	0x0
13:12	order	Order for this assignment. If the ingress VID can be assigned from other packet field ranges, the one with the highest order wins.	0x0

28.9.40 Ingress VID MAC Range Search Data

This MAC address range can be used to assign the ingress VID. The search starts from entry 0 and returns the first match to lookup in the **Ingress VID MAC Range Assignment Answer** table.

Number of Entries :	4
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	83450 to 83465

Field Description

Bits	Field Name	Description	Default Value
11:0	ports	Ports that this range search is activated on.	0x0
12	saOrDa	Is this rule for source or destination MAC address. 0 = Source MAC 1 = Destination MAC	0x0
60:13	start	Start of MAC address range.	0x0
108:61	end	End of MAC address range.	0x0

28.9.41 Ingress VID Outer VID Range Assignment Answer

The ingress VID to be assigned when the corresponding range matched.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : **Ingress VID Outer VID Range Search Data** hit index
Address Space : 4875 to 4878

Field Description

Bits	Field Name	Description	Default Value
11:0	ingressVid	Ingress VID.	0x0
13:12	order	Order for this assignment. If the ingress VID can be assigned from other packet field ranges, the one with the highest order wins.	0x0

28.9.42 Ingress VID Outer VID Range Search Data

If a packet has an outer VLAN tag, this outer VID range can be used to assign the ingress VID. The search starts from entry 0 and returns the first match to lookup in the **Ingress VID Outer VID Range Assignment Answer** table.

Number of Entries : 4
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Addressing : All entries are read out in parallel
Address Space : 83954 to 83961

Field Description

Bits	Field Name	Description	Default Value
11:0	ports	Ports that this range search is activated on.	0x0
12	vtype	Shall this entry match S-Type or C-Type VLAN. 0 = C-Type 1 = S-Type	0x0
24:13	start	Start of VID range.	0x0
36:25	end	End of VID range.	0x0

28.9.43 L2 Aging Collision Shadow Table

This table traces the **valid** field of the **L2 Aging Collision Table** and is used by L2 forwarding to check if a hit in the **L2 Lookup Collision Table** is valid. Any software write to this table shall be updated to the **valid** field of the **L2 Aging Collision Table**.

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	**L2 Lookup Collision Table** hit index
Address Space :	83240 to 83255

Field Description

Bits	Field Name	Description	Default Value
0	valid	If this is set, then the corresponding **L2 Lookup Collision Table** entry is valid.	0x0

28.9.44 L2 Aging Collision Table

This table holds the status of the entries in the **L2 Lookup Collision Table**. Any software write to the **valid** field in this table shall be done in the **L2 Aging Collision Shadow Table**.

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	**L2 Lookup Collision Table** hit index
Address Space :	246 to 261

Field Description

Bits	Field Name	Description	Default Value
0	valid	If this is set, then the corresponding **L2 Lookup Collision Table** entry is valid.	0x0
1	stat	If this is set, then the corresponding **L2 Lookup Collision Table** entry will not be aged out.	0x0
2	hit	If this is set, then the corresponding **L2 Lookup Collision Table** entry has a L2 SA/DA search hit since the last aging scan.	0x0

28.9.45 L2 Aging Status Shadow Table

This table traces the **valid** field of the **L2 Aging Table** and is used by L2 forwarding to check if a hit in the **L2 DA Hash Lookup Table** is valid. Any software write to this table shall be updated to the **valid** field of the **L2 Aging Table**. Any software write to this table shall be copied to the **L2 Aging Status Shadow Table - Replica**

Number of Entries :	4096
Type of Operation :	Read/Write
Addressing :	address[0:9] : hash of {GID, destination MAC} address[10:11] : bucket number
Address Space :	50007 to 54102

Field Description

Bits	Field Name	Description	Default Value
0	valid	If this is set, then the corresponding hash table entry is valid.	0x0

28.9.46 L2 Aging Status Shadow Table - Replica

This table traces the **valid** field of the **L2 Aging Table** and is used by L2 forwarding to check if a hit in the **L2 SA Hash Lookup Table** is valid. Content of this table shall be identical as the **L2 Aging Status Shadow Table**.

Number of Entries : 4096
Type of Operation : Read/Write
Addressing :

address[0:9] :	hash of {GID, source MAC}
address[10:11] :	bucket number

Address Space : 74615 to 78710

Field Description

Bits	Field Name	Description	Default Value
0	valid	If this is set, then the corresponding hash table entry is valid.	0x0

28.9.47 L2 Aging Table

This table uses the same addressing as the **L2 DA Hash Lookup Table** to show the status of each entries in that table. Any software write to any valid field in this table shall be done in the **L2 Aging Status Shadow Table**. Any software write to this table shall be copied to the **L2 Aging Status Shadow Table - Replica**

Number of Entries : 4096
Type of Operation : Read/Write
Addressing :

address[0:9] :	hash of {GID, destination MAC}
address[10:11] :	bucket number

Address Space : 274 to 4369

Field Description

Bits	Field Name	Description	Default Value
0	valid	If set, then the corresponding hash table entry is valid.	0x0
1	stat	If set, then the corresponding hash table entry will not be aged out.	0x0
2	hit	If set, then the corresponding hash table entry has a L2 DA search hit since the last aging scan.	0x0

28.9.48 L2 DA Hash Lookup Table

The L2 table is used for hash search based on the destination MAC address and a GID from the **VLAN Table**. When performing a L2 destination port lookup, {GID, destination MAC} is used as key for a hash calculation (see Section MAC Table Hashing). The hash is then used as index into this table to read out the 4 buckets. The incoming {GID, destination MAC} are compared to all the buckets. If any of the buckets match then address was known. The result of the lookup will be read from the **L2 Destination Table** at the same address as the matching hash index and bucket. Any software write to this table shall be copied to the **L2 SA Hash Lookup Table**.

Number of Entries :	4096
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	address[0:9] : hash of {GID, destination MAC} address[10:11] : bucket number
Address Space :	54103 to 62294

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddr	MAC address.	0x0
59:48	gid	Global identifier from the VLAN Table.	0x0

28.9.49 L2 Destination Table

This table contains either a destination port or a pointer to the L2 multicast table.Any software write to this table shall be copied to the **L2 Destination Table - Replica**.

Number of Entries :	4112
Type of Operation :	Read/Write
Addressing :	address 0 to 4095**L2 DA Hash Lookup Table** address : address 4096 to 4111 :**L2 Lookup Collision Table** address
Address Space :	62295 to 66406

Field Description

Bits	Field Name	Description	Default Value
0	uc	Unicast if set; multicast if cleared. Multicast means that a lookup to the **L2 Multicast Table** will occur and determine a list of destination ports.	0x0
6:1	destPort_or_mcAddr	Destination port number or pointer into the **L2 Multicast Table**.	0x0
7	pktDrop	If set, the packet will be dropped and the **L2 Lookup Drop** incremented.	0x0

28.9.50 L2 Destination Table - Replica

This table is replicated from the **L2 Destination Table** and used by the learning engine allowing the learning engine and packet forwarding to process in parallel. Content of this table shall be identical as the **L2 Destination Table**.

Number of Entries :	4112
Type of Operation :	Read/Write
Addressing :	address 0 to 4095**L2 SA Hash Lookup Table** address : address 4096 to 4111 :**L2 Lookup Collision Table** address
Address Space :	78711 to 82822

Field Description

Bits	Field Name	Description	Default Value
0	uc	Unicast if set; multicast if cleared. Multicast means that a lookup to the **L2 Multicast Table** will occur and determine a list of destination ports.	0x0
6:1	destPort_or_mcAddr	Destination port number or pointer into the **L2 Multicast Table**.	0x0
7	pktDrop	If set, the packet will be dropped and the **L2 Lookup Drop** incremented.	0x0

28.9.51 L2 Lookup Collision Table

Collision table for the **L2 DA Hash Lookup Table**. If there is a hash collision and all the buckets for that hash index are occupied then additional entries can be stored in the collision table. When searching this table, all entries are compared in parallel and the matching entry with the lowest address will be used as a match result. Chapter Learning and Aging describes how to search and write to this table.

Number of Entries :	16
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	83906 to 83937

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddr	MAC address	0x0
59:48	gid	Global identifier for learning	0x0

28.9.52 L2 Lookup Collision Table Masks

Masks for collision memory for the MAC address and the global identifier. Only the first 4entries has masks on them.

Number of Entries :	4
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	83898 to 83905

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddr	MAC address mask	$2^{48}-1$
59:48	gid	Global identifier for learning mask	0xfff

28.9.53 L2 Multicast Handling

Exceptions for L2 multicast flag handling, only valid for the Multicast Broadcast Storm Control and the Ingress Egress Port Packet Type Filter. The switch sets by default a L2 multicast flag when DA is an Ethernet multicast address (i.e. DA with the least-significant bit of the first octet equals 1 (e.g. 01:80:c2:00:00:00) but not equal to ff:ff:ff:ff:ff:ff).

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	82832

Field Description

Bits	Field Name	Description	Default Value
0	exclIPv4Mc	If set, IPv4 packets with IPv4 multicast MAC address will NOT have a L2 multicast flag.	0x0
1	exclIPv6Mc	If set, IPv6 packets with IPv6 multicast MAC address will NOT have a L2 multicast flag.	0x0
2	inclL2McLut	If set, packets that are forwarded by **L2 Multicast Table** will internally be treated as the L2 multicast bit in the L2 DA address would have been set to one.	0x1
3	inclMultiPorts	If set, packets that end up in more than one destination port but not due to broadcast or flooding will have a L2 multicast flag. Observe that mirroring is not a valid multiport destination.	0x0
4	unknownL2McFilterRule	Select the filtering rules for unknown L2 multicast MAC DA in the **Ingress Egress Port Packet Type Filter**. 0 = **dropL2FloodingFrames** 1 = **dropL2MulticastFrames**	0x0

28.9.54 L2 Multicast Table

L2 multicast table.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	mcAddr field from **L2 Destination Table** or from **Next Hop Table**
Address Space :	83176 to 83239

Field Description

Bits	Field Name	Description	Default Value
11:0	mcPortMask	L2 portmask entry members. If set, the port is part of multicast group and shall be transmitted to.	0xfff

28.9.55 L2 Reserved Multicast Address Action

If the higher bits of the incoming packets MAC DA address matches the **L2 Reserved Multicast Address Base** then the lower bits are used as index into this table. The action can be to drop the packet, send the packet to the CPU or just process the packet in the normal L2 pipeline.

Number of Entries : 256
Type of Operation : Read/Write
Addressing : MAC DA[7:0]
Address Space : 4487 to 4742

Field Description

Bits	Field Name	Description	Default Value
11:0	dropMask	Determines which source ports that are not allowed to receive this multicast address. Each bit set to 1 will result in dropping this multicast address on that source port. Bit 0 is port 0, bit 1 is port 1 etc. Each drop will be counted in **L2 Reserved Multicast Address Drop**.	0x0
23:12	sendToCpuMask	Received packets on these source ports will be sent to the CPU. Bit 0 represents port 0, bit 1 represents port 1 etc. LLDP frames sent to the CPU takes priority over this.	0x0

28.9.56 L2 Reserved Multicast Address Base

Certain L2 Destination MAC addresses shall be treated special when entering the switch. If the first 40 bits of the Destination MAC address matches the macBase field then the lowest 8 bits are used as index into the **L2 Reserved Multicast Address Action** table.

Number of Entries : 1
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Address Space : 83894

Field Description

Bits	Field Name	Description	Default Value
39:0	macBase	The first 40 bits of the reserved MAC address, and the lower 16 bits of it can be masked. The default is 01:80:c2:00:00	0x180c20000
55:40	mask	Bit comparison mask for the lower 2 bytes in macBase (marked with XX as in 01:80:c2:XX:XX). If a bit is set in the mask then the corresponding bit will be compared. Otherwise the bits are dont care.	0xffff

28.9.57 L2 SA Hash Lookup Table

L2 table used for hash search based on the source MAC and a GID from the **VLAN Table**. When performing a SA MAC learning check {GID, Source MAC} is used as key for a hash function (see Section MAC Table Hashing) which calculates a hash index. The hash index points to this table that has 4 buckets. The incoming {GID, source MAC} are compared to all the 4 buckets. If any of the buckets match then address was known. The result of the lookup will be read from the **L2 Destination Table - Replica** at the same address as the matching hash index and bucket. Content of this table shall be identical as the **L2 DA Hash Lookup Table**.

Number of Entries : 4096
Number of Addresses per Entry : 2
Type of Operation : Read/Write
Addressing :

address[0:9] :	hash of {GID, source MAC}
address[10:11] :	bucket number

Address Space : 66423 to 74614

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddr	MAC address.	0x0
59:48	gid	Global identifier from the VLAN Table.	0x0

28.9.58 L3 LPM Result

This is the longest prefix routing table result. The index into the table is the hit index from the **L3 Routing TCAM**.

Number of Entries : 16
Type of Operation : Read/Write
Addressing : **L3 Routing TCAM** hit index
Address Space : 13095 to 13110

Field Description

Bits	Field Name	Description	Default Value
0	useECMP	Enables the use of ECMP hash to calculate the next hop pointer. 0 = Use ECMP hash. 1 = Do not use ECMP hash.	0x0
8:1	ecmpMask	How many bits of the ECMP hash will be used when calculating the ECMP offset. This byte is AND:ed with the ECMP hash to determine which bits shall be used as offset.	0x0
11:9	ecmpShift	How many bits the masked ECMP hash will be right shifted.	0x0
21:12	nextHopPointer	Index into the **Next Hop Table** for this destination.	0x0

28.9.59 L3 Routing Default

The default router to be used if the destination lookup in L3 tables fails, i.e does not match either the LPM or the hash tables.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : **vrf**
Address Space : 83304 to 83307

Field Description

Bits	Field Name	Description	Default Value
9:0	nextHop	The default next hop to be used. Index into the **Next Hop Table**.	0x0
10	pktDrop	If set the packet will be drop and the **L3 Lookup Drop** counter incremented.	0x0
11	sendToCpu	If set then the packet will be sent to the CPU.	0x0

28.9.60 L3 Routing TCAM

This is the longest prefix match routing table used to determine the next hop. This table is compared from the highest address and downwards. The match which has the highest entry number is selected. The selected entry number is used to index the**L3 LPM Result** table to provide the next hop result. For each entry the mask determines which bits that shall be compared. An entry contains three parts: valid flag, comparison fields and field masks. Each comparison field is associated with a mask to optionally ignore several bits or even the entire field during comparison. To allow any value on a certain bit, the corresponding bit in the mask shall be set to 1. As a consequence, the field will have that bit nailed to 0 if read and ignored during lookup. Hit in multiple entries will return the first hit index (lowest address/index) to lookup in the result table.

Number of Entries : 16
Number of Addresses per Entry : 16
Type of Operation : Read/Write
Addressing : Entry number
Address Space : 84475 to 84730

Field Description

Bits	Field Name	Description	Default Value
1:0	proto	Select if this is an IPv4, IPv6 or MPLS entry. 0 = Reserved 1 = MPLS Entry. 2 = IPv4 entry. 3 = IPv6 entry. protoMaskN determines the bits in the field that can be ignored for comparison.	0x0
3:2	vrf	This entries VRF. The packets assigned VRF will be compared with this field. vrfMaskN determines the bits in the field that can be ignored for comparison.	0x0
131:4	destIPAddr	The IP or MPLS address to be matched. If the entry is an IPv4 entry then bits [31:0] should be set to the IPv4 address. If the entry is an MPLS entry then bits [4-1:0] should contain the source port while bits [4+19:4] should contain the MPLS label. destIPAddrMaskN determines the bits in the field that can be ignored for comparison.	0x0
133:132	protoMaskN	Mask for the proto field. For each bit in the mask, 0 means the bit is valid for comparison, 1 means the comparison ignores this bit.	0x0
135:134	vrfMaskN	Mask for the vrf field. For each bit in the mask, 0 means the bit is valid for comparison, 1 means the comparison ignores this bit.	0x0
263:136	destIPAddrMaskN	Mask for the destIPAddr field. For each bit in the mask, 0 means the bit is valid for comparison, 1 means the comparison ignores this bit.	0x0
264	valid	If set, this entry is valid	0x0

28.9.61 LLDP Configuration

A LLDP packet is identified as a LLDP frame if the packets MAC DA matches one of the mac1-mac3 fields and the packets EtherType matches eth. The portmask field determines if an identified LLDP packet will bypass the normal packet processing and instead be sent to the CPU or if the packet should pass through normal packet processing.

Number of Entries :	1
Number of Addresses per Entry :	8
Type of Operation :	Read/Write
Address Space :	83630

Field Description

Bits	Field Name	Description	Default Value
47:0	mac1	DA MAC address to match for LLDP packet.	0x180c200000e
95:48	mac2	DA MAC address to match for LLDP packet.	0x180c2000003
143:96	mac3	DA MAC address to match for LLDP packet.	0x180c2000000
159:144	eth	The Ethernet Type for a LLDP	0x88cc

Bits	Field Name	Description	Default Value
160	bpduOption	If both LLDP and BPDU are valid, because the BPDU has same MAC address as LLDP, then this option allows the BPDU identification to be turned off 0 = Don't do anything. Both LLDP and BPDU can be valid at same time. 1 = Remove BPDU valid causing that the packet will only be seen as a LLDP packet and not a BPDU frame and the new frame will not be sent to the CPU because the switch will no longer consider it a BPDU frame, this includes Rapid Spanning Tree BPDUs also.	0x0
172:161	portmask	One bit per source port, bit 0 for port 0, bit 1 for port 1 etc. 0 = Do not sent a matched LLDP packet to the CPU from this port. Packet will pass through normal packet processing. 1 = Send a matched LLDP packet to CPU from this source port and hence bypassing normal processing.	0x7ff

28.9.62 Learning And Aging Enable

Enable/Disable the learning and aging function. If software needs to take fully control over learning and aging tables by writting to the FIB directly, the learning and aging units should be completely turned off, which means all fields in this register have to be cleared to 0, partly reset is not allowed.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 231

Field Description

Bits	Field Name	Description	Default Value
0	learningEnable	If set, the learning unit will be activated.	0x1
1	agingEnable	If set, the aging unit will be activated.	0x1
2	daHitEnable	If set, MAC DA hit in the forwarding information base will update the hit bit for non-static entries.	0x1
3	lru	If set, the learning unit will try to overwrite a least recently used non-static entry in either the hash table or the collision table when there is no free entry to use. Otherwise the learning unit will try to overwrite a non-static entry in the collision table.	0x0

28.9.63 Learning Conflict

Status register for the failed port move operation. A valid status means the L2 Forwarding Information Base cannot bind the existing GID, MAC to a new port. Once the status register is updated from the hardware, no more fails can be updated untill the software clears the valid field.

Number of Entries :	1
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Address Space :	223

Field Description

Bits	Field Name	Description	Default Value
0	valid	Indicates hardware has written a learning conflict to this status register. Write 0 to clear.	0x0
48:1	macAddr	MAC address.	0x0
60:49	gid	Global identifier from the VLAN Table.	0x0
64:61	port	Port number.	0x0

28.9.64 Learning Overflow

Status register for the failed hardware learning operation. A valid status means the L2 Forwarding Information Base cannot find an available slot for the unknown GID, MAC. Once the status register is updated from the hardware, no more fails can be updated untill the software clears the valid field.

Number of Entries :	1
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Address Space :	227

Field Description

Bits	Field Name	Description	Default Value
0	valid	Indicates hardware has written a learning overflow to this status register, Write 0 to clear.	0x0
48:1	macAddr	MAC address.	0x0
60:49	gid	Global identifier from the VLAN Table.	0x0
64:61	port	Port number.	0x0

28.9.65 Link Aggregate Weight

The link aggregate hash will index into this table to determine which physical port within the aggregate that a packet should be output to. The number of bits set for a port will determine the ratio of packets that will go out on that port. For each hash index only one of the ports that belong to the same link aggregate must be set. The number of bits set divided by number of hash values determines the ratio of traffic going to that port. All link aggregates share this table since each physical port can only belong to one link aggregate. When a link aggregate only has one port then all bits for that port must be set.

Number of Entries :	256
Type of Operation :	Read/Write
Addressing :	The link aggregate hash.
Address Space :	82860 to 83115

Field Description

Bits	Field Name	Description	Default Value
11:0	ports	One bit per physical port.	0x0

28.9.66 Link Aggregation Ctrl

This register controls whether link aggregation is enabled and which packet header fields that will be used to calculate the link aggregate hash value.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82823

Field Description

Bits	Field Name	Description	Default Value
0	enable	Is Link aggregation enabled or not. 0 = Link Aggregation is disabled 1 = Link Aggregation is enabled	0x0
1	useSaMacInHash	The packets source MAC address shall be part of the hash key when calculating the link aggregate hash value	0x0
2	useDaMacInHash	The packets destination MAC addresses shall be part of the hash key when calculating the link aggregate hash value	0x0
3	useIpInHash	The packets IP source and destination addresses shall be part of the hash key when calculating the link aggregate hash value	0x0
4	useL4InHash	The packets L4 SP / DP and L4 protocol byte shall be part of the hash key when calculating the link aggregate hash value	0x0
5	useTosInHash	The incoming packets TOS byte shall be part of the hash key when calculating the link aggregate hash value	0x0
6	useNextHopInHash	For routed packets the next hop entry shall be part of the hash key when calculating the link aggregate hash value.	0x0
7	useVlanIdInHash	The packets VLAN Identifier tag shall be part of the hash key when calculating the link aggregate hash value.	0x0

28.9.67 Link Aggregation Membership

This register is used to determine which link aggregation a specific source port is membership of. If link aggregation is enabled then this port number is used for all source lookups instead of the port where the packet enterned the switch.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Ingress port
Address Space : 83326 to 83337

Field Description

Bits	Field Name	Description	Default Value
3:0	la	The Link aggregation which this port is a member of	0x0

28.9.68 Link Aggregation To Physical Ports Members

This link aggregate portmasks are setup to determine which physical ports are members of each link aggregate.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : The link aggregate number.
Address Space : 82848 to 82859

Field Description

Bits	Field Name	Description	Default Value
11:0	members	Physical ports that are members of this link aggregate. One bit per port.	0x0

28.9.69 MPLS EXP Field To Egress Queue Mapping Table

Mapping table from MPLS EXP priority fields to egress queues.

Number of Entries : 8
Type of Operation : Read/Write
Addressing : Incoming packets MPLS EXP priority bits
Address Space : 83256 to 83263

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueue	Egress queue	0x1

28.9.70 MPLS EXP Field To Packet Color Mapping Table

Mapping table from MPLS EXP priority fields to packet initial color.

Number of Entries : 8
Type of Operation : Read/Write
Addressing : Incoming packets MPLS EXP priority bits
Address Space : 49991 to 49998

Field Description

Bits	Field Name	Description	Default Value
1:0	color	Packet initial color	0x0

28.9.71 Next Hop Packet Modifications

Determines the VLAN operations to perform on the packet exiting the router. One or two VLAN headers can be added to the outgoing packet.

Number of Entries :	1024
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	**nextHopPacketMod**
Address Space :	46903 to 48950

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this a valid entry. If the router points to an entry with this field cleared the packet will be sent to CPU. 0 = Invalid 1 = Valid	0x0
1	outerVlanAppend	Insert/push an outer VLAN header in the packet. The information used to create the new VLAN header is controlled by the fields **outerVid**, **outerPcpSel**, **outerCfiDeiSel** and **outerEthType**. If the selected outermost VLAN header field doesn't exist in the packet then the new VLAN header field will be taken from **Router Egress Queue To VLAN Data**. 0 = No operation. 1 = Insert/push an outer VLAN tag.	0x0
3:2	outerPcpSel	Selects which PCP bits to use when building an outer VLAN header. 0 = From outermost VLAN header in the original packet (if any). 1 = From this entrie's **outerPcp** field. 2 = From **Router Egress Queue To VLAN Data**.	0x0
5:4	outerCfiDeiSel	Selects which CFI/DEI bit to use when building an outer VLAN header. 0 = From outermost VLAN header in the original packet (if any). 1 = From this entrie's **outerCfiDei** field. 2 = From **Router Egress Queue To VLAN Data**.	0x0
7:6	outerEthType	Pointer to the VLAN type. 0 = C-VLAN - 0x8100. 1 = S-VLAN - 0x88A8. 2 = User defined VLAN.	0x0
19:8	outerVid	The VID used when building an outer VLAN header.	0x0
22:20	outerPcp	The PCP bits to use when building an outer VLAN header. If selected by **outerPcpSel**.	0x0

Bits	Field Name	Description	Default Value
23	outerCfiDei	The CFI/DEI bit to use when building an outer VLAN header. If selected by **outerCfiDeiSel**.	0x0
24	innerVlanAppend	Insert/push an inner VLAN header in the packet. The information used to create the new VLAN header is controlled by the fields **innerVid**, **innerPcpSel**, **innerCfiDeiSel** and **innerEthType**. If the selected innermost VLAN header field doesn't exist in the packet then the new VLAN header field will be taken from **Router Egress Queue To VLAN Data**. 0 = No operation 1 = Insert/push an inner VLAN tag.	0x0
26:25	innerPcpSel	Selects which PCP bits to use when building an inner VLAN header. 0 = From innermost VLAN header in the original packet (if any). 1 = From this entrie's **innerPcp** field. 2 = From **Router Egress Queue To VLAN Data**.	0x0
28:27	innerCfiDeiSel	Selects which CFI/DEI bit to use when building an inner VLAN header. 0 = From innermost VLAN header in the original packet (if any). 1 = From this entrie's **innerCfiDei** field. 2 = From **Router Egress Queue To VLAN Data**.	0x0
30:29	innerEthType	Pointer to the VLAN type. 0 = C-VLAN - 0x8100. 1 = S-VLAN - 0x88A8. 2 = User defined VLAN.	0x0
42:31	innerVid	The VID used when building an inner VLAN header.	0x0
45:43	innerPcp	The PCP bits to use when building an inner VLAN header. If selected by **innerPcpSel**.	0x0
46	innerCfiDei	The CFI/DEI bit to use when building an inner VLAN header. If selected by **innerCfiDeiSel**.	0x0
50:47	msptPtr	The multiple spanning tree to be used by packets for egress spanning tree check for this next hop. Points to an entry in **Egress Multiple Spanning Tree State**	0x0

28.9.72 Next Hop Table

Forwarding decision for a routed packet including destination port(s), or if packet shall be dropped or sent to the CPU port.

Number of Entries : 1024
Type of Operation : Read/Write
Addressing : Next Hop Pointer
Address Space : 45879 to 46902

Field Description

Bits	Field Name	Description	Default Value
0	validEntry	Is this a valid entry or not. If the entry is not valid then the packet shall be sent to the CPU for further processsing	0x0

Bits	Field Name	Description	Default Value
10:1	nextHopPacketMod	Pointer into the **Next Hop Packet Modifications** table and the **Next Hop DA MAC** table.	0x0
11	l2Uc	L2 unicast or multicast. A multicast means that a lookup in the **L2 Multicast Table** will take place to determine the destination portmask. 0 = L2 multicast. 1 = L2 unicast.	0x0
17:12	destPort_or_mcAddr	Destination port number or a pointer into the **L2 Multicast Table**	0x0
18	pktDrop	If set then the packet will be dropped and the **L3 Lookup Drop** incremented.	0x0
19	sendToCpu	If set then the packet will be sent to the CPU.	0x0

28.9.73 Port Move Options

Determine if port move is allowed on static entries.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 82831

Field Description

Bits	Field Name	Description	Default Value
11:0	allowPortMoveOnStatic	This field configures which source ports that are allowed to change their static GID and MAC to other ports. One bit for each port where bit 0 corresponds to port 0. When the L2 forwarding information base identifies a GID, MAC SA and source port combination that conflicts with a existing static entry, if the previous binded port has a coressponding bit set to 1 in this field, it allows the learning engine to update the GID and MAC to the current source port.	0xfff

28.9.74 Reserved Destination MAC Address Range

The mac addresses ranges that the packets destination MAC address are compared with and the corresponding actions. A range is matched if the packets MAC address is $\geq$ *startAddr* and the address is $\leq$ *stopAddr*. The table is searched starting from entry 0. When a range is matched the corresponding actions (drop, send to cpu, force egress queue) will be activated. If multiple ranges are matched, any matching range that sets drop will cause a drop. Any match that sets sendToCpu will cause send to CPU (this has priority over drop). When multiple ranges that match has set the forceQueue field then the highest numbered entry will determine the value.

Number of Entries : 4
Number of Addresses per Entry : 4
Type of Operation : Read/Write
Addressing : All entries are read out in parallel
Address Space : 83610 to 83625

Field Description

Bits	Field Name	Description	Default Value
47:0	startAddr	The start MAC address of the range. A packets destination MAC address must be equal or greater than this value to match the range.	0x0
95:48	stopAddr	The end MAC address of the range. A packets destination MAC address must be equal or less than this value to match the range.	0x0
96	dropEnable	If the MAC address was within the range the packet shall be dropped and the **Reserved MAC DA Drop** counter incremented.	0x0
97	sendToCpu	If the MAC address was within the range the packet shall be sent to the CPU.	0x0
98	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 17.1	0x0
101:99	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
103:102	color	Inital color of the packet.	0x0
104	forceColor	If set, the packet shall have a forced color.	0x0
105	mmpValid	If set, this entry contains a valid MMP pointer	0x0
109:106	mmpPtr	Ingress MMP pointer.	0x0
111:110	mmpOrder	Ingress MMP pointer order.	0x0
123:112	enable	Enable the reserved MAC DA check per source port. One bit for each port where bit 0 corresponds to port 0. If a bit is set to one, the reserved MAC DA range is activated for that source port.	0x0

28.9.75 Reserved Source MAC Address Range

The mac addresses ranges that the packets source MAC address are compared with and the corresponding actions. A range is matched if the packets MAC address is $\geq$ *startAddr* and the address is $\leq$ *stopAddr*. The table is searched starting from entry 0. When a range is matched the corresponding actions (drop, send to cpu, force egress queue) will be activated. If multiple ranges are matched, any matching range that sets drop will cause a drop. Any match that sets sendToCpu will cause send to CPU (this has priority over drop). When multiple ranges that match has set the forceQueue then the highest numbered entry will determine the value.

Number of Entries :	4
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	83594 to 83609

Field Description

Bits	Field Name	Description	Default Value
47:0	startAddr	The start MAC address of the range. A packets source MAC address must be equal or greater than this value to match the range.	0x0

Bits	Field Name	Description	Default Value
95:48	stopAddr	The end MAC address of the range. A packets source MAC address must be equal or less than this value to match the range.	0x0
96	dropEnable	If the MAC address was within the range the packet shall be dropped and the **Reserved MAC SA Drop** counter incremented.	0x0
97	sendToCpu	If the MAC address was within the range the packet shall be sent to the CPU.	0x0
98	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 17.1	0x0
101:99	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
103:102	color	Inital color of the packet.	0x0
104	forceColor	If set, the packet shall have a forced color.	0x0
105	mmpValid	If set, this entry contains a valid MMP pointer	0x0
109:106	mmpPtr	Ingress MMP pointer.	0x0
111:110	mmpOrder	Ingress MMP pointer order.	0x0
123:112	enable	Enable the reserved source MAC check per source port. One bit for each port where bit 0 corresponds to port 0. If a bit is set to one, the reserved source MAC range is activated for that source port.	0x0

28.9.76 Router Egress Queue To VLAN Data

Map from egress queue number to VLAN PCP and CFI/DEI values to be used in router VLAN operations selected by **Next Hop Packet Modifications**.

Number of Entries : 8
Type of Operation : Read/Write
Addressing : Egress Queue
Address Space : 49999 to 50006

Field Description

Bits	Field Name	Description	Default Value
0	cfiDei	Map from egress queue to CFI/DEI	0x0
3:1	pcp	Map from egress queue to PCP	0x0

28.9.77 Router MTU Table

An MTU check is done on each routed packet by comparing the IPv4 Total Length field with the **maxIPv4MTU** limit. Correspondingly IPv6 Payload Length field is compared with **maxIPv6MTU**. If the length field exceeds the limit the packet will be sent to the CPU. Each router VRF has a MTU limit for each port.

Number of Entries : 48
Type of Operation : Read/Write
Addressing : destination-port * 4 + VRF
Address Space : 83128 to 83175

Field Description

Bits	Field Name	Description	Default Value
15:0	maxIPv4MTU	The maximum MTU allowed for IPv4 packets	0xffff
31:16	maxIPv6MTU	The maximum MTU allowed for IPv6 packets	0xffff

28.9.78 Router Port MAC Address

The incoming packets destination MAC address is compared against all the entries in the table. If there is a match after the macMask has been applied the packet will enter the routing function with the VRF identifier assigned from the matching entry. The table must be configured so that only one match is possible.

Number of Entries :	16
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	83386 to 83449

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddress	The base destination MAC address that is used to identify packets to the router.	0x0
95:48	macMask	Each bit says if the bit in the DA MAC shall be compared. 0 = Dont compare bit. 1 = Compare bit.	0x0
96	valid	If set, this entry is valid for comparison.	0x0
98:97	vrf	The VRF to use for this router	0x0

28.9.79 SMON Set Search

If both source port and VLAN ID match one of the entries, the corresponding SMON counter will be updated.

Number of Entries :	2
Type of Operation :	Read/Write
Addressing :	SMON set number
Address Space :	83324 to 83325

Field Description

Bits	Field Name	Description	Default Value
3:0	srcPort	Source port	0x0
15:4	vid	VLAN ID	0x0

28.9.80 Send to CPU

Configuration of MAC addresses used to redirect packets to CPU.

Number of Entries :	1
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Address Space :	83626

Field Description

Bits	Field Name	Description	Default Value
11:0	allowBpdu	Send to CPU portmask, bit 0 port 0, bit 1 port 1 etc. If source port bit is set then packets that have the destination MAC address equal to 01:80:C2:00:00:00 are sent to the CPU port.	0xfff
23:12	allowRstBpdu	Send to CPU portmask, bit 0 port 0, bit 1 port 1 etc. If the source port bit is set then packets that have the destination MAC address equal to 01:00:0C:CC:CC:CD are sent to the CPU port.	0xfff
35:24	uniqueCpuMac	If set then unicast packets can not be switched or routed to the CPU port. Other mechanism for sending to the CPU port are not affected (e.g. ACL's). This also enables detection of a specific MAC address, **cpuMacAddr**, that will be sent to the CPU.	0x0
83:36	cpuMacAddr	Packets with this destination MAC address will be sent to the CPU. Only valid if **uniqueCpuMac** on the source port is set.	0x0

28.9.81 Source Port Table

This table configures various functions that are dependent on which port the packet enters the switch. A VLAN operation (e.g. push, pop, swap) to be performed can be selected by the **vlanSingleOp** field in **Source Port Table**. For the push and swap operations the information used to create the new VLAN header is controlled by the fields **vidSel**, **cfiDeiSel**, **pcpSel** and **typeSel**. Other configurations are VLAN LUT index, input mirroring, spanning tree state, Ingress VID offset, special VID treatment, multicast learning, min/max number of VLANs and L3 priority selection.

Number of Entries :	12
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	83338 to 83385

Field Description

Bits	Field Name	Description	Default Value
0	learningEn	If hardware learning is turned on and this is set to one, the unknown source MAC address from this port will be learned.	0x1

Bits	Field Name	Description	Default Value
1	dropUnknownDa	If set to one packets with unknown destination MAC address from this port will be dropped.	0x0
2	prioFromL3	If the packet is IP/MPLS and this is set the egress queue will be selected from Layer 3 decoding described in Determine Egress Queue.	0x0
3	colorFromL3	If the packet is IP/MPLS and this bit is set the packet initial color will be selected from Layer 3 decoding.	0x0
6:4	vlanSingleOp	The source port VLAN operation to perform on the packet. 0 = No operation. 1 = Swap. 2 = Push. 3 = Pop. 4 = Penultimate pop(remove all VLAN headers).	0x0
8:7	vidSel	Selects which VID to use when building a new VLAN header in a source port push or swap operation. If the selected VLAN header doesn't exist in the packet then this table entry's **defaultVid** will be used. 0 = From outermost VLAN in the original packet (if any). 1 = From this table entry's **defaultVid**. 2 = From the second VLAN in the original packet (if any).	0x0
10:9	cfiDeiSel	Selects which CFI/DEI to use when building a new VLAN header in a source port push or swap operation. If the selected VLAN header doesn't exist in the packet then this table entry's **defaultCfiDei** will be used. 0 = From outermost VLAN in the original packet (if any). 1 = From this table entry's **defaultCfiDei**. 2 = From the second VLAN in the original packet (if any).	0x0
12:11	pcpSel	Selects which PCP to use when building a new VLAN header in a source port push or swap operation. If the selected VLAN header doesn't exist in the packet then this table entry's **defaultPcp** will be used. 0 = From outermost VLAN in the original packet. (if any) 1 = From this table entry's **defaultPcp**. 2 = From the second VLAN in the original packet (if any).	0x0
14:13	typeSel	Selects which TPID to use when building a new VLAN header in a source port push or swap operation. 0 = C-VLAN - 0x8100. 1 = S-VLAN - 0x88A8. 2 = User defined VLAN type from register **Egress Ethernet Type for VLAN tag**.	0x0

Bits	Field Name	Description	Default Value
16:15	vlanAssignment	Controls how a packets Ingress VID is assigned. If the selected source is from a VLAN header in the incoming packet and the packet doesn't have that header, then this table entry's **defaultVid** will be used. 0 = packet based - the Ingress VID is assigned from the incoming packets outermost VLAN header. 1 = port-based - the packets Ingress VID is assigned from this table entry's **defaultVid** 2 = mixed - if there are two VLANs in the incoming packet, the inner VLAN is chosen. If the incoming packet has only 0 or 1 VLAN, then it will select this table entry's **defaultVid**	0x0
28:17	defaultVid	The default VID. This is used in source port VLAN operations (see **vidSel**). It is used to assign Ingress VID (see **vlanAssignment**). It is used when creating an internal VLAN header for incoming packets that has no VLAN header.	0x0
29	defaultCfiDei	The default CFI / DEI bit. This is used in source port VLAN operations (see **cfiDeiSel**). It is used when creating an internal VLAN header for incoming packets that has no VLAN header.	0x0
32:30	defaultPcp	The default PCP bits. This is used in source port VLAN operations (see .**pcpSel**). It is used when creating an internal VLAN header for incoming packets that has no VLAN header.	0x0
34:33	defaultVidOrder	When a new hit is done in the result in the L2,L3,L4 VID range checks the ingress VID will only be changed if the result has a higher order value.	0x0
36:35	minAllowedVlans	The minimum number of VLAN headers a packet must have to be allowed on this port. Otherwise the packet will be dropped and the **Minimum Allowed VLAN Drop** will be incremented. 0 = All packets are accepted. 1 = 1 or more tags are accepted. 2 = 2 or more tags are accepted. 3 = No packets are accepted.	0x0
38:37	maxAllowedVlans	The maximum number of VLAN headers a packet is allowed to have to enter on this port. Otherwise the packet will be dropped and the **Maximum Allowed VLAN Drop** will be incremented. 0 = Only untagged packets are accepted. 1 = 0 to 1 tags are accepted. 2 = Any number of VLANs are accepted. 3 = Any number of VLANs are accepted.	0x2
39	ignoreVlanMembership	By default packets on non-VLAN member source port are dropped before entering the L2 lookup process. Set this field to one to ignore the VLAN membership check on the source port. However L2 lookup can never forward packets to non-VLAN member destinations.	0x0
40	learnMulticastSaMac	If set, the learning engine allows Ethernet multicast source MAC addresses to be learned.	0x0

Bits	Field Name	Description	Default Value
41	inputMirrorEnabled	If set, input mirroring is enabled on this port. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the **destInputMirror** port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0
42	imUnderVlanMembership	If set, input mirroring to a destination that not a member of the VLAN will be ignored.	0x0
43	imUnderPortIsolation	If set, input mirroring to a destination that isolated the source port in the **srcPortFilter** will be ignored.	0x0
47:44	destInputMirror	Destination physical port for input mirroring. Only valid if **inputMirrorEnabled** is set.	0x0
50:48	spt	The spanning tree state for this ingress port. The state Disabled implies that spanning tree protocol is not enabled and hence frames will be forwarded on this egress port. 0 = Disabled. 1 = Blocking. 2 = Listening. 3 = Learning. 4 = Forwarding.	0x0
51	enablePriorityTag	An outer VLAN tag with VID matching **priorityVid** will have PCP bits extracted and used to determine output queue but in remaining VLAN processing this tag will not be treated as a VLAN tag. If the packet has an inner VLAN tag this will be treated as an outer VLAN tag in the following VLAN processing. The VID will only be matched in a VLAN header located immediately after DA and SA MAC, i.e. no custom tags allowed. In egress processing the outer VLAN tag will be removed. 0 = Disable comparison. 1 = Enable comparison.	0x0
63:52	priorityVid	The VID used in the outer VLAN tag comparison, see **enablePriorityTag**.	0x0
64	disableRouting	On this source port are the packets allowed to do L3 routing. 0 = No 1 = Yes	0x0

28.9.82 Time to Age

Interval period after which FIB entries are aged out.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	244

Field Description

Bits	Field Name	Description	Default Value
31:0	tickCnt	Number of ticks (see Chapter Tick) between starts of the aging process.	$2^{32}-1$
34:32	tick	Select one of the 5 available ticks. The tick frequencies are configured globaly in the **Core Tick Configuration** register.	0x0

28.9.83 VLAN PCP And DEI To Color Mapping Table

Mapping table from VLAN PCP and DEI field to packet initial color.

Number of Entries : 16
Type of Operation : Read/Write
Addressing :

address[0:2] :	PCP
address[3] :	DEI

Address Space : 49463 to 49478

Field Description

Bits	Field Name	Description	Default Value
1:0	color	Packet initial color.	0x0

28.9.84 VLAN PCP To Queue Mapping Table

Mapping table from VLAN PCP priority bits to ingress/egress queues.

Number of Entries : 8
Type of Operation : Read/Write
Addressing : Incoming packets VLAN priority bits
Address Space : 83264 to 83271

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueue	Egress queue.	0x1

28.9.85 VLAN Table

Defines the VLAN port membership, which GID to use in L2 lookups, the MSPT to use, if routing is allowed and a VLAN operation (e.g. push, pop, swap) to be performed.
The VLAN operation is selected by the **vlanSingleOp** field. For the push and swap operations the information used to create the new VLAN header is controlled by the fields **vidSel**, **cfiDeiSel**, **pcpSel** and **typeSel**.

Number of Entries :	4096
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	The packet's Ingress VID plus offset as defined in **Source Port Table**.
Address Space :	4883 to 13074

Field Description

Bits	Field Name	Description	Default Value
11:0	vlanPortMask	VLAN membership portmask. The packets source port must be a member of the VLAN, otherwise the packet will be dropped and the **VLAN Member Drop** will be incremented. The membership mask will also limit the destination ports for L2 unicast, multicast, broadcast and flooding. If this results in an empty destination port mask then the packet is dropped and the **Empty Mask Drop** will be incremented.	0xfff
23:12	gid	The packet will be assigned a global identifier that is used during L2 lookup to allow multiple VLANs to share the same L2 tables.	0x0
24	mmpValid	If set, this entry contains a valid MMP pointer	0x0
28:25	mmpPtr	Ingress MMP pointer.	0x0
30:29	mmpOrder	Ingress MMP pointer order.	0x0
34:31	msptPtr	The multiple spanning tree to be used by packets on this VLAN. Points to entries in the **Ingress Multiple Spanning Tree State** and **Egress Multiple Spanning Tree State** tables	0x0
37:35	vlanSingleOp	The ingress VLAN operation to perform on the packet. 0 = No operation. 1 = Swap. 2 = Push. 3 = Pop. 4 = Penultimate Pop(remove all VLANS).	0x0
39:38	vidSel	Selects which VID to use when building a new VLAN header in a push or swap operation. If the selected VLAN header doesn't exist in the packet then this table entry's **vid** will be used. 0 = From the outermost VLAN in the original packet (if any). 1 = From this table entry's **vid**. 2 = From the second VLAN in the original packet (if any).	0x0
41:40	cfiDeiSel	Selects which CFI/DEI to use when building a new VLAN header in a push or swap operation. If the selected VLAN header doesn't exist in the packet then this table entry's **cfiDei** will be used. 0 = From outermost VLAN in the original packet (if any). 1 = From this table entry's **cfiDei**. 2 = From the second VLAN in the original packet (if any).	0x0

Bits	Field Name	Description	Default Value
43:42	pcpSel	Selects which PCP to use when building a new VLAN header in a push or swap operation. If the selected VLAN header doesn't exist in the packet then this table entry's **pcp** will be used. 0 = From outermost VLAN in the original packet. (if any) 1 = From this table entry's **pcp**. 2 = From the second VLAN in the original packet (if any).	0x0
55:44	vid	The VID used in VLAN push or swap operation if selected by **vidSel**.	0x0
58:56	pcp	The PCP used in VLAN push or swap operation if selected by **pcpSel**.	0x0
59	cfiDei	The CFI/DEI used in VLAN push or swap operation if selected by **cfiDeiSel**	0x0
61:60	typeSel	Selects which TPID to use when building a new VLAN header in a push or swap operation. 0 = C-VLAN - 0x8100. 1 = S-VLAN - 0x88A8. 2 = User defined VLAN type from register **Egress Ethernet Type for VLAN tag** field **typeValue**.	0x0
62	allowRouting	Allow routing. 0 = The router will not process the packet but L2 processing will be done normally. 1 = Packet will be processed by the router.	0x1
63	sendIpMcToCpu	Send all IPv4 and IPv6 multicast packets to CPU, bypassing L2 processing and L3 routing.	0x0

28.10 MBSC

28.10.1 L2 Broadcast Storm Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for L2 Broadcast Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 124 to 135

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	0x5c8

28.10.2 L2 Broadcast Storm Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for L2 Broadcast Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 136 to 147

Field Description

Bits	Field Name	Description	Default Value
15:0	threshold	Minimum number of tokens in bucket for the status to be set to accept.	0x2e4

28.10.3 L2 Broadcast Storm Control Enable

Bitmask to turn L2 Broadcast Storm Control ON/OFF (1/0) for Egress Ports

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 148

Field Description

Bits	Field Name	Description	Default Value
11:0	enable	Bitmask where the index is the Egress Ports	0x0

28.10.4 L2 Broadcast Storm Control Rate Configuration

Token Bucket rate Configuration for L2 Broadcast Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 112 to 123

Field Description

Bits	Field Name	Description	Default Value
0	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x1
12:1	tokens	The number of tokens added each tick	0x4a
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	0x2
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

28.10.5 L2 Flooding Storm Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for L2 Flooding Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 198 to 209

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	0x5c8

28.10.6 L2 Flooding Storm Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for L2 Flooding Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 210 to 221

Field Description

Bits	Field Name	Description	Default Value
15:0	threshold	Minimum number of tokens in bucket for the status to be set to accept.	0x2e4

28.10.7 L2 Flooding Storm Control Enable

Bitmask to turn L2 Flooding Storm Control ON/OFF (1/0) for Egress Ports

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 222

Field Description

Bits	Field Name	Description	Default Value
11:0	enable	Bitmask where the index is the Egress Ports	0x0

28.10.8 L2 Flooding Storm Control Rate Configuration

Token Bucket rate Configuration for L2 Flooding Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 186 to 197

Field Description

Bits	Field Name	Description	Default Value
0	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x1
12:1	tokens	The number of tokens added each tick	0x4a
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	0x2
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

28.10.9 L2 Multicast Storm Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for L2 Multicast Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 161 to 172

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	0x5c8

28.10.10 L2 Multicast Storm Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for L2 Multicast Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 173 to 184

Field Description

Bits	Field Name	Description	Default Value
15:0	threshold	Minimum number of tokens in bucket for the status to be set to accept.	0x2e4

28.10.11 L2 Multicast Storm Control Enable

Bitmask to turn L2 Multicast Storm Control ON/OFF (1/0) for Egress Ports

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 185

Field Description

Bits	Field Name	Description	Default Value
11:0	enable	Bitmask where the index is the Egress Ports	0x0

28.10.12 L2 Multicast Storm Control Rate Configuration

Token Bucket rate Configuration for L2 Multicast Storm Control

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Ports
Address Space : 149 to 160

Field Description

Bits	Field Name	Description	Default Value
0	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x1
12:1	tokens	The number of tokens added each tick	0x4a
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	0x2
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

28.11 Scheduling

28.11.1 Output Disable

Bitmask for disabling the egress queues on egress ports.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress port
Address Space : 86212 to 86223

Field Description

Bits	Field Name	Description	Default Value
0	egressQueue0Disabled	If set, stop scheduling new packets for output from queue 0 on this egress port.	0x0
1	egressQueue1Disabled	If set, stop scheduling new packets for output from queue 1 on this egress port.	0x0
2	egressQueue2Disabled	If set, stop scheduling new packets for output from queue 2 on this egress port.	0x0
3	egressQueue3Disabled	If set, stop scheduling new packets for output from queue 3 on this egress port.	0x0
4	egressQueue4Disabled	If set, stop scheduling new packets for output from queue 4 on this egress port.	0x0
5	egressQueue5Disabled	If set, stop scheduling new packets for output from queue 5 on this egress port.	0x0
6	egressQueue6Disabled	If set, stop scheduling new packets for output from queue 6 on this egress port.	0x0
7	egressQueue7Disabled	If set, stop scheduling new packets for output from queue 7 on this egress port.	0x0

28.12 Shared Buffer Memory

28.12.1 Buffer Free

The number of cells available in the buffer memory for incoming packets.

Number of Entries : 1
Type of Operation : Read Only
Address Space : 1

Field Description

Bits	Field Name	Description	Default Value
12:0	cells	Number of free cells.	0x1000

28.12.2 Egress Port Depth

Number of packets available in the buffer memory for each egress port.

Number of Entries : 12
Type of Operation : Read Only
Addressing : Egress Port
Address Space : 86103 to 86114

Field Description

Bits	Field Name	Description	Default Value
12:0	packets	Number of packet currently queued.	0x0

28.12.3 Egress Queue Depth

Number of packets available in the buffer memory for each egress queue.

Number of Entries : 96
Type of Operation : Read Only
Addressing : Global queue number
Address Space : 86115 to 86210

Field Description

Bits	Field Name	Description	Default Value
12:0	packets	Number of packets currently queued.	0x0

28.12.4 Minimum Buffer Free

Minimum number of cells available in the buffer memory

Number of Entries : 1
Type of Operation : Read Only
Address Space : 86211

Field Description

Bits	Field Name	Description	Default Value
12:0	cells	Number of cells.	0x1000

28.12.5 Packet Buffer Status

Queue status of the packet buffer

Number of Entries : 1
Type of Operation : Read Only
Address Space : 86100

Field Description

Bits	Field Name	Description	Default Value
11:0	empty	Empty flags for the egress ports	0xfff

28.13 Statistics: ACL

28.13.1 Ingress L2 ACL Match Counter

Number of packets hit in entries of **Ingress L2 ACL Match Data Entries**.
In Figure 23.1, **ippAcl** with process sequence **11** represents the internal location of this counter.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	See **Ingress L2 ACL Match Data Entries** for how ACL rules are mapped to counters.
Address Space :	84763 to 84794

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.13.2 Ingress L3 ACL Match Counter

Number of packets hit in entries of **Ingress L3/L4 ACL Match Data Entries**.
In Figure 23.1, **ippAcl** with process sequence **11** represents the internal location of this counter.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	**Ingress L3/L4 ACL Match Data Entries** entry number
Address Space :	85823 to 85854

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.14 Statistics: Debug

28.14.1 EPP PM Drop

Number of drops due to FIFO overflows in EPP PM.
In Figure 23.1, **epmOverflow** with process sequence **22** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	86307

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.14.2 IPP PM Drop

Number of drops due to FIFO overflows in IPP PM.
In Figure 23.1, **ipmOverflow** with process sequence **12** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4464

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.14.3 PS Error Counter

Number of errors occured in the PS-converter.
In Figure 23.1, **psError** with process sequence **25** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress port
Address Space : 95250 to 95261

Field Description

Bits	Field Name	Description	Default Value
7:0	underrun	Number of packets which have empty cycles caused by the internal PS-converter but not the external halt during packet transmissions.	0x0
15:8	overflow	Number of FIFO overflows in the PS-converter. This error will cause packet corruptions.	0x0

28.14.4 SP Overflow Drop

Number of packets dropped due to: FIFO overflow in the SP-converter.
In Figure 23.1, **spOverflow** with process sequence **5** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read Only
Addressing : Ingress port
Address Space : 4416 to 4427

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets on this ingress port.	0x0

28.15 Statistics: EPP Egress Port Drop

28.15.1 Egress Port Disabled Drop

Number of packets dropped due to egress port disabled.
In Figure 23.1, **epppDrop** with process sequence **19** represents the internal location of this counter.

Number of Entries :	12
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	86283 to 86294

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.15.2 Egress Port Filtering Drop

Number of packets dropped due to egress port filtering.
In Figure 23.1, **epppDrop** with process sequence **19** represents the internal location of this counter.

Number of Entries :	12
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	86295 to 86306

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.15.3 Unknown Egress Drop

Number of packets dropped during egress packet processing due to unknown reasons. Internal error caused by packet drop with an invalid Drop ID.
In Figure 23.1, **epppDrop** with process sequence **19** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress port
Address Space : 86271 to 86282

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.16 Statistics: IPP Egress Port Drop

28.16.1 Egress Spanning Tree Drop

Number of packets dropped due to egress spanning tree check configured in **Egress Spanning Tree State** and **Egress Multiple Spanning Tree State**
In Figure 23.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Port (not aggregated)
Address Space : 85867 to 85878

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.16.2 Ingress-Egress Packet Filtering Drop

Number of packets dropped due to ingress-egress packet filtering configured in **Ingress Egress Port Packet Type Filter**.
In Figure 23.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Port (not aggregated)
Address Space : 85891 to 85902

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.16.3 MBSC Drop

Number of packets dropped due to MBSC. When the egress port exceeds the multicast/broadcast traffic limits any multicast/broadcast packets will be dropped.
In Figure 23.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Port (not aggregated)
Address Space : 85879 to 85890

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.16.4 Queue Off Drop

Number of packets dropped due to the queue being turned off.
In Figure 23.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Port (not aggregated)
Address Space : 85855 to 85866

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17 Statistics: IPP Ingress Port Drop

28.17.1 Empty Mask Drop

Number of packets dropped due to an empty destination port mask.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4467

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.2 Expired TTL Drop

Number of packets dropped due to expired TTL.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4480

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.3 IP Checksum Drop

Number of packets dropped due to incorrect IP checksum.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4482

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.4 Ingress L2 ACL Drop

Number of packets dropped due to the L2 ingress ACL.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4473

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.5 Ingress Packet Filtering Drop

Number of packets dropped due to ingress port packet type filtering as configured in **Ingress Port Packet Type Filter**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4472

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.6 Ingress Spanning Tree Drop: Blocking

Number of packets dropped due to that a ports's ingress spanning tree protocol state was **Blocking** or that port and packet VLAN's ingress multiple spanning tree instance state was **Discarding**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4470

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.7 Ingress Spanning Tree Drop: Learning

Number of packets dropped due to that a port's ingress spanning tree protocol state was **Learning** or that port and packet VLAN's ingress multiple spanning tree instance state was **Learning**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4469

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.8 Ingress Spanning Tree Drop: Listen

Number of packets dropped due to that a port's ingress spanning tree protocol state was **Listening**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4468

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.9 Invalid Routing Protocol Drop

Number of packets dropped due to invalid routing protocol. This occurs when a packet enters the router port but the protocol type is not allowed to be routed as configured in **Ingress Router Table**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4479

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.10 L2 Lookup Drop

Number of packets dropped in the L2 destination port lookup process. Either due to a drop flag in an **L2 Destination Table** entry, or due to destination port not being member of the VLAN .
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4471

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.11 L2 Reserved Multicast Address Drop

Number of packets dropped due to the L2 Reserved Multicast Addresses on counter 0
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4484

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.12 L3 ACL Drop

Number of packets dropped due to matching an L3 ACL entry with **dropEnable** operation set.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4483

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.13 L3 Lookup Drop

Number of packets dropped due to a drop flag in **L3 Routing Default** or **Next Hop Table**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4481

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.14 Maximum Allowed VLAN Drop

Number of packets dropped due to too many VLAN tags. Packets are dropped if number of VLANS is above the limit setup in the **Source Port Table**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4478

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.15 Minimum Allowed VLAN Drop

Number of packets dropped due to insufficient VLAN tags. Packets are dropped if number of VLANS is below the limit setup in the **Source Port Table**.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4477

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.16 Reserved MAC DA Drop

Number of packets dropped due to the packets destination MAC address match a **Reserved Destination MAC Address Range** that is configured to be dropped.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4474

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.17 Reserved MAC SA Drop

Number of packets dropped due to the packets source MAC address match a **Reserved Source MAC Address Range** that is configured to be dropped.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4475

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.18 Unknown Ingress Drop

Number of packets dropped during ingress packet processing due to unknown reasons. Internal error caused by packet drop with an invalid Drop ID.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4466

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.17.19 VLAN Member Drop

Number of packets dropped due to the packets source port notbeing part of the packets VLAN membership.
In Figure 23.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4476

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.18 Statistics: Misc

28.18.1 Buffer Overflow Drop

Counter for the number of packets dropped due to the shared buffer memory being full.
In Figure 23.1, **bmOverflow** with process sequence **16** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 86101

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.18.2 Drain Port Drop

Number of packets dropped due to the port is drained.
In Figure 23.1, **drain** with process sequence **21** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress port
Address Space : 86259 to 86270

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.18.3 Egress Resource Manager Drop

Number of packets dropped by the egress resource manager.
In Figure 23.1, **erm** with process sequence **15** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read/Write
Addressing : Egress Port
Address Space : 86088 to 86099

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.18.4 Flow Classification And Metering Drop

Number of packets dropped due to flow classification and metering.
In Figure 23.1, **mmp** with process sequence **14** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 85903

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.18.5 IPP Empty Destination Drop

Number of drops due to the determined destination is cleared during post-ingress packet processing and causing no cell to be enqueued in the buffer memory. This happens on single cell packet with end-of-packet drop actions.
In Figure 23.1, **eopDrop** with process sequence **14** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4465

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

28.18.6 MAC RX Broken Packets

Number of broken packets dropped.
In Figure 23.1, **macBrokenPkt** with process sequence **3** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read Only (unreliable)
Addressing : Ingress Port
Address Space : 48 to 59

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.18.7 MAC RX Short Packet Drop

Number of packets dropped due to length below 60 bytes.
In Figure 23.1, **macRxMin** with process sequence **4** represents the internal location of this counter.

Number of Entries : 12
Type of Operation : Read Only (unreliable)
Addressing : Ingress Port
Address Space : 60 to 71

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.18.8 Re-queue Overflow Drop

Counter for the number of packets dropped due to a FIFO overflow in re-queue.
In Figure 23.1, **rqOverflow** with process sequence **24** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 86102

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets	0x0

28.19 Statistics: Packet Datapath

28.19.1 EPP Packet Head Counter

Number of packet first cells through the Egress Packet Process module.
In Figure 23.1, **eppTxPkt** with process sequence **24** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 86308

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

28.19.2 EPP Packet Tail Counter

Number of packet last cells through the Egress Packet Process module.
In Figure 23.1, **eppTxPkt** with process sequence **24** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 86309

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

28.19.3 IPP Packet Head Counter

Number of packet first cells through the Ingress Packet Process module.
In Figure 23.1, **ippTxPkt** with process sequence **13** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4485

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

28.19.4 IPP Packet Tail Counter

Number of packet last cells through the Ingress Packet Process module.
In Figure 23.1, **ippTxPkt** with process sequence **13** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 4486

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

28.19.5 PB Packet Head Counter

Number of packet first cells through the Shared Buffer Memory module.
In Figure 23.1, **pbTxPkt** with process sequence **18** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 86256

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

28.19.6 PB Packet Tail Counter

Number of packet last cells through the Shared Buffer Memory module.
In Figure 23.1, **pbTxPkt** with process sequence **18** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 86257

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

28.19.7 PS Packet Head Counter

Number of packet first cells through the Parallel to Serial module.
In Figure 23.1, **psTxPkt** with process sequence **25** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 95248

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

28.19.8 PS Packet Tail Counter

Number of packet last cells through the Parallel to Serial module.
In Figure 23.1, **psTxPkt** with process sequence **25** represents the internal location of this counter.

Number of Entries : 1
Type of Operation : Read/Write
Address Space : 95249

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

28.20 Statistics: Routing

28.20.1 Next Hop Hit Status

Status bit is set if a packet was routed using the corresponding entry in the **Next Hop Table**.
In Figure 23.1, **nextHop** with process sequence **11** represents the internal location of this counter.

Number of Entries : 1024
Type of Operation : Read/Write
Addressing : Next Hop
Address Space : 84799 to 85822

Field Description

Bits	Field Name	Description	Default Value
0	ipv4	The next hop entry was hit with an IPv4 packet.	0x0
1	ipv6	The next hop entry was hit with an IPv6 packet.	0x0
2	mpls	The next hop entry was hit with an MPLS packet.	0x0

28.20.2 Received Packets on Ingress VRF

Number of packets enter a VRF on ingress.
In Figure 23.1, **vrfIn** with process sequence **11** represents the internal location of this counter.

Number of Entries : 4
Type of Operation : Read/Write
Addressing : vrf
Address Space : 84795 to 84798

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.20.3 Transmitted Packets on Egress VRF

Number of packets leave a VRF on egress.
In Figure 23.1, **vrfOut** with process sequence **19** represents the internal location of this counter.

Number of Entries :	4
Type of Operation :	Read/Write
Addressing :	vrf
Address Space :	95197 to 95200

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.21 Statistics: SMON

28.21.1 SMON Set 0 Byte Counter

Number of bytes counted in SMON Set 0.
In Figure 23.1, **smon** with process sequence **11** represents the internal location of this counter.

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	VLAN PCP
Address Space :	84747 to 84754

Field Description

Bits	Field Name	Description	Default Value
23:0	bytes	Number of bytes.	0x0

28.21.2 SMON Set 0 Packet Counter

Number of packets counted in SMON Set 0.
In Figure 23.1, **smon** with process sequence **11** represents the internal location of this counter.

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	VLAN PCP
Address Space :	84731 to 84738

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

28.21.3 SMON Set 1 Byte Counter

Number of bytes counted in SMON Set 1.
In Figure 23.1, **smon** with process sequence **11** represents the internal location of this counter.

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	VLAN PCP
Address Space :	84755 to 84762

Field Description

Bits	Field Name	Description	Default Value
23:0	bytes	Number of bytes.	0x0

28.21.4 SMON Set 1 Packet Counter

Number of packets counted in SMON Set 1.
In Figure 23.1, **smon** with process sequence **11** represents the internal location of this counter.

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	VLAN PCP
Address Space :	84739 to 84746

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0