Packet Architects AB

Ethernet Switch TSN 7x1G + 2x3G User Guide

Core Revision unknown

Datasheet Revision unknown

Contents

List of Figures

List of Tables

Chapter 1

Overview

This L2 Ethernet Switching Core offers full wire-speed on all 9 ports. Each port has 8 egress queues which are controlled by a multi-level 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 150 bytes. In total the buffer memory has a capacity of 1536 cells.

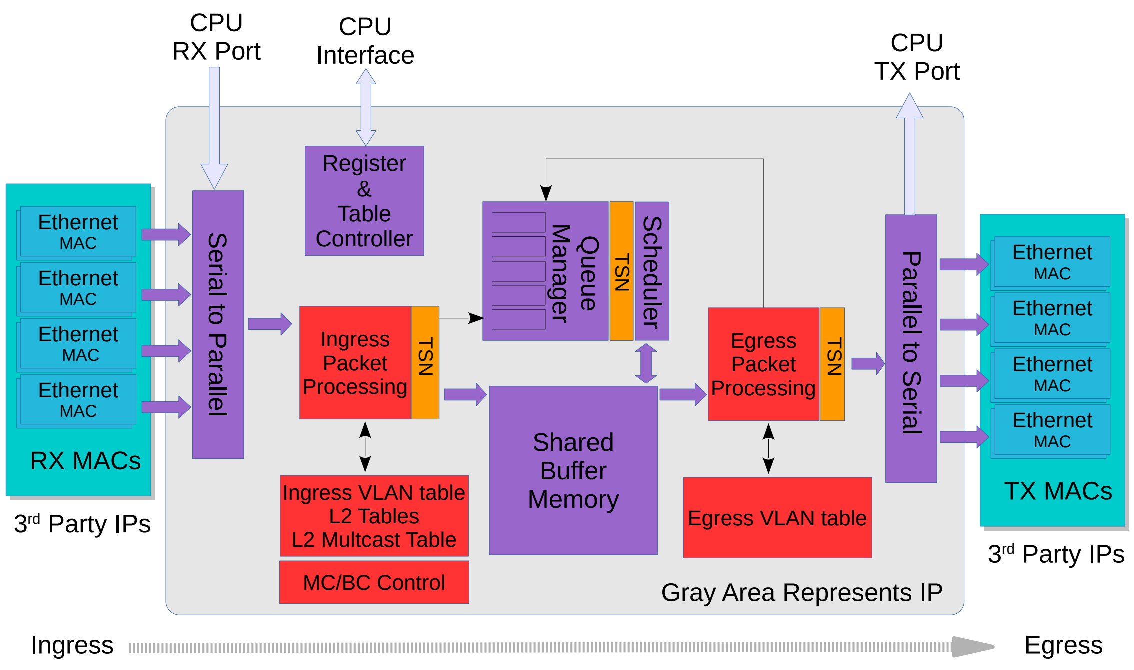

Figure 1.1: Switch Core Overview

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

- 7 ports of 1 Gigabit Ethernet.
- 2 ports of 3 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 32749 bytes.
- Passes maximum overlap mesh test (RFC2899) using all ports for all packet sizes up to 1518 bytes.
- Time-Sensitive Networking:
  - IEEE Std 802.1Qci-2017: Per-Stream Filtering and Policing
  - IEEE Std 802.1CB-2017: Frame Replication and Elimination for Reliability
  - IEEE Std 802.1Qbv-2015: Enhancements for Scheduled Traffic
  - IEEE Std 802.1Qav-2009: Credit Based Shaper
- 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.
- RSPAN - Remote Switch Port Analyzer
- 4 source MAC address ranges with a number of different actions.
- 4 destination MAC address ranges with a number of different actions.
- 1,024 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.
- Egress VLAN translation table allowing unique VID-to-VID translation per egress port.
- VLAN priority tag can bypass VLAN processing and be popped on egress.
- 496 entries of ingress classification / ACL Lookups. The classification / ACL keys are configurable for each source port and the fields are selected from a incoming packets L2, L3 or L4 fields. The selection is described in 10.2 The classificaiton / ACL key can be up to 372 bits long. The classification / ACL lookup is based on a combination of hash and TCAM. The actions which can be done is listed below:
  - Multiple actions can be assigned to each result. All results can be done in parallel if the user so wishes.

  - Result action can be to drop a packet.
  - Result action can be to send a packet to the CPU port.
  - Result action can be to send a packet to a specific port.
  - Result action can be to update a counter. There are 32 counters which can be used by the classification / ACL engine.
  - Result action can be to force packet to a specific queue on a egress port.
  - Result action can be to assign a meter/market/policer to measure the packet bandwidth.
  - Result action can be to assign a color to the packet which is used by the meter/marker/policer.
  - Result action can be to force the packet to use a specific VID when doing the VLAN table lookup.
  - Result action can be to do a input mirror on a packet.
  - Result action can be to not allow the packet to be learned in L2 MAC table.
- The ingress configurable classification / ACL engine can use the type and code fields from ICMP frames.
- The ingress configurable classification / ACL engine can use the fields, including the group address, from IGMP frames.
- 1843200 bits shared packet buffer memory for all ports divided into 1536 cells each of 150 bytes size
- 8 priority queues per egress port.
- Configurable mapping of egress queue from IP TOS, MPLS exp/tc or VLAN PCP bits.
- 32 ingress admission control entries.
- Deficit Weighted Round Robin Scheduler.
- Bandwidth shapers per port.
- Individual bandwidth shapers for each priority on each port.
- Individual bandwidth shapers for each queue on each port.
- 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.
- Attack prevention by TCP flag rules combined with TCP-port and IP address checks, this also includes IMCP length attack checks.

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 serial to parallel converter accumulates 150 bytes to build a cell, and the cell is sent to ingress processing, if a packet consists of more than 150 bytes then a new cell is built. This is repeated until the *end of packet* signal is asserted.
4. 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.
5. If the packet matches a certain traffic type whose bandwidth is monitored by the core, it will be pointed to one of the 32 meter-marker-droppers to do the rate measurement. The result may drop the packet or change the packet color.
6. 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.
7. Unless the packet is dropped, the packet is written cell-by-cell into the buffer memory with the I2E header appended.
8. 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.
9. 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.
10. 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.
11. 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.
12. A port with packets available for transmission, will only transmit a new packet if the port shaper allows it to.
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.
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. Data is transmitted on the output port.
18. Egress port counters are updated.

1.2 Port Numbering Table

Table 1.1 shows the port numbering. Register **CPU Port** determines the port that can serve as a CPU port, the default CPU port number is 8.

Interface Number	**BW**	**Clock**	**Clock Frequency**	**Sync With Core Clock**	**Port Number & Multicast Table Bit**	**CPU Port**
0	3.0Gbit/s	clk_mac_rx/tx_0	125.00MHz	Yes	0	Optional
1	3.0Gbit/s	clk_mac_rx/tx_1	125.00MHz	Yes	1	Optional
2	1.0Gbit/s	clk_mac_rx/tx_2	125.00MHz	Yes	2	Optional
3	1.0Gbit/s	clk_mac_rx/tx_3	125.00MHz	Yes	3	Optional
4	1.0Gbit/s	clk_mac_rx/tx_4	125.00MHz	Yes	4	Optional
5	1.0Gbit/s	clk_mac_rx/tx_5	125.00MHz	Yes	5	Optional
6	1.0Gbit/s	clk_mac_rx/tx_6	125.00MHz	Yes	6	Optional
7	1.0Gbit/s	clk_mac_rx/tx_7	125.00MHz	Yes	7	Optional
8	1.0Gbit/s	clk_mac_rx/tx_8	125.00MHz	Yes	8	Default

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*.

```
      |
      |
      +-->[ MAC DA == BPDU         ]---+
      +-->[ MAC DA == SSTP         ]---+
      +-->[ MAC DA == cpuMacAddr   ]---+
      +-->[ MAC DA == other        ]---+
      +-->[ MAC DA == LLDP.mac1/2/3]---+
      +-->[ MAC DA == LACP.mac     ]---+
                                       |
      +--------------------------------+
      |
  [ MAC SA ]
      |
      +---[ EType==fromCpu    ]
      |   [ 5 byte CPU tag    ]-----+
      |                             |
      +<----------------------------+
      |
      +<----------------------------+
      |                             |
      |      0,1,2 VLAN tags        |
      +---[ EType==C-/S-VLAN TPID ]-+
      |   [   2 byte VLAN TCI     ]
      |
      +---[ EType==R-TAG.eth ]
      |   [ 4 byte R-TAG     ]------+
      |                             |
      +<----------------------------+
      |       |
      +-->[ EType==LLDP.eth]--> X
      +-->[ EType==IEEE_1722_AVTP.eth]--> X
      +-->[ EType==ARP.eth]--> X
      +-->[ EType==RARP.eth]--> X
```

```
+-->[ EType==ieee1588EthType.eth]--> X
+-->[ EType==ieee8021xEthType.eth]--> X
+---[ EType==MPLS ]
|   [ MPLS tag 1  ]--> X
|
+-->[ EType==unknown ]--> X
|
+-->[ EType==PPPoE ]
|   [ PPPoE header ]
|     |
|     +-->[ EType!=IPv6 or EType !=IPv4 ]--> X
|     +-->[ EType==IPv6 ]-----+
|     +-->[ EType==IPv4 ]     |
|                    |        |
+-->[ EType==IPv6 ]-----------+
|                    |        |
+-->[ EType==IPv4 ]----+      |
                     | |      |
                     v v      v
           [ IPv4 Header ]  [ IPv6 Header ]
                  |           |
+-----------------+-----------+
|
+-->[ TCP Header                    ]--> X
+-->[ L4Proto == ahHeader.l4Proto   ]--> X
+-->[ L4Proto == espHeader.l4Proto  ]--> X
+-->[ L4Proto == gre.l4Proto        ]--> X
+-->[ L4Proto == sctp.l4Proto       ]--> X
+-->[ IGMP Header                   ]--> X
+-->[ ICMP Header                   ]--> X
+-->[ UDP Header                    ]----+
                                         |
+----------------------------------------+
|
+-->[ UDP Dest Port == bootp.udp1/udp2  ] --> X
+-->[ UDP Dest Port == capwap.udp1/udp2 ] --> X
+-->[ UDP Dest Port == gre.udp1/udp2    ] --> X
+-->[ UDP Dest Port == Unknown          ] --> 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.

   (e) If the DA MAC is equal to the register **LACP Packet Decoder Options** field **mac** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

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) ARP
   If the Ethernet Type field is equal to the **ARP Packet Decoder Options** field **eth** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

   (c) RARP
   If the Ethernet Type field is equal to the register **RARP Packet Decoder Options** field **eth** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

   (d) Redundancy Tag
   If the Ethernet Type field value is 0xF1C1 then the packet carries a sequence number. Redundancy tag is used for Frame Replication and Elimination for Reliability (FRER)

   (e) 802.1X and EAPOL Packets
   If the Ethernet Type field is equal to register **IEEE 802.1X and EAPOL Packet Decoder Options** field **eth** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped. The drop counter is located in **IEEE 802.1X and EAPOL Decoder Drop**.

   (f) IEEE 1588 L2 Ethernet Type
   If the Ethernet Type field is equal to register **IEEE 1588 L2 Packet Decoder Options** field **eth** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

   (g) 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**.

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

(i) From CPU Tags
Packets from CPU will use a Ethernet type value of 0x9988.The From CPU Tag is further described in Chapter 30.

(j) 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.

(k) 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.

  iii. AH Header
  If the next protocol field in IPv4 or IPv6 is equal to the register **AH Header Packet Decoder Options** field **l4Proto** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

  iv. ESP Header
  If the next protocol field in IPv4 or IPv6 is equal to the register **ESP Header Packet Decoder Options** field **l4Proto** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

  v. GRE
  If the next protocol field in IPv4 or IPv6 is equal to the register **GRE Packet Decoder Options** field **l4Proto** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

  vi. SCTP
  If the next protocol field in IPv4 or IPv6 is equal to the register **SCTP Packet Decoder Options** field **l4Proto** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

(l) UDP or TCP Source or Destination Port Checks

  i. GRE
  If the Destination Port in UDP is equal to the **GRE Packet Decoder Options** field **udp1** or field **udp2** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

  ii. DNS
  If the Destination Port in UDP or TCP is equal to the **DNS Packet Decoder Options** field **l4Port** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

  iii. BOOTP or DHCP
  If the Destination Port in UDP is equal to the register **BOOTP and DHCP Packet**

Decoder Options field **udp1** or field **udp2** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

iv. CAPWAP
If the Destination Port in UDP is equal to the register **CAPWAP Packet Decoder Options** field **udp1** or field **udp2** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

v. IEEE 1588 L4
If the Destination Port, and IPv4 or IPv6 and the UDP is equal to the register **IEEE 1588 L4 Packet Decoder Options** then the field source port bit in the **toCpu** determines if the packet shall be sent directly to the CPU, bypassing normal forwarding process. The source port bit in the field **drop** determines if the packet shall be dropped.

(m) 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. Configurable ACL
   The incoming packet is classified on a configurable selection of L2, L3 and L4 fields. The ACL lookup is a d-left hash search, described in Dleft Lookup. There are numerous actions that can be applied when a packet matches an ACL entry. For details see the **Configurable ACL Engine** section.

8. 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.

9. 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.

10. 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.

11. IPv4 checksum check and drop.
For IPv4 packets calculate the checksum value and optionally drop the packet with wrong checksum value.

12. Attack prevention drop
TCP/UDP packets are checked by **TCP/UDP Flag Rules** to prevent security or DOS attacks.

13. L2 Switching
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. L2 Action Table Lookup
The L2 Action Table Lookups provides a extra level of controll over what shall be done with the L2 packets. It can be used to archive 802.1X compliance and be used to secure the switch. The functionality has a enable bit in the **Source Port Table** field **enableL2ActionTable**. Depending on the result from both the L2 SA Lookup , L2 DA Lookup and status on source port (**l2ActionTablePortState**) and destination port(s) **L2 Action Table Egress Port State** a address is formed to read out L2 Action Tables. The **L2 Action Table** is based on the packets destiantion ports, while **L2 Action Table Source Port** is based on the packets incoming source port. If the packet is going to no egress port (portmask==0) then none of the **L2 Action Table** actions will be done while the **L2 Action Table Source Port** is always carried out (When function is enabled).

15. 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.

16. 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.

17. Learning Lookup
The source MAC address is searched in the **L2 DA 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.

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

3. 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.

4. Egress VLAN Translation
   Potentially replace the outgoing VID and Ethernet Type on a specific port with a specific VID. Uses a Dleft lookup in **Egress VLAN Translation Small Table**, **Egress VLAN Translation Large Table** and **Egress VLAN Translation TCAM**.

5. RSPAN
   Perform a push or pop of an RSPAN tag if enabled in **Egress RSPAN Configuration**.

6. 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 18 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 4 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 two places (t1-t2) in the core where latency jitter can be introduced. See Figure 4.1 on page 32.

t1 In the SP the ports are visited in a fixed order, thus introducing a jitter the size of the port visitation period.

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

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

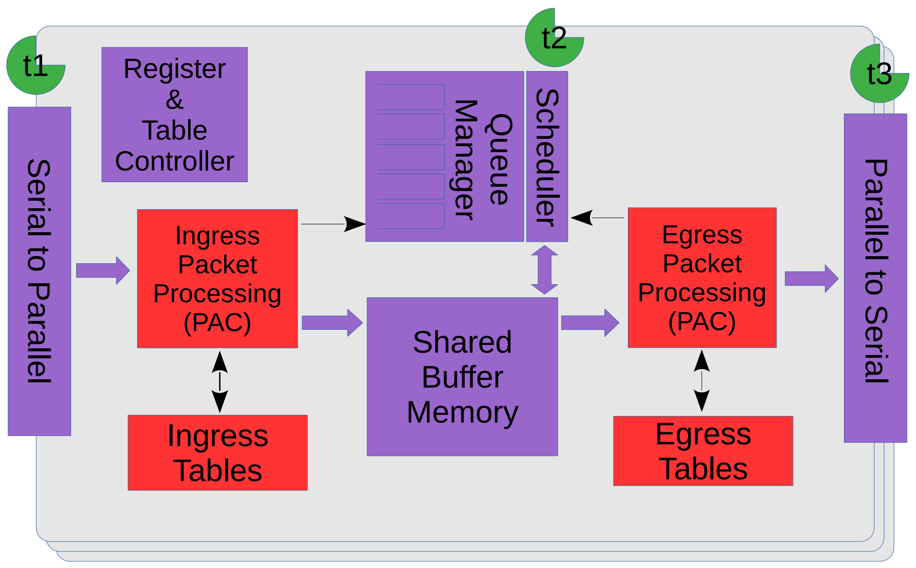

Figure 4.1: Jitter Overview

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 Ingress Configurable ACL

The ACL engine has an option to override the ingress VID assigned above. If the forceVidValid field in the **Ingress Configurable ACL N Small Table** is set to 1, the corresponding forceVid field will be used as the new ingress VID value. The same applies to the **Ingress Configurable ACL N Large Table** and **Ingress Configurable ACL N TCAM Answer** tables. The detailed L2 ACL match and action are described in the **Configurable ACL Engine** 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.

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 18.1).

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:

```
+-----------------+-----------------------------+
|                 |            TCI              |
|     TPID        +-------+-------+-------------+
|                 |  PCP  |  DEI  |     VID     |
+-----------------+-------+-------+-------------+
```

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.

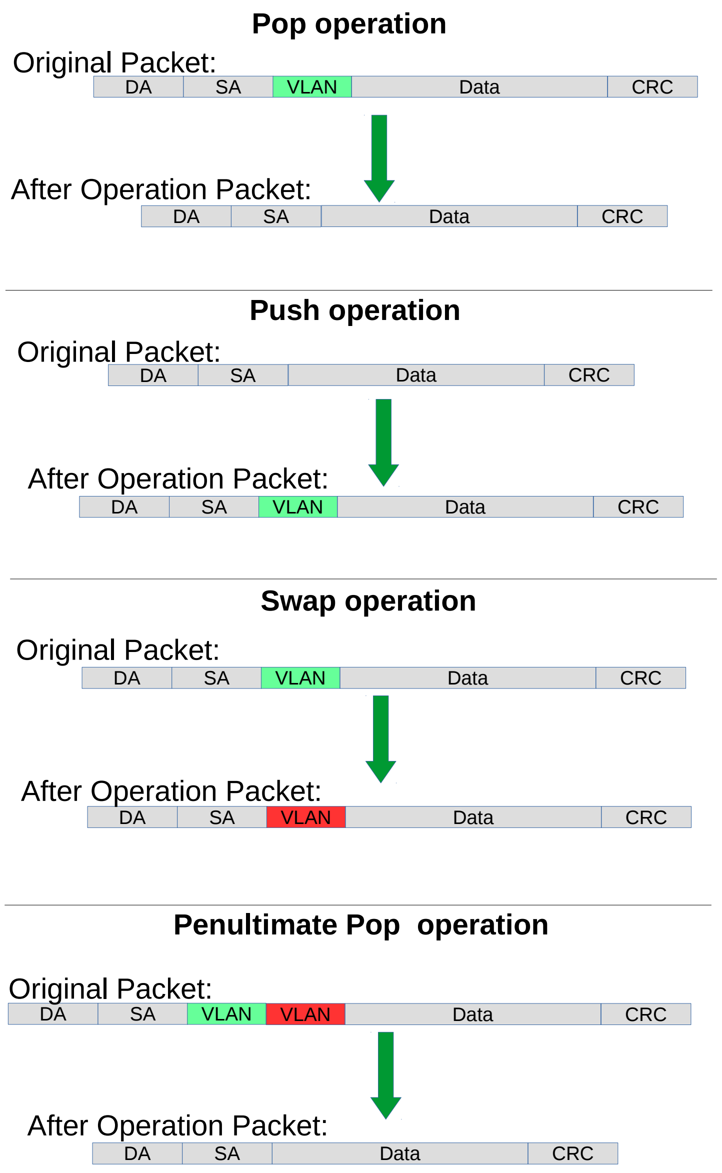

Figure 5.1: VLAN Packet Operations

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**.

5.3.3 Operation Based On Incoming Packets Number of VLANs

There exists a option which overrides the default **vlanSingleOp** field depending on the number of VLANs the packet has. This operation allows a user to set a specific operation depeding on the number of VLANs the incoming packet has. This VID operation then overrides the default VID operation. This operation is setup in field **nrVlansVidOperationIf**.

5.3.4 Configurable ACL VLAN Swap Operation

The **Ingress Configurable ACL N Small Table** , **Ingress Configurable ACL N Large Table** and **Ingress Configurable ACL N TCAM Answer** tables provides three fields updateVid, updatePcp and updateCfiDei to perform a VLAN swap operation. The VLAN type can also be changed using the updateEType. VLAN push and pop operations are not supported in this ACL.

5.3.5 VLAN Table Operation

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

5.3.6 VLAN Table VID Operation Based On the Packets Number of VLANs

There exists a option which overrides the default **vlanSingleOp** field depending on the number of VLANs the packet has. This operation allows a user to set a specific operation depeding on the number of VLANs the incoming packet has after the source port operation push/pop/swap/penultmate pop has been done. The VID operation then overrides the default VID operation specified in field **vlanSingleOp** and all its data fields. This operation is setup in field **nrVlansVidOperationIf**. This setting is done on a per port basis allowing each source port to have its own setting. Source port 0 is represented in bits [1:0] , Source port 1 is represented in bits [3:2] and so on.

5.3.7 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**.

5.3.8 Egress Port VID Operation

Egress Port VID Operation provides an option to override the default **vlanSingleOp** depending on the number of VLANs the packet has and the ingress VID of the packet. Each entry of the **Egress Port VID Operation** register compares the egress port, ingress VID and VLAN tagging conditions and activate the corresponding VLAN operation from the first hit.

5.3.9 Egress Vlan Translation

This operation which is located in the egress path allows a replacement of the outermost VLAN Identifier in the packet. The egress port, the outermost VID of the packet after all VLAN operations and the outermost VID type (C or S tag) creates a lookup key to be used in a Dleft lookup using the **Egress VLAN Translation Small Table**, **Egress VLAN Translation Large Table** and **Egress VLAN Translation TCAM** Tables. If multiple hits the **Egress VLAN Translation Selection** can be used to determine which result to select. It is possible to mask the search data using **Egress VLAN Translation Search Mask**..

5.3.10 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 18.1 **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.

5.3.11 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.
2. One VLAN swap operation from:
   - **updateVid**, **updatePcp**, **updateCfiDei** or **updateEType** in the **Configurable ACL Engine**.
3. One of the four VLAN operations from:
   - **VLAN Table** VLAN operation.
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.

5.3.12 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]
```

5.3.13 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 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** and the **L2 Aging Status Shadow Table**. 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**.
- 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.
- Normally the source port is excluded from the destination portmask. If that results in an empty destination port mask then the packet is dropped and counted in the **L2 Lookup Drop** register.

  This behaviour can be changed using the **Hairpin Enable** register, allowing a packet to be switched to the same port it came in.
- 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 the **Flooding Action Send to Port** is enabled on this source port (using **enable** set to one) and the packet is flooded then the packet is sent to the destination port pointed to by the field **destPort** instead of being flooded to all ports part of the packets VLAN. The destination port does not need to be part of the packets VLAN group membership.
- 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.

6.3 L2 Action Table

There is two tables which allows detailed control for each packet depending on the source L2 MAC table result, the destination L2 MAC table result and the ingress and egress port which each has a configurable state. This the L2 Action Table used for each egress port which the packet shall be sent to is defined in **L2 Action Table** and secondly the **L2 Action Table Source Port**. Both tables used a number of bits from the source port table, egress port state, SA and DA MAC lookups to form a address into the tables which is then read out and acted on. Each source port enables if the L2 Action tables shall be used or not using the field **enableL2ActionTable**. The L2 Action Tables can be used to permit specific frames from certain source ports to other destination ports using a filter defined in **Allow Special Frame Check For L2 Action Table**. There are 4 rules which are shared among all ports and pointed from the L2 Action Tables as a result by setting **useSpecialAllow** to one and then pointing to the rule using field **allowPtr**.

If a packet is going to no egress ports (portmask==0) then none of the actions in the **L2 Action Table** will be carried out, while the **L2 Action Table Source Port** will always be carrie out since a packet always comes in on a source port. Because of this the addressing is slightly different for these two table lookups.

The use cases for the tables is described below. Both tables have the same result actions.

6.3.1 Learning Unicast and Learning Multicast

As stated before the L2 Action Table can be used to stop learning on certain frames. There is a additional setting allowing the user to define if the learning is not to be allowed for unicast or multicast packets. Since a learning lookup is based on the Source MAC address this is also what is compared against. If the SA MAC is a multicast address then the **noLearningMc** field will be used to determine if the packet shall be learned or if SA MAC address is a unicast then the **noLearningUc** will determine if the packet shall be learned or not.

6.3.2 Drop and Learning

If a packet is dropped by the L2 Action Table the packet will be still be learned. If you want the packets not to be learned then both **dropAll** and **noLearningUc** and **noLearningMc** should be turned on (set to one).

6.3.3 Priorities Between Actions

There are multiple actions from the L2 action table this section explains the order between them.

1. The drop special packet is first carried out and drops all instances of the packet
2. The drop port move then takes priority and drops all instance of the packet
3. The drop-all drops all instances of a packet however special type packets can still be accepted if they are setup to do so.
4. After the drops the send-to-CPU is carried out. Only a single copy will be sent to the CPU.

6.3.4 Using L2 Action Table for 802.1X

Simple Port Authentication

By using the source port bit **l2ActionTablePortState** and the egress port state bit in register **L2 Action Table Egress Port State** to indicate if a port is authenticated or not packets can be limited to communicate with other ports. This is done by setting up the different addresses in the L2 Action Table to do drop operations when a packet comes in from a non-authenticated port going to a authenticated port.

Port Authentication with MAC addresses

In order to allow already existing computers (MAC address) allow to pass through the switch without any problems the SA lookup result bit **l2ActionTableSaStatus** can be used indicate if this source MAC address (i.e. computer/end-station) has been authenticated or not on this port. A non-authenticated computer shall still be able to communicate with other ports which are not authenticated. Since the three bits partly forms the address into the L2 Action Table it is possible to form rules which when a packet is allowed to access other ports depending on what the state of these ports are and if the computer it wants to communicate with is known to the switch or not. The field **l2ActionTableDaStatus** can be used to further enhance the security wheather or not two computers shall be able to communicate.

Port Authentication Enhancements with Learning and Port-Move

As the network security needs to be enhanced further the L2 Action Table allows setting up rules if a packet coming in and going to different ports shall be able be able to be learned or if a already existing MAC address shall be able to be port moved.

Port Authentication Enhancements only allow certain traffic types

As the last enhancement there can be special rules formed which allows only certain packet types to pass on a port combination using the result options **useSpecialAllow** and **allowPtr**. This allowPtr points to general rules of which packet types to drop or to allow. This rules are setup in **Allow Special Frame Check For L2 Action Table**.

Chapter 7

Mirroring

This core supports both input and output mirroring.

7.1 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.
- Packets hit in the **Configurable ACL Engine** can send an input mirror copy to the port configured in ACL's **`destInputMirror`** field if there is an enabled **`inputMirror`** action.

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**. If a packet has an input mirror action from the ACL and its source port also enables input mirroring, the destination port of that copy is determined by the ACL result.

7.2 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

7.2.1 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 8

RSPAN - Remote Switch Port Analyzer

RSPAN is a function that allows mirroring traffic to other switches by encapsulating the packets in a VLAN tag.

An RSPAN network consists of switches with three roles.

1. *Source Device*
   The source device is where the mirrored traffic originates. It uses the normal mirroring functions to send the mirror copies. The mirrored packets are encapsulated in a RSPAN tag and output on a port.

2. *Intermediate Device*
   An intermediate device just forwards the RSPAN tagged packets.

3. *Destination Device*
   The destination device removes the RSPAN tag and output the packet on a port.

8.1 Source Device

Input and output mirroring can be used to create the mirror copies. A dedicated RSPAN port, reflector port, is used. On this port only mirror traffic should be sent. No other traffic should be switched to this port, i.e. normal switching functions should not use this port as a destination.

The reflector port must be configured to push a RSPAN tag by setting **pushRspanTag** in **Egress RSPAN Configuration**.

The RSPAN tag is a normal VLAN tag and the content of the tag is configured in **Egress RSPAN Configuration**.

A switch can have multiple reflector ports.

8.2 Intermediate Device

An intermediate device must be configured to allow receiving RSPAN tagged packets and to forward them to a dedicated port. This can be accomplished by setting up a source port VLAN with a GID only used for this purpose. The VLAN will have two member ports, the RSPAN ingress port and the RSPAN egress port. Learning should be disabled for the ingress port. The ingress packets will then be flooded to the egress port.

8.3 Destination Device

The destination device receives the RSPAN packet on a dedicated ingress port and forwards them to the dedicated monitor port. This forwarding can be done in the same way as an intermediate device.

On the egress port the RSPAN tag is popped by setting **popRspanTag** in **Egress RSPAN Configuration**.

Chapter 9

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.

9.0.1 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.

9.1 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
```

9.2 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.

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,
              saIp, useIpInHash,
              daIp,
              srcPort):
    # This function builds the key to be
    #   used for calculating the hash.
    final_data = 0
    if useDaMacInHash==0:
        daMac = 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 10

Classification

10.1 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.
- L2 Source MAC Drop is setup in table **L2 Destination Table** using field **pktDropSa**. This will drop all packets which matches this SA MAC address.
- 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.

10.2 Configurable Ingress ACL Engine

The ingress ACL engine uses a configurable selection of fields from the incoming packet headers, from L2 fields to L4 fields. From the selected fields a hash table lookup is then done using D-left hashing. The hashing is combined with a TCAM to resolve hash collisions and to enable per entry masking of data. Each of the hash tables can also be masked, but only a single mask can be applied for all data in a hash table.

There are 2 parallell ACL engines that each can perform one lookup per packet. All lookups are done in parallel and then there is a post processing of all the matching results to determine what actions to perform. There can be multiple actions taken for a single packet. How the actions are determined when there are multiple matches are described below.

10.2.1 Field Selection

For each source port the **useAcl***N* field in the **Source Port Table** configures if the incoming packets shall be subjected to an ACL lookup. By default the ACL is turned off.

If the ACL is turned on then the field **aclRule***N* is used as a pointer into **Ingress Configurable ACL N Rules Setup** . This determines which fields that are used in the ACL lookup for this source port.

Each ACL engine has its own unique fields which can be selected. These are listed below. A field is selected by setting the corresponding bit in the fieldSelectBitMask.

ACL Engine	Width of Search Data	Fields to select from	Nr of Rules (Fields) to maximum use	Number of Parallel Hash Tables	Small Table Entries	Large Table Entries	TCAM Entries
0	208	32	6	4	64	128	16
1	372	32	10	2	16	256	16

Table 10.1: Ingress ACL Engine Settings

Pre Lookup for Configurable Ingress ACL Table 0

This ACL engine has a pre-lookup. This is done to enable a different rule on how to build the ACL fields to be selected. If this lookup does not result in a valid rule poitner then the rule pointer from the source port table will be selected. The prelookup is setup in **Ingress Configurable ACL 0 Pre Lookup**

Packet Field	Size in Bits	Description
Source Port Bits	2 bits	The source port bits from source port table **preLookupAclBits**.
L2 Protocol	1 bits	The packets L2 Type 0 = Other than this list. 1 = IEEE 1722 AVTP
Type of L3 Packet	2 bits	The packets L3 Type 0 = IPv4 1 = IPv6 2 = MPLS 3 = Others.
Type of L4 Packet	3 bits	The packets L4 Type 0 = Not known. 1 = Is IPv4 or IPv6 but type is not any L4 type in this list. 2 = UDP 3 = TCP 4 = IGMP 5 = ICMP 6 = ICMPv6 7 = MLD

Fields for Configurable Ingress ACL Table 0

The following fields can be selected for Configurable Ingress ACL Table 0, the column Bit in Select Bitmask is the number which is set in the bitmask to select the field.

Bit in Select Bitmask	Field Name	Size in Bits	When is field valid?	Description
0	MAC DA	48	Always valid	The packets destination MAC address.
1	MAC SA	48	Always valid	The packets source MAC address
2	Outer VID	12	When packet has a VLAN.	The packets outermost VLAN Identifier (VID)
3	Has VLANs	1	Always valid	Does the packet have any VLAN tags 0 = No VLAN in packet 1 = One or more VLANs in packet
4	Outer VLAN Tag Type	1	When packet has an outer VLANs.	When the packet has an outer VLAN what Ethernet Type is this VLAN? 0 = Customer VLAN Tag 1 = Service VLAN Tag
5	Inner VLAN Tag Type	1	When packet has an inner VLAN.	When the packet has an inner VLAN what Ethernet Type is this VLAN? 0 = Customer VLAN Tag 1 = Service VLAN Tag
6	Outer PCP	3	When packet has a VLAN.	The packets outermost VLAN PCP field.
7	Outer DEI	1	When packet has a VLAN.	The packets outermost VLAN DEI field.
8	Inner VID	12	When packet has a two VLANs.	The packes innermost VLAN Identifier (VID).

Bit in Select Bit-mask	Field Name	Size in Bits	When is field valid?	Description
9	Inner PCP	3	When packet has a two VLANs.	The packets innermost VLAN PCP field.
10	Inner DEI	1	When packet has a two VLANs.	The packets innermost VLAN DEI field.
11	IPv4 SA	32	When L2 frame holds a IPv4 packet.	IPv4 Source Address.
12	IPv4 DA	32	When L2 frame holds a IPv4 packet.	IPv4 Destination Address.
13	IPv6 SA	128	When L2 frame holds a IPv6 packet.	IPv6 Source Address.
14	IPv6 DA	128	When L2 frame holds a IPv6 packet.	IPv6 Destination Address.
15	Outer MPLS	20	When L2 frame holds a MPLS packet.	Outermost MPLS label.
16	TOS	8	When packet is a IPv4 or IPv6	IPv4 or IPv6 Type-Of-Service (TOS) byte.
17	TTL	8	When packet is a IPv4,IPv6 or MPLS	IPv4, IPv6 or MPLS Time-To-Live (TTL) byte.
18	L4 Source Port	16	When packet is a IPv4 or IPv6 and UDP or TCP L4 protocol is present	L4 TCP or UDP packets source port.
19	L4 Destination Port	16	When packet is a IPv4 or IPv6 and UDP or TCP L4 protocol is present	L4 TCP or UDP packets destination port.
20	MLD Address	128	When packet is a IPv6 and the ICMPv6 type is equal to 130,131,132	The MLD headers Multicast Address field.
21	ICMP Type	8	When L4 packet is a ICMP packet	ICMP Type.
22	ICMP Code	8	When L4 packet is a ICMP packet	ICMP Code.
23	IGMP Type	8	When L4 packet is a IGMP	IGMP Type.
24	IGMP Group Address	32	When L4 packet is a IGMP	IGMP Group Address.
25	AVTP Data	32	When a packet is a IEEE 1722 AVTP packet.	The first 32 bits after the Ethernet Type which contains the fields CD, sub-type,sv, version and type-specific data.
26	L4 Protocol	8	When packet is a IPv4 or IPv6	IPv4, IPv6 L4 protocol type byte.

Bit in Select Bit-mask	Field Name	Size in Bits	When is field valid?	Description
27	Ethernet Type	16	Always valid	The packets Ethernet Type after VLANs.
28	L4 Type	3	Always valid	The type of an L4 packet. 0 = Not any type in this list. 1 = IPv6 or IPv4 packet but L4 protocol is not UDP, TCP, IGMP, ICMP, ICMPv6 or MLD 2 = UDP in IPv4/6 3 = TCP in IPv4/6 4 = IGMP in IPv4/6 5 = ICMP in IPv4/6 6 = ICMPv6 in IPv6, excluding MLD 7 = MLD - sub protocol of ICMPv6
29	L3 Type	2	Always valid	The type of an L3 packet. 0 = IPv4 1 = IPv6 2 = MPLS 3 = Not IPv4,IPv6 or MPLS.
30	Source Port	4	Always valid	The source port of the packet.
31	Rule Pointer	3	Always valid	The rule pointer (index in the **Ingress Configurable ACL N Rules Setup**).

10.2.2 Example Of Selecting Fields For Configurable Ingress ACL Table 0

Since this ACL engine can select up to 6 fields. This is done by setting bits in the rule pointers fieldSelectBitmask. Lets look at a few examples of the layout of the 208 bits in search key looks like when different fields are selected.

Example ACL with Ethernet Type

In this example we only want to create a rule with one field which is the Ethernet Type. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 1000000000000000000000000000 in binary format (Hex value of 0x8000000) and the lookup data will be located as follows:

0 - 17	Ethernet Type Width : 16	Valid 1
	16 1	0 0

Table 10.4: Hash Key Example for Ethernet Type

Example with Destiantion MAC Address and Outer VLAN VID

In this example we want to create a rule which with two fields which are destiantion MAC address and outermost VLAN Identifier. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 101 in binary format (Hex value of 0x5) and the lookup data will be located as follows:

0 - 62	MAC DA Width : 48	Outer VID Width : 12	Valid 2
	61 14	13 2	1 0

Table 10.5: Hash Key Example for Destiantion MAC Address and Outer LAN VID

Example of Simple L2 ACL

In this example we want to create a rule which with three L2 fields which are Destiantion MAC address, source MAC address and Ethernet Type. Typically this is a L2 ACL Engine. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 1000000000000000000000000011 in binary format (Hex value of 0x8000003) and the lookup data will be located as follows:

0	Ethernet Type	MAC DA	MAC SA	Valid
-	Width : 16	Width : 48	Width : 48	3
115	114 99	98 51	50 3	2 0

Table 10.6: Hash Key Example for Simple L2 ACL

Example of L3 IPv4 ACL

In this example we want to create a rule which with four L3 fields which are Destiantion IPv4 address, source IPv4 address, L3 Packet Type and L4 Protocol. Typically this is a L3 ACL Engine. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 100100000000000001100000000000 in binary format (Hex value of 0x24001800) and the lookup data will be located as follows:

0	L3 Type	IPv4 DA	IPv4 SA	L4 Protocol	Valid
-	Width : 2	Width : 32	Width : 32	Width : 8	4
78	77 76	75 44	43 12	11 4	3 0

Table 10.7: Hash Key Example for L3 IPv4 ACL

Example of L4 ACL

In this example we want to create a rule which with five fields which are source port, L4 destiantion Port, L4 source port, L3 Packet Type and L4 Protocol. Typically this is a L4 ACL Engine. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 11001000000110000000000000000000 in binary format (Hex value of 0x640c0000) and the lookup data will be located as follows:

0	Source Port	L3 Type	L4 Protocol	L4 Destination Port	L4 Source Port	Valid
-	Width : 4	Width : 2	Width : 8	Width : 16	Width : 16	5
51	50 47	46 45	44 37	36 21	20 5	4 0

Table 10.8: Hash Key Example for L4 ACL

Example of Openflow Entry

In this example we want to create a rule which looks like an Openflow entry. This can be done by selecing source port, destiantion MAC, source MAC, Ethernet Type, inner VLAN, outer VLAN, L3 Type, IPv4 SA, IPv4 DA, L4 protocol, L4 Source port and L4 Destiantion port and finally the rule pointer. All in all 13 fields are selected. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 11101100000011000001100100000111 in binary format (Hex value of 0xec0c1907) and the lookup data will be located as follows:

0	Source Port	MAC DA	MAC SA	Outer VID	Inner VID	Ethernet Type	L3 Type
-	Width : 4	Width : 48	Width : 48	Width : 12	Width : 12	Width : 16	Width : 2
262	261 258	257 210	209 162	161 150	149 138	137 122	121 120

IPv4 SA	IPv4 DA	L4 Protocol	L4 Destination Port	L4 Source Port	Rule Pointer	Valid
Width : 32	Width : 32	Width : 8	Width : 16	Width : 16	Width : 3	13
119 88	87 56	55 48	47 32	31 16	15 13	12 0

Table 10.9: Hash Key Example for Openflow Entry

Pre Lookup for Configurable Ingress ACL Table 1

This ACL engine has a pre-lookup. This is done to enable a different rule on how to build the ACL fields to be selected. If this lookup does not result in a valid rule poitner then the rule pointer from the source port table will be selected. The prelookup is setup in **Ingress Configurable ACL 1 Pre Lookup**

Packet Field	Size in Bits	Description
Source Port Bits	2 bits	The source port bits from source port table **preLookupAclBits**.
L2 Protocol	1 bits	The packets L2 Type 0 = Other than this list. 1 = IEEE 1722 AVTP
Type of L3 Packet	2 bits	The packets L3 Type 0 = IPv4 1 = IPv6 2 = MPLS 3 = Others.
Type of L4 Packet	3 bits	The packets L4 Type 0 = Not known. 1 = Is IPv4 or IPv6 but type is not any L4 type in this list. 2 = UDP 3 = TCP 4 = IGMP 5 = ICMP 6 = ICMPv6 7 = MLD

Fields for Configurable Ingress ACL Table 1

The following fields can be selected for Configurable Ingress ACL Table 1, the column Bit in Select Bitmask is the number which is set in the bitmask to select the field.

Bit in Select Bitmask	Field Name	Size in Bits	When is field valid?	Description
0	MAC DA	48	Always valid	The packets destination MAC address.
1	MAC SA	48	Always valid	The packets source MAC address
2	Outer VID	12	When packet has a VLAN.	The packets outermost VLAN Identifier (VID)
3	Has VLANs	1	Always valid	Does the packet have any VLAN tags 0 = No VLAN in packet 1 = One or more VLANs in packet
4	Outer VLAN Tag Type	1	When packet has an outer VLANs.	When the packet has an outer VLAN what Ethernet Type is this VLAN? 0 = Customer VLAN Tag 1 = Service VLAN Tag
5	Inner VLAN Tag Type	1	When packet has an inner VLAN.	When the packet has an inner VLAN what Ethernet Type is this VLAN? 0 = Customer VLAN Tag 1 = Service VLAN Tag
6	Outer PCP	3	When packet has a VLAN.	The packets outermost VLAN PCP field.
7	Outer DEI	1	When packet has a VLAN.	The packets outermost VLAN DEI field.
8	Inner VID	12	When packet has a two VLANs.	The packes innermost VLAN Identifier (VID).

Bit in Select Bit-mask	**Field Name**	**Size in Bits**	**When is field valid?**	**Description**
9	Inner PCP	3	When packet has a two VLANs.	The packets innermost VLAN PCP field.
10	Inner DEI	1	When packet has a two VLANs.	The packets innermost VLAN DEI field.
11	IPv4 SA	32	When L2 frame holds a IPv4 packet.	IPv4 Source Address.
12	IPv4 DA	32	When L2 frame holds a IPv4 packet.	IPv4 Destination Address.
13	IPv6 SA	128	When L2 frame holds a IPv6 packet.	IPv6 Source Address.
14	IPv6 DA	128	When L2 frame holds a IPv6 packet.	IPv6 Destination Address.
15	Outer MPLS	20	When L2 frame holds a MPLS packet.	Outermost MPLS label.
16	TOS	8	When packet is a IPv4 or IPv6	IPv4 or IPv6 Type-Of-Service (TOS) byte.
17	TTL	8	When packet is a IPv4,IPv6 or MPLS	IPv4, IPv6 or MPLS Time-To-Live (TTL) byte.
18	L4 Source Port	16	When packet is a IPv4 or IPv6 and UDP or TCP L4 protocol is present	L4 TCP or UDP packets source port.
19	L4 Destination Port	16	When packet is a IPv4 or IPv6 and UDP or TCP L4 protocol is present	L4 TCP or UDP packets destination port.
20	MLD Address	128	When packet is a IPv6 and the ICMPv6 type is equal to 130,131,132	The MLD headers Multicast Address field.
21	ICMP Type	8	When L4 packet is a ICMP packet	ICMP Type.
22	ICMP Code	8	When L4 packet is a ICMP packet	ICMP Code.
23	IGMP Type	8	When L4 packet is a IGMP	IGMP Type.
24	IGMP Group Address	32	When L4 packet is a IGMP	IGMP Group Address.
25	AVTP Data	32	When a packet is a IEEE 1722 AVTP packet.	The first 32 bits after the Ethernet Type which contains the fields CD, sub-type,sv, version and type-specific data.
26	L4 Protocol	8	When packet is a IPv4 or IPv6	IPv4, IPv6 L4 protocol type byte.

Bit in Select Bit-mask	**Field Name**	**Size in Bits**	**When is field valid?**	**Description**
27	Ethernet Type	16	Always valid	The packets Ethernet Type after VLANs.
28	L4 Type	3	Always valid	The type of an L4 packet. 0 = Not any type in this list. 1 = IPv6 or IPv4 packet but L4 protocol is not UDP, TCP, IGMP, ICMP, ICMPv6 or MLD 2 = UDP in IPv4/6 3 = TCP in IPv4/6 4 = IGMP in IPv4/6 5 = ICMP in IPv4/6 6 = ICMPv6 in IPv6, excluding MLD 7 = MLD - sub protocol of ICMPv6
29	L3 Type	2	Always valid	The type of an L3 packet. 0 = IPv4 1 = IPv6 2 = MPLS 3 = Not IPv4,IPv6 or MPLS.
30	Source Port	4	Always valid	The source port of the packet.
31	Rule Pointer	3	Always valid	The rule pointer (index in the **Ingress Configurable ACL N Rules Setup**).

10.2.3 Example Of Selecting Fields For Configurable Ingress ACL Table 1

Since this ACL engine can select up to 10 fields. This is done by setting bits in the rule pointers fieldSelectBitmask. Lets look at a few examples of the layout of the 372 bits in search key looks like when different fields are selected.

Example ACL with IPv4 DA

In this example we only want to create a rule with one field which is the IP DA. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 1000000000000 in binary format (Hex value of 0x1000) and the lookup data will be located as follows:

0	IPv4 DA	Valid
-	Width : 32	1
33	32 1	0 0

Table 10.12: Hash Key Example for IPv4 DA

Example with Destiantion MAC Address and Outer VLAN VID

In this example we want to create a rule which with two fields which are destiantion MAC address and outermost VLAN Identifier. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 101 in binary format (Hex value of 0x5) and the lookup data will be located as follows:

0	MAC DA	Outer VID	Valid
-	Width : 48	Width : 12	2
62	61 14	13 2	1 0

Table 10.13: Hash Key Example for Destiantion MAC Address and Outer LAN VID

Example of Simple L2 ACL

In this example we want to create a rule which with three L2 fields which are Destiantion MAC address, source MAC address and Ethernet Type. Typically this is a L2 ACL Engine. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 1000000000000000000000000011 in binary format (Hex value of 0x8000003) and the lookup data will be located as follows:

0	Ethernet Type	MAC DA	MAC SA	Valid
-	Width : 16	Width : 48	Width : 48	3
115	114 99	98 51	50 3	2 0

Table 10.14: Hash Key Example for Simple L2 ACL

Example of L3 IPv4 ACL

In this example we want to create a rule which with four L3 fields which are Destiantion IPv4 address, source IPv4 address, L3 Packet Type and L4 Protocol. Typically this is a L3 ACL Engine. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 100100000000000001100000000000 in binary format (Hex value of 0x24001800) and the lookup data will be located as follows:

0	L3 Type	IPv4 DA	IPv4 SA	L4 Protocol	Valid
-	Width : 2	Width : 32	Width : 32	Width : 8	4
78	77 76	75 44	43 12	11 4	3 0

Table 10.15: Hash Key Example for L3 IPv4 ACL

Example of L4 ACL

In this example we want to create a rule which with five fields which are source port, L4 destiantion Port, L4 source port, L3 Packet Type and L4 Protocol. Typically this is a L4 ACL Engine. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 11001000000110000000000000000000 in binary format (Hex value of 0x640c0000) and the lookup data will be located as follows:

0	Source Port	L3 Type	L4 Protocol	L4 Destination Port	L4 Source Port	Valid
-	Width : 4	Width : 2	Width : 8	Width : 16	Width : 16	5
51	50 47	46 45	44 37	36 21	20 5	4 0

Table 10.16: Hash Key Example for L4 ACL

Example of Openflow Entry

In this example we want to create a rule which looks like an Openflow entry. This can be done by selecing source port, destiantion MAC, source MAC, Ethernet Type, inner VLAN, outer VLAN, L3 Type, IPv4 SA, IPv4 DA, L4 protocol, L4 Source port and L4 Destiantion port and finally the rule pointer. All in all 13 fields are selected. This means that the fieldSelectBitmask, which is 32 bits , will be set as follows 11101100000011000001100100000111 in binary format (Hex value of 0xec0c1907) and the lookup data will be located as follows:

0	Source Port	MAC DA	MAC SA	Outer VID	Inner VID	Ethernet Type	L3 Type
-	Width : 4	Width : 48	Width : 48	Width : 12	Width : 12	Width : 16	Width : 2
262	261 258	257 210	209 162	161 150	149 138	137 122	121 120

IPv4 SA	IPv4 DA	L4 Protocol	L4 Destination Port	L4 Source Port	Rule Pointer	Valid
Width : 32	Width : 32	Width : 8	Width : 16	Width : 16	Width : 3	13
119 88	87 56	55 48	47 32	31 16	15 13	12 0

Table 10.17: Hash Key Example for Openflow Entry

10.2.4 ACL Search

The hash key is used to perform a lookup using the D-left hashing function described in detail in chapter D-left Lookup.

Before the hash key is used the mask in **Ingress Configurable ACL N Search Mask** is applied.

D-left calculates two hash values from the hash key. These hash values are then used to index the **Ingress Configurable ACL N Small Table** and **Ingress Configurable ACL N Large Table** . The hash calculations are described in section Hash function for Configurable ACL.

In addition to the D-left search the hash key is also used to search in the **Ingress Configurable ACL N TCAM** .

10.2.5 ACL Actions

Once a hit has been determined by any of the searches above, the answer is read out from the corresponding answer entry. If it was a D-left hash hit then the answer actions is part of the hash memories (**Ingress Configurable ACL N Small Table** , **Ingress Configurable ACL N Large Table**). If it was a hit in the TCAM then the **Ingress Configurable ACL N TCAM Answer** is used.

The behavior for multiple hits is configured in **Ingress Configurable ACL N Selection** .

The statistics counter which can be updated are located in the **Ingress Configurable ACL Match Counter**

10.3 Multiple ACL Lookups

The section above describes a single ACL Lookup. There are however 2parallel ACL lookups. The functionality in the different lookup engines is the same with the exception that ACL engine 0 has seperate keys for IGMP, ICMP or MLD packets which are not available in the other engines.

Each of the ACL engines has its own rule configuration as well as its own hash and TCAM tables. The hash and TCAM table sizes and search data width for the different engines are as follows.

By using the same rules for multiple engines the table space for a rule can be extended.

10.3.1 Multiple Actions

If the parallel ACL engines have multiple matches the result actions from each search engine can take effect. How multiple actions are handled depends on the type of action.

Any Match

If one or more ACL engines matches and has this action set then the action will take effect.

Action Field	Ingress Acl 0 Has Action	Ingress Acl 1 Has Action
noLearning	Yes	Yes
dropEnable	Yes	Yes
sendToCpu	Yes	Yes

Table 10.18: Actions that will take effect if one or more is set.

First Match or Priority

If multiple ACL engines matches and has this action set then the value from the lowest numbered engine will be used.If an entry has the priority field set this value will be used and the values which do not have priority set will be ignored.If multiple matches have the priority field set then value from the highest numbered engine will be used.

Enable Field	Priority Field	Value Field	Ingress Acl 0 Has Action	Ingress Acl 1 Has Action
forceVidValid	forceVidPrio	forceVid	Yes	Yes
forceQueue	forceQueuePrio	eQueue	Yes	Yes
forceColor	forceColorPrio	color	Yes	Yes
mmpValid	mmpOrder	mmpPtr	Yes	Yes
updateCfiDei	cfiDeiPrio	newCfiDeiValue	Yes	Yes
updatePcp	pcpPrio	newPcpValue	Yes	Yes
updateVid	vidPrio	newVidValue	Yes	Yes
updateEType	ethPrio	newEthType	Yes	Yes
imPrio	inputMirror	destInputMirror	Yes	Yes
streamValid	N/A	streamHandle	No	Yes
sendToPort	N/A	destPort	Yes	Yes
updateCounter	N/A	counter	Yes	Yes

Table 10.19: The lowest numbered takes effect if no priority else the highest numbered with priority set.

Counter Update

All matches that have counter update action, **updateCounter** set will take effect. Each counter pointed to will be updated. If multiple actions point to the same counter then the counter value will only be incremented by one.

Send To Port

All matches that have an action **sendToPort** will take effect by setting the port number in the packet destination port mask, possibly resulting in a multicast.

Send To CPU

If any match has the **sendToCpu** action set it will take effect. When the To CPU Tag is used the reason code will indicate table index in the lowest numbered engine.

Ingress Admission Control Pointer

If there are multiple matches with actions to set the MMP pointer, mmpPointer then the selection will be done based on the mmpOrder field. This selection is described in Ingress Admission Control.

10.3.2 Default Port ACL action

When a port has the field **enableDefaultPortAcl** set then once a packet misses the ingress ACL lookup, on this source port, this action will be carried out. The action to be carried out is specified in the register **Source Port Default ACL Action**. The actions are the same which can be done for the ACL Lookup. If the bit is set in field **forcePortAclAction** then all packets coming in on this source port are subjected to the actions specified in **Source Port Default ACL Action**. This force ACL default action overrides all other ingress ACL actions/decisions.

Chapter 11

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 12

Attack Prevention

The switch has the possibility to decode TCP/UDP packets and detect and drop packets that matches patterns in order to prevent security or DOS attacks.

If a packet is a TCP/UDP packet (IPv4 or IPv6) the TCP/UDP flags will be compared to all the **TCP/UDP Flag Rules**. The flag comparison can also be combined with a check if the IP Source address equals the IP Destination address. There is also a check if the TCP/UDP source port number matches the TCP/UDP destination port number.

The switch also provides a length check for ICMP packets. **ICMP Length Check** allows the packet to be dropped if the ICMP protocol data size is more than a certain bytes.

If a packet matches any of these rules the packet will be dropped and the **Attack Prevention Drop** will be incremented. When a packet fails either the ICMP length check or the TCP/UDP flag check, the ACL rules can still be hit. However, the ACL action to send the packet to the CPU or any egress port will only override drop decisions based on TCP/UDP rules. In other words, if a packet fails the ICMP length check, it cannot be redirected to an egress port using ACL actions.

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 8 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 47% (including/excluding multicast addresses does not change this). When learning sequential MAC addresses (such as in the RFC2889) the utilization is 99%.

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: 57 bits hash key
          key[56:48] = GID
          key[47:0]  = MAC
      fold count = 8
      returns: 8 bits hash value
  """
  hashval = key & 0b11111111
  hashval = hashval ^ (key>>8)
  hashval = hashval & 0b11111111
  hashval = hashval ^ (key>>16)
  hashval = hashval & 0b11111111
  hashval = hashval ^ (key>>24)
  hashval = hashval & 0b11111111
  hashval = hashval ^ (key>>32)
  hashval = hashval & 0b11111111
  hashval = hashval ^ (key>>40)
  hashval = hashval & 0b11111111
  hashval = hashval ^ (key>>48)
  hashval = hashval & 0b11111111
  hashval = hashval ^ (key>>56)
  hashval = hashval & 0b11111111
  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 & 0x1ff) << 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:380
    key = (380)<< 48 | 0x466162bc84dd
    hashval = calc_l2_hash(key) # the hash value is used as index into the L2 DA Hash T
    assert hashval == 221
```

13.1.2 Hash function for Ingress Configurable ACL 0

The hash function recevies the lookup key created by selecting the fields from the packet determined by the **Ingress Configurable ACL 0 Rules Setup** The lookup key is up to 208 bits wide. 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.

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

```
def calc_confAcl_small0_hash( key ):
  """ key: 208 bits hash key
      fold count = 52
      returns: 4 bits hash value
  """
  hashval = key & 0b1111
  hashval = hashval ^ (key>>4)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>8)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>12)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>16)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>20)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>24)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>28)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>32)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>36)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>40)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>44)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>48)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>52)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>56)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>60)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>64)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>68)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>72)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>76)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>80)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>84)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>88)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>92)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>96)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>100)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>104)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>108)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>112)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>116)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>120)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>124)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>128)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>132)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>136)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>140)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>144)
```

```
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>148)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>152)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>156)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>160)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>164)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>168)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>172)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>176)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>180)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>184)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>188)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>192)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>196)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>200)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>204)
  hashval = hashval & 0b1111
  return hashval

def confAcl_small0_hash( destination_address ):
  """ Calculate index into confAcl_small0 hash table from
      the Destination Address. The parameter must be an integer. """
  key = destination_address & 0xffffffffffffffffffffffffffffffffffffffffffffffffffff
  return calc_confAcl_small0_hash( key )


def calc_confAcl_large0_hash( key ):
  """ key: 208 bits hash key
      fold count = 42
      returns: 5 bits hash value
  """
  hashval = key & 0b11111
  hashval = hashval ^ (key>>5)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>10)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>15)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>20)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>25)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>30)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>35)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>40)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>45)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>50)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>55)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>60)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>65)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>70)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>75)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>80)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>85)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>90)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>95)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>100)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>105)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>110)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>115)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>120)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>125)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>130)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>135)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>140)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>145)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>150)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>155)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>160)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>165)
hashval = hashval & 0b11111
hashval = hashval ^ (key>>170)
hashval = hashval & 0b11111
```

```
  hashval = hashval ^ (key>>175)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>180)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>185)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>190)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>195)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>200)
  hashval = hashval & 0b11111
  hashval = hashval ^ (key>>205)
  hashval = hashval & 0b11111
  return hashval

def confAcl_large0_hash( destination_address ):
  """ Calculate index into confAcl_large0 hash table from
      the Destination Address. The parameter must be an integer. """
  key = destination_address & 0xffffffffffffffffffffffffffffffffffffffffffffffffffff
  return calc_confAcl_large0_hash( key )



def confAcl0_hash_test():
    key = 547849758642249131348928414228049098856428225774588745595 16492
    hashval = confAcl_small0_hash(key)
    assert hashval == 7

    hashval = confAcl_large0_hash(key)
    assert hashval == 14
```

13.1.3 Hash function for Ingress Configurable ACL 1

The hash function recevies the lookup key created by selecting the fields from the packet determined by the **Ingress Configurable ACL 1 Rules Setup** The lookup key is up to 372 bits wide. 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.

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

```
def calc_confAcl_small1_hash( key ):
  """ key: 372 bits hash key
      fold count = 124
      returns: 3 bits hash value
  """
  hashval = key & 0b111
  hashval = hashval ^ (key>>3)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>6)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>9)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>12)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>15)
  hashval = hashval & 0b111
hashval = hashval ^ (key>>18)
hashval = hashval & 0b111
hashval = hashval ^ (key>>21)
hashval = hashval & 0b111
hashval = hashval ^ (key>>24)
hashval = hashval & 0b111
hashval = hashval ^ (key>>27)
hashval = hashval & 0b111
hashval = hashval ^ (key>>30)
hashval = hashval & 0b111
hashval = hashval ^ (key>>33)
hashval = hashval & 0b111
hashval = hashval ^ (key>>36)
hashval = hashval & 0b111
hashval = hashval ^ (key>>39)
hashval = hashval & 0b111
hashval = hashval ^ (key>>42)
hashval = hashval & 0b111
hashval = hashval ^ (key>>45)
hashval = hashval & 0b111
hashval = hashval ^ (key>>48)
hashval = hashval & 0b111
hashval = hashval ^ (key>>51)
hashval = hashval & 0b111
hashval = hashval ^ (key>>54)
hashval = hashval & 0b111
hashval = hashval ^ (key>>57)
hashval = hashval & 0b111
hashval = hashval ^ (key>>60)
hashval = hashval & 0b111
hashval = hashval ^ (key>>63)
hashval = hashval & 0b111
hashval = hashval ^ (key>>66)
hashval = hashval & 0b111
hashval = hashval ^ (key>>69)
hashval = hashval & 0b111
hashval = hashval ^ (key>>72)
hashval = hashval & 0b111
hashval = hashval ^ (key>>75)
hashval = hashval & 0b111
hashval = hashval ^ (key>>78)
hashval = hashval & 0b111
hashval = hashval ^ (key>>81)
hashval = hashval & 0b111
hashval = hashval ^ (key>>84)
hashval = hashval & 0b111
hashval = hashval ^ (key>>87)
hashval = hashval & 0b111
hashval = hashval ^ (key>>90)
hashval = hashval & 0b111
hashval = hashval ^ (key>>93)
hashval = hashval & 0b111
hashval = hashval ^ (key>>96)
hashval = hashval & 0b111
hashval = hashval ^ (key>>99)
hashval = hashval & 0b111
hashval = hashval ^ (key>>102)
hashval = hashval & 0b111
hashval = hashval ^ (key>>105)
hashval = hashval & 0b111
hashval = hashval ^ (key>>108)
hashval = hashval & 0b111
hashval = hashval ^ (key>>111)
hashval = hashval & 0b111
hashval = hashval ^ (key>>114)
hashval = hashval & 0b111
hashval = hashval ^ (key>>117)
hashval = hashval & 0b111
hashval = hashval ^ (key>>120)
hashval = hashval & 0b111
hashval = hashval ^ (key>>123)
hashval = hashval & 0b111
hashval = hashval ^ (key>>126)
hashval = hashval & 0b111
hashval = hashval ^ (key>>129)
hashval = hashval & 0b111
hashval = hashval ^ (key>>132)
hashval = hashval & 0b111
hashval = hashval ^ (key>>135)
hashval = hashval & 0b111
hashval = hashval ^ (key>>138)
hashval = hashval & 0b111
hashval = hashval ^ (key>>141)
hashval = hashval & 0b111
hashval = hashval ^ (key>>144)
hashval = hashval & 0b111
hashval = hashval ^ (key>>147)
hashval = hashval & 0b111
hashval = hashval ^ (key>>150)
hashval = hashval & 0b111
hashval = hashval ^ (key>>153)
hashval = hashval & 0b111
hashval = hashval ^ (key>>156)
hashval = hashval & 0b111
hashval = hashval ^ (key>>159)
hashval = hashval & 0b111
hashval = hashval ^ (key>>162)
hashval = hashval & 0b111
hashval = hashval ^ (key>>165)
hashval = hashval & 0b111
hashval = hashval ^ (key>>168)
hashval = hashval & 0b111
hashval = hashval ^ (key>>171)
hashval = hashval & 0b111
hashval = hashval ^ (key>>174)
hashval = hashval & 0b111
hashval = hashval ^ (key>>177)
hashval = hashval & 0b111
hashval = hashval ^ (key>>180)
hashval = hashval & 0b111
hashval = hashval ^ (key>>183)
hashval = hashval & 0b111
hashval = hashval ^ (key>>186)
hashval = hashval & 0b111
hashval = hashval ^ (key>>189)
hashval = hashval & 0b111
hashval = hashval ^ (key>>192)
hashval = hashval & 0b111
hashval = hashval ^ (key>>195)
hashval = hashval & 0b111
hashval = hashval ^ (key>>198)
hashval = hashval & 0b111
hashval = hashval ^ (key>>201)
hashval = hashval & 0b111
hashval = hashval ^ (key>>204)
hashval = hashval & 0b111
hashval = hashval ^ (key>>207)
hashval = hashval & 0b111
hashval = hashval ^ (key>>210)
hashval = hashval & 0b111
hashval = hashval ^ (key>>213)
hashval = hashval & 0b111
hashval = hashval ^ (key>>216)
hashval = hashval & 0b111
hashval = hashval ^ (key>>219)
hashval = hashval & 0b111
hashval = hashval ^ (key>>222)
hashval = hashval & 0b111
hashval = hashval ^ (key>>225)
hashval = hashval & 0b111
hashval = hashval ^ (key>>228)
hashval = hashval & 0b111
hashval = hashval ^ (key>>231)
hashval = hashval & 0b111
hashval = hashval ^ (key>>234)
hashval = hashval & 0b111
hashval = hashval ^ (key>>237)
hashval = hashval & 0b111
hashval = hashval ^ (key>>240)
hashval = hashval & 0b111
hashval = hashval ^ (key>>243)
hashval = hashval & 0b111
hashval = hashval ^ (key>>246)
hashval = hashval & 0b111
hashval = hashval ^ (key>>249)
hashval = hashval & 0b111
hashval = hashval ^ (key>>252)
hashval = hashval & 0b111
hashval = hashval ^ (key>>255)
hashval = hashval & 0b111
hashval = hashval ^ (key>>258)
hashval = hashval & 0b111
hashval = hashval ^ (key>>261)
hashval = hashval & 0b111
hashval = hashval ^ (key>>264)
hashval = hashval & 0b111
hashval = hashval ^ (key>>267)
hashval = hashval & 0b111
hashval = hashval ^ (key>>270)
hashval = hashval & 0b111
hashval = hashval ^ (key>>273)
hashval = hashval & 0b111
hashval = hashval ^ (key>>276)
hashval = hashval & 0b111
hashval = hashval ^ (key>>279)
hashval = hashval & 0b111
hashval = hashval ^ (key>>282)
hashval = hashval & 0b111
hashval = hashval ^ (key>>285)
hashval = hashval & 0b111
hashval = hashval ^ (key>>288)
hashval = hashval & 0b111
hashval = hashval ^ (key>>291)
hashval = hashval & 0b111
hashval = hashval ^ (key>>294)
hashval = hashval & 0b111
hashval = hashval ^ (key>>297)
hashval = hashval & 0b111
hashval = hashval ^ (key>>300)
hashval = hashval & 0b111
hashval = hashval ^ (key>>303)
hashval = hashval & 0b111
hashval = hashval ^ (key>>306)
hashval = hashval & 0b111
hashval = hashval ^ (key>>309)
hashval = hashval & 0b111
hashval = hashval ^ (key>>312)
hashval = hashval & 0b111
hashval = hashval ^ (key>>315)
hashval = hashval & 0b111
hashval = hashval ^ (key>>318)
hashval = hashval & 0b111
hashval = hashval ^ (key>>321)
hashval = hashval & 0b111
hashval = hashval ^ (key>>324)
hashval = hashval & 0b111
hashval = hashval ^ (key>>327)
hashval = hashval & 0b111
hashval = hashval ^ (key>>330)
hashval = hashval & 0b111
hashval = hashval ^ (key>>333)
hashval = hashval & 0b111
hashval = hashval ^ (key>>336)
hashval = hashval & 0b111
hashval = hashval ^ (key>>339)
hashval = hashval & 0b111
hashval = hashval ^ (key>>342)
hashval = hashval & 0b111
hashval = hashval ^ (key>>345)
hashval = hashval & 0b111
hashval = hashval ^ (key>>348)
hashval = hashval & 0b111
hashval = hashval ^ (key>>351)
hashval = hashval & 0b111
hashval = hashval ^ (key>>354)
hashval = hashval & 0b111
hashval = hashval ^ (key>>357)
hashval = hashval & 0b111
```

```
  hashval = hashval ^ (key>>360)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>363)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>366)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>369)
  hashval = hashval & 0b111
  return hashval

def confAcl_small1_hash( destination_address ):
  """ Calculate index into confAcl_small1 hash table from
      the Destination Address. The parameter must be an integer. """
  key = destination_address & 0xffffffffffffffffffffffffffffffffffffffffffffffffffffffff
  return calc_confAcl_small1_hash( key )


def calc_confAcl_large1_hash( key ):
  """ key: 372 bits hash key
      fold count = 54
      returns: 7 bits hash value
  """
  hashval = key & 0b1111111
  hashval = hashval ^ (key>>7)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>14)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>21)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>28)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>35)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>42)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>49)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>56)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>63)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>70)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>77)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>84)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>91)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>98)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>105)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>112)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>119)
  hashval = hashval & 0b1111111
hashval = hashval ^ (key>>126)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>133)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>140)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>147)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>154)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>161)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>168)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>175)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>182)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>189)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>196)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>203)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>210)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>217)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>224)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>231)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>238)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>245)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>252)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>259)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>266)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>273)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>280)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>287)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>294)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>301)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>308)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>315)
hashval = hashval & 0b1111111
hashval = hashval ^ (key>>322)
```

```
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>329)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>336)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>343)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>350)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>357)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>364)
  hashval = hashval & 0b1111111
  hashval = hashval ^ (key>>371)
  hashval = hashval & 0b1111111
  return hashval

def confAcl_large1_hash( destination_address ):
  """ Calculate index into confAcl_large1 hash table from
      the Destination Address. The parameter must be an integer. """
  key = destination_address & 0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
  return calc_confAcl_large1_hash( key )



def confAcl1_hash_test():
    key = 893451498969389116019371810925526767673135522704478449701464938994221562327680
    hashval = confAcl_small1_hash(key)
    assert hashval == 4

    hashval = confAcl_large1_hash(key)
    assert hashval == 11
```

13.1.4 Hash function for Egress Vlan Translation

The hash function receives the outermost VID of the modified packet at egress, the egress port number, along with the VLAN Ethernet type (C or S tag). 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.

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

```
def calc_egressVlanTranslation_small_hash( outermostVidType,
                                           outermostVid,
                                           dstPort ):
  """ key: 17 bits hash key
      fold count = 6
      returns: 3 bits hash value
  """
  key = 0
  key = key << 1 | (outermostVidType & 0x1)
  key = key << 12 | (outermostVid & 0xfff)
  key = key << 4 | (dstPort & 0xf)
  hashval = key & 0b111
  hashval = hashval ^ (key>>3)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>6)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>9)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>12)
  hashval = hashval & 0b111
  hashval = hashval ^ (key>>15)
  hashval = hashval & 0b111
  return hashval
def egressVlanTranslation_small_hash( outermostVidType,
                                      outermostVid,
                                      dstPort ):
  """ Calculate index into egressVlanTranslation_small hash table from
      the different fields. The parameter must be an integer. """

  return calc_egressVlanTranslation_small_hash( outermostVidType=outermostVidType,
                                                outermostVid=outermostVid,
                                                dstPort=dstPort )



def calc_egressVlanTranslation_large_hash( outermostVidType,
                                           outermostVid,
                                           dstPort ):
  """ key: 17 bits hash key
      fold count = 5
      returns: 4 bits hash value
  """
  key = 0
  key = key << 1 | (outermostVidType & 0x1)
  key = key << 12 | (outermostVid & 0xfff)
  key = key << 4 | (dstPort & 0xf)
  hashval = key & 0b1111
  hashval = hashval ^ (key>>4)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>8)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>12)
  hashval = hashval & 0b1111
  hashval = hashval ^ (key>>16)
  hashval = hashval & 0b1111
  return hashval
def egressVlanTranslation_large_hash( outermostVidType,
                                      outermostVid,
                                      dstPort ):
  """ Calculate index into egressVlanTranslation_large hash table from
      the different fields. The parameter must be an integer. """

  return calc_egressVlanTranslation_large_hash( outermostVidType=outermostVidType,
                                                outermostVid=outermostVid,
                                                dstPort=dstPort )



def egressVlanTranslation_hash_test():
    dstPort = 14
    outermostVid = 274
    outermostVidType = 1
```

```
hashval = egressVlanTranslation_small_hash( outermostVidType=outermostVidType,
                                            outermostVid=outermostVid,
                                            dstPort=dstPort)
assert hashval == 4

hashval = egressVlanTranslation_large_hash( outermostVidType=outermostVidType,
                                            outermostVid=outermostVid,
                                            dstPort=dstPort)
assert hashval == 13
```

Chapter 14

D-left Lookup

D-left is a hash table search algorithm that reduces the risk of hash collisions by using two hash tables each indexed by a separate hash key.

This implementation uses two hash tables, one smaller and one larger, combined with a synthesized TCAM to resolve hash collisions. This is shown in figure 14.1.

The hash search is done by taking a hash key and calculating two hashes from that. The two hash values are used as index into the small and large hash tables.

Each table has a number of buckets for each hash index. All buckets for the selected index are read out in parallel. The hash key is then compared with the compareData from each bucket. There is a hit if one of the buckets compareData matches the hash key. If multiple buckets matches then the highest numbered bucket is used.

This is done in parallel for both the small and the large table.

In addition the hash key is also searched in the TCAM. In the TCAM search all entries are compared with the hash and if there are multiple matches then the lowest numbered entry is used.

Since a single search can result in multiple hits in all three tables there is configuration that selects which table shall be used in this case.

The two hash tables have separate masks which allows some bits to be masked away. For the TCAM there is a mask per entry.

14.1 Functions using D-left

The following functions use D-left Lookup.

14.1.1 Egress VLAN Translation

The Egress VLAN Translation table:

- The hash tables are **Egress VLAN Translation Small Table** and **Egress VLAN Translation Large Table**. Each of the the hash tables has 2 buckets for each hash index.
- The search data/hash key is the egress port, the outermost VID and the outermost VID Type, a C-tag (0) or S-tag (1).
- The TCAM is **Egress VLAN Translation TCAM**.
- The hash functions used to index the **Egress VLAN Translation Small Table** and **Egress VLAN Translation Large Table** are described in section Hash function for Egress VLAN Translation.
- The masks for the hash tables are **Egress VLAN Translation Search Mask**.
- The configuration for resolving multiple hits is in **Egress VLAN Translation Selection**.

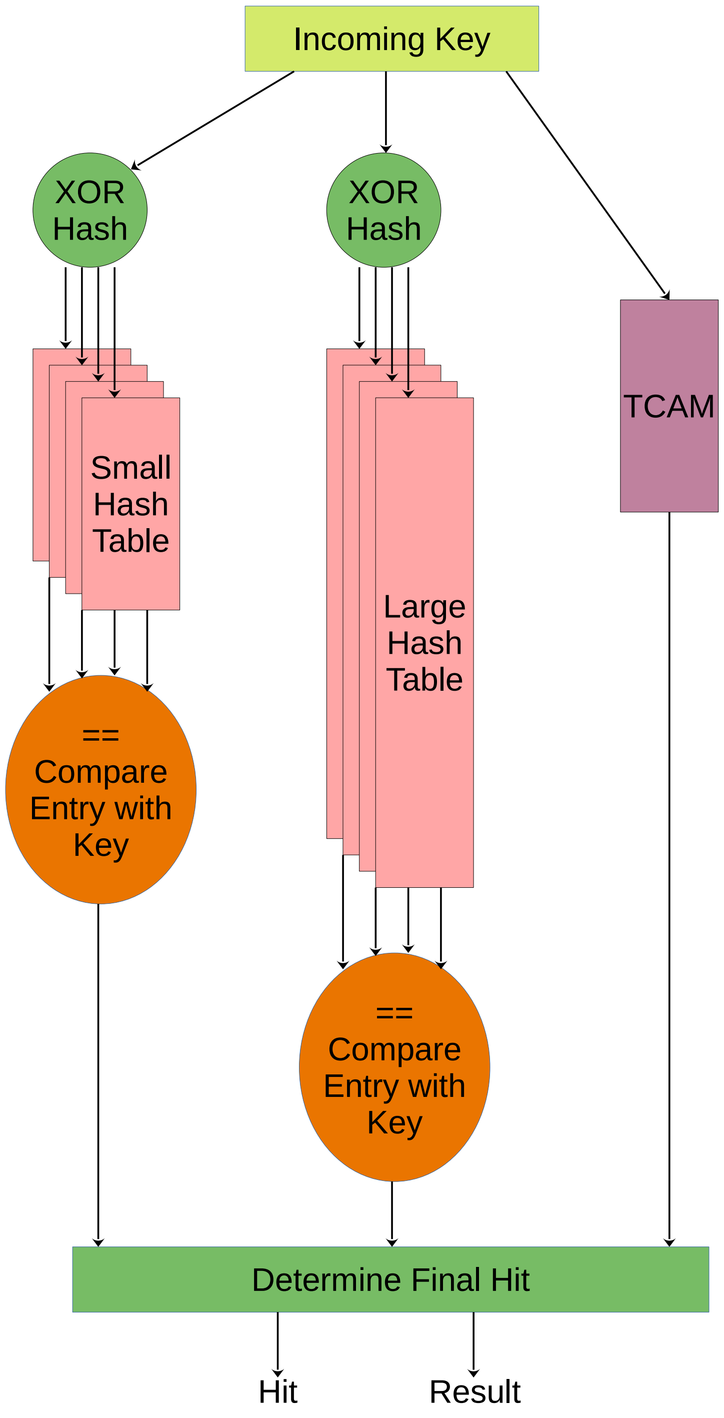

Figure 14.1: D-left Function

- While the hash tables stores the answer in the same memories as the lookup key, the TCAM has a seperate table holding the answer: **Egress VLAN Translation TCAM Answer**.

14.1.2 Ingress Configurable ACL

The ingress configurable ACL is setup by using the following registers and tables.

- The search data/hash key is the selected packet header fields (see Selectable Packet Fields).
- Hash tables

  - The hash functions used to index the hash tables are described in section Hash function for Configurable ACL.
  - **Ingress Configurable ACL 0 Small Table**
  - **Ingress Configurable ACL 0 Large Table**
  - **Ingress Configurable ACL 1 Small Table**
  - **Ingress Configurable ACL 1 Large Table**
- TCAM
  - **Ingress Configurable ACL 0 TCAM**
  - **Ingress Configurable ACL 1 TCAM**
- Masks for the hash tables
  - **Ingress Configurable ACL 0 Search Mask**
  - **Ingress Configurable ACL 1 Search Mask**
- Configuration for resolving multiple hits
  - **Ingress Configurable ACL 0 Selection**
  - **Ingress Configurable ACL 1 Selection**
- The ACL actions are stored in the hash tables but the actions for TCAM hits are stored in a separate tables
  - **Ingress Configurable ACL 0 TCAM Answer**
  - **Ingress Configurable ACL 1 TCAM Answer**

Chapter 15

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.

15.1 L2 Forwarding Information Base (FIB)

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

15.1.1 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.

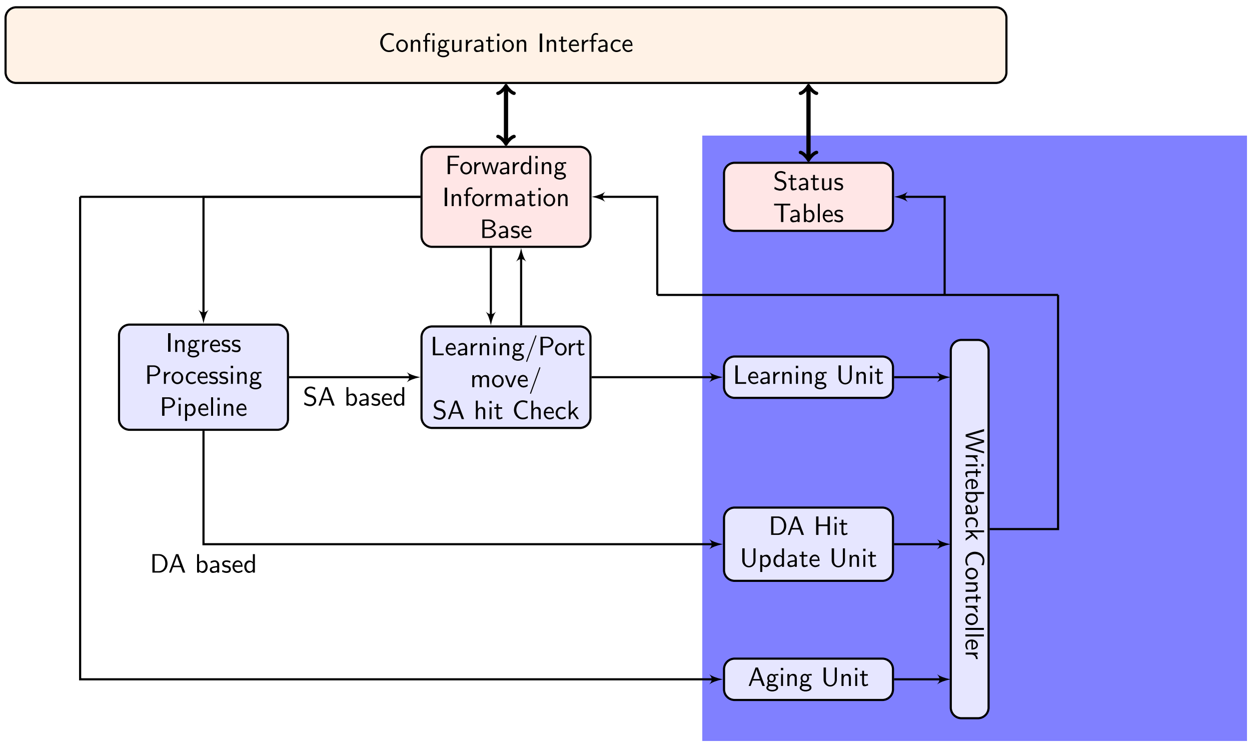

Figure 15.1: Learning and Aging Engine

   (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 57-bit hash key and the hash function returns a 8-bit hash value.

15.1.2 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.

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 15.1 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. Unexpected behavior will occur if the tables do not have the same content.

15.1.3 Hash Collision Accommodation

In order to solve hash collisions, the **L2 DA Hash Lookup Table** has 4 buckets each with 256 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 10-bit hit index with the hash value in the lower 8 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 + 1,024).

15.2 Hardware Learning and Aging

15.2.1 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.

15.2.2 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.
  - If the packet hit in the **Configurable ACL Engine** with **noLearning** enabled.
- 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**.

[1]Check all reason codes in Table 30.2
[2]See more in Chapter Spanning Tree.

15.2.3 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 15.1. 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.

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

Table 15.1: Hardware Aging Operations

15.2.4 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 15.1, 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.

15.3 Software Learning and Aging

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

15.3.1 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

[3]The system ticks are described in Chapter Tick.

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.

15.3.2 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 16

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 30

16.1 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.

16.2 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.

16.3 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 17

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

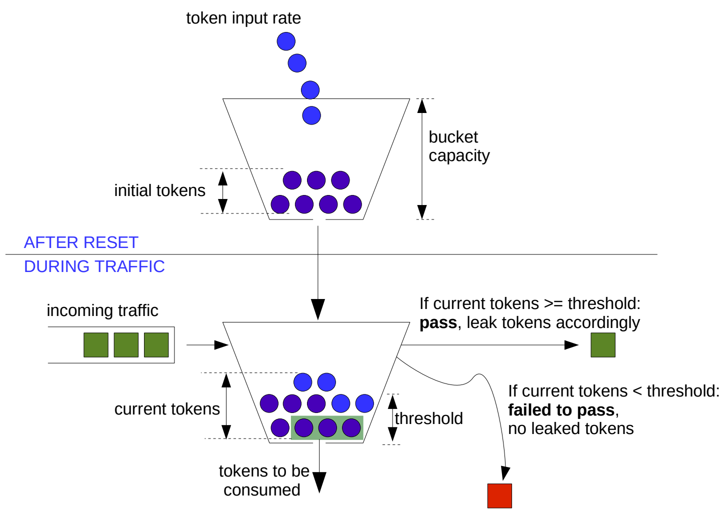
token input rate
bucket capacity
initial tokens
AFTER RESET
DURING TRAFFIC
incoming traffic
If current tokens >= threshold: **pass**, leak tokens accordingly
current tokens
threshold
If current tokens < threshold: **failed to pass**, no leaked tokens
tokens to be consumed

Figure 17.1: General Token Bucket Illustration

Figure 17.1 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:

- ingress rate control
- multicast broadcast storm control
- queue shaper
- prio shaper
- egress port shaper

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 or five PTP ticks.

Chapter 18

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.

Each egress queue has a *priority* (or prio) attribute, ranging from zero to seven. There are no limitations to how the priorities are assiged. All egress queues may have the same priority, or they may all have different priorities (if there are enough priorities to go around). If at all possible, an egress queue with a higher[2] priority will always get to output a packet before a queue with a lower priority. Egress queues with the same priority will be selected in a round robin manner by the DWRR scheduler.

The egress queue is determined by the ingress packet processing. If a packet is forwarded to multiple egress ports, each packet instance will have a separate process of egress queue assignment based on its target egress port.

18.1 Determine Egress Queue

Figure 18.1 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. Hit in the **L4 Port Range to Queue Assignment** with **force** set to one
2. Hit in the **IP Address To Queue Assignment** with **force** set to one.
3. Hit in the **VID to Queue Assignment** with **force** set to one
4. Hit in the **DA or SA MAC to Queue Assignment** with **force** set to one
5. Hit in the **L4 Protocol to Queue Assignment** with **force** set to one
6. Hit in the **Ethernet Type to Queue Assignment** with **force** set to one
7. **Configurable ACL Engine** has a forceQueue action enabled.

[1]That is unless the packet is to be dropped, because then the pointer is instead added to the end of the throw queue.
[2]Priorities are numbered backward, so zero is the highest priority

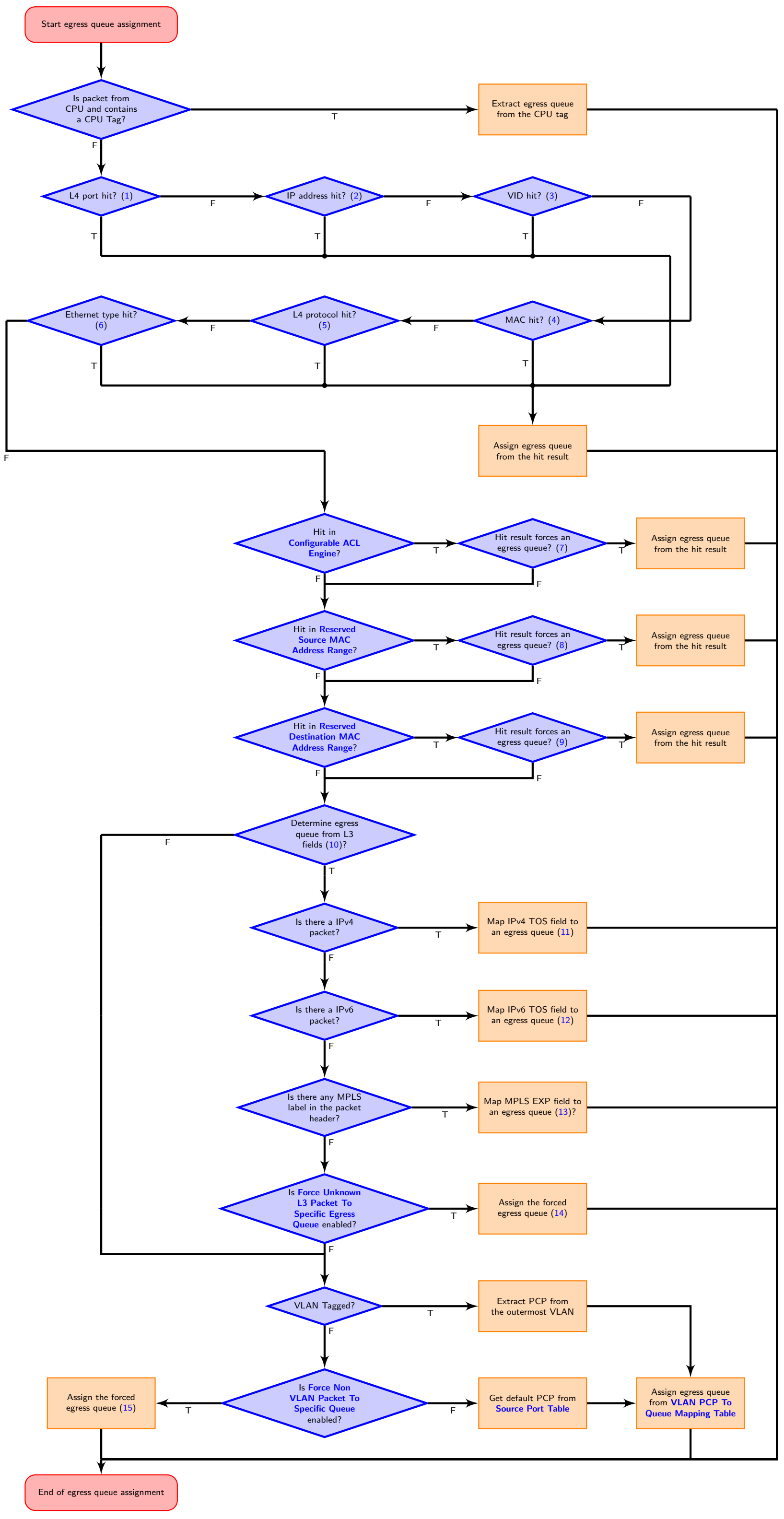
Start egress queue assignment
Is packet from CPU and contains a CPU Tag?
Extract egress queue from the CPU tag
L4 port hit? (1)
IP address hit? (2)
VID hit? (3)
Ethernet type hit? (6)
L4 protocol hit? (5)
MAC hit? (4)
Assign egress queue from the hit result
Hit in Configurable ACL Engine?
Hit result forces an egress queue? (7)
Hit in Reserved Source MAC Address Range?
Hit result forces an egress queue? (8)
Hit in Reserved Destination MAC Address Range?
Hit result forces an egress queue? (9)
Determine egress queue from L3 fields (10)?
Is there a IPv4 packet?
Map IPv4 TOS field to an egress queue (11)
Is there a IPv6 packet?
Map IPv6 TOS field to an egress queue (12)
Is there any MPLS label in the packet header?
Map MPLS EXP field to an egress queue (13)?
Is Force Unknown L3 Packet To Specific Egress Queue enabled?
Assign the forced egress queue (14)
VLAN Tagged?
Extract PCP from the outermost VLAN
Is Force Non VLAN Packet To Specific Queue enabled?
Assign the forced egress queue (15)
Get default PCP from Source Port Table
Assign egress queue from VLAN PCP To Queue Mapping Table
End of egress queue assignment
T
F

Figure 18.1: Egress Queue Selection Diagram. This process is done individually for each egress port.

8. **forceQueue** in **Reserved Source MAC Address Range**
9. **forceQueue** in **Reserved Destination MAC Address Range**
10. **Egress Queue Priority Selection**
11. **IPv4 TOS Field To Egress Queue Mapping Table**
12. **IPv6 Class of Service Field To Egress Queue Mapping Table**
13. **MPLS EXP Field To Egress Queue Mapping Table**
14. **eQueue** in **Force Unknown L3 Packet To Specific Egress Queue**
15. **forceQueue** in **Force Non VLAN Packet To Specific Queue**

This process is completed separately for each egress port. The input to the process can come from:

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

The available fields for comparision are:

- L4 source or destination port
- IPv4 or IPv6 source or destination
- VID
- DA or SA MAC
- Ethernet type

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.

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

18.2.1 Remap Egress Queue to Packet Headers

This core supports remapping determined egress queues to outgoing packets' headers.

- 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.

18.3 Priority Mapping

Each queue is mapped to one of eight egress priorities in the **Map Queue to Priority** register. Thus each priority will have between none and all queues as members. The priority mapping affects the scheduling of the packets. See Section 18.7, below for the details.

The priorities are ranked in descending order, from the highest priority (zero), to the lowest (seven).

Note that the priority mapping must not be changed for any queue that has packets queued. Doing so would make the ERM counters irrevocably corrupted, necessitating a reset for the core to continue normal operation.

18.4 Timed Gates For Egress Queues

For each egress queue in the buffer memory, the core features a time-aware egress transmission gate to selectively expose the queue status to subsequent scheduling modules. Egress transmission gate provides a mechanism that can guarantee the transmission of enqueued packets in a specific egress queue within a specific time window. In order to synchronize the gate status over the entire network, the time which the gate execution is based on shall be accurate. See more details in **PTP Tick**.

18.4.1 Initialization

Egress transmission gate is turned off by default, to bring it up on an egress port for the first time after reset, the following steps must be done:

1. Make sure that the PTP Tick is enabled in the **PTP Tick Select** register.
2. Write 1 to **Egress Transmission Gate Enabled**.
3. Set up **Egress Transmission Gate Configuration**.
4. Set up in total **adminControlListLength** entries in the **Egress Transmission Gate List**, starting from the **adminStartAddr** in ascending order.
5. Only write 1 to **Egress Transmission Gate Update** when the above steps are done, failure to follow the sequence may result in invalid configurations.

18.4.2 Admin Configurations

While configuring the egress transmission gate over the network, it is important to execute the gate operations under an accurate global time. Considerations shall be taken among multiple settings to make the transmission gate work properly.

- Hardware Time:
  Egress Transmission Gate Current Time represents the current hardware time, the granularity is based on **Egress Transmission Gate Base Tick** which is derived from the master PTP tick. Notice that the granularity shall not be changed when the transmission gate is operating.

- Base Time:
  adminBaseTime determines the time at which the **Egress Transmission Gate Configuration** is loaded into the hardware and affects the start time of each subsequent gate cycle. The base time has the same granularity with the hardware time, software determines the correspondence between the hardware time and the real time by reading **Egress Transmission Gate Current Time** and comparing it with its own accurate time. Knowledge of the correspondence and the hardware time granularity allows software to calculate the base time from a real time.

  Even in the initialization phase, the base time needs to be a future time for the hardware, the core itself can not adjust the configuration loading time based on a past time.

- Pending Update:
  Once **Egress Transmission Gate Update** is written by 1 from the software, a pending process is triggered and the transmission gate waits for the base time to occur. The admin configurations are loaded as hardware operation configurations when the hardware current time reaches the based time.

  - **`operBaseTime`** = adminBaseTime
  - **`operTick`** = adminTick
  - **`operCycleTime`** = adminCycleTime
  - **`operCycleTimeExtension`** = adminCycleTimeExtension
  - **`operControlListLength`** = adminControlListLength
  - **`operStartAddr`** = adminStartAddr

  Egress Transmission Gate Update Status is pulled high during the pending period, and any modification on the configuration table will affect future loading. When the admin configurations are loaded, the status register is pulled low by the hardware and modifications on the configuration table will not affect the current gate executions.

- Cycle Time, Time Interval and List Length:
  adminCycleTime has the same granularity as the hardware time and the base time. It determines the time for one gate cycle and it is used to restart a new gate cycle iteration again when the hardware time meets **`operBaseTime`** + N * **`operCycleTime`** where N is an integer.

  The maximum execution time to complete the gate list can be calculated by the sum of all **timeInterval** among the given **`operControlListLength`**. The granularity of the time interval is selected by **`operTick`** which can be different from the hardware time. If the maximum execution time is longer than the cycle time, uncompleted list entries will be terminated accordingly. If the maximum execution time is less than the cycle time, the gate status from the last given entry will be retained until the next gate cycle.

- Time Interval and Jitter:
  Theoretically, if the cycle time is the same as the maximum execution time, a new gate cycle will start exactly when the last entry in the given list is completed. However, the actual execution time of each entry is subject to delay jitter caused by the port scheduler. When the granularity of the time interval is close to the core clock, the actual time required may be slightly longer than the theoretical time. In a scenario that the last entry in the given gate list is used as guard band (close all queues so that the ongoing packet can be transmitted before the start of the next gate cycle), this jitter will be automatically eliminated. It is important to note that the time interval should not be less than the maximum jitter caused by the port scheduler, as this may result in the corresponding gate list entry not being executed.

- List Start Address:
  Every gate cycle starts at **`operStartAddr`** in the gate list, when the last entry in the whole **Egress**

Transmission Gate List is executed in the middle of one gate cycle, the execution will continue from entry 0 in the gate list.

18.4.3 Runtime Reconfiguration

Once the egress transmission gate has done the bring up sequence for the first time, it can be turned on or off by only using the **Egress Transmission Gate Enabled** register. If a runtime reconfiguration is needed, apply the same operating sequence as the bring up stage.

18.4.4 Impact of Output Disable

The result from the egress transmission gate will be checked by **Output Disable** again. In order to only let the egress transmission gate open and close egress queues, an egress port with transmission gate turned on shall not use the **Output Disable** functionality.

18.5 Shapers

For a queue to be eligable for sending a packet there has to be a packet available in the queue and the average bandwidth for the queue, as measured by the token buckets in the queue shaper, has to be below the threshold set up in the **Queue Shaper Rate Configuration** registers.

Additionaly the average bandwidth of the priority to which the queue is mapped has to be below the threshold set up in the **Prio Shaper Rate Configuration** registers.

18.5.1 Queue Shaper

The egress queue rates are shaped by token buckets configured in the **Queue Shaper Rate Configuration** registers. While the token bucket level (**Queue Shaper Current Size**) is below the threshold configured in the **Queue Shaper Bucket Threshold Configuration** register, no new packets are scheduled for the corresponding egress queue. Ongoing packets are not affected by the shaping bucket status.

The queue shapers are enabled using the **Queue Shaper Enable** register, and the saturation level of the queue shaper buckets is controlled by the **Queue Shaper Bucket Capacity Configuration** register.

When you enable the queue shapers you must make sure to also enable the PTP Tick using the **PTP Tick Select** register, because the PTP Tick is disabled by default.

18.5.2 Prio Shaper

The egress prio rates are shaped by token buckets configured in the **Prio Shaper Rate Configuration** registers. While the token bucket level (**Prio Shaper Current Size**) is below the threshold configured in the **Prio Shaper Bucket Threshold Configuration** register, no new packets are scheduled for the corresponding egress prio. Ongoing packets are not affected by the shaping bucket status.

The prio shapers are enabled using the **Prio Shaper Enable** register, and the saturation level of the prio shaper buckets is controlled by the **Prio Shaper Bucket Capacity Configuration** register.

When you enable the prio shapers you must make sure to also enable the PTP Tick using the **PTP Tick Select** register, because the PTP Tick is disabled by default.

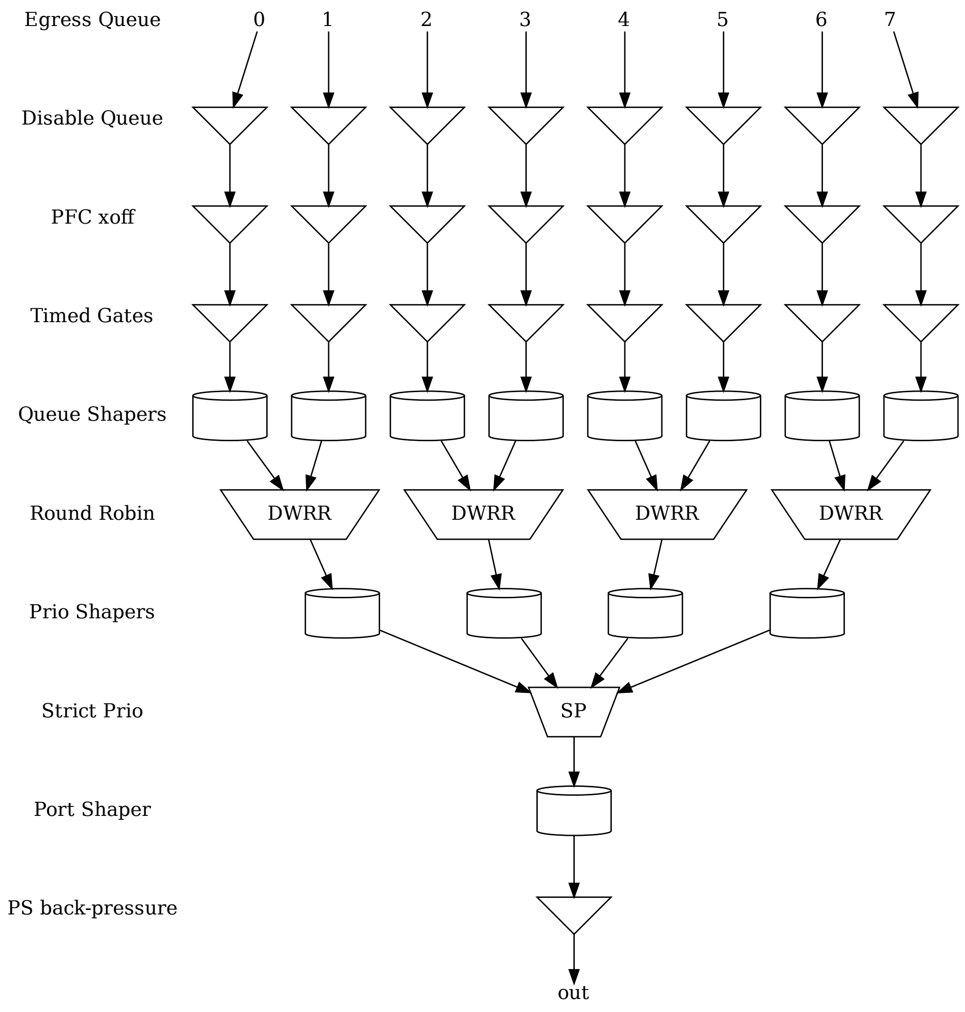

Figure 18.2: Egress Queue Scheduling example. Here using half the priorities, with two queues mapped to each.

18.6 Scheduling

The egress queue scheduling is accomplished by a combination of strict priority schedulers for the priorities and round robin queue schedulers for the queues mapped to the same priority. A visual representation of this is can be found in Figure 18.2. This figure is an example where half the priorities are used and two queues map to each priority[3].

For a priority to be allowed to output a packet it must have mapped queues with available packets. It must also:

- be allowed to send by the prio shaper
- not be paused
- not be halted

From the priorities getting through the above needle's eye the highest priority is selected, and then the available queues mapped to that priority are selected by a byte-based deficit weighted round robin scheduler (described below).

18.7 DWRR Scheduler

The DWRR scheduler only acts on queues mapped to the same priority. Within each group of such queues it selects the most appropriate queue to output by comparing the number of bytes output for each queue with the weights set up for the queues.

This is accomplished using one byte counting bucket per queue and port. The non-empty queue with the highest bucket count in the group is selected. Bytes are subtracted from the corresponding bucket when a packet is sent out. Whenever the value in a bucket goes below the value configured in the **threshold** field of the **DWRR Bucket Misc Configuration** register, the buckets for all the queues belonging to the same priority will be replenished. The number of bytes added to each bucket is **weight** $<< X$, where weight is taken from the **DWRR Weight Configuration** register, and X is a multiplier (for all queues) that is calculated to make sure that at least one cell worth of bytes is added to the queue that went below the threshold.

$X = max(0, highestSetBit(cellBytes) - highestSetBit(weigth))$

If a queue has no data to send, its bucket will eventually saturate at the cap set in the **DWRR Bucket Capacity Configuration** register.

The value in the **ifg** field of the **DWRR Bucket Misc Configuration** is additionally subtracted from the buckets for each packet.

18.8 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

[3]So other similar diagrams would result with different settings in the **Map Queue to Priority** register.

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.

18.9 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 19

Packet Coloring

19.1 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 19.1.

Color	**Code**
Green	0
Yellow	1
Red	2

Table 19.1: 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 18.1.

1. **Configurable ACL Engine** has a forceColor action enabled.
2. **forceColor** in **Reserved Source MAC Address Range**
3. **forceColor** in **Reserved Destination MAC Address Range**
4. **colorFromL3** in **Source Port Table**
5. **IPv4 TOS Field To Packet Color Mapping Table**
6. **IPv6 Class of Service Field To Packet Color Mapping Table**
7. **MPLS EXP Field To Packet Color Mapping Table**
8. **forceColor** in **Force Unknown L3 Packet To Specific Color**
9. **forceColor** in **Force Non VLAN Packet To Specific Color**

A diagram in Figure 19.1 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.

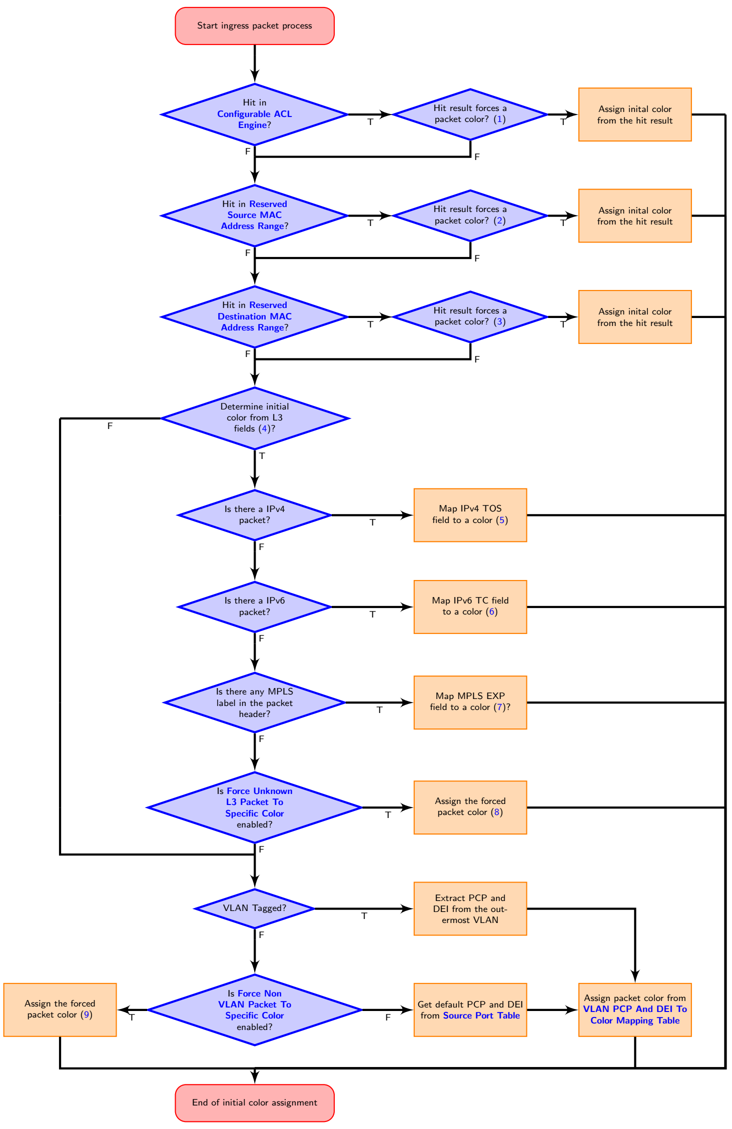
Start ingress packet process
Hit in Configurable ACL Engine?
Hit result forces a packet color? (1)
Assign inital color from the hit result
Hit in Reserved Source MAC Address Range?
Hit result forces a packet color? (2)
Assign inital color from the hit result
Hit in Reserved Destination MAC Address Range?
Hit result forces a packet color? (3)
Assign inital color from the hit result
Determine initial color from L3 fields (4)?
Is there a IPv4 packet?
Map IPv4 TOS field to a color (5)
Is there a IPv6 packet?
Map IPv6 TC field to a color (6)
Is there any MPLS label in the packet header?
Map MPLS EXP field to a color (7)?
Is Force Unknown L3 Packet To Specific Color enabled?
Assign the forced packet color (8)
VLAN Tagged?
Extract PCP and DEI from the outermost VLAN
Is Force Non VLAN Packet To Specific Color enabled?
Assign the forced packet color (9)
Get default PCP and DEI from Source Port Table
Assign packet color from VLAN PCP And DEI To Color Mapping Table
End of initial color assignment
T
F

Figure 19.1: Packet Initial Color Selection Diagram

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.

19.2 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 20

Admission Control

20.1 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.

20.1.1 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 20.1. 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. ACL rules:
   Hit in the **Configurable ACL Engine**.

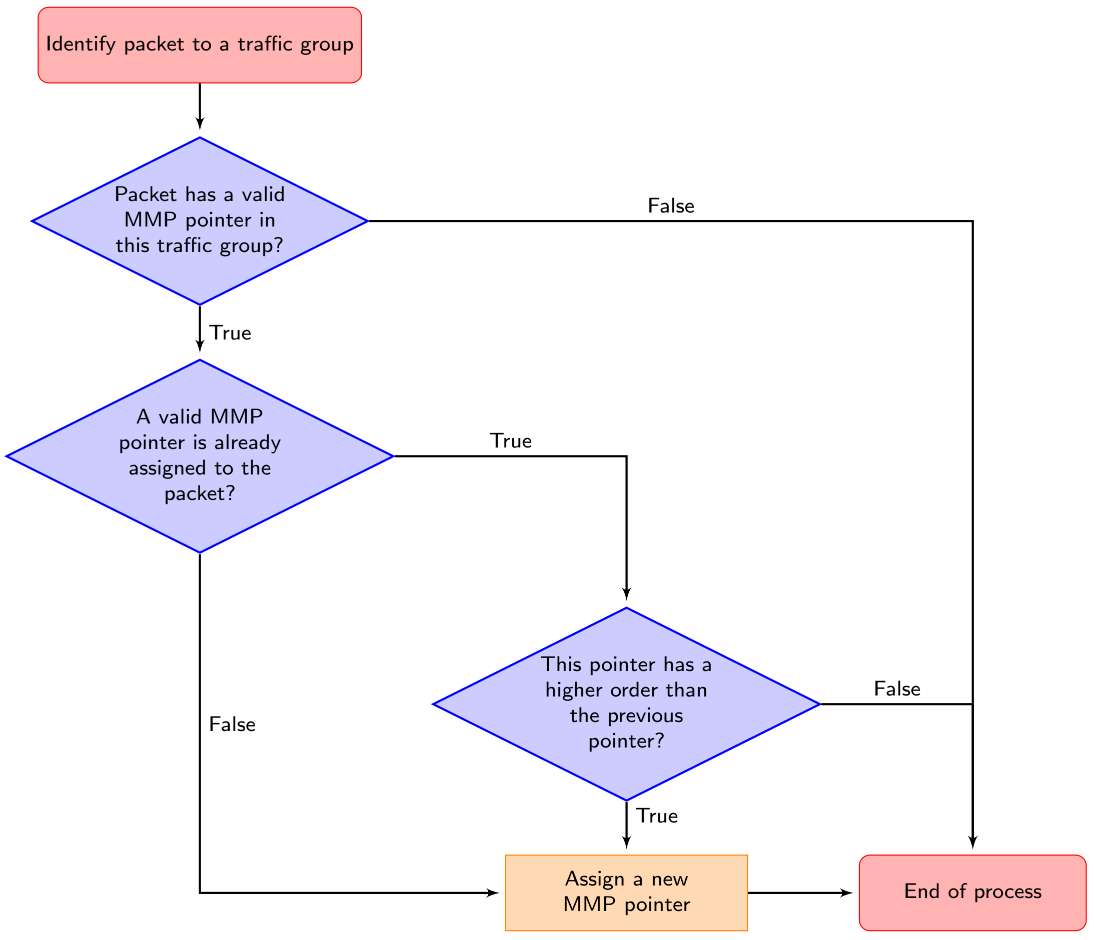
Identify packet to a traffic group
Packet has a valid MMP pointer in this traffic group?
False
True
A valid MMP pointer is already assigned to the packet?
True
False
This pointer has a higher order than the previous pointer?
False
True
Assign a new MMP pointer
End of process

Figure 20.1: MMP pointer Selection Diagram

5. Ingress VID:
   Lookup in **VLAN Table** based on the **ingress VID**.
6. PSFP:
   Hit in **Stream Filter Lookup Table**

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.

20.2 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 32 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 20.1. 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 20.1: Rate Configuration Example (Assume tickFreqList = [1MHz, 100KHz, 10KHz, 1KHz, 100Hz])

Chapter 21

Per-Stream Filtering and Policing

Per-Stream Filtering and Policing (PSFP) performs a series of filtering and policing operations on incoming packets to protect the network from overload. PSFP operates through three sequential modules, including Stream Filter, Stream Gate, and optional Flow Meter. Note that if a packet is dropped at a certain stage, subsequent modules will no longer process the packet. For example, a packet that is discarded in the stream filter will no longer be inspected by the stream gate and the flow meter.

In this core, PSFP is implemented as part of the post-ingress processing, PSFP will not act on any corrupted packets or packets that have been discarded by the forwarding logic.

21.1 Stream Filter

Due to the sequential execution of PSFP, packets need to first match the stream filter before initiating filtering and policing operations. The stream filter relies on two parameters to hit an entry in the **Stream Filter Lookup Table** and determine whether a PSFP process is needed for the packet:

1. Stream Handle: Defined in IEEE Std 802.1CB-2017, in this core the stream handle is implemented by the ACL.
2. Priority: For VLAN-tagged packets, priority is derived from the Priority Code Point (PCP) in the VLAN header. For non-VLAN packets, priority comes from the **defaultPcp** field in the **Source Port Table**.

Wildcard Support

Both of these parameters support wildcards, and when there are multiple hits, the result of the first hit is returned. This means that the last entry in the **Stream Filter Lookup Table** can use two wildcards for both the stream handle and the priority to implement the default PSFP action so that all traffic through the core are inspected by PSFP. If a packet does not hit any entry in the stream filter, it will bypass the PSFP checks.

Result on Hit

When a packet hits an entry in the **Stream Filter Lookup Table**, it returns three IDs / pointers:

- Stream Filter ID / Max SDU Filter Pointer
- Stream Gate ID / Ingress Transmission Gate Pointer
- Flow Meter ID / Ingress MMP Pointer (optional)

21.1.1 Max SDU Filter

A stream filter ID can be used to point to an entry in the **Max SDU Filter** to perform a packet length check and discard packets exceeding a specified threshold (**maxSDU**). Note that if a packet is discarded by the max SDU filter, it will not consume the granted L2 payload count (**maxMSDU**) in the subsequent stream gate.

21.2 Stream Gate / Ingress Transmission Gate

A stream gate ID points to an ingress transmission gate which is configured in the **Ingress Transmission Gate Configuration** and allows periodic open and closed status changes based on the **Ingress Transmission Gate List**.

- Gate Closed: Packets are discarded when the gate is closed.
- Gate Open: Packets pass through the open gate with an option to alter the previously assigned egress queue. Meanwhile an open gate can regulate the maximum allowable L2 payload (**maxMSDU**) within the current time interval, and it will be considered closed when the total octets exceeds this limit.

21.2.1 Configuration and Constraints

Regarding specific gate control list configurations and global time synchronization, please refer to Timed Gates For Egress Queues. **Ingress Transmission Gate Current Time** represents the hardware time for PSFP stream gates and related configuration registers are:

- **Ingress Transmission Gate Base Tick**
- **Ingress Transmission Gate Configuration**
- **Ingress Transmission Gate Enabled**
- **Ingress Transmission Gate Update**

Each gate ID has two types of status that can be observed from the conf bus:

- **Ingress Transmission Gate Update Status** shows the pending configuration updates.
- **Ingress Transmission Gate Current Status** shows the current gate list entry.

Similar to the refresh of each egress gate status, the background process that refreshes the status of each ingress stream gate ID is also executed periodically. For a specific stream gate ID, the time interval should be much larger than the refresh period.

Due to the fact that packets in the core are divided into cells, the gate operation to do the egress queue update will take place in the first cell, while packet length checks will occur in the last cell of the packet. The design allows the gate state to change at most once during the transmission of a packet. More than one change in state will render the additional state between the first and last cell unobservable.

Additionally, if a packet passes through two consecutively open states of a gate and **maxMSDU** check is enabled, the packet will bypass the L2 payload check of the first open state and undergo it at the second open state. If a packet experiences a change from an open to closed state or vice versa, the packet will be discarded.

21.3 Flow Meter

Optionally, PSFP can be configured to associate with a flow meter ID. In this design, the functionality of the flow meter shares the same set of MMPs from ingress admission control. MMP and its configuration details are in Meter-Marker-Policer.

A packet can only have one MMP pointer so a PSFP flow metered packet can not be combined with other ingress admission control policing. Reserve the maximum **mmpOrder** for PSFP to ensure that the flow meter ID takes precedence over other ingress admission control traffic groups.

21.4 Stream Blocking

In PSFP, each of the three modules has an optional stream blocking mechanism. This means that when packet drop occurs, the module is placed in a blocked state, discarding all traffic sent to it after the drop decision. This relies on system level configuration to ensure that all traffic correctly passes through this core. When packet drop occurs, software intervention is required to restore the operation of PSFP.

Each blocking mechanism consists of a pair of Booleans:

- Max SDU Filter
  - **blockingEn** in **Max SDU Filter** and **Max SDU Filter Blocking**
- Stream Gate
  - **invalidRxBlockingEn** in **Stream Gate Blocking Enable** and **Stream Gate Invalid RX Blocking**
  - **maxMsduBlockingEn** in **Stream Gate Blocking Enable** and **Stream Gate Max MSDU Blocking**
- Flow Meter
  - **Ingress Admission Control Mark All Red Enable** and **Ingress Admission Control Mark All Red**

When both of the two parameters are 1 then the corresponding stream filter ID / stream gate ID / flow meter ID is put under a blocking state until software clears the values.

21.5 Statistics

The statistics of PSFP are based on the stream filter ID. This design provides the following statistics:

- **PSFP Matching Frame Counter**
- **PSFP Passing SDU Counter**
- **PSFP Not Passing SDU Counter**
- **PSFP Passing Frame Counter**
- **PSFP Not Passing Frame Counter**
- **PSFP Red Frames Counter**

For a given stream filter ID, the sum of **PSFP Passing SDU Counter** and **PSFP Not Passing SDU Counter** equals **PSFP Matching Frame Counter**. The sum of **PSFP Passing Frame Counter** and **PSFP Not Passing Frame Counter** equals **PSFP Passing SDU Counter**.

Chapter 22

Frame Replication and Elimination for Reliability

22.1 Enabling FRER

This core supports IEEE 802.1CB Frame Replication and Elimination for Reliability (FRER). To enable FRER for this IP, you need to first define ACL rules for the relevant streams with action to obtain the corresponding Stream Handle (**`streamHandle`**) before proceeding with further processing. This core in a network can act as either a frame replication node or a frame elimination node for a specific stream. Due to the involvement of multiple register configurations in FRER and its typical cooperation with other network nodes, careful consideration is needed when configuring FRER. This document primarily focuses on the configurations of the two modes inside the switch, the end-to-end functional configuration of FRER at the system level is beyond the scope of this document.

Once a packet is assigned a **`streamHandle`**, the **Stream Handle To FRER Mapping Table** associates it with a FRER ID (**`frerId`**). In the default configuration, all stream handles are linked to FRER ID 0 and are not activated. For a given **`frerId`**, set the **mode** field in the **FRER Configuration** to enable the generation mode for frame replications or the recovery mode for frame eliminations.

22.2 Generation Mode

- When **`frerId`** is set to generation mode, normally the **`frerId`** is associated with only one **`streamHandle`**, which is used to identify compound streams that require FRER.
- This core supports adding a Redundancy tag (R-TAG) when packets under the generation mode is transmitted out from the core. The sequence number in the R-TAG is from the **FRER Sequence Number** register, and each time an R-TAG is added, the corresponding sequence number for the next received packet increments by 1. To reset the sequence number, users can write the new value to the register directly. When the sequence number reaches the maximum value of 65535, the next sequence number restarts from 0.
- Configure the L2 forwarding function to split the compound stream to member streams to multiple egress ports. This can be achieved through L2 multicast setup in the FIB, flooding based on VLAN membership or other possible ways. The principle is that the ACL rule used to identify the compound stream shall be synchronized with how the forwarding decision is made.
- If an FRER network is using reserved VLAN IDs to represent member streams, this core can utilize **Egress VLAN Translation TCAM** to assign specific VLAN IDs to different member streams. Note that the R-TAG will be placed after all VLANs recognized by this core.

22.3 Recovery Mode

- When an `frerId` is set to recovery mode, this `frerId` usually binds to more than one `streamHandle`. Each `streamHandle` represents a member stream and the source port is typically part of the ACL rule in order to identify member streams from different source ports. This helps when previous nodes have not made any special treatment to mark member streams.
- Recovery mode must have at least **individualRecovery** in the **Individual Recovery Config** set to True or **sequenceRecovery** in the **Sequence Recovery Config** set to True. Individual recovery is used to eliminate redundant frames with the same stream handles, using the match recovery algorithm as the default. Sequence recovery is used to eliminate redundant frames with the same FRER ID, using the vector recovery algorithm as the default.
- Individual recovery and sequence recovery can be reset separately by the **Individual Recovery Reset** and the **Sequence Recovery Reset** register.
- Packet dropped by the ingress packet processing (including MMP) will not execute the recovery algorithms and will not update the status of sequence number history.
- Recovery mode must specify which egress port will merge the member streams into one compound stream. Only one egress port can be chosen for a given FRER ID under the recovery mode. In other words, the recovery mode requires cooperation from the forwarding logic. When packets suppose to perform recovery operations send out on other egress ports, those egress ports will not drop redundant packets, packets are transmitted with or without the R-TAG depending on the egress port configurations.
- Use **delSeqNum** to remove the R-TAG on a per egress port basis.
- Latent error detection only applies to the sequence recovery function. The detected error is reported to the **Latent Error Detection Status** and needs to be cleared by the software.
- **latentErrorPaths** in the **Latent Error Detection Configuration** needs to be in consistent with the **Stream Handle To FRER Mapping Table**.
- The recovery function and the latent error detection function have separate tick configurations in **Recovery Tick** and **Latent Error Detection Tick** for their time measurement purposes.

22.4 Internal State

Both the generation mode and the recovery mode holds its own internal state. The state can be controlled without the global reset.

- Under the generation mode, **FRER Sequence Number** holds the sequence number for the next packet hence a re-configuration or reset can be done by writing new values to it directly.
- Under the recovery mode, the internal state of the individual recovery and the sequence recovery can be controlled separately. A reset can be triggered by either **Individual Recovery Reset** or **Sequence Recovery Reset** from the conf bus, or an internal timeout when an FRER ID stops receiving packets for a certain period (`timeoutCnt` in either **Individual Recovery Config** or **Sequence Recovery Config**). A reset puts the recovery function under the state to take any sequence number again.

22.5 Redundancy Tag

There are several standards for the purpose of seamless redundancy, this core only supports adding and removing the redundancy tag (R-TAG) defined in IEEE 802.1CB. An R-TAG is always added for packets under the generation mode, and the removal of R-TAG is egress port based and independent from FRER mode settings. The **delSeqNum** field from the **Egress Port Configuration** only applies to R-TAGs on the received packets, it does not affect the R-TAG that will be inserted by the FRER generation mode.

22.6 Statistics

- **Individual Recovery Discarded Counter**
- **Individual Recovery Lost Counter**
- **Individual Recovery Out Of Order Counter**
- **Individual Recovery Passed Counter**
- **Individual Recovery Rogue Counter**
- **Individual Recovery Tagless Counter**
- **Sequence Recovery Discarded Counter**
- **Sequence Recovery Lost Counter**
- **Sequence Recovery Out Of Order Counter**
- **Sequence Recovery Passed Counter**
- **Sequence Recovery Rogue Counter**
- **Sequence Recovery Tagless Counter**

Chapter 23

Tick

All token buckets - and all other functions dependent on measuring time - in the core are basing their time measurements on the system ticks. This core has two sets of ticks. The core ticks and the PTP ticks. The PTP ticks are used for:

- Ingress Transmission Gates
- Egress Transmission Gates
- Port Shapers
- Queue Shapers
- Prio Shapers

All other blocks use the core ticks. Both sets of ticks work the same way, in that there is a master tick and a number of, slower, derived ticks.

23.1 Core 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.

23.2 PTP 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 **PTP 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 **PTP Tick Configuration** register, so `tick1` is clkDivider slower than `tick0`, `tick2` is clkDivider slower than `tick1`, and so on.

If the **PTP Tick Configuration** is updated during runtime, all features relying on the ptp 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 PTP Tick divider is disabled. Using the **PTP Tick Select** register the input to the tick divider can be chosen to be driven from the core clock or from pin 1 of the *debug_write_data* interface.

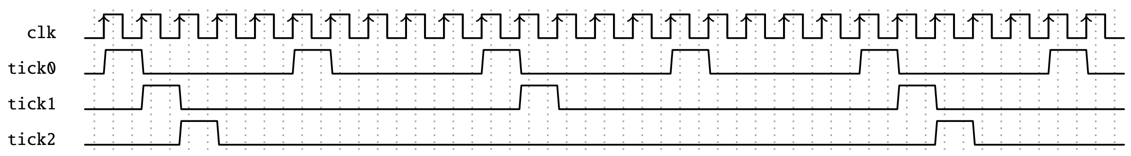

Figure 23.1: Ticks when clkDivider=5 and stepDivider=2

Chapter 24

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 17.1). 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 four kinds of traffic are checked by the MBSC unit:

- L2 Broadcast
- L2 Unknown Multicast Flooding
- L2 Unknown Unicast 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).

24.1 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 Unknown Multicast: A Packet that will be L2 switchecd but the DA is unknown. The unknown DA MAC has Ethernet multicast bit set to 1. In this case the packet is flooded to all VLAN member ports.
  - Token bucket configurations:
    * **L2 Unknown Multicast Storm Control Enable**
    * **L2 Unknown Multicast Storm Control Bucket Capacity Configuration**
    * **L2 Unknown Multicast Storm Control Bucket Threshold Configuration**
    * **L2 Unknown Multicast Storm Control Rate Configuration**
- L2 Unknown Unicast: A Packet that will be L2 switchecd but the DA is unknown. The unknown DA MAC has Ethernet multicast bit set to 0. In this case the packet is flooded to all VLAN member ports.
  - Token bucket configurations:
    * **L2 Unknown Unicast Storm Control Enable**
    * **L2 Unknown Unicast Storm Control Bucket Capacity Configuration**
    * **L2 Unknown Unicast Storm Control Bucket Threshold Configuration**
    * **L2 Unknown Unicast 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**

24.2 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:

```
#!/usr/bin/python

rate     = 0.05

minLen   = 64 # bytes
slice    = 1 # switch slices
ifg      = 20 # bytes
pnb      = 1 # = packet mode
portBW   = 3000 # 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}
```

[1]The system ticks are described in Chapter 23.

Chapter 25

Ingress Rate Control

Similar to the multicast/broadcast storm control unit that applied to each egress port, the core features another ingress rate control unit for each ingress port. The metering is done separately for different ingress queues and there are in total six inspected traffic types can be supported.

25.1 Determine ingress queue

- For VLAN tagged packet, the packet ingress queue is mapped from 802.1Q priority bits through the **VLAN PCP To Queue Mapping Table**.
- For non-VLAN tagged packet, the packet ingress queue can be mapped from **defaultPcp** through the **VLAN PCP To Queue Mapping Table**, or forced by **Force Non VLAN Packet To Specific Queue**.

25.2 Inspected Traffic

Ingress Rate Control Type defines the supported traffic types that can be metered on each ingress queue.

- L2 unicast hit
- L2 unicast miss
- L2 multicast hit
- L2 multicast miss
- Broadcast
- Reserved MAC DA

25.3 Configuration

Ingress rate control uses the same type of token bucket as MBSC, hence the configurations are done in the same way:

- **Ingress Rate Control Enable**
- **Ingress Rate Control Bucket Capacity Configuration**
- **Ingress Rate Control Bucket Threshold Configuration**
- **Ingress Rate Control Rate Configuration**

Chapter 26

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 1536 cells, each 150 byte wide, available in the buffer memory. A packet is written to the buffer memory with the original packet data plus a 18 byte ingress to egress header, thus a 1600 byte packet will have 1618 bytes and occupy eleven 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 133 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.

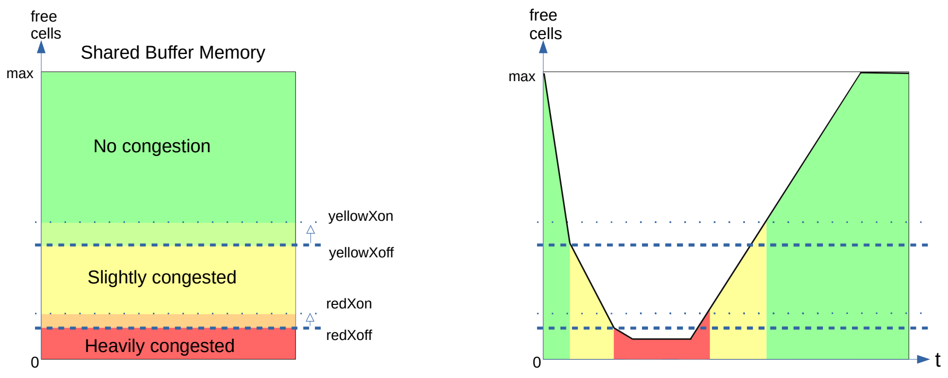

Figure 26.1: Buffer memory congestion zones

26.1 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**.

26.2 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**.

26.3 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.

26.4 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.

26.5 Restrictions

Be aware that the **Map Queue to Priority** settings need to be done when there is no traffic on any port. Update with ongoing traffic may provide a wrong enqueuing snapshot to the ERM and cause inconsistencies that can not be recovered without a reset.

Chapter 27

Flow Control

The purpose of flow control is to give access to storage in the packet buffer in an fair manner between the ports sending packets to this switch. No single source port or, if configured for it, traffic class, shall be able to behave in a way that punishes other source ports (or traffic classes). For this purpose flow control has two tools at its disposition: Pausing and tail-drop.

27.1 Pausing

Pausing, or Ethernet flow control, is a method of remote controlling the far-end interface's transmissions to this switch using dedicated pause frames. Hence, for successful pause operation the far-end interface also needs to be set up properly. The remote control is done by regularly sending pause frames (by this switch's MACs) to the far-end interfaces.

The switch core will only provide the MACs with a vector of the current pause state. It is up to the MAC to detect state changes and send the appropriate pause frames. The interface for the pause state vector is described in Section 31.4.

The pause frames are entirely handled by the MAC. It both creates frames and consumes incoming frames. The switch does not expect any pause frames on the packet interface from MAC, and the switch will not create any pause frames.

The beauty of pausing is that it can be used to set up flow control without packet drops. If the size of the packet buffer is large enough to cope with the data in flight from all the far end interfaces, and they all support pausing, it is possible to configure a completely drop-less system.

If, however, some far end interfaces do not support pausing, or the amount of data in flight is too large, it is necessary to make use of tail dropping.

27.2 Tail-Drop

Tail-drop is an implicit flow-control scheme. By deliberately dropping incoming packets (tail refers to the tail of the queue) there is an induced limitation of flows by Layer 3 transport protocols with flow control (e.g. TCP). So in contrast to Pausing, Tail-drop is not reliant on features of neighboring interfaces, but on features of higher level protocols. Transport protocols without flow control (e.g. UDP) will not limit their flows due to drops, but tail-drop will still prevent those flows, when misbehaving, from interfering with traffic from other source ports (or traffic classes).

Note that for flow control to function correctly all source ports have to be set up for either pausing or tail-drop (or both). If a single source port is not configured properly, it can starve all the others of buffering resources.

27.2.1 Tail-drop as police for Pausing

Even on Pause-enabled ports it may be useful to set up tail dropping as back-up for Pausing. By setting the tail-drop threshold at a level where we would have stopped receiving data from a Pausing-enabled source port, had it observed our pause frame, we can protect our packet buffering resources even in the case that a remote interface fails to act on the pause frame.

27.3 Buffer partitioning

The packet buffer space is partitioned into reserved and free-for-all (FFA) areas. Properly configured tail-drop will never drop a packet so long as only the reserved areas are used. Below I will use "resource" to mean "source port" on a non-PFC port and "source port/traffic class" on a PFC-enabled port.

The number of FFA cells that are are allowed to be consumed by each resource before it will be hit by flow control is configured individually per resource. When the number of used free-for-all cells reaches the configured Xoff threshold, the pause state will be set to Xoff. And when the tail-drop threshold is exceeded a packet may be dropped (depending on whether there are reserves left).

The flow control decision will only be made once the last cell of a packet is about to be written to the packet buffer. Thus the thresholds need to be set so that there is space for one maximum packet per source port set aside.

27.3.1 Reserves

The tail-drop and the pausing share the reserved settings and the counters but the meaning of reserve is different between them. For tail-drop a reserve is really a reserve. Meaning that if, for instance, a source port still has reserves left it will not drop even if the global threshold is exceeded. For pausing, when an Xoff threshold is reached it will cause pausing whether or not there are reserves left. So when the global Xoff threshold is reached all ports with pausing enabled will be paused. Even those that have reserves left.

The reason that tail drop and pausing work differently is that pausing needs hysteresis between Xoff and Xon, and tail drop does not. It would be difficult to maintain the hysteresis if the reserves were observed for pausing.

Each port can be set up to work in either PFC-mode, and non-PFC-mode. In PFC-mode the accounting is done per port and traffic class, while in non-PFC-mode the accounting is only per port.

27.4 Non-PFC mode

In non-PFC mode the traffic class is disregarded, and accounting is only done per source port. The mode is controlled individually per source port by the **Port Pause Settings:mode** fields for pausing and by the **Port Tail-Drop Settings:mode** fields for tail-drop. The **Port Reserved** registers define the number of cells reserved per source port.

These counters are used in non-PFC mode:

- **FFA Used PFC**: The total number of free-for-all cells occupied by ports in PFC-mode
- **FFA Used non-PFC**: Total number of free-for-all cells occupied by ports in non-PFC-mode
- **Port Used**: Number of cells occupied by each source port

Note that the global threshold is for the sum of FFA cells, that is the sum of **FFA Used PFC** and **FFA Used non-PFC**

27.5 PFC-mode

In PFC mode accounting is additionally done per traffic class. The **Port/TC Reserved** registers define the number of cells reserved for each specific source port and traffic class combination.

Figure 27.1 illustrates the partitioning of reserved and FFA areas.

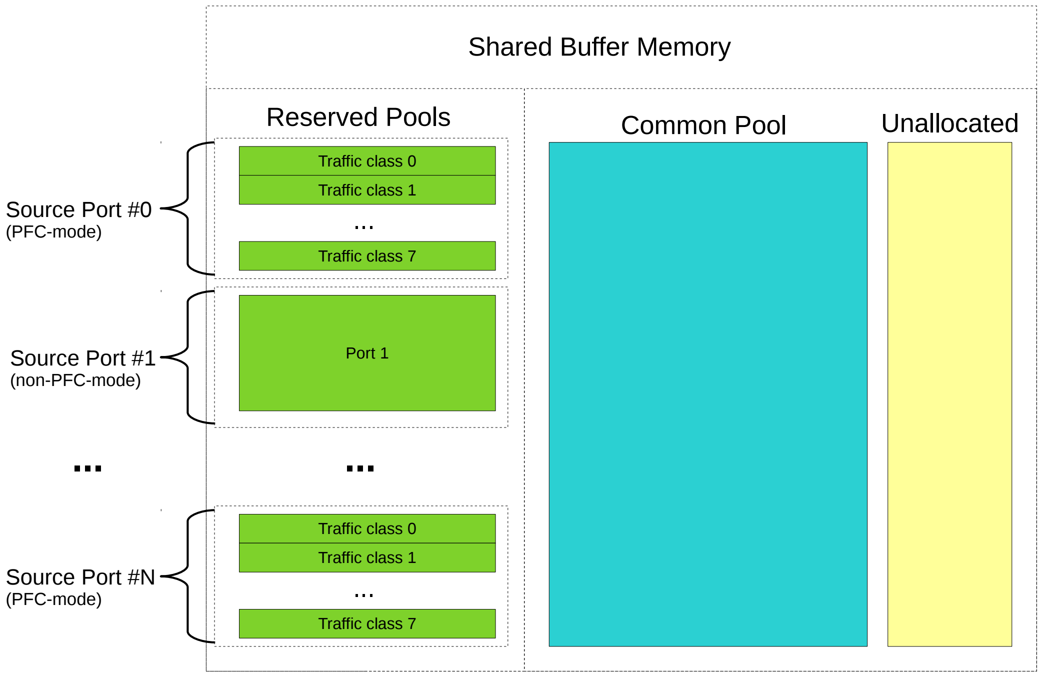

Figure 27.1: The buffer memory is partitioned into Reserved and FFA areas. The unallocated area is the space set aside for the currently incoming packets.

These counters are used in PFC mode:

- **FFA Used PFC**: The total number of free-for-all cells occupied by ports in PFC-mode
- **FFA Used non-PFC**: Total number of free-for-all cells occupied by ports in non-PFC-mode
- **Port FFA Used**: The number of free-for-all cells occupied for each source port
- **TC FFA Used**: The number of free-for-all cells occupied for each traffic class
- PFC Inc/Dec Counters: The cell counters per Port/TC are comprised of separate increment and decrement counters per Port/TC. The current counter value is calculated by taking the increment minus the decrement modulo the counter size.

27.5.1 Pausing Thresholds

For tail-drop there is a single set of thresholds above which packets are dropped. For pausing there are two sets of thresholds, Xon thresholds and Xoff thresholds, thus forming a hysteresis area to avoid bursts of pause frames at the threshold. Going above the Xoff threshold will produce a pause frame turning off the packet flow at the remote interface, but to produce a pause frame turning it back on requires going all the way down below the Xon threshold.

These are the pausing thresholds:

- **Xoff FFA Threshold**: When the total number of used FFA cells is at or above this threshold the global pause state is set to paused.
- **Xon FFA Threshold**: When the total number of used FFA cells goes below this threshold the global pause state is set to un-paused.
- **TC Xoff FFA Threshold**: When the total number of used FFA cells for a traffic class is at or above this threshold the traffic class state is set to paused
- **TC Xon FFA Threshold**: When the total number of used FFA cells for a traffic class goes below below this threshold the traffic class state is set to un-paused.

- **Port Xoff FFA Threshold**: When the total number of used FFA cells for a source port is at or above this threshold the source port state will be set to paused.
- **Port Xon FFA Threshold**: When the total number of used FFA cells for a source port goes below this threshold the source port state is set to un-paused.
- **Port/TC Xoff Total Threshold**: When the sum of the FFA and Reserved cells used for a specific source port and traffic class combination is at or above this threshold, the state of this specific source port and traffic class combination will be set to paused.
- **Port/TC Xon Total Threshold**: When the sum of the FFA and Reserved cells used for a specific source port and traffic class combination goes below this threshold the state for this specific source port and traffic class combination is set to un-paused

Note that all thresholds are for the number of FFA cells used, except for the Port/TC threshold which is for the total number of cells used.

In non-PFC-mode each resource is affected by two thresholds: The source port threshold and the global threshold. Both need to be in the un-paused state for the source port to the set to un-paused.

In PFC-mode each resource (source port and traffic class) is affected by four thresholds:

- Source Port/Traffic Class
- Source Port
- Traffic Class
- Global

All four need to be in the un-paused state for the source port and traffic class combination to be set to un-paused.

27.5.2 Tail-drop Thresholds

For tail-drop there is no hysteresis so there is only a single set of thresholds:

- **Tail-Drop FFA Threshold**: When the total number of used FFA cells is above this threshold all packets will be dropped from the tail-drop-enabled ports that have no reserved cells left to spend
- **Port Tail-Drop FFA Threshold**: When the total number of used FFA cells for a source port is above this threshold incoming packets from this source port will be dropped unless the port is in PFC-mode and there are reserved cells left to spend
- **TC Tail-Drop FFA Threshold**: When the total number of used FFA cells for a traffic class is above this threshold any incoming packet belonging to the traffic class will be dropped unless the port/TC has reserved cells left to spend. Only valid in PFC-mode
- **Port/TC Tail-Drop Total Threshold**: When the sum of the FFA and Reserved cells used for a specific source port and traffic class combination is above this threshold any incoming packet from this source port assigned to this traffic class will be dropped. Only valid in PFC-mode

The **Tail-Drop FFA Threshold**, **TC Tail-Drop FFA Threshold** and **Port Tail-Drop FFA Threshold** are not obeyed strictly. The first packet exceeding the threshold may be accepted, causing a one-packet over-shoot.

27.6 Enabling Tail-Drop

Tail-drop is enabled per source port using the **Port Tail-Drop Settings:enable** fields. The individual thresholds are enabled using the enable fields in each threshold register. See Section 27.5.1 above.

27.7 Enabling Pausing

Pausing is enabled per source port using **Port Pause Settings:enable** fields. The individual thresholds are enabled using the enable fields in each threshold register. See Section 27.5.1 above.

27.8 Dropped packets

Packets that are dropped will still consume resources while they are waiting for deallocation. This applies even to broken packets, for instance packets with CRC errors.

The packets dropped due to exceeding the Tail-Drop thresholds are counted in the **Ingress Resource Manager Drop** register.

27.9 Reconfiguration

The Xon, Xoff and tail-drop thresholds can be reconfigured at any time. The reserved settings, however, cannot be changed on any source port on which there is traffic. The reserved settings also cannot be changed for any source port that has packets queued. If the reserved settings are changed in these cases the flow control counters will be irrevocably corrupted, necessitating a reset for the core to continue normal operation.

27.10 Debug Features

Each threshold can be forced to trigger using the trip fields of the threshold registers. For tail-drop only drop can be forced this way, but accept can of course be assured by disabling the threshold using the enable field.

For pausing a specific pause state can be forced using the force and pattern fields of the **Port Pause Settings** register.

Chapter 28

Egress Port Shaper

The egress port rates are shaped by token buckets configured in the **Port Shaper Rate Configuration** registers. While the token bucket level (**Port Shaper Current Size**) is below the threshold configured in the **Port Shaper Bucket Threshold Configuration** register, no new packets are scheduled for the corresponding egress port. Ongoing packets are not affected by the shaping bucket status.

The port shapers are enabled using the **Port Shaper Enable** register, and the saturation level of the port shaper buckets is controlled by the **Port Shaper Bucket Capacity Configuration** register.

When you enable the port shaper you must make sure to also enable the PTP Tick using the **PTP Tick Select** register, because the PTP Tick is disabled by default.

An illustration of a token bucket can be seen in Figure 17.1 (despite what the illustration says the shaper will of course never drop any packets).

Chapter 29

Statistics

Short Name	Register Name
3. macBrokenPkt	MAC RX Broken Packets
4. macRxMin	MAC RX Short Packet Drop
4. macRxMax	MAC RX Long 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 Rate Control Drop Ingress Packet Filtering Drop Reserved MAC DA Drop Reserved MAC SA Drop VLAN Member Drop Minimum Allowed VLAN Drop Maximum Allowed VLAN Drop IP Checksum Drop L2 Reserved Multicast Address Drop Ingress Configurable ACL Drop Attack Prevention Drop ARP Decoder Drop RARP Decoder Drop L2 IEEE 1588 Decoder Drop L4 IEEE 1588 Decoder Drop IEEE 802.1X and EAPOL Decoder Drop SCTP Decoder Drop LACP Decoder Drop AH Decoder Drop ESP Decoder Drop DNS Decoder Drop BOOTP and DHCP Decoder Drop CAPWAP Decoder Drop GRE Decoder Drop L2 Action Table Special Packet Type Drop L2 Action Table Drop L2 Action Table Port Move Drop L2 Destination Table SA Lookup Drop Source Port Default ACL Action Drop

Short Name	Register Name
	Ingress Received and Dropped Counter
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 Configurable ACL Match Counter
11. preEppDrop	Queue Off Drop Egress Spanning Tree Drop MBSC Drop Ingress-Egress Packet Filtering Drop L2 Action Table Per Port Drop
11. ippReception	Ingress MAC SA Change Counter
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
14. psfp	PSFP Matching Frame Counter PSFP Passing SDU Counter PSFP Not Passing SDU Counter PSFP Passing Frame Counter PSFP Not Passing Frame Counter PSFP Red Frames Counter
14. frer	Individual Recovery Passed Counter Individual Recovery Discarded Counter Individual Recovery Out Of Order Counter Individual Recovery Rogue Counter Individual Recovery Lost Counter Individual Recovery Tagless Counter Sequence Recovery Passed Counter Sequence Recovery Discarded Counter Sequence Recovery Out Of Order Counter Sequence Recovery Rogue Counter Sequence Recovery Lost Counter Sequence Recovery Tagless Counter Latent Error Detection Status FRER Drop
15. erm	Egress Resource Manager Drop
16. bmOverflow	Buffer Overflow Drop
16. irm	Ingress Resource Manager Drop
18. pbTxPkt	PB Packet Head Counter PB Packet Tail Counter
19. epppDrop	Unknown Egress Drop Egress Port Disabled Drop Egress Port Filtering Drop
20. dequeued	Dequeued Packets Dequeued Bytes
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

Short Name	Register Name

Table 29.1: 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 29.1 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 29.1 can be found through Table 29.1.

29.1 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.

- 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.

29.2 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.

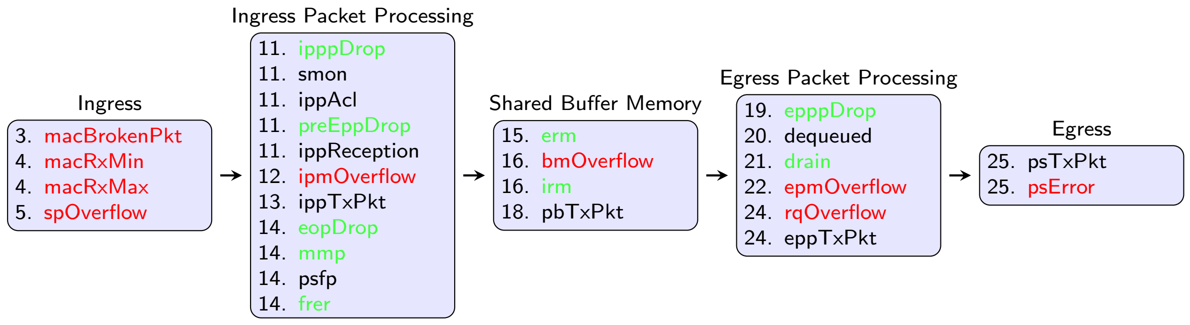

Figure 29.1: Location of Statistics Counters

All the related counters are in Section Statistics: ACL.

29.3 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.

29.4 Ingress Port Receive Statistics

Section Statistics: IPP Ingress Port Receive lists available statistics for good received packets on a per ingress port basis.

- Good received but IPP dropped packets
  - **Ingress Received and Dropped Counter**
- Good received broadcast packets with MAC SA changed
  - **Ingress MAC SA Change Counter**

29.5 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.

29.6 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.

29.7 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.

Chapter 30

Packets To And From The CPU

The CPU port (number 8 by default) 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.

30.1 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	[8:0] 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 30.1: 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 30.1). 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 30.1.

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.

30.1.1 Identify the From CPU Tag

By default, only packets that are received on the CPU port will be able to support identifying the specific Ethernet type for the from CPU tag. This means that packets with this Ethernet type that are received on other ports of the switch will be treated as unknown and will not enter the packet processing based on the from CPU tag.

[1]The ordering described in 30.1 is the receive/transmit order.

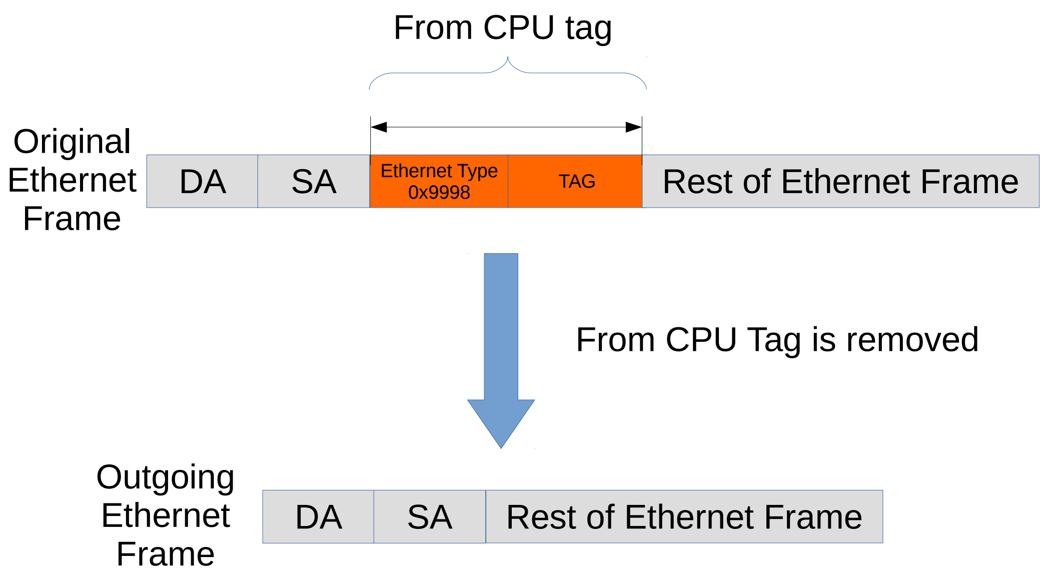

Figure 30.1: Packet from CPU with CPU tag

If non-CPU ports need to identify the from CPU tag, it can be achieved by the **enableFromCpuTag** from the **Source Port Table**. Notice the CPU port is not affected by this setting and always decode the from CPU tag.

30.1.2 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.
- 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 802.1X-EAPOL AH ARP AVTP DHCP CAPWAP DNS ESP GRE L2 1588 L4 1588 LACP RARP SCTP 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.

30.2 Packets To the CPU

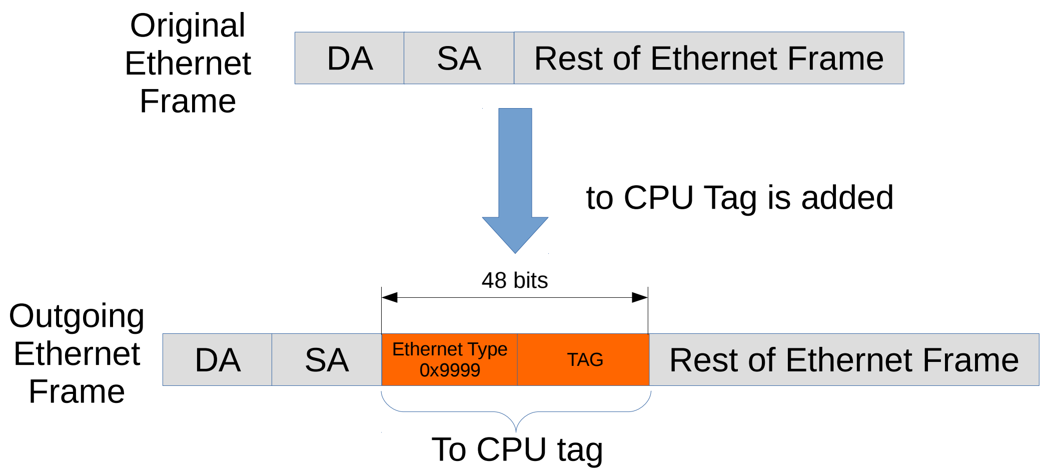

Figure 30.2: Packet to CPU with CPU tag

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 30.2). 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 8 in this core), the packets are tagged with a specific Ethernet Type (type 39321). Figure 30.2 shows the Ethernet type field followed by a tag, and together 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 to 2	Reason for packet sent to CPU. See table 30.3. Byte 1 is the msb of the reason code.
2	Reserved
3	Reserved

Table 30.2: To CPU tag format

30.2.1 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 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 source port default ACL action sent the packet to the CPU..Index to source port which sent the packet in.
21 + HitIndex	The TCAM in the configurable ingress ACL engine 0 sent the packet to the CPU..Index to rule.
37 + HitIndex	The small table in the configurable ingress ACL engine 0 sent the packet to the CPU..Index to rule.
101 + HitIndex	The large table in the configurable ingress ACL engine 0 sent the packet to the CPU..Index to rule.
229 + HitIndex	The TCAM in the configurable ingress ACL engine 1 sent the packet to the CPU..Index to rule.
245 + HitIndex	The small table in the configurable ingress ACL engine 1 sent the packet to the CPU..Index to rule.
261 + HitIndex	The large table in the configurable ingress ACL engine 1 sent the packet to the CPU..Index to rule.
517	This is an L2 1588 frame.
518	This is an L4 1588 frame.
519	This is an ARP frame.
520	This is an RARP frame.
521	This is an LLDP frame.
522	This is an 802.1X EAPOL frame.
523	This is an GRE frame.
524	This is an SCTP frame.
525	This is an LCAP frame.
526	This is an AH frame.
527	This is an ESP frame.
528	This is an DNS frame.
529	This is a BOOTP or DHCP frame.
530	This is an CAPWAP frame.
531	Packet matched an L2 Multicast Reserved Address
532	The **L2 Action Table** has determined that this packet shall be sent to the CPU.

Table 30.3: Reason for packet sent to CPU

The possible reasons are listed in Table 30.3.

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 the **Configurable ACL Engine** with a sendToCpu action.

30.2.2 Reason Code Operations

If the packet is sent to the CPU port with a non-zero reason code, the **CPU Reason Code Operation** register allows extra actions based on the corresponding reason code. The reason code number is checked in 4 given ranges from the first entry to the last entry. If the reason code has multiple hits, different operations can be done in parallel and the same operation in the latter one will override the previous hit.

- **mutableCpu** allows the packets that are sent to the CPU port use another port number for the CPU port. In this case the to CPU tag is always inserted to the packet and will not be controlled by **Disable CPU tag on CPU Port**.
- **forceQueue** alters the egress queue of the packets that are sent to the CPU port.

Chapter 31

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.

31.1 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). The two *clk_mult* are higher frequency clocks, syncronous with the core clock, that are used in a few places in the core where a higher clock gives a substantial area savings.

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 18.9, while making sure that the packet interface halt is kept low.

Signal Name	Size	In Out	Description
clk	1	In	Core clock. For 13 Gbit/s wire-speed throughput use a core clock frequency of 25.0 MHz
rstn	1	In	Global asynchronous reset (active low)
clk_mac_rx_**N**	1	In	Clock for the RX packet interface for port **N**. Synchronous with the core clock.
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**. Synchronous with the core clock.
rstn_mac_tx_**N**	1	In	Asynchronous reset (active low) for the TX packet interface for port **N**
clk_mult_0	1	In	A 125.0MHz clock, synchronous with the core clock.
clk_mult_1	1	In	A 50.0MHz clock, synchronous with the core clock.
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 31.1: 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 31.1. 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.

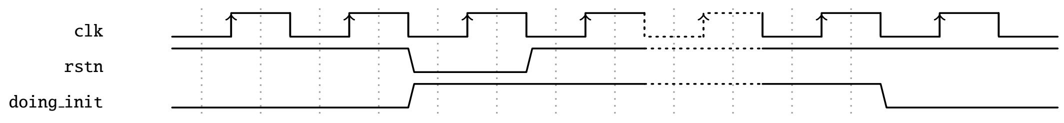

Figure 31.1: Core Initialization

31.1.1 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, but to figure out what went wrong you will probably need to use the debug interface.

31.2 Packet Interface

There are 9 packet interfaces, or ports for short, each divided into a reception part and a transmission part. The ports are numbered from 0 to 8.

Pin	Size	Direction	Description
`idata_sp_`**N**	24	`In`	Packet data.
`ivalid_bytes_sp_`**N**	2	`In`	Indicates the number of valid data bytes. For all transactions where *last* is not high, this shall be equal to the data width in bytes.
`ifirst_sp_`**N**	1	`In`	Start-of-packet flag.
`ilast_sp_`**N**	1	`In`	End-of-packet flag. The *last* field is also used to signal broken packets. For a correctly transmitted packet *last* is asserted for the last data transaction of the packet. If *last* is set high when *valid_bytes* is zero, the packet is marked as broken, and will be dropped by the core.

Table 31.2: Packet RX interface for ports 0 and 1. **N** is the ingress interface number.

Pin	Size	Direction	Description
`odata_ps_`**N**	24	`Out`	Packet data.
`ovalid_bytes_ps_`**N**	2	`Out`	Indicates the number of valid data bytes. For all transactions where *last* is not high, this is equal to the data width in bytes.
`ofirst_ps_`**N**	1	`Out`	Start-of-packet flag.
`olast_ps_`**N**	1	`Out`	End-of-packet flag. For a correctly transmitted packet *last* is asserted for the last data transaction of the packet. If *last* is set high when *valid_bytes* is zero, the packet shall be dropped or terminated with an error by the MAC.
`tx_halt_ps_`**N**	1	`In`	Interrupt the data transmission from egress port **N**.

Table 31.3: Packet TX interface for ports 0 and 1. **N** is the egress interface number.

The port interfaces are not all the same. There are two different port interface variants in this core, each with an RX and a TX direction:

1. Ports 0 and 1: RX-interface see Table 31.2 on page 159, TX-interface see Table 31.3 on page 159
2. Ports 2-8: RX-interface see Table 31.4 on page 160, TX-interface see Table 31.5 on page 160

Each direction of a packet interface consists of *first*, *last*, *valid_bytes*, and *data* fields. The transmit direction has an additional *halt* 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 number indicated in *valid_bytes*. Unless the *last* flag is set all bytes or no bytes must be valid.

Pin	Size	Direction	Description
idata_sp_**N**	8	In	Packet data.
ivalid_bytes_sp_**N**	1	In	Indicates the number of valid data bytes. For all transactions where *last* is not high, this shall be equal to the data width in bytes.
ifirst_sp_**N**	1	In	Start-of-packet flag.
ilast_sp_**N**	1	In	End-of-packet flag. The *last* field is also used to signal broken packets. For a correctly transmitted packet *last* is asserted for the last data transaction of the packet. If *last* is set high when *valid_bytes* is zero, the packet is marked as broken, and will be dropped by the core.

Table 31.4: Packet RX interface for ports 2-8. **N** is the ingress interface number.

Pin	Size	Direction	Description
odata_ps_**N**	8	Out	Packet data.
ovalid_bytes_ps_**N**	1	Out	Indicates the number of valid data bytes. For all transactions where *last* is not high, this is equal to the data width in bytes.
ofirst_ps_**N**	1	Out	Start-of-packet flag.
olast_ps_**N**	1	Out	End-of-packet flag. For a correctly transmitted packet *last* is asserted for the last data transaction of the packet. If *last* is set high when *valid_bytes* is zero, the packet shall be dropped or terminated with an error by the MAC.
tx_halt_ps_**N**	1	In	Interrupt the data transmission from egress port **N**.

Table 31.5: Packet TX interface for ports 2-8. **N** is the egress interface number.

Sending and Receiving packets

Data transmission, either to or from the core, begins with a transaction where the *first* field is high and the *valid_bytes* field is non-zero, and ends with a data transmission where the *last* field is high. Idle transactions—where valid_bytes, *first* and *last* 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 32749 bytes. If this length was allowed to be exceeded either on the ingress or the egress it would 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 32749-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 16325-byte packets.

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 *halt* interface to prevent overflows.

Broken packets

A packet ending with *last* set high and *valid_bytes* set to zero 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**.

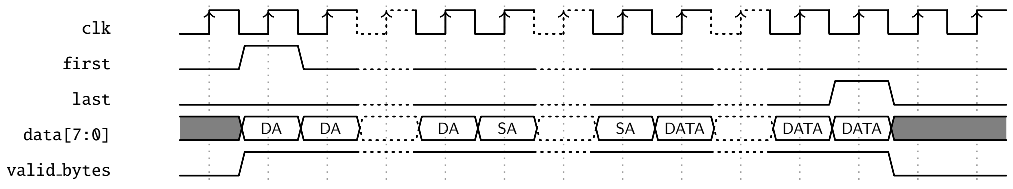

Figure 31.2: Sending and Receiving packets without error (8-bit)

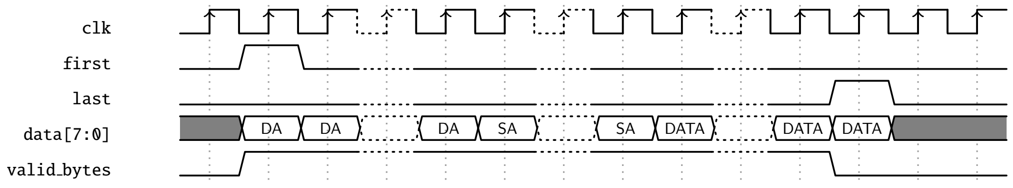

Figure 31.3: Sending and Receiving packets with error (8-bit)

Halts

Data transmission from the transmit interface of the core can be interrupted individually per egress port using the *halt* signals. A high halt signal on the positive edge of mac clock, will cause the transmission to be idle for the corresponding egress port on the same positive edge. Data transmission will resume on the next positive edge of mac clock when halt is again low.

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]

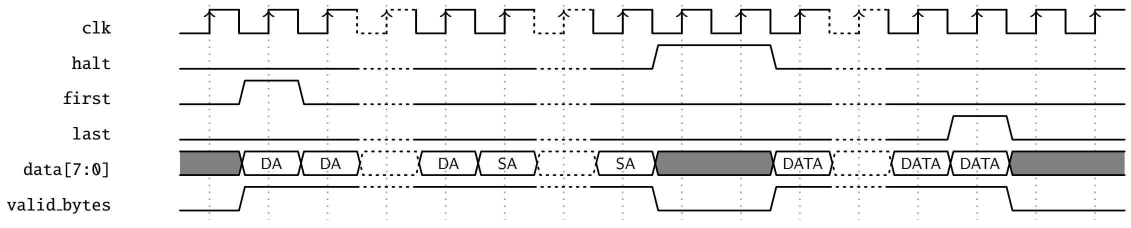

Figure 31.4: Halted transmit packet (8-bit)

followed by byte 1 on bits data[15:8] and so on.

The *valid_bytes* 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 *valid_bytes* can indicate less than the full bus width. In this case the byte order is still the same as previous transfers. For example when *valid_bytes* is 1 the last byte of the packet is placed on bits [7:0] and with *valid_bytes* of 2 the last byte of the packet is placed on bits [15:8] and the second to last is on bits [7:0].

31.3 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 31.6 below.

A user guide for the configuration interface follows in Chapter 32.

Pin	Size	Direction	Description
`request_data_N`	32	`In`	The request data
`request_address_N`	17	`In`	The request address
`request_re_N`	1	`In`	Read enable for the transaction. Active high
`request_we_N`	1	`In`	Write enable for the transaction. Active high
`request_type_N`	2	`In`	The request type 0 `Default` 1 `Accumulator` 2 *`Reserved`* 3 *`Reserved`*
`request_id_N`	1	`In`	The request identifier.
`reply_status_N`	2	`Out`	The reply status 0 Idle (`NONE`) 1 Read OK (`ROK`) 2 Write OK (`WOK`) 3 Fail (`FAIL`)
`reply_id_N`	1	`Out`	The reply identifier
`reply_data_N`	32	`Out`	The reply data.

Table 31.6: The signals for an instance of the configuration interface

31.4 Pause Interfaces

There are separate pause interfaces for sending status information from the switch to the MAC, *opfc_status*, and from the MAC to the switch, *iext_pause*. Note that these interfaces are in the core clock domain.

31.4.1 PFC Status

The ipfc_status interface is used to transfer pause status from the switch resource manager to the MAC, so the MAC can generate pause frames.

The switch will merely indicate its current pause status, it is up to the MAC to generate the necessary pause frames to keep the far end switch in the desired pausing state.

In port mode the status interface will send 0 in unpaused state, and 0xff in paused state.

31.4.2 External Pause

The *iext_pause* interface is used to transfer PFC pause status received by the MAC to the switch egress scheduler. When the status is XOFF the switch egress scheduler will not send any new packets. Ongoing packets are not affected. There is one iext_pause interface for each packet interface. Even when priority pause is not enabled the external pause interface is still operating per priority.

Pin	**Direction**	**Size**	**Description**
`iext_pause_`**N**	In	8	Xoff=1, Xon=0 status for each PFC channel (0..7)
`opfc_status_`**N**`[7:0]`	Out	8	Xoff=1, Xon=0 status for each PFC channel (0..7)

Table 31.7: ThePFC status and External Pause interfaces, where **N** is the packet interface number

31.5 Debug Read Interface

The debug read interface outputs internal debug signals on the *debug_read_data* port. Which signals to observe is selected with the *debug_read_select* port. The mapping between select value and debug signal is described in Table 31.9. Both these signals are pipelined.

Pin	**Direction**	**Size**	**Description**
`debug_read_select`	In	9	Selects the signal to monitor. See Table 31.9.
`debug_read_data`	In	32	The debug output data.

Table 31.8: The Debug Read interface

id	**instance**	**signal**
0	pa_top.switch.mactop	constant-0
1	—"—	rx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
2	—"—	tx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
3	—"—	rx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
4	—"—	tx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
5	—"—	rx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
6	—"—	tx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
7	—"—	rx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
8	—"—	tx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
9	—"—	rx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
10	—"—	tx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
11	—"—	rx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
12	—"—	tx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
13	—"—	rx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
14	—"—	tx_pkt_bus {8'data, 1'valid_bytes, 1'last, 1'first}
15	—"—	rx_pkt_bus {24'data, 2'valid_bytes, 1'last, 1'first}
16	—"—	tx_pkt_bus {24'data, 2'valid_bytes, 1'last, 1'first}
17	—"—	rx_pkt_bus {24'data, 2'valid_bytes, 1'last, 1'first}
18	—"—	tx_pkt_bus {24'data, 2'valid_bytes, 1'last, 1'first}
19	—"—	constant-19
20	pa_top.switch.ipp0	constant-20
21	—"—	ipp_ipkt_bus {18'data, 8'valid_bytes, 4'id, 1'last, 1'first}
22	—"—	ipp_opkt_bus {18'data, 8'valid_bytes, 4'id, 1'last, 1'first}
23	—"—	pass_da_0
24	—"—	pass_da_1
25	—"—	dut_iIpp_iDropper_dbg_drop
26	—"—	dut_iIpp_iDropper_dbg_ifirst

id	instance	signal
27	—"—	dut_iIpp_iDropper_dbg_ilast
28	—"—	pass_sa_0
29	—"—	pass_sa_1
30	—"—	constant-30
31	pa_top.switch.ipp0.ippp	constant-31
32	—"—	dut_iIppp_iExtiRateCtrlReqBlock_iBucket_reg_stat
33	—"—	constant-33
34	pa_top.switch.ipp0.pm	constant-34
35	—"—	pm_fifo_overflow
36	—"—	dut_dbg_fifo_full
37	—"—	halt_from_pm
38	—"—	dut_iFifoa_debug_in
39	—"—	dut_iFifoa_debug_out
40	—"—	constant-40
41	pa_top.switch.sp0	constant-41
42	—"—	dut_iSpbridge_iBridge_8_iSyncFifo_iF_iFifos_zFcnt_pop_empty
43	—"—	dut_iSpbridge_iBridge_8_iSyncFifo_iF_iFifos_zFcnt_push_full
44	—"—	dut_iSpbridge_iBridge_7_iSyncFifo_iF_iFifos_zFcnt_pop_empty
45	—"—	dut_iSpbridge_iBridge_7_iSyncFifo_iF_iFifos_zFcnt_push_full
46	—"—	dut_iSpbridge_iBridge_6_iSyncFifo_iF_iFifos_zFcnt_pop_empty
47	—"—	dut_iSpbridge_iBridge_6_iSyncFifo_iF_iFifos_zFcnt_push_full
48	—"—	dut_iSpbridge_iBridge_5_iSyncFifo_iF_iFifos_zFcnt_pop_empty
49	—"—	dut_iSpbridge_iBridge_5_iSyncFifo_iF_iFifos_zFcnt_push_full
50	—"—	dut_iSpbridge_iBridge_4_iSyncFifo_iF_iFifos_zFcnt_pop_empty
51	—"—	dut_iSpbridge_iBridge_4_iSyncFifo_iF_iFifos_zFcnt_push_full
52	—"—	dut_iSpbridge_iBridge_3_iSyncFifo_iF_iFifos_zFcnt_pop_empty
53	—"—	dut_iSpbridge_iBridge_3_iSyncFifo_iF_iFifos_zFcnt_push_full
54	—"—	dut_iSpbridge_iBridge_2_iSyncFifo_iF_iFifos_zFcnt_pop_empty
55	—"—	dut_iSpbridge_iBridge_2_iSyncFifo_iF_iFifos_zFcnt_push_full
56	—"—	dut_iSpbridge_iBridge_1_iSyncFifo_iF_iFifos_zFcnt_pop_empty
57	—"—	dut_iSpbridge_iBridge_1_iSyncFifo_iF_iFifos_zFcnt_push_full
58	—"—	dut_iSpbridge_iBridge_0_iSyncFifo_iF_iFifos_zFcnt_pop_empty
59	—"—	dut_iSpbridge_iBridge_0_iSyncFifo_iF_iFifos_zFcnt_push_full
60	—"—	dut_iSpbridge_assert_reset_sp_bridge
61	—"—	dut_iSpbridge_assert_reset_sp_bridge
62	—"—	dut_iSpbridge_assert_reset_sp_bridge
63	—"—	dut_iSpbridge_assert_reset_sp_bridge
64	—"—	dut_iSpbridge_assert_reset_sp_bridge
65	—"—	dut_iSpbridge_assert_reset_sp_bridge
66	—"—	dut_iSpbridge_assert_reset_sp_bridge
67	—"—	dut_iSpbridge_assert_reset_sp_bridge
68	—"—	dut_iSpbridge_assert_reset_sp_bridge
69	—"—	constant-69
70	pa_top.switch.pb0	constant-70
71	—"—	dut_iPbu_debug_refc_inc
72	—"—	dut_iPbu_debug_port_sch
73	—"—	dut_iPbu_dmux_wrr
74	—"—	dut_iPbu_debug_qenext
75	—"—	dut_iPbu_assert_qediff
76	—"—	dut_iPbu_assert_reque_sp
77	—"—	Mask of currently receiving packets that have been broken due to BM full
78	—"—	dut_iPbu_follow_pfc_accept
79	—"—	dut_iPbu_iAssertpacket_0_assert_out
80	—"—	pa.top.switch.pb0.iAssertpacket0 {8'valid_bytes, 4'port, 1'last, 1'first}
81	—"—	dut_iPbu_iPortshaper_iBuckets_reg_stat
82	—"—	dut_iPbu_zPassdbgqeread_0_o
83	—"—	dut_iPbu_iRequeue_iReFifo_8_iF_iFifos_zFcnt_pop_empty
84	—"—	dut_iPbu_iRequeue_iReFifo_8_iF_iFifos_zFcnt_push_full
85	—"—	dut_iPbu_iRequeue_iReFifo_7_iF_iFifos_zFcnt_pop_empty
86	—"—	dut_iPbu_iRequeue_iReFifo_7_iF_iFifos_zFcnt_push_full
87	—"—	dut_iPbu_iRequeue_iReFifo_6_iF_iFifos_zFcnt_pop_empty
88	—"—	dut_iPbu_iRequeue_iReFifo_6_iF_iFifos_zFcnt_push_full
89	—"—	dut_iPbu_iRequeue_iReFifo_5_iF_iFifos_zFcnt_pop_empty
90	—"—	dut_iPbu_iRequeue_iReFifo_5_iF_iFifos_zFcnt_push_full
91	—"—	dut_iPbu_iRequeue_iReFifo_4_iF_iFifos_zFcnt_pop_empty
92	—"—	dut_iPbu_iRequeue_iReFifo_4_iF_iFifos_zFcnt_push_full
93	—"—	dut_iPbu_iRequeue_iReFifo_3_iF_iFifos_zFcnt_pop_empty
94	—"—	dut_iPbu_iRequeue_iReFifo_3_iF_iFifos_zFcnt_push_full
95	—"—	dut_iPbu_iRequeue_iReFifo_2_iF_iFifos_zFcnt_pop_empty
96	—"—	dut_iPbu_iRequeue_iReFifo_2_iF_iFifos_zFcnt_push_full
97	—"—	dut_iPbu_iRequeue_iReFifo_1_iF_iFifos_zFcnt_pop_empty
98	—"—	dut_iPbu_iRequeue_iReFifo_1_iF_iFifos_zFcnt_push_full
99	—"—	dut_iPbu_iRequeue_iReFifo_0_iF_iFifos_zFcnt_pop_empty
100	—"—	dut_iPbu_iRequeue_iReFifo_0_iF_iFifos_zFcnt_push_full
101	—"—	dut_iPbu_iRefc_refc_mem_debug
102	—"—	dut_iPbu_zPassqesp_zPasslist_0_o
103	—"—	Filter mask for packets dropped by ERM
104	—"—	dut_iPbu_debug_pb_drop
105	—"—	constant-105
106	pa_top.switch.pb0.erm.dut_iEql	constant-106

id	instance	signal
107	—"—	red_zone
108	—"—	constant-108
109	pa_top.switch.pb0.pfc	constant-109
110	—"—	dut_debug_sp_above_rsv
111	—"—	constant-111
112	pa_top.switch.pb0.qe0	constant-112
113	—"—	dut_assert_dfifo
114	—"—	dut_assert_firstflag
115	—"—	dut_assert_reset_next
116	—"—	dut_drop_cnt
117	—"—	dut_send_cnt
118	—"—	dut_iDfifo_iF_iFifoml_pop_empty
119	—"—	dut_iDfifo_iF_iFifoml_push_full
120	—"—	dut_iDfifo_iF_iFifoml_zFcnt_pop_empty
121	—"—	dut_iDfifo_iF_iFifoml_zFcnt_push_full
122	—"—	dut_iDfifo_iF_iFifoml_iFifo2_zFcnt_pop_empty
123	—"—	dut_iDfifo_iF_iFifoml_iFifo2_zFcnt_push_full
124	—"—	dut_iDfifo_iF_iFifoml_iFifo1_zFcnt_pop_empty
125	—"—	dut_iDfifo_iF_iFifoml_iFifo1_zFcnt_push_full
126	—"—	dut_ipkt_fifo_8_debug_in
127	—"—	dut_ipkt_fifo_8_debug_out
128	—"—	dut_ipkt_fifo_7_debug_in
129	—"—	dut_ipkt_fifo_7_debug_out
130	—"—	dut_ipkt_fifo_6_debug_in
131	—"—	dut_ipkt_fifo_6_debug_out
132	—"—	dut_ipkt_fifo_5_debug_in
133	—"—	dut_ipkt_fifo_5_debug_out
134	—"—	dut_ipkt_fifo_4_debug_in
135	—"—	dut_ipkt_fifo_4_debug_out
136	—"—	dut_ipkt_fifo_3_debug_in
137	—"—	dut_ipkt_fifo_3_debug_out
138	—"—	dut_ipkt_fifo_2_debug_in
139	—"—	dut_ipkt_fifo_2_debug_out
140	—"—	dut_ipkt_fifo_1_debug_in
141	—"—	dut_ipkt_fifo_1_debug_out
142	—"—	dut_ipkt_fifo_0_debug_in
143	—"—	dut_ipkt_fifo_0_debug_out
144	—"—	dut_pfifo_level
145	—"—	dut_pfifo_level
146	—"—	dut_pfifo_level
147	—"—	dut_pfifo_level
148	—"—	dut_pfifo_level
149	—"—	dut_pfifo_level
150	—"—	dut_pfifo_level
151	—"—	dut_pfifo_level
152	—"—	dut_pfifo_level
153	—"—	constant-153
154	pa_top.switch.pb0.wrr	constant-154
155	—"—	dut_debug_below
156	—"—	dut_zPassdebugbvalpipe_zPasslist_7_o
157	—"—	dut_zPassdebugbvalpipe_zPasslist_6_o
158	—"—	dut_zPassdebugbvalpipe_zPasslist_5_o
159	—"—	dut_zPassdebugbvalpipe_zPasslist_4_o
160	—"—	dut_zPassdebugbvalpipe_zPasslist_3_o
161	—"—	dut_zPassdebugbvalpipe_zPasslist_2_o
162	—"—	dut_zPassdebugbvalpipe_zPasslist_1_o
163	—"—	dut_zPassdebugbvalpipe_zPasslist_0_o
164	—"—	dut_reg_bval
165	—"—	dut_reg_bval
166	—"—	dut_reg_bval
167	—"—	dut_reg_bval
168	—"—	dut_reg_bval
169	—"—	dut_reg_bval
170	—"—	dut_reg_bval
171	—"—	dut_reg_bval
172	—"—	dut_reg_bval
173	—"—	dut_reg_bval
174	—"—	dut_reg_bval
175	—"—	dut_reg_bval
176	—"—	dut_reg_bval
177	—"—	dut_reg_bval
178	—"—	dut_reg_bval
179	—"—	dut_reg_bval
180	—"—	dut_reg_bval
181	—"—	dut_reg_bval
182	—"—	dut_reg_bval
183	—"—	dut_reg_bval
184	—"—	dut_reg_bval
185	—"—	dut_reg_bval
186	—"—	dut_reg_bval

id	instance	signal
187	—"—	dut_reg_bval
188	—"—	dut_reg_bval
189	—"—	dut_reg_bval
190	—"—	dut_reg_bval
191	—"—	dut_reg_bval
192	—"—	dut_reg_bval
193	—"—	dut_reg_bval
194	—"—	dut_reg_bval
195	—"—	dut_reg_bval
196	—"—	dut_reg_bval
197	—"—	dut_reg_bval
198	—"—	dut_reg_bval
199	—"—	dut_reg_bval
200	—"—	dut_reg_bval
201	—"—	dut_reg_bval
202	—"—	dut_reg_bval
203	—"—	dut_reg_bval
204	—"—	dut_reg_bval
205	—"—	dut_reg_bval
206	—"—	dut_reg_bval
207	—"—	dut_reg_bval
208	—"—	dut_reg_bval
209	—"—	dut_reg_bval
210	—"—	dut_reg_bval
211	—"—	dut_reg_bval
212	—"—	dut_reg_bval
213	—"—	dut_reg_bval
214	—"—	dut_reg_bval
215	—"—	dut_reg_bval
216	—"—	dut_reg_bval
217	—"—	dut_reg_bval
218	—"—	dut_reg_bval
219	—"—	dut_reg_bval
220	—"—	dut_reg_bval
221	—"—	dut_reg_bval
222	—"—	dut_reg_bval
223	—"—	dut_reg_bval
224	—"—	dut_reg_bval
225	—"—	dut_reg_bval
226	—"—	dut_reg_bval
227	—"—	dut_reg_bval
228	—"—	dut_reg_bval
229	—"—	dut_reg_bval
230	—"—	dut_reg_bval
231	—"—	dut_reg_bval
232	—"—	dut_reg_bval
233	—"—	dut_reg_bval
234	—"—	dut_reg_bval
235	—"—	dut_reg_bval
236	—"—	dut_reg_rank
237	—"—	dut_reg_rank
238	—"—	dut_reg_rank
239	—"—	dut_reg_rank
240	—"—	dut_reg_rank
241	—"—	dut_reg_rank
242	—"—	dut_reg_rank
243	—"—	dut_reg_rank
244	—"—	dut_reg_rank
245	—"—	constant-245
246	pa_top.switch.pb0.qshp	constant-246
247	—"—	dut_iPrioshaper_reg_stat
248	—"—	dut_iQueueshaper_reg_stat
249	—"—	constant-249
250	pa_top.switch.bm0	constant-250
251	—"—	dut_bm_ifree_debug_free
252	—"—	constant-252
253	pa_top.switch.ps0	constant-253
254	—"—	halt_from_ps
255	—"—	dut_iPs2_zPsAssert_item
256	—"—	dut_iPs2_iBridge_8_iSyncFifo_iF_iFifos_zFcnt_pop_empty
257	—"—	dut_iPs2_iBridge_8_iSyncFifo_iF_iFifos_zFcnt_push_full
258	—"—	dut_iPs2_iBridge_7_assert_reset
259	—"—	dut_iPs2_iBridge_7_iSyncFifo_iF_iFifos_zFcnt_pop_empty
260	—"—	dut_iPs2_iBridge_7_iSyncFifo_iF_iFifos_zFcnt_push_full
261	—"—	dut_iPs2_iBridge_6_assert_reset
262	—"—	dut_iPs2_iBridge_6_iSyncFifo_iF_iFifos_zFcnt_pop_empty
263	—"—	dut_iPs2_iBridge_6_iSyncFifo_iF_iFifos_zFcnt_push_full
264	—"—	dut_iPs2_iBridge_5_assert_reset
265	—"—	dut_iPs2_iBridge_5_iSyncFifo_iF_iFifos_zFcnt_pop_empty
266	—"—	dut_iPs2_iBridge_5_iSyncFifo_iF_iFifos_zFcnt_push_full

id	instance	signal
267	—"—	dut_iPs2_iBridge_4_assert_reset
268	—"—	dut_iPs2_iBridge_4_iSyncFifo_iF_iFifos_zFcnt_pop_empty
269	—"—	dut_iPs2_iBridge_4_iSyncFifo_iF_iFifos_zFcnt_push_full
270	—"—	dut_iPs2_iBridge_3_assert_reset
271	—"—	dut_iPs2_iBridge_3_iSyncFifo_iF_iFifos_zFcnt_pop_empty
272	—"—	dut_iPs2_iBridge_3_iSyncFifo_iF_iFifos_zFcnt_push_full
273	—"—	dut_iPs2_iBridge_2_assert_reset
274	—"—	dut_iPs2_iBridge_2_iSyncFifo_iF_iFifos_zFcnt_pop_empty
275	—"—	dut_iPs2_iBridge_2_iSyncFifo_iF_iFifos_zFcnt_push_full
276	—"—	dut_iPs2_iBridge_1_assert_reset
277	—"—	dut_iPs2_iBridge_1_iSyncFifo_iF_iFifos_zFcnt_pop_empty
278	—"—	dut_iPs2_iBridge_1_iSyncFifo_iF_iFifos_zFcnt_push_full
279	—"—	dut_iPs2_iBridge_0_assert_reset
280	—"—	dut_iPs2_iBridge_0_iSyncFifo_iF_iFifos_zFcnt_pop_empty
281	—"—	dut_iPs2_iBridge_0_iSyncFifo_iF_iFifos_zFcnt_push_full
282	—"—	dut_iPs2_iSplitter_1_assert_noend
283	—"—	dut_iPs2_iSplitter_1_assert_ptr
284	—"—	dut_iPs2_iSplitter_0_assert_noend
285	—"—	dut_iPs2_iSplitter_0_assert_ptr
286	—"—	dut_iPs2_iSplitter_1_used_mem
287	—"—	dut_iPs2_iSplitter_1_used_mem
288	—"—	dut_iPs2_iSplitter_1_used_mem
289	—"—	dut_iPs2_iSplitter_1_used_mem
290	—"—	dut_iPs2_iSplitter_1_used_mem
291	—"—	dut_iPs2_iSplitter_1_used_mem
292	—"—	dut_iPs2_iSplitter_1_used_mem
293	—"—	dut_iPs2_iSplitter_0_used_mem
294	—"—	dut_iPs2_iSplitter_0_used_mem
295	—"—	constant-295
296	pa_top.switch.epp0	constant-296
297	—"—	dut_iEpp_assert_ipkt
298	—"—	dut_iEpp_assert_opkt
299	—"—	epp_ipkt_bus {18'data, 8'valid_bytes, 4'id, 1'last, 1'first}
300	—"—	epp_opkt_bus {18'data, 8'valid_bytes, 4'id, 1'last, 1'first}
301	—"—	dut_iEpp_iDropper_da_0
302	—"—	dut_iEpp_iDropper_da_1
303	—"—	dut_iEpp_iDropper_dbg_drop
304	—"—	dut_iEpp_iDropper_dbg_ifirst
305	—"—	dut_iEpp_iDropper_dbg_ilast
306	—"—	dut_iEpp_iDropper_sa_0
307	—"—	dut_iEpp_iDropper_sa_1
308	—"—	pa.top.switch.epp0.iPacketassertpm {8'valid_bytes, 4'port, 1'last, 1'first}
309	—"—	pa.top.switch.epp0.iPacketassertin {8'valid_bytes, 4'port, 1'last, 1'first}
310	—"—	constant-310
311	pa_top.switch.epp0.pm	constant-311
312	—"—	pm_fifo_overflow
313	—"—	dut_dbg_fifo_full
314	—"—	halt_from_pm
315	—"—	dut_iFifoa_debug_in
316	—"—	dut_iFifoa_debug_out
317	—"—	constant-317
318	pa_top.switch.ingress_common	constant-318
319	—"—	dut_iLearnage_iHitUpdate_iFifo_0_iF_iFifos_zFcnt_pop_empty
320	—"—	dut_iLearnage_iHitUpdate_iFifo_0_iF_iFifos_zFcnt_push_full
321	—"—	dut_iMbsc_iFloodMc_reg_stat
322	—"—	dut_iMbsc_iFloodUc_reg_stat
323	—"—	dut_iMbsc_iMc_reg_stat
324	—"—	dut_iMbsc_iBc_reg_stat
325	—"—	constant-325
326	pa_top.switch.interface_common	constant-326
327	—"—	dut_zFaii_iMf_zMf_1_item
328	—"—	dut_zFaip_iMf_zMf_1_item
329	—"—	dut_zFaie_iMf_zMf_1_item
330	—"—	dut_zFaiq_iMf_zMf_1_item
331	—"—	dut_zFais_iMf_zMf_1_item
332	—"—	constant-332

Table 31.9: Debug Selection Map

31.6 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** and **PTP Tick Select**.

Pin	Direction	Size	Description
`debug_write_data`	In	2	The debug write input data. Must be tied low for normal switch operation.

Table 31.10: The Debug Write interface

Chapter 32

Configuration Interface

The configuration interface is used for monitoring the core and for configuration of internal registers and tables. The pins are described in Table 31.6 on page 162.

Even if you are just doing a quick and dirty bus implementation, please read the short implementation note, Section 32.5, at the end of this chapter.

32.1 Request Types

Requests can be of either read or write type. Asserting the read- and write-enables concurrently is not supported. Reads and writes can be of `DEFAULT` or `ACCUMULATOR` type. Although registers and tables where the data width is less than or equal to the configuration interface data width support only the `DEFAULT` type. The purpose of the `ACCUMULATOR` request type is to access data that is wider than the bus without the risk of data inconsistency.

Requests for registers which exceed the bus width are discussed in more detail in Section 32.4 below.

32.2 Reply Types

A write access will produce either a `WOK`, reply indicating that the write was successful, or a `FAIL` reply, indicating that the write failed. A read access will similarly produce either a `ROK` or a `FAIL` response. When the response is `ROK`, the read data is available on the data pins. All valid requests will result in a reply, but no reply, not even a fail, will be produced for an access to an unmapped address.

If the core clock frequency is set below the recommended frequency and the core is running at full capacity then a request to a memory may take an infinite time to complete. In practice the recommended frequency is set so that there are sufficient cycles for firmware accesses even under full load.

Figure 32.1 shows two write accesses to the same register taking different time to complete.

32.3 Transaction Identifier

This core has only a single transaction ID, so the request_id shall be tied low. Normally you should always wait for a transaction to return a reply before issuing a new transaction, because issuing concurrent transactions can cause the loss of replies. But for writes to registers it is relatively safe to re-use the same ID for back-to-back accesses. The replies may be inconsistent, but since registers (unlike tables) will never block an access, the writes will succeed.

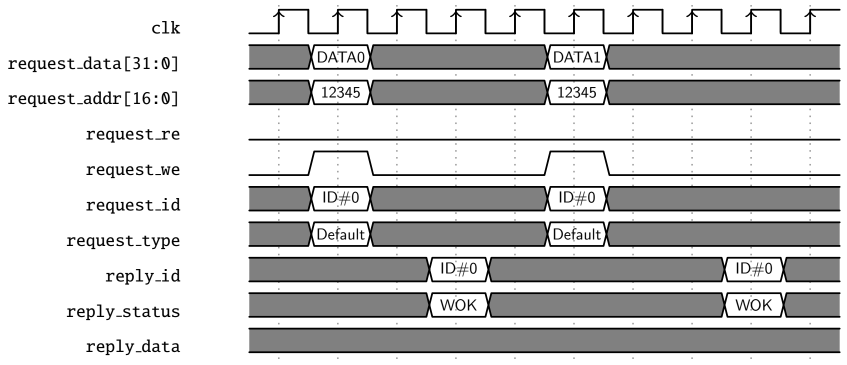

Figure 32.1: Completion time, even to the same register, may vary

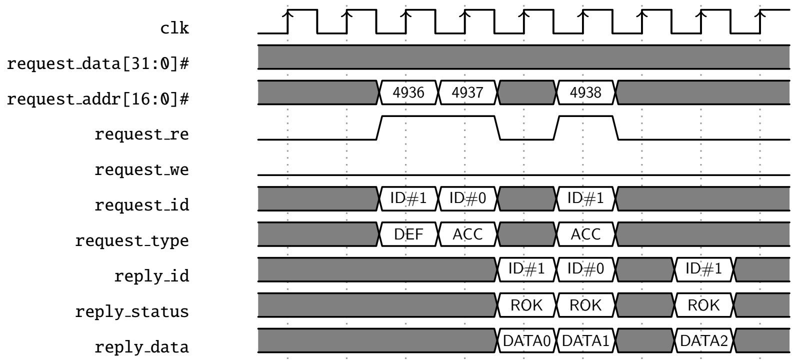

Figure 32.2: Read from a wide register

32.4 Atomic Wide Access - Accumulator Access

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 tough the bus width is narrower than the data. Accesses to the accumulator are done using the same address that would be used to directly access the data it shadows, the only difference being that the request type is set to `ACCUMULATOR`.

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 request_data, and fill it out with the data in the accumulator. Thus writing data wider than the bus cannot be done without taking the accumulator into account.

If only a part of the data is to be written, the most efficient approach is to do a default read (loading the accumulator) followed by a default write. The accesses should be issued as close as possible, to minimize the risk of the core updating the memory data while the accumulator is loaded. 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.

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 table will result in index 1 starting at address 4, with address 3 unused and address 2 only containing a single valid bit.

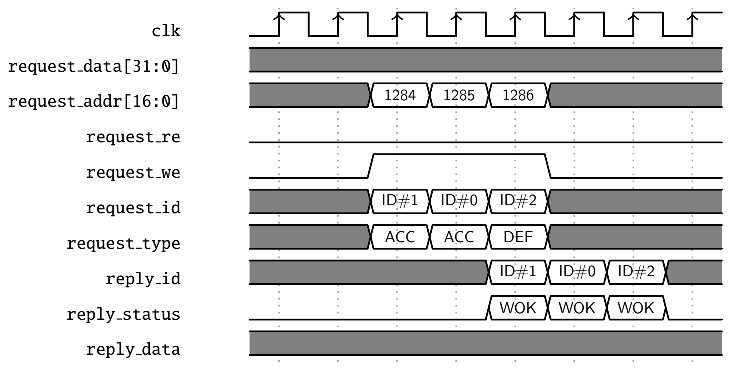

Figure 32.3: Write to a wide register

32.5 Implementation note

Mapping the pins of the configuration bus to your bus of choice is usually a non-trivial task. We recommend two things:

- Map the request_type pin to an unused address bit
- Do all requests within wide registers or tables in consecutive order.

The first makes it easy to do default and accumulator accesses using any bus protocol. The second makes it easier to later optimize your access performance should the need arise.

So, a read from a wide register would start with a default read on the lowest address of the register, and then continue upward using accumulator accesses. A write would start with an accumulator write to the lowest address, continue upward with accumulator writes until the last address where a default write finishes the transaction. The software API implementation provided with the switch supports both of these thereby hiding it completely for the software that use the API.

Note that for this to work your bus needs to be set up to guarantee the order of accesses in the bridge to the configuration bus.

Chapter 33

Implementation

33.1 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.

33.1.1 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 ↔ IPP	0
IPP ↔ PB/BM	0
PB ↔ BM	0
BM ↔ EPP	0
EPP ↔ PS	0

Table 33.1: 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 33.2: The settings for input and output flops for the floorplan blocks

The pipeline settings used when generating this core are shown in Table 33.1, and the input/output flops are listed in Table 33.2[1].

33.1.2 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.

33.1.3 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.

33.2 Memory wrappers

The memories in the core are instantiated using the verilog_memory.v wrapper. It is expected that this wrapper is replaced, or modified, by the customer to instanciate appropriate memory macros. The macros needed are listed in Table 33.2. For memories with the *write_through* attribute set, simultaneous reading and writing the of same address is expected to yield the write data as read result. For memories with *write_through* set to 0 simultaneous reading and writing to the same address shall not occur.

type	**width**	**depth**	**write through**	**write mask**	**input flops**	**output flops**
dp	240	40	1	None	0	1
dp	120	140	1	None	0	1
dp	286	32	1	None	0	1
dp	286	16	1	None	0	1
dp	457	128	1	None	0	1
dp	457	8	1	None	0	1
dp	102	4096	1	None	0	1
dp	27	256	1	None	0	1
dp	57	256	1	None	0	1
dp	57	256	0	None	0	1
dp	11	1040	1	None	0	1
dp	11	1040	0	None	0	1
dp	18	128	1	None	0	1
dp	64	32	1	None	0	1
dp	92	32	1	None	0	1
dp	114	32	1	None	0	1
dp	1056	11	0	None	0	1

[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.

dp	279	190	0	None	0	1
dp	240	45	0	None	0	1
dp	4	1536	0	None	0	1
dp	8	1536	0	None	0	1
dp	20	1536	0	None	0	1
dp	17	1536	0	None	0	1
dp	12	1536	0	None	0	1
dp	31	1536	1	None	0	1
dp	1200	1536	0	None	0	1
dp	33	512	0	None	1	1
dp	255	415	0	None	0	1
dp	240	80	0	None	0	1
dp	120	280	0	None	0	1

Table 33.3: The memory macros needed for this core. dp=two ports, one read and one write, running on the same clock.

Only memories with 2048 bits or more have been generated as a memory instance. Smaller memories are created as arrays of flops in the verilog source code. To change the criterium for making a memory as an instance or as an array of flops, new RTL has to be generated[2].

33.3 Dual ported memories

All memories are dual ported. Unless the frequency would be prohibitively high, the best approach is to implement the memories using single ported memory macros clocked at twice the speed. Note in the example timing diagram that the write is done in the first clock cycle to satisfy the *write_through* criterium. For memories that are not *write_through* it may be desirable for timing reasons to have the read in the first clock cycle.

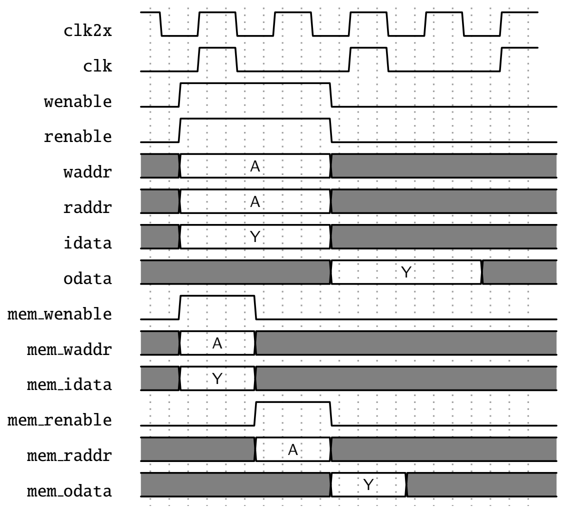

Figure 33.1: Timing diagram for a single ported memory used in the dual ported memory wrapper. In this case a concurrent read and write to the same address of a memory wrapper set for one cycle latency and with the write through attribute set.

[2]Although, any instantiated memory wrapper can of course be left as is, and thus be implemented as an array of flops in synthesis.

There is no dedicated double frequency clock connected to the memories, it has to be provided using the *meminst_in busses to the memory wrappers.

33.4 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

Which setting is used for each memory instance can be seen in the *input flops* and *output flops* columns of Table 33.2.

33.5 Lint set up

For spyglass linting the following settings are assumed:

- set_parameter ignore_local_variables yes
- set_parameter handle_zero_padding “W362”

33.5.1 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 34

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 32, and the pins for the configuration interfaces are described in Section 31.3.

34.1 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 34.1. The total address space used by this core is 38908 addresses.

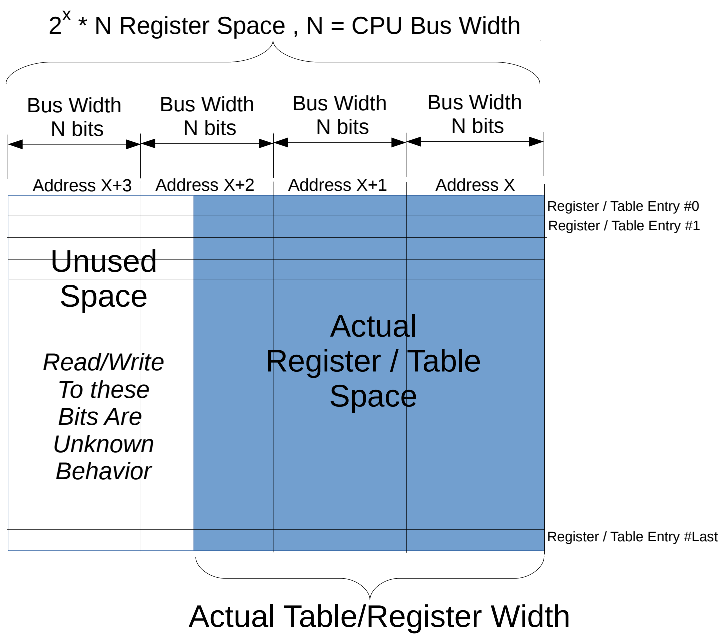
2^x * N Register Space , N = CPU Bus Width
Bus Width N bits
Bus Width N bits
Bus Width N bits
Bus Width N bits
Address X+3
Address X+2
Address X+1
Address X
Register / Table Entry #0
Register / Table Entry #1
Unused Space
Read/Write To these Bits Are Unknown Behavior
Actual Register / Table Space
Register / Table Entry #Last
Actual Table/Register Width

Figure 34.1: Address space usage by tables

34.2 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*.

34.3 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 PTP Tick Configuration PTP Tick Select CPU Port Scratch
rx_length_ref	MAC RX Maximum Packet Length[0..8]
rx_length_drop	MAC RX Broken Packets[0..8] MAC RX Short Packet Drop[0..8] MAC RX Long Packet Drop[0..8]
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_values	L2 Broadcast Storm Control Current Size
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_values	L2 Multicast Storm Control Current Size
l2_multicast_storm_control_misc	L2 Multicast Storm Control Enable
l2_unknown_unicast_storm_control_rate_settings	L2 Unknown Unicast Storm Control Rate Configuration
l2_unknown_unicast_storm_control_bucket_settings	L2 Unknown Unicast Storm Control Bucket Capacity Configuration L2 Unknown Unicast Storm Control Bucket Threshold Configuration
l2_unknown_unicast_storm_control_values	L2 Unknown Unicast Storm Control Current Size
l2_unknown_unicast_storm_control_misc	L2 Unknown Unicast Storm Control Enable

Bank Name	**Connected Registers or Tables**
l2_unknown_multicast_storm_control_rate_settings	L2 Unknown Multicast Storm Control Rate Configuration
l2_unknown_multicast_storm_control_bucket_settings	L2 Unknown Multicast Storm Control Bucket Capacity Configuration L2 Unknown Multicast Storm Control Bucket Threshold Configuration
l2_unknown_multicast_storm_control_values	L2 Unknown Multicast Storm Control Current Size
l2_unknown_multicast_storm_control_misc	L2 Unknown Multicast Storm Control Enable
le_ae_status	Learning Conflict Learning Overflow
le_ae_control	Learning And Aging Enable Hardware Learning Configuration[0..8] Time to Age
age_cam_register_bank	L2 Aging Collision Table[0..15]
mac_cnt_register_bank	Hardware Learning Counter[0..8]
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 Ingress Spanning Tree Drop: Blocking L2 Lookup Drop Ingress Rate Control Drop Ingress Packet Filtering Drop Reserved MAC DA Drop Reserved MAC SA Drop VLAN Member Drop Minimum Allowed VLAN Drop Maximum Allowed VLAN Drop IP Checksum Drop L2 Reserved Multicast Address Drop Ingress Configurable ACL Drop Attack Prevention Drop ARP Decoder Drop RARP Decoder Drop L2 IEEE 1588 Decoder Drop L4 IEEE 1588 Decoder Drop IEEE 802.1X and EAPOL Decoder Drop SCTP Decoder Drop LACP Decoder Drop AH Decoder Drop ESP Decoder Drop DNS Decoder Drop BOOTP and DHCP Decoder Drop CAPWAP Decoder Drop GRE Decoder Drop L2 Action Table Special Packet Type Drop L2 Action Table Drop L2 Action Table Port Move Drop L2 Destination Table SA Lookup Drop Source Port Default ACL Action Drop
count_opkt_pa top switch ipp0 conf	IPP Packet Head Counter IPP Packet Tail Counter

Bank Name	**Connected Registers or Tables**
Ingress Configurable ACL 0 Pre Lookup	Ingress Configurable ACL 0 Pre Lookup
Ingress Configurable ACL 0 Large Table	Ingress Configurable ACL 0 Large Table
Ingress Configurable ACL 0 Small Table	Ingress Configurable ACL 0 Small Table
Ingress Configurable ACL 0 TCAM Answer	Ingress Configurable ACL 0 TCAM Answer
Ingress Configurable ACL 1 Large Table	Ingress Configurable ACL 1 Large Table
Ingress Configurable ACL 1 Small Table	Ingress Configurable ACL 1 Small Table
VLAN Table	VLAN Table
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
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
L2 Action Table	L2 Action Table
L2 Action Table Source Port	L2 Action Table Source Port
Ingress Rate Control Type	Ingress Rate Control Type
ipp_register_bank_ss0	Link Aggregation Ctrl Ingress Ethernet Type for VLAN tag ICMP Length Check SCTP Packet Decoder Options AH Header Packet Decoder Options ESP Header Packet Decoder Options Ingress Configurable ACL 0 Selection Ingress Configurable ACL 1 Selection Check IPv4 Header Checksum Force Non VLAN Packet To Specific Color Force Unknown L3 Packet To Specific Color Egress Spanning Tree State Forward From CPU Port Move Options L2 Action Table Egress Port State L2 Multicast Handling Ingress MMP Drop Mask Debug srcPort Debug dstPortmask FRER Sequence Number FRER Configuration Stream Handle To FRER Mapping Table Stream Filter Lookup Table Enable Enqueue To Ports And Queues Flooding Action Send to Port Link Aggregation To Physical Ports Members Link Aggregate Weight Ingress Egress Port Packet Type Filter Allow Special Frame Check For L2 Action Table Egress Multiple Spanning Tree State Hairpin Enable L2 Multicast Table L2 Aging Collision Shadow Table MPLS EXP Field To Packet Color Mapping Table IPv6 Class of Service Field To Packet Color Mapping Table IPv4 TOS Field To Packet Color Mapping Table VLAN PCP And DEI To Color Mapping Table

Bank Name	Connected Registers or Tables
	VID to Queue Assignment L4 Protocol to Queue Assignment Ethernet Type to Queue Assignment Egress Queue Priority Selection Force Unknown L3 Packet To Specific Egress Queue Force Non VLAN Packet To Specific Queue VLAN PCP To Queue Mapping Table MPLS EXP Field To Egress Queue Mapping Table TCP/UDP Flag Rules Ingress Multiple Spanning Tree State Ingress VID Ethernet Type Range Assignment Answer Ingress VID Inner VID Range Assignment Answer Ingress VID Outer VID Range Assignment Answer Ingress VID MAC Range Assignment Answer Ingress Configurable ACL 1 Rules Setup Ingress Configurable ACL 1 Pre Lookup Ingress Configurable ACL 0 Rules Setup Ingress Port Packet Type Filter SMON Set Search Ingress Admission Control Initial Pointer L2 Reserved Multicast Address Action Link Aggregation Membership Source Port Table DA or SA MAC to Queue Assignment Ingress VID MAC Range Search Data Source Port Default ACL Action Ingress Configurable ACL 1 TCAM Answer Reserved Source MAC Address Range Reserved Destination MAC Address Range Send to CPU LACP Packet Decoder Options LLDP Configuration L2 Reserved Multicast Address Base ARP Packet Decoder Options RARP Packet Decoder Options IEEE 1588 L2 Packet Decoder Options IEEE 802.1X and EAPOL Packet Decoder Options GRE Packet Decoder Options DNS Packet Decoder Options BOOTP and DHCP Packet Decoder Options CAPWAP Packet Decoder Options Ingress Transmission Gate Current Status CPU Reason Code Operation L2 Lookup Collision Table Masks L2 Lookup Collision Table L4 Port Range to Queue Assignment Ingress VID Ethernet Type Range Search Data Ingress VID Inner VID Range Search Data Ingress VID Outer VID Range Search Data IEEE 1588 L4 Packet Decoder Options Ingress Configurable ACL 1 Search Mask Ingress Configurable ACL 1 TCAM Ingress Configurable ACL 0 Search Mask IP Address To Queue Assignment Ingress Configurable ACL 0 TCAM

Bank Name	Connected Registers or Tables
ipp_register_bank_misc_ss0	Ingress Drop Options
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_aclConfStatisticsBlock	Ingress Configurable ACL Match Counter[0..31]
ingress_rate_control_rate_settings	Ingress Rate Control Rate Configuration
ingress_rate_control_bucket_settings	Ingress Rate Control Bucket Capacity Configuration Ingress Rate Control Bucket Threshold Configuration
ingress_rate_control_values	Ingress Rate Control Current Size
ingress_rate_control_misc	Ingress Rate Control Enable
count_ipp0_egressDropStatisticsBlock	Queue Off Drop[0..8] Egress Spanning Tree Drop[0..8] MBSC Drop[0..8] Ingress-Egress Packet Filtering Drop[0..8] L2 Action Table Per Port Drop[0..8]
count_dropipp0_igrPortMibBlock	Ingress Received and Dropped Counter[0..8]
count_saipp0_igrPortMibBlock	Ingress MAC SA Change Counter[0..8]
regbank_ipp0_ingressGateCtrlBlock gate	Ingress Transmission Gate Base Tick Ingress Transmission Gate Update Status Ingress Transmission Gate Enabled[0..31] Ingress Transmission Gate Current Time Ingress Transmission Gate Configuration[0..31]
Ingress Transmission Gate List	Ingress Transmission Gate List
regbank_ctrl_ipp0_ingressGateCtrlBlock gate	Ingress Transmission Gate Update[0..31]
bk_stream_filter_0	Max SDU Filter
bk_sdu_blocking_0	Max SDU Filter Blocking
bk_gate_blocking_en_0	Stream Gate Blocking Enable
bk_gate_rx_blocking_0	Stream Gate Invalid RX Blocking
bk_gate_msdu_blocking_0	Stream Gate Max MSDU Blocking
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_psfp_match_0	PSFP Matching Frame Counter
bk_psfp_sdu_pass_0	PSFP Passing SDU Counter
bk_psfp_sdu_drop_0	PSFP Not Passing SDU Counter
bk_psfp_gate_pass_0	PSFP Passing Frame Counter
bk_psfp_gate_drop_0	PSFP Not Passing Frame Counter
bk_psfp_mmp_drop_0	PSFP Red Frames Counter
config_frer_0	Individual Recovery Reset Sequence Recovery Reset Recovery Tick Latent Error Detection Tick Individual Recovery Config Sequence Recovery Config Latent Error Detection Configuration
frer_stat_individual_passed_0	Individual Recovery Passed Counter
frer_stat_individual_discarded_0	Individual Recovery Discarded Counter
frer_stat_individual_out_of_order_0	Individual Recovery Out Of Order Counter

Bank Name	**Connected Registers or Tables**
frer_stat_individual_rogue_0	Individual Recovery Rogue Counter
frer_stat_individual_lost_0	Individual Recovery Lost Counter
frer_stat_individual_tagless_0	Individual Recovery Tagless Counter
frer_stat_sequence_passed_0	Sequence Recovery Passed Counter
frer_stat_sequence_discarded_0	Sequence Recovery Discarded Counter
frer_stat_sequence_out_of_order_0	Sequence Recovery Out Of Order Counter
frer_stat_sequence_rogue_0	Sequence Recovery Rogue Counter
frer_stat_sequence_lost_0	Sequence Recovery Lost Counter
frer_stat_sequence_tagless_0	Sequence Recovery Tagless Counter
frer_led_status_0	Latent Error Detection Status
bk_frer_stat_0	FRER Drop
bk_erm_ss0	ERM Yellow Configuration Resource Limiter Set[0..3] ERM Red Configuration Egress Resource Manager Pointer[0..8]
count_erm_ss0	Egress Resource Manager Drop[0..8]
pb_info_regbank_ss0	Packet Buffer Status
count_drop_pa top switch pb0	Buffer Overflow Drop Ingress Resource Manager Drop
pb_queue_manage_register_bank_ss0	Map Queue to Priority[0..8]
count_drop_pa top switch pb0 iRequeue	Re-queue Overflow Drop
pfc_regbank_rsv_size_ss0	Port/TC Reserved[0..71]
pfc_regbank_port_rsv_size_ss0	Port Reserved[0..8]
PFC Inc Counters for ingress ports 0 to 8	PFC Inc Counters for ingress ports 0 to 8
PFC Dec Counters for ingress ports 0 to 8	PFC Dec Counters for ingress ports 0 to 8
pfc_regbank_cmn_misc_ss0	Port FFA Used[0..8] Port Used[0..8] TC FFA Used[0..7] FFA Used PFC FFA Used non-PFC
pfc_regbank_pause_settings1_ss0	Port Pause Settings[0..8]
pfc_regbank_taildrop_settings0_ss0	Port Tail-Drop Settings[0..8]
pfc_regbank_misc_ss0	Xon FFA Threshold Xoff FFA Threshold Tail-Drop FFA Threshold TC Xon FFA Threshold[0..7] TC Xoff FFA Threshold[0..7] TC Tail-Drop FFA Threshold[0..7] Port Xon FFA Threshold[0..8] Port Xoff FFA Threshold[0..8] Port Tail-Drop FFA Threshold[0..8] Port/TC Xon Total Threshold[0..71] Port/TC Xoff Total Threshold[0..71] Port/TC Tail-Drop Total Threshold[0..71]
qe_register_bank_ss0_sp0	Egress Port Depth[0..8] Egress Queue Depth[0..71]
pb_r_register_bank_ss0	Minimum Buffer Free
disable_queue_output_register_bank_ss0	Output Disable[0..8]
dwrr_bucket_capacity_settings_ss0	DWRR Bucket Capacity Configuration[0..8]
dwrr_bucket_misc_settings_ss0	DWRR Bucket Misc Configuration[0..8]
dwrr_weight_settings_ss0	DWRR Weight Configuration[0..71]
dwrr_values_ss0	DWRR Current Size[0..71]
dwrr_rank_ss0	DWRR Rank[0..8]
regbank_pa top switch pb0 gate	Egress Transmission Gate Base Tick

Bank Name	Connected Registers or Tables
	Egress Transmission Gate Update Status Egress Transmission Gate Enabled[0..8] Egress Transmission Gate Current Time Egress Transmission Gate Configuration[0..8]
Egress Transmission Gate List	Egress Transmission Gate List
regbank_ctrl_pa top switch pb0 gate	Egress Transmission Gate Update[0..8]
queue_shaper_rate_settings	Queue Shaper Rate Configuration
queue_shaper_bucket_settings	Queue Shaper Bucket Capacity Configuration Queue Shaper Bucket Threshold Configuration
queue_shaper_values	Queue Shaper Current Size
queue_shaper_misc	Queue Shaper Enable
prio_shaper_rate_settings	Prio Shaper Rate Configuration
prio_shaper_bucket_settings	Prio Shaper Bucket Capacity Configuration Prio Shaper Bucket Threshold Configuration
prio_shaper_values	Prio Shaper Current Size
prio_shaper_misc	Prio Shaper Enable
port_shaper_rate_settings	Port Shaper Rate Configuration
port_shaper_bucket_settings	Port Shaper Bucket Capacity Configuration Port Shaper Bucket Threshold Configuration
port_shaper_values	Port Shaper Current Size
port_shaper_misc	Port Shaper Enable
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..8]
count_pa top switch epp0 conf	Unknown Egress Drop[0..8] Egress Port Disabled Drop[0..8] Egress Port Filtering Drop[0..8] EPP PM Drop
count_pktpa top switch epp0 conf	Dequeued Packets
count_bytpa top switch epp0 conf	Dequeued Bytes
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 Queue To PCP And CFI/DEI Mapping Table	Egress Queue To PCP And CFI/DEI Mapping Table
Egress VLAN Translation Large Table	Egress VLAN Translation Large Table
Egress VLAN Translation Small Table	Egress VLAN Translation Small Table
Egress VLAN Translation TCAM Answer	Egress VLAN Translation TCAM Answer
epp_register_bank_ss0	Output Mirroring Table Egress Ethernet Type for VLAN tag Egress VLAN Translation Selection Disable CPU tag on CPU Port Egress Port VID Operation Egress VLAN Translation Search Mask Egress RSPAN Configuration Egress VLAN Translation TCAM
count_opkt_pa top switch ps0 ps_wrap_bridge	PS Packet Head Counter PS Packet Tail Counter

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

34.4 Registers and Tables in Alphabetical Order

Name	Address Range
AH Decoder Drop	1450
AH Header Packet Decoder Options	30547
ARP Decoder Drop	1443
ARP Packet Decoder Options	32852
Allow Special Frame Check For L2 Action Table	30998 - 31001
Attack Prevention Drop	1442
BOOTP and DHCP Decoder Drop	1453
BOOTP and DHCP Packet Decoder Options	32864
Buffer Free	1
Buffer Overflow Drop	36750
CAPWAP Decoder Drop	1454
CAPWAP Packet Decoder Options	32866
CPU Port	6
CPU Reason Code Operation	32932 - 32939
Check IPv4 Header Checksum	30551
Color Remap From Egress Port	38556 - 38573
Color Remap From Ingress Admission Control	38574 - 38637
Core Tick Configuration	2
Core Tick Select	3
Core Version	0
DA or SA MAC to Queue Assignment	32542 - 32685
DNS Decoder Drop	1452
DNS Packet Decoder Options	32862
DWRR Bucket Capacity Configuration	37394 - 37402
DWRR Bucket Misc Configuration	37403 - 37411
DWRR Current Size	37484 - 37555
DWRR Rank	37556 - 37564
DWRR Weight Configuration	37412 - 37483
Debug dstPortmask	30561
Debug srcPort	30560
Dequeued Bytes	38464 - 38535
Dequeued Packets	38392 - 38463
Disable CPU tag on CPU Port	38761
Drain Port	38354
Drain Port Drop	38355 - 38363
EPP PM Drop	38391
EPP Packet Head Counter	38536
EPP Packet Tail Counter	38537
ERM Red Configuration	36730
ERM Yellow Configuration	36720
ESP Decoder Drop	1451
ESP Header Packet Decoder Options	30548
Egress Ethernet Type for VLAN tag	38759

Name	Address Range
Egress Multiple Spanning Tree State	31002 - 31017
Egress Port Configuration	38538 - 38555
Egress Port Depth	37303 - 37311
Egress Port Disabled Drop	38373 - 38381
Egress Port Filtering Drop	38382 - 38390
Egress Port VID Operation	38762 - 38825
Egress Queue Depth	37312 - 37383
Egress Queue Priority Selection	31751 - 31759
Egress Queue To PCP And CFI/DEI Mapping Table	38638 - 38645
Egress RSPAN Configuration	38828 - 38845
Egress Resource Manager Drop	36740 - 36748
Egress Resource Manager Pointer	36731 - 36739
Egress Spanning Tree Drop	34866 - 34874
Egress Spanning Tree State	30554
Egress Transmission Gate Base Tick	37565
Egress Transmission Gate Configuration	37578 - 37613
Egress Transmission Gate Current Time	37576
Egress Transmission Gate Enabled	37567 - 37575
Egress Transmission Gate List	37614 - 37677
Egress Transmission Gate Update	37678 - 37686
Egress Transmission Gate Update Status	37566
Egress VLAN Translation Large Table	38646 - 38709
Egress VLAN Translation Search Mask	38826
Egress VLAN Translation Selection	38760
Egress VLAN Translation Small Table	38710 - 38741
Egress VLAN Translation TCAM	38846 - 38861
Egress VLAN Translation TCAM Answer	38742 - 38749
Empty Mask Drop	1427
Enable Enqueue To Ports And Queues	30706 - 30714
Ethernet Type to Queue Assignment	31715 - 31750
FFA Used PFC	37013
FFA Used non-PFC	37014
FRER Configuration	30594 - 30625
FRER Drop	36687
FRER Sequence Number	30562 - 30593
Flooding Action Send to Port	30715 - 30723
Flow Classification And Metering Drop	35308
Force Non VLAN Packet To Specific Color	30552
Force Non VLAN Packet To Specific Queue	31769 - 31777
Force Unknown L3 Packet To Specific Color	30553
Force Unknown L3 Packet To Specific Egress Queue	31760 - 31768
Forward From CPU	30555
GRE Decoder Drop	1455
GRE Packet Decoder Options	32860
Hairpin Enable	31018 - 31026
Hardware Learning Configuration	281 - 289
Hardware Learning Counter	308 - 316
ICMP Length Check	30545
IEEE 1588 L2 Packet Decoder Options	32856
IEEE 1588 L4 Packet Decoder Options	33076
IEEE 802.1X and EAPOL Decoder Drop	1447
IEEE 802.1X and EAPOL Packet Decoder Options	32858
IP Address To Queue Assignment	33668 - 34243

Name	Address Range
IP Checksum Drop	1439
IPP Empty Destination Drop	1425
IPP PM Drop	1424
IPP Packet Head Counter	1461
IPP Packet Tail Counter	1462
IPv4 TOS Field To Egress Queue Mapping Table	25591 - 25846
IPv4 TOS Field To Packet Color Mapping Table	31371 - 31626
IPv6 Class of Service Field To Egress Queue Mapping Table	25847 - 26102
IPv6 Class of Service Field To Packet Color Mapping Table	31115 - 31370
Individual Recovery Config	35759 - 35886
Individual Recovery Discarded Counter	36143 - 36206
Individual Recovery Lost Counter	36335 - 36398
Individual Recovery Out Of Order Counter	36207 - 36270
Individual Recovery Passed Counter	36079 - 36142
Individual Recovery Reset	35661 - 35724
Individual Recovery Rogue Counter	36271 - 36334
Individual Recovery Tagless Counter	36399 - 36462
Ingress Admission Control Current Status	35533 - 35564
Ingress Admission Control Initial Pointer	32113 - 32240
Ingress Admission Control Mark All Red	35341 - 35372
Ingress Admission Control Mark All Red Enable	35309 - 35340
Ingress Admission Control Reset	35501 - 35532
Ingress Admission Control Token Bucket Configuration	35373 - 35500
Ingress Configurable ACL 0 Large Table	1719 - 3766
Ingress Configurable ACL 0 Pre Lookup	1463 - 1718
Ingress Configurable ACL 0 Rules Setup	32094 - 32101
Ingress Configurable ACL 0 Search Mask	33652
Ingress Configurable ACL 0 Selection	30549
Ingress Configurable ACL 0 Small Table	3767 - 4790
Ingress Configurable ACL 0 TCAM	34244 - 34499
Ingress Configurable ACL 0 TCAM Answer	4791 - 4854
Ingress Configurable ACL 1 Large Table	4855 - 8950
Ingress Configurable ACL 1 Pre Lookup	31838 - 32093
Ingress Configurable ACL 1 Rules Setup	31830 - 31837
Ingress Configurable ACL 1 Search Mask	33108
Ingress Configurable ACL 1 Selection	30550
Ingress Configurable ACL 1 Small Table	8951 - 9206
Ingress Configurable ACL 1 TCAM	33140 - 33651
Ingress Configurable ACL 1 TCAM Answer	32738 - 32801
Ingress Configurable ACL Drop	1441
Ingress Configurable ACL Match Counter	34533 - 34564
Ingress Drop Options	34500
Ingress Egress Port Packet Type Filter	30989 - 30997
Ingress Ethernet Type for VLAN tag	30544
Ingress MAC SA Change Counter	34911 - 34919
Ingress MMP Drop Mask	30559
Ingress Multiple Spanning Tree State	31798 - 31813
Ingress Packet Filtering Drop	1433
Ingress Port Packet Type Filter	32102 - 32110
Ingress Rate Control Bucket Capacity Configuration	34637 - 34708
Ingress Rate Control Bucket Threshold Configuration	34709 - 34780
Ingress Rate Control Current Size	34781 - 34852
Ingress Rate Control Drop	1432

Name	Address Range
Ingress Rate Control Enable	34853
Ingress Rate Control Rate Configuration	34565 - 34636
Ingress Rate Control Type	30471 - 30542
Ingress Received and Dropped Counter	34902 - 34910
Ingress Resource Manager Drop	36751
Ingress Spanning Tree Drop: Blocking	1430
Ingress Spanning Tree Drop: Learning	1429
Ingress Spanning Tree Drop: Listen	1428
Ingress Transmission Gate Base Tick	34920
Ingress Transmission Gate Configuration	34956 - 35083
Ingress Transmission Gate Current Status	32868 - 32931
Ingress Transmission Gate Current Time	34954
Ingress Transmission Gate Enabled	34922 - 34953
Ingress Transmission Gate List	35084 - 35147
Ingress Transmission Gate Update	35148 - 35179
Ingress Transmission Gate Update Status	34921
Ingress VID Ethernet Type Range Assignment Answer	31814 - 31817
Ingress VID Ethernet Type Range Search Data	33052 - 33059
Ingress VID Inner VID Range Assignment Answer	31818 - 31821
Ingress VID Inner VID Range Search Data	33060 - 33067
Ingress VID MAC Range Assignment Answer	31826 - 31829
Ingress VID MAC Range Search Data	32686 - 32701
Ingress VID Outer VID Range Assignment Answer	31822 - 31825
Ingress VID Outer VID Range Search Data	33068 - 33075
Ingress-Egress Packet Filtering Drop	34884 - 34892
L2 Action Table	30215 - 30342
L2 Action Table Drop	1457
L2 Action Table Egress Port State	30557
L2 Action Table Per Port Drop	34893 - 34901
L2 Action Table Port Move Drop	1458
L2 Action Table Source Port	30343 - 30470
L2 Action Table Special Packet Type Drop	1456
L2 Aging Collision Shadow Table	31091 - 31106
L2 Aging Collision Table	292 - 307
L2 Aging Status Shadow Table	26103 - 27126
L2 Aging Table	317 - 1340
L2 Broadcast Storm Control Bucket Capacity Configuration	137 - 145
L2 Broadcast Storm Control Bucket Threshold Configuration	146 - 154
L2 Broadcast Storm Control Current Size	155 - 163
L2 Broadcast Storm Control Enable	164
L2 Broadcast Storm Control Rate Configuration	128 - 136
L2 DA Hash Lookup Table	27127 - 29174
L2 Destination Table	29175 - 30214
L2 Destination Table SA Lookup Drop	1459
L2 IEEE 1588 Decoder Drop	1445
L2 Lookup Collision Table	32948 - 32979
L2 Lookup Collision Table Masks	32940 - 32947
L2 Lookup Drop	1431
L2 Multicast Handling	30558
L2 Multicast Storm Control Bucket Capacity Configuration	174 - 182
L2 Multicast Storm Control Bucket Threshold Configuration	183 - 191
L2 Multicast Storm Control Current Size	192 - 200
L2 Multicast Storm Control Enable	201

Name	Address Range
L2 Multicast Storm Control Rate Configuration	165 - 173
L2 Multicast Table	31027 - 31090
L2 Reserved Multicast Address Action	32241 - 32496
L2 Reserved Multicast Address Base	32850
L2 Reserved Multicast Address Drop	1440
L2 Unknown Multicast Storm Control Bucket Capacity Configuration	248 - 256
L2 Unknown Multicast Storm Control Bucket Threshold Configuration	257 - 265
L2 Unknown Multicast Storm Control Current Size	266 - 274
L2 Unknown Multicast Storm Control Enable	275
L2 Unknown Multicast Storm Control Rate Configuration	239 - 247
L2 Unknown Unicast Storm Control Bucket Capacity Configuration	211 - 219
L2 Unknown Unicast Storm Control Bucket Threshold Configuration	220 - 228
L2 Unknown Unicast Storm Control Current Size	229 - 237
L2 Unknown Unicast Storm Control Enable	238
L2 Unknown Unicast Storm Control Rate Configuration	202 - 210
L4 IEEE 1588 Decoder Drop	1446
L4 Port Range to Queue Assignment	32980 - 33051
L4 Protocol to Queue Assignment	31679 - 31714
LACP Decoder Drop	1449
LACP Packet Decoder Options	32838
LLDP Configuration	32842
Latent Error Detection Configuration	35951 - 36078
Latent Error Detection Status	36655 - 36686
Latent Error Detection Tick	35758
Learning And Aging Enable	280
Learning Conflict	276
Learning Overflow	278
Link Aggregate Weight	30733 - 30988
Link Aggregation Ctrl	30543
Link Aggregation Membership	32497 - 32505
Link Aggregation To Physical Ports Members	30724 - 30732
MAC RX Broken Packets	57 - 65
MAC RX Long Packet Drop	75 - 83
MAC RX Maximum Packet Length	48 - 56
MAC RX Short Packet Drop	66 - 74
MBSC Drop	34875 - 34883
MPLS EXP Field To Egress Queue Mapping Table	31786 - 31793
MPLS EXP Field To Packet Color Mapping Table	31107 - 31114
Map Queue to Priority	36752 - 36760
Max SDU Filter	35180 - 35195
Max SDU Filter Blocking	35196 - 35211
Maximum Allowed VLAN Drop	1438
Minimum Allowed VLAN Drop	1437
Minimum Buffer Free	37384
Output Disable	37385 - 37393
Output Mirroring Table	38750 - 38758
PB Packet Head Counter	38352
PB Packet Tail Counter	38353
PFC Dec Counters for ingress ports 0 to 8	36915 - 36986

Name	Address Range
PFC Inc Counters for ingress ports 0 to 8	36843 - 36914
PS Error Counter	38898 - 38906
PS Packet Head Counter	38896
PS Packet Tail Counter	38897
PSFP Matching Frame Counter	35565 - 35580
PSFP Not Passing Frame Counter	35629 - 35644
PSFP Not Passing SDU Counter	35597 - 35612
PSFP Passing Frame Counter	35613 - 35628
PSFP Passing SDU Counter	35581 - 35596
PSFP Red Frames Counter	35645 - 35660
PTP Tick Configuration	4
PTP Tick Select	5
Packet Buffer Status	36749
Port FFA Used	36987 - 36995
Port Move Options	30556
Port Pause Settings	37015 - 37023
Port Reserved	36834 - 36842
Port Shaper Bucket Capacity Configuration	38280 - 38288
Port Shaper Bucket Threshold Configuration	38289 - 38297
Port Shaper Current Size	38298 - 38306
Port Shaper Enable	38307
Port Shaper Rate Configuration	38271 - 38279
Port Tail-Drop FFA Threshold	37078 - 37086
Port Tail-Drop Settings	37024 - 37032
Port Used	36996 - 37004
Port Xoff FFA Threshold	37069 - 37077
Port Xon FFA Threshold	37060 - 37068
Port/TC Reserved	36762 - 36833
Port/TC Tail-Drop Total Threshold	37231 - 37302
Port/TC Xoff Total Threshold	37159 - 37230
Port/TC Xon Total Threshold	37087 - 37158
Prio Shaper Bucket Capacity Configuration	38051 - 38122
Prio Shaper Bucket Threshold Configuration	38123 - 38194
Prio Shaper Current Size	38195 - 38266
Prio Shaper Enable	38267
Prio Shaper Rate Configuration	37979 - 38050
Queue Off Drop	34857 - 34865
Queue Shaper Bucket Capacity Configuration	37759 - 37830
Queue Shaper Bucket Threshold Configuration	37831 - 37902
Queue Shaper Current Size	37903 - 37974
Queue Shaper Enable	37975
Queue Shaper Rate Configuration	37687 - 37758
RARP Decoder Drop	1444
RARP Packet Decoder Options	32854
Re-queue Overflow Drop	36761
Recovery Tick	35757
Reserved Destination MAC Address Range	32818 - 32833
Reserved MAC DA Drop	1434
Reserved MAC SA Drop	1435
Reserved Source MAC Address Range	32802 - 32817
Resource Limiter Set	36722 - 36729
SCTP Decoder Drop	1448
SCTP Packet Decoder Options	30546

Name	Address Range
SMON Set 0 Byte Counter	34517 - 34524
SMON Set 0 Packet Counter	34501 - 34508
SMON Set 1 Byte Counter	34525 - 34532
SMON Set 1 Packet Counter	34509 - 34516
SMON Set Search	32111 - 32112
SP Overflow Drop	1376 - 1384
Scratch	7
Send to CPU	32834
Sequence Recovery Config	35887 - 35950
Sequence Recovery Discarded Counter	36495 - 36526
Sequence Recovery Lost Counter	36591 - 36622
Sequence Recovery Out Of Order Counter	36527 - 36558
Sequence Recovery Passed Counter	36463 - 36494
Sequence Recovery Reset	35725 - 35756
Sequence Recovery Rogue Counter	36559 - 36590
Sequence Recovery Tagless Counter	36623 - 36654
Source Port Default ACL Action	32702 - 32737
Source Port Default ACL Action Drop	1460
Source Port Table	32506 - 32541
Stream Filter Lookup Table	30690 - 30705
Stream Gate Blocking Enable	35212 - 35243
Stream Gate Invalid RX Blocking	35244 - 35275
Stream Gate Max MSDU Blocking	35276 - 35307
Stream Handle To FRER Mapping Table	30626 - 30689
TC FFA Used	37005 - 37012
TC Tail-Drop FFA Threshold	37052 - 37059
TC Xoff FFA Threshold	37044 - 37051
TC Xon FFA Threshold	37036 - 37043
TCP/UDP Flag Rules	31794 - 31797
Tail-Drop FFA Threshold	37035
Time to Age	290
Unknown Egress Drop	38364 - 38372
Unknown Ingress Drop	1426
VID to Queue Assignment	31643 - 31678
VLAN Member Drop	1436
VLAN PCP And DEI To Color Mapping Table	31627 - 31642
VLAN PCP To Queue Mapping Table	31778 - 31785
VLAN Table	9207 - 25590
Xoff FFA Threshold	37034
Xon FFA Threshold	37033

34.5 Active Queue Manager

34.5.1 ERM Red Configuration

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

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	36730

Field Description

Bits	Field Name	Description	Default Value
10:0	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.	0x99
21:11	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.	0x180
29:22	redMaxCells	Maximum allowed packet length in cells when the buffer memory congestion status is red.	0xe

34.5.2 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 :	36720

Field Description

Bits	Field Name	Description	Default Value
10:0	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.	0x218
21:11	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.	0x361
30:22	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.	0x1ff
41:31	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.	0x156

34.5.3 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 :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	36731 to 36739

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

34.5.4 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 :	36722 to 36729

Field Description

Bits	Field Name	Description	Default Value
10:0	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**.	0x39
21:11	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.	0x6e
32:22	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.	0x23

Bits	Field Name	Description	Default Value
40:33	maxCells	Maximum allowed packet length in cells for this limiter. Packet with cells more than this value will be dropped.	0xff

34.6 Core Information

34.6.1 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

34.7 Egress Packet Processing

34.7.1 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 :	9
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	38556 to 38573

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

34.7.2 Color Remap From Ingress Admission Control

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

Number of Entries :	32
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Meter Pointer
Address Space :	38574 to 38637

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

34.7.3 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 :	38761

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

34.7.4 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 :	38354

Field Description

Bits	Field Name	Description	Default Value
8:0	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

34.7.5 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 :	38759

Field Description

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

34.7.6 Egress Multiple Spanning Tree State

Table of egress Multiple Spanning Tree Protocol Instances. The field **msptPtr** in the **VLAN Table** is used to address the instance/entry in this table. Each entry contains the egress spanning tree states for all ports in this MSTI.

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	**msptPtr** from **VLAN Table**
Address Space :	31002 to 31017

Field Description

Bits	Field Name	Description	Default Value
17:0	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

34.7.7 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 :	9
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38538 to 38555

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

Bits	Field Name	Description	Default Value
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
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

Bits	Field Name	Description	Default Value
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
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	delSeqNum	If set, remove the R-tag on this port for sequence recovery.	0x0

34.7.8 Egress Port VID Operation

This search table checks the ingress VID and the number of VLANs before the egress port VLAN operation. If both ingress VID and number of VLANs are in the defined range then the VLAN operation in this table will override egress port VLAN operations. In case of multiple hit, VLAN operation from the first hit takes effect.

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 :	38762 to 38825

Field Description

Bits	Field Name	Description	Default Value
2:0	vlanSingleOpIf	If this entry is hit, then this VLAN operation will override egress port VLAN operation. 0 = No operation. 1 = Swap. 2 = Push. 3 = Pop. 4 = Penultimate pop(remove all VLAN headers).	0x0
4:3	typeSelIf	If this entry is hit, 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
6:5	vidSelIf	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 vidIf will be used. 0 = From outermost VLAN in the packet (if any). 1 = From this table entry's **vidIf**. 2 = From the Ingress VID as selected in the **Source Port Table**.	0x0
8:7	cfiDeiSelIf	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 **cfiDeiIf**. 2 = From **Egress Queue To PCP And CFI/DEI Mapping Table**.	0x0
10:9	pcpSelIf	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 **cfiDeiIf** 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
22:11	vidIf	VID used in VLAN push or swap operation if vidSelIf chooses VID from this table.	0x0
23	cfiDeiIf	CFI/DEI used in VLAN push or swap operation if cfiDeiSelIf chooses CFI/DEI from this table.	0x0
26:24	pcpIf	PCP used in VLAN push or swap operation if pcpSelIf chooses PCP from this table.	0x0
38:27	startVid	Start of ingress VID to hit.	0x0

Bits	Field Name	Description	Default Value
50:39	endVid	End of ingress VID to hit.	0x0
53:51	minNrVlans	Minimum number of VLANs to hit	0x0
56:54	maxNrVlans	Maximum number of VLANs to hit	0x0
65:57	validPorts	Determine the valid egress port list.	0x0

34.7.9 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 :	38638 to 38645

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

34.7.10 Egress RSPAN Configuration

Configuration for RSPAN tags on each egress port. When configured to push or pop a RSPAN tag then all packets will unconditionally be subject to this operation. When pushing an RSPAN tag the content of the tag is specified in this register.

Number of Entries :	9
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38828 to 38845

Field Description

Bits	Field Name	Description	Default Value
0	pushRspanTag	Push an RSPAN tag to all packets on this port.	0x0
1	popRspanTag	Pop an RSPAN tag from all packets on this port.	0x0
17:2	rspanTagEthType	The EtherType used when pushing an RSPAN tag.	0x0
29:18	rspanTagVid	The VID used when pushing an RSPAN tag.	0x0
30	rspanTagCfiDei	The DEI used when pushing an RSPAN tag.	0x0
33:31	rspanTagPcp	The PCP used when pushing an RSPAN tag.	0x0

34.7.11 Egress VLAN Translation Large Table

The outermost VID and VID Ethernet Type (Service tag or Customer tag types) of the outgoing packet is compared.. If multiple buckets match then the result from the highest entry is selected.

Number of Entries :	32
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	address[3:0] : hash of { dstPort outermostVid outermostVid-Type } address[4:4] : bucket number
Address Space :	38646 to 38709

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
4:1	dstPort	This is a field which is used as search data. The destination port which the packet is going out on	0x0
16:5	outermostVid	This is a field which is used as search data. The outermost VID of the modified packet.	0x0
17	outermostVidType	This is a field which is used as search data. The outermost VID is a S-tag or C-Tag. 0 = Customer tag 1 = Service tag	0x0
29:18	newVid	This is a result field used when this entry is hit. The new VID for the outgoing packet.	0x0
45:30	ethType	This is a result field used when this entry is hit. The new Ethernet Type for the outgoing packet	0x0

34.7.12 Egress VLAN Translation Search Mask

Before the hashing and searching is done in the **Egress VLAN Translation Large Table** and **Egress VLAN Translation Small Table** The search data is AND:ed with this mask. If a bit in the mask is set to zero then this bit in the lookup will be viewed as do not care. Seperate masks exists for both small and large tables.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	38826

Field Description

Bits	Field Name	Description	Default Value
3:0	dstPort_mask_small	Which bits to compare in the field dstPort in **Egress VLAN Translation Small Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	0xf

Bits	Field Name	Description	Default Value
7:4	dstPort_mask_large	Which bits to compare in the field dstPort **Egress VLAN Translation Large Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	0xf
19:8	outermostVid_mask_small	Which bits to compare in the field outermostVid in **Egress VLAN Translation Small Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	0xfff
31:20	outermostVid_mask_large	Which bits to compare in the field outermostVid **Egress VLAN Translation Large Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	0xfff
32	outermostVidType_mask_small	Which bits to compare in the field outermostVidType in **Egress VLAN Translation Small Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	0x1
33	outermostVidType_mask_large	Which bits to compare in the field outermostVidType **Egress VLAN Translation Large Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	0x1

34.7.13 Egress VLAN Translation Selection

This register selects which result to use when there are multiple hits.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38760

Field Description

Bits	Field Name	Description	Default Value
0	selectTcamOrTable	If set to zero then TCAM answer is selected. If set to one then hash table answer is selected.	0x0
1	selectSmallOrLarge	If set to zero then small hash table is selected. If set to one then large hash table is selected.	0x0

34.7.14 Egress VLAN Translation Small Table

The outermost VID and VID Ethernet Type (Service tag or Customer tag types) of the outgoing packet is compared.. If multiple buckets match then the result from the highest entry is selected.

Number of Entries :	16	
Number of Addresses per Entry :	2	
Type of Operation :	Read/Write	
Addressing :	address[2:0] :	hash of { dstPort outermostVid outermostVid-Type }
	address[3:3] :	bucket number
Address Space :	38710 to 38741	

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
4:1	dstPort	This is a field which is used as search data. The destination port which the packet is going out on	0x0
16:5	outermostVid	This is a field which is used as search data. The outermost VID of the modified packet.	0x0
17	outermostVidType	This is a field which is used as search data. The outermost VID is a S-tag or C-Tag. 0 = Customer tag 1 = Service tag	0x0
29:18	newVid	This is a result field used when this entry is hit. The new VID for the outgoing packet.	0x0
45:30	ethType	This is a result field used when this entry is hit. The new Ethernet Type for the outgoing packet	0x0

34.7.15 Egress VLAN Translation TCAM

The outermost VID and VID Ethernet Type (Service tag or Customer tag types) of the outgoing packet is compared.

Number of Entries :	8
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	38846 to 38861

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
4:1	dstPort_mask	Mask for dstPort.	0xf
8:5	dstPort	The destination port which the packet is going out on	0x0
20:9	outermostVid_mask	Mask for outermostVid.	0xfff
32:21	outermostVid	The outermost VID of the modified packet.	0x0
33	outermostVidType_mask	Mask for outermostVidType.	0x1

Bits	Field Name	Description	Default Value
34	outermostVidType	The outermost VID is a S-tag or C-Tag. 0 = Customer tag 1 = Service tag	0x0

34.7.16 Egress VLAN Translation TCAM Answer

This is the table holding the answer for the **Egress VLAN Translation TCAM**.

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	**Egress VLAN Translation TCAM** hit index
Address Space :	38742 to 38749

Field Description

Bits	Field Name	Description	Default Value
11:0	newVid	The new VID for the outgoing packet.	0x0
27:12	ethType	The new Ethernet Type for the outgoing packet	0x0

34.7.17 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 :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38750 to 38758

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
5	omUnderVlanMembership	If set, output mirroring to a destination that not a member of the VLAN will be ignored.	0x0

34.8 Flow Control

34.8.1 FFA Used PFC

Total number of cells from the common pool used by ports in PFC-mode.

Number of Entries :	1
Type of Operation :	Read Only
Address Space :	37013

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.2 FFA Used non-PFC

Total number of cells used from the common pool by ports in non-PFC mode.

Number of Entries :	1
Type of Operation :	Read Only
Address Space :	37014

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.3 PFC Dec Counters for ingress ports 0 to 8

Wrapping counters of deallocated cells. The number of currently used cells is the allocated minus the deallocated modulo the counter size.

Number of Entries :	72
Type of Operation :	Read Only
Addressing :	8*(Source port) + Traffic class
Address Space :	36915 to 36986

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.4 PFC Inc Counters for ingress ports 0 to 8

Wrapping counters of allocated cells. The number of currently used cells is the allocated minus the deallocated modulo the counter size.

Number of Entries :	72
Type of Operation :	Read Only
Addressing :	8*(Source port) + Traffic class
Address Space :	36843 to 36914

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.5 Port FFA Used

Number of cells used from the common pool for this source port

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Source port
Address Space :	36987 to 36995

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.6 Port Pause Settings

Pause settings per source port.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Source port
Address Space :	37015 to 37023

Field Description

Bits	Field Name	Description	Default Value
0	enable	0 = Pausing disabled 1 = Pausing enabled	0x0
1	mode	On a port where both pausing and tail-drop is enabled the modes must match for the calculation of used FFA to be correct. 0 = Priority mode 1 = Port mode	0x0
3:2	reserved	Reserved.	0x0
11:4	force	Each bit refers to one traffic class (bit 0 = TC 0) 0 = No force 1 = Force the pause state to that set in the pattern field Only valid if pausing is enabled.	0x0
19:12	pattern	Each bit refers to one traffic class (bit 0 = TC 0) 0 = Not paused 1 = Paused	0x0

34.8.7 Port Reserved

Number of cells reserved in the buffer memory for this source port. Shall be set to zero for prio-mode ports Note that this setting can only be changed for an empty port.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Source port
Address Space :	36834 to 36842

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0xb

34.8.8 Port Tail-Drop FFA Threshold

Settings for the Port Tail-Drop FFA Threshold

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Source port
Address Space :	37078 to 37086

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Tail-drop threshold in number of cells. When the FFA cells used by the source port reaches this threshold no further packets will be accepted for this source port	0x600
11	enable	0 = This tail-drop threshold is disabled 1 = This tail-drop threshold is enabled	0x0
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this tail-drop threshold is enabled.	0x0

34.8.9 Port Tail-Drop Settings

Tail-drop settings per source port.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Source port
Address Space :	37024 to 37032

Field Description

Bits	Field Name	Description	Default Value
0	enable	0 = Tail-drop is disabled for this source port 1 = Tail-drop is enabled for this source port	0x0
1	mode	On a port where both pausing and tail-drop is enabled the modes must match for the calculation of used FFA to be correct. 0 = Priority mode 1 = Port mode	0x0

34.8.10 Port Used

Total number of cells used for this source port

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Source port
Address Space :	36996 to 37004

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.11 Port Xoff FFA Threshold

Settings for Port Xoff FFA Threshold

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Source port
Address Space :	37069 to 37077

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xoff threshold for the number of used FFA cells for this source port	0x0
11	enable	0 = This Xoff threshold is disabled 1 = This Xoff threshold is enabled	0x0
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this Xoff threshold is enabled.	0x0

34.8.12 Port Xon FFA Threshold

Settings for Port Xon FFA Threshold

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Source port
Address Space :	37060 to 37068

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xon threshold for the number of used FFA cells for this source port	0x0

34.8.13 Port/TC Reserved

Number of cells reserved in the buffer memory for this source port and traffic class. For ports set to port-mode this should be 0 for all queues. Note that this setting can only be changed for an empty port.

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	8 * Source port + Traffic class
Address Space :	36762 to 36833

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.14 Port/TC Tail-Drop Total Threshold

Settings for Port/TC Tail-Drop Total Threshold

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	8 * Source port + Traffic class
Address Space :	37231 to 37302

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Tail-drop threshold in number of cells. When the sum of reserved and FFA cells used by this specific source port and traffic class combination reaches this threshold no further packets will be accepted for this source port and traffic class	0x600
11	enable	0 = This tail-drop threshold is disabled 1 = This tail-drop threshold is enabled	0x0

Bits	Field Name	Description	Default Value
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this tail-drop threshold is enabled.	0x0

34.8.15 Port/TC Xoff Total Threshold

Settings for Port/TC Xoff Total Threshold

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	8 * Source port + Traffic class
Address Space :	37159 to 37230

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xoff threshold for the sum of reserved and FFA cells used for this source port and traffic class combination	0x0
11	enable	0 = This Xoff threshold is disabled 1 = This Xoff threshold is enabled	0x0
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this Xoff threshold is enabled.	0x0

34.8.16 Port/TC Xon Total Threshold

Settings for Port/TC Xon Total Threshold

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	8 * Source port + Traffic class
Address Space :	37087 to 37158

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xon threshold for the sum of reserved and FFA cells used for this source port and traffic class combination	0x0

34.8.17 TC FFA Used

Number of cells used from the common pool for this traffic class.

Number of Entries :	8
Type of Operation :	Read Only
Addressing :	Traffic class
Address Space :	37005 to 37012

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells	0x0

34.8.18 TC Tail-Drop FFA Threshold

Settings for TC Tail-Drop FFA Threshold

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	Traffic class
Address Space :	37052 to 37059

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Tail-drop threshold in number of cells. When the FFA cells used by the traffic class reaches this threshold no further packets will be accepted for this traffic class	0x600
11	enable	0 = This tail-drop threshold is disabled 1 = This tail-drop threshold is enabled	0x0
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this tail-drop threshold is enabled.	0x0

34.8.19 TC Xoff FFA Threshold

Settings for TC Xoff FFA Threshold

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	Traffic class
Address Space :	37044 to 37051

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xoff threshold for the number of used FFA cells for this traffic class	0x0
11	enable	0 = This Xoff threshold is disabled 1 = This Xoff threshold is enabled	0x0
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this Xoff threshold is enabled.	0x0

34.8.20 TC Xon FFA Threshold

Settings for TC Xon FFA Threshold

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	Traffic class
Address Space :	37036 to 37043

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xon threshold for the number of used FFA cells for this traffic class	0x0

34.8.21 Tail-Drop FFA Threshold

Settings for Tail-Drop FFA Threshold

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	37035

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Tail-drop threshold in number of cells. When the total number of FFA cells used reaches this threshold no further packets will be accepted.	0x592
11	enable	0 = This tail-drop threshold is disabled 1 = This tail-drop threshold is enabled	0x0
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this tail-drop threshold is enabled.	0x0

34.8.22 Xoff FFA Threshold

Settings for Xoff FFA Threshold

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	37034

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xoff threshold for the total number of used FFA cells	0x0

Bits	Field Name	Description	Default Value
11	enable	0 = This Xoff threshold is disabled 1 = This Xoff threshold is enabled	0x0
12	trip	0 = Normal operation 1 = Force this threshold to be counted as exceeded Only valid if this Xoff threshold is enabled.	0x0

34.8.23 Xon FFA Threshold

Settings for Xon FFA Threshold

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	37033

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Xon threshold for the total number of used FFA cells	0x0

34.9 Global Configuration

34.9.1 CPU Port

Select which port is the CPU port.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	6

Field Description

Bits	Field Name	Description	Default Value
3:0	port	Port number	0x8

34.9.2 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
22:0	clkDivider	The master Core Tick will be issued once every $rg\_tick\_div.clkDivider/256$ core clock cycles. If set to zero, there will be no tick.	0x1900
26:23	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

34.9.3 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

34.9.4 MAC RX Maximum Packet Length

Packets with length above this value will be dropped.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Ingress Port
Address Space :	48 to 56

Field Description

Bits	Field Name	Description	Default Value
23:0	bytes	Number of bytes.	0x3fff

34.9.5 PTP Tick Configuration

Global register for setting the frequency of the ptp tick

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	4

Field Description

Bits	Field Name	Description	Default Value
22:0	clkDivider	The master PTP Tick will be issued once every $rg\_tick\_div.clkDivider/256$ core clock cycles. If set to zero, there will be no tick.	0x186a
26:23	stepDivider	The four ticks derived from the master ptp 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

34.9.6 PTP Tick Select

Global register for setting clock input to the ptp tick divider

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	5

Field Description

Bits	Field Name	Description	Default Value
1:0	clkSelect	Select the source clock for the PTP Tick divider. 0: disabled, 1: core clock, 2: debug_write_data[1], 3: reserved	0x0

34.9.7 Scratch

Scratch Register

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	7

Field Description

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

34.10 Ingress Packet Processing

34.10.1 AH Header Packet Decoder Options

The L4 protocol number which is used to determine if the packet has a Authentical Header, the underlaying packet must be a IPv4 or IPv6 packet.. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30547

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
8:1	l4Proto	The value to be used to find this packet type.	0x33
17:9	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
26:18	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.2 ARP Packet Decoder Options

The Ethernet type used to determine if a packet is a ARP packet.. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32852

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
16:1	eth	The value to be used to find this packet type.	0x806

Bits	Field Name	Description	Default Value
25:17	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
34:26	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.3 Allow Special Frame Check For L2 Action Table

The result in **L2 Action Table** is a pointer field **allowPtr** which allows result from the L2 SA Action Table to setup rules of which types of packets/frames are allowed to be sent in on a port. If any of there is a match and packet is not allowed then all instances are dropped of this packet. The drop counter **L2 Action Table Special Packet Type Drop** is updated.

Number of Entries :	4
Type of Operation :	Read/Write
Addressing :	Result from **L2 Action Table**
Address Space :	30998 to 31001

Field Description

Bits	Field Name	Description	Default Value
0	dontAllowBPDU	Allow BPDU frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
1	dontAllow8021X_EAPOL	Allow 802.1X EAPOL frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
2	dontAllowCAPWAP	Allow CAPWAP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
3	dontAllowARP	Allow ARP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
4	dontAllowRARP	Allow RARP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
5	dontAllowDNS	Allow DNS frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
6	dontAllowBOOTP_DHCP	Allow BOOTP_DHCP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
7	dontAllowSCTP	Allow STCP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
8	dontAllowLLDP	Allow LLDP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0

Bits	Field Name	Description	Default Value
9	dontAllowGRE	Allow GRE frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
10	dontAllowESP	Allow ESP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
11	dontAllowAH	Allow AH frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
12	dontAllowL2_1588	Allow L2 1588 frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
13	dontAllowL4_1588	Allow L4 1588 frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
14	dontAllowICMP	Allow ICMP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
15	dontAllowIGMP	Allow IGMP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
16	dontAllowL2McReserved	Allow L2 Reserved Da frames, see register **L2 Reserved Multicast Address Base**. 0 = Allow frame. 1 = Do not allow frame.	0x0
17	dontAllowIPV4	Allow IPV4 frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
18	dontAllowIPV6	Allow IPV6 frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
19	dontAllowUDP	Allow UDP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
20	dontAllowTCP	Allow TCP frames. 0 = Allow frame. 1 = Do not allow frame.	0x0
21	dontAllowMPLS	Allow MPLS frames. 0 = Allow frame. 1 = Do not allow frame.	0x0

34.10.4 BOOTP and DHCP Packet Decoder Options

The UDP port 1 number used by the BOOTP protocol, the underlaying packet must be a IPv4 packet. If L4 Source Port is this value then L4 Destination Port must be egisterbootpUdpPort2 value and vice versa. . If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32864

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
16:1	udp1	The value to be used to find this packet type.	0x43
32:17	udp2	The value to be used to find this packet type.	0x44
41:33	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
50:42	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.5 CAPWAP Packet Decoder Options

The fields needs to determine if a packet is a CAPWAP packet the underlaying packet must be a IPv4 or IPv6 packet. . If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32866

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
16:1	udp1	The value to be used to find this packet type.	0x147e
32:17	udp2	The value to be used to find this packet type.	0x147f
41:33	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
50:42	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.6 CPU Reason Code Operation

When a packet raises a send to CPU action during the ingress packet process, follow-up operations can be performed based on the reason code. In this table 4 ranges are searched in order and the same action hit in the latter range overrides the previous hit.

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 :	32932 to 32939

Field Description

Bits	Field Name	Description	Default Value
0	mutableCpu	Force the packet to another port instead of the CPU port when the CPU reason code hit in the range.	0x0
4:1	port	The new destination to replace the CPU port.	0x0
5	forceQueue	Force the packet to the CPU port with a new egress queue when the CPU reason code hit in the range.	0x0
8:6	eQueue	Egress queue	0x0
24:9	start	Start of CPU reason code.	0x0
40:25	end	End of CPU reason code.	0x0

34.10.7 Check IPv4 Header Checksum

This register provides an option to drop the IPv4 packet if its header checksum field has an incorrect value.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30551

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

34.10.8 DA or SA MAC to Queue Assignment

This register allows each egress port to determine the queue based on a source or destination MAC address. If hit in multiple entries with queue assignment enabled, the value is assigned from the highest numbered hit entry.

Number of Entries :	36
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	egress port * 4 + entry number
Address Space :	32542 to 32685

Field Description

Bits	Field Name	Description	Default Value
47:0	mac	The MAC Address to match. Bits that are masked must be 0 in this field.	0x0
95:48	mask	The MAC Address Mask. Setting a bit to zero mean this bit will not be compared. The corresponding bit in the mac field must be 0.	$2^{48} - 1$
96	saOrDa	Select if this entry should be compared with Source or Destination MAC address. 0 = SA MAC 1 = DA MAC	0x0
97	force	Force the queue if the MAC address is a match.	0x0
100:98	queue	The queue to assign for this port	0x0

34.10.9 DNS Packet Decoder Options

The TCP/UDP destination port number used to determine if a packet is a DNS packet, the underlaying packet must be a IPv4 or IPv6 packet.. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32862

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
16:1	l4Port	The value to be used to find this packet type.	0x35
25:17	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
34:26	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.10 Debug dstPortmask

Packet processing pipeline status for dstPortmask.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30561

Field Description

Bits	Field Name	Description	Default Value
8:0	value	Status from last processed packet.	0x0

34.10.11 Debug srcPort

Packet processing pipeline status for srcPort.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30560

Field Description

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

34.10.12 ESP Header Packet Decoder Options

The L4 protocol number which is used to determine if the packet has a Authentical Header, the underlaying packet must be a IPv4 or IPv6 packet.. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30548

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
8:1	l4Proto	The value to be used to find this packet type.	0x32
17:9	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
26:18	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.13 Egress Queue Priority Selection

How to select the egress queue priority.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	31751 to 31759

Field Description

Bits	Field Name	Description	Default Value
0	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

34.10.14 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
Type of Operation :	Read/Write
Address Space :	30554

Field Description

Bits	Field Name	Description	Default Value
26:0	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

34.10.15 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 :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	30706 to 30714

Field Description

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

34.10.16 Ethernet Type to Queue Assignment

This register allows each egress port determine the queue based on a Ethernet type.

Number of Entries :	36
Type of Operation :	Read/Write
Addressing :	egress port * 4 + entry number
Address Space :	31715 to 31750

Field Description

Bits	Field Name	Description	Default Value
15:0	ethType	The Ethernet Type to match.	0x0
16	force	Force the queue if the ethernet type is a match.	0x0
19:17	queue	The queue to assign for this port	0x0

34.10.17 FRER Configuration

Determine the mode for each FRER ID. FRER IDs are mapped from stream handles. Packets with FRER ID in generation mode will be tagged with a redundancy tag when it is out from the core. Packets with FRER ID in recovery mode will be checked to remove the redundant copies.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	30594 to 30625

Field Description

Bits	Field Name	Description	Default Value
1:0	mode	0 = Disabled 1 = Generation 2 = Recovery	0x0
5:2	seqRecovPort	Sequence recovery port. When the FRER ID is in recovery mode, the sequence recovery will be applied to this egress port. Note that multicast packets have individual recovery on all destinations, but sequence recovery on a single port.	0x0

34.10.18 FRER Sequence Number

Show current sequence number.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	30562 to 30593

Field Description

Bits	Field Name	Description	Default Value
15:0	seq	Sequence number.	0x0

34.10.19 Flooding Action Send to Port

If a packet is flooded and this function is enabled on the source port then the packet is send to a single egress port instead of being flooded to all ports part of the packets VLAN membership.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Source Port
Address Space :	30715 to 30723

Field Description

Bits	Field Name	Description	Default Value
0	enable	Enable sent to port instead of flooding. 0 = Disable 1 = Enable	0x0
4:1	destPort	Once enabled this is the destination port to sent the packet to in case of flooding.	0x0

34.10.20 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 :	30552

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

34.10.21 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 :	9
Type of Operation :	Read/Write
Addressing :	Ingress/Egress Port
Address Space :	31769 to 31777

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 18.1	0x0
3:1	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0

34.10.22 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 :	30553

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

34.10.23 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 :	9
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	31760 to 31768

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 18.1	0x0
3:1	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0

34.10.24 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 :	30555

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

34.10.25 GRE Packet Decoder Options

The L4 protocol number which is used to detemine if the packet has a GRE header. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32860

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
8:1	l4Proto	The value to be used to find this packet type.	0x2f
24:9	udp1	The value to be used to find this packet type.	0x1292
40:25	udp2	The value to be used to find this packet type.	0x1293
49:41	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
58:50	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.26 Hairpin Enable

Decide if the L2 switching allows a packet to be switched back on the same port it entered the switch. There are separate controls for flooding due to unknown MAC DA, multicast and unicast.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	31018 to 31026

Field Description

Bits	Field Name	Description	Default Value
0	allowFlood	Allow flooding to source port.	0x0
1	allowMc	Allow multicast to source port.	0x0
2	allowUc	Allow unicast to source port.	0x1

34.10.27 Hardware Learning Configuration

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

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Ingress Port
Address Space :	281 to 289

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
13:3	learnLimit	Maximum number of entries can be learned on this port. 0 means no limit.	0x0
14	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
15	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

34.10.28 Hardware Learning Counter

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

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Ingress Port
Address Space :	308 to 316

Field Description

Bits	Field Name	Description	Default Value
10:0	cnt	Number of learned L2 entries.	0x0

34.10.29 ICMP Length Check

Length check for IP packets carrying ICMP protocol data. IP payload length larger than the maximum size defined in this register can cause the packet get dropped.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30545

Field Description

Bits	Field Name	Description	Default Value
0	dropMaxICMPv4	If set, the IPv4 packet carrying ICMPv4 data size larger than the defined maximum length will be dropped	0x0
14:1	maxICMPv4Bytes	Maximum size of ICMPv4	0x200
15	dropMaxICMPv6	If set, the IPv6 packet carrying ICMPv6 data size larager than the defined maximum length will be dropped	0x0
29:16	maxICMPv6Bytes	Maximum size of ICMPv6	0x200

34.10.30 IEEE 1588 L2 Packet Decoder Options

The Ethernet type used to determine if a packet is a IEEE 1588 L2 Packet. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32856

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
16:1	eth	The value to be used to find this packet type.	0x88f7
25:17	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0

Bits	Field Name	Description	Default Value
34:26	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.31 IEEE 1588 L4 Packet Decoder Options

IEEE 1588 L4 packet is determined by this register. Fields from L2/L3/L4 are required for the comparison, including two optional DA MAC, five optional IPv4 DA, two optional IPv6 DA with the first one maskable, and two optional UDP destination ports. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	32
Type of Operation :	Read/Write
Address Space :	33076

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
48:1	da_mac1	DA MAC to match.	0x11b19000000
96:49	da_mac2	DA MAC to match.	0x180c200000e
128:97	da_ipv4_addr1	IPv4 DA to match.	0xe0000181
160:129	da_ipv4_addr2	IPv4 DA to match.	0xe0000182
192:161	da_ipv4_addr3	IPv4 DA to match.	0xe0000183
224:193	da_ipv4_addr4	IPv4 DA to match.	0xe0000184
256:225	da_ipv4_addr5	IPv4 DA to match.	0xe000016b
384:257	da_ipv6_addr1	IPv6 DA to match. This address is maskable.	0x18100000000000000000000000000ff0
512:385	da_ipv6_mask1	Bit mask for da_ipv6_addr1. For each bit of the mask, 1 means valid for comparison.	0xfff0ffffffffffffffffffffffffffff
640:513	da_ipv6_addr2	IPv6 DA to match.	0x6b0000000000000000000000000ff02
656:641	udp1	UDP destination to match.	0x13f
672:657	udp2	UDP destination to match.	0x140
681:673	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
690:682	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.32 IEEE 802.1X and EAPOL Packet Decoder Options

The Ethernet type used to determine if a packet is a 802.1X or EAPOL packet. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32858

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
16:1	eth	The value to be used to find this packet type.	0x888e
25:17	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
34:26	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.33 IP Address To Queue Assignment

This register allows each egress port determine the queue based on a IPv4 or IPv6 source or destination address.

Number of Entries :	36
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Addressing :	egress port * 4 + entry number
Address Space :	33668 to 34243

Field Description

Bits	Field Name	Description	Default Value
127:0	ip	The IP Address to match. IPv4 is located in bits [31:0]. Bits that are masked must be 0 in this field.	0x0
255:128	mask	The IP Address Mask. Setting a bit to zero mean this bit will not be compared. The corresponding bit in the ip field must be 0	$2^{128} - 1$
256	saOrDa	Select if this entry shall match a Source or Destination IP address. 0 = Source Address 1 = Destination Address	0x0
257	ipv4OrIpv6	Select if this entry shall match an IPv4 or IPv6 address. 0 = IPv4 1 = IPv6	0x0
258	force	Force the queue if the IP address matches.	0x0
261:259	queue	The queue to assign for this port	0x0

34.10.34 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 :	25591 to 25846

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueuePort_0	Egress queue for port 0.	0x1
5:3	pQueuePort_1	Egress queue for port 1.	0x1
8:6	pQueuePort_2	Egress queue for port 2.	0x1
11:9	pQueuePort_3	Egress queue for port 3.	0x1
14:12	pQueuePort_4	Egress queue for port 4.	0x1
17:15	pQueuePort_5	Egress queue for port 5.	0x1
20:18	pQueuePort_6	Egress queue for port 6.	0x1
23:21	pQueuePort_7	Egress queue for port 7.	0x1
26:24	pQueuePort_8	Egress queue for port 8.	0x1

34.10.35 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 :	31371 to 31626

Field Description

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

34.10.36 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 :	25847 to 26102

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueuePort_0	Egress queue for port 0.	0x1
5:3	pQueuePort_1	Egress queue for port 1.	0x1
8:6	pQueuePort_2	Egress queue for port 2.	0x1

Bits	Field Name	Description	Default Value
11:9	pQueuePort_3	Egress queue for port 3.	0x1
14:12	pQueuePort_4	Egress queue for port 4.	0x1
17:15	pQueuePort_5	Egress queue for port 5.	0x1
20:18	pQueuePort_6	Egress queue for port 6.	0x1
23:21	pQueuePort_7	Egress queue for port 7.	0x1
26:24	pQueuePort_8	Egress queue for port 8.	0x1

34.10.37 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 :	31115 to 31370

Field Description

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

34.10.38 Individual Recovery Config

Configurations for the individual recovery function.

Number of Entries :	64
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Stream handle
Address Space :	35759 to 35886

Field Description

Bits	Field Name	Description	Default Value
0	individualRecovery	Apply individual recovery	0x1
1	algo	Individual recovery algorithm. 0 = Vector 1 = Match	0x1
5:2	historyLen	Specify the valid number of bits in the sequence history. Only valid for the vector recovery algorighm and the minimum value is 2.	0x8
37:6	timeoutCnt	Number of ticks (see Chapter Tick) for the timeout period.	0x0
38	takeNoSequence	0 = Drop packets without sequence number. 1 = Accept packets without sequence number.	0x0

34.10.39 Individual Recovery Reset

Reset the sequence history of the individual recovery function and allow any sequence number for the next packet.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	Stream handle
Address Space :	35661 to 35724

Field Description

Bits	Field Name	Description	Default Value
0	reset	Set to one to reset the individual recovery function. Hardware clears the reset after one clock cycle.	0x0

34.10.40 Ingress Admission Control Current Status

Number of tokens currently in the token bucket.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Meter Pointer
Address Space :	35533 to 35564

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

34.10.41 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 :	32113 to 32240	

Field Description

Bits	Field Name	Description	Default Value
0	mmpValid	If set, this entry contains a valid MMP pointer	0x0

Bits	Field Name	Description	Default Value
5:1	mmpPtr	Initial pointer to the ingress MMP.	0x0
7:6	mmpOrder	Order of the initial ingress MMP pointer.	0x0

34.10.42 Ingress Admission Control Mark All Red

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

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Meter Pointer
Address Space :	35341 to 35372

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

34.10.43 Ingress Admission Control Mark All Red Enable

Option to block metering after MMP packet drops.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Meter Pointer
Address Space :	35309 to 35340

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

34.10.44 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 :	32
Type of Operation :	Read/Write
Addressing :	Meter Pointer
Address Space :	35501 to 35532

Field Description

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

34.10.45 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 :	32
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	Meter Pointer
Address Space :	35373 to 35500

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
81:67	maxLength	Maximum allowed packet length in bytes. Packets with bytes larger than this value will be dropped before metering.	0x7fff

Bits	Field Name	Description	Default Value
83:82	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
91:84	byteCorrection	Extra bytes per packet for IFG correction, only valid under byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

34.10.46 Ingress Configurable ACL 0 Large Table

This table is used for the configurable ACL lookup. A hash is calculated on the selected fields from the packet header. The hash is then used as index into this table.. If multiple buckets match then the result from the highest entry is selected.

Number of Entries :	128
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Addressing :	address[4:0] : hash of {compareData } address[6:5] : bucket number
Address Space :	1719 to 3766

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
208:1	compareData	The data which shall be compared in this entry.	0x0
209	sendToCpu	This is a result field used when this entry is hit. If set, the packet shall be sent to the CPU port.	0x0
210	dropEnable	This is a result field used when this entry is hit. If set, the packet shall be dropped and the **Ingress Configurable ACL Drop** counter is incremented.	0x0
211	sendToPort	This is a result field used when this entry is hit. Send the packet to a specific port. 0 = Disabled. 1 = Send to port configured in destPort.	0x0
215:212	destPort	This is a result field used when this entry is hit. The port which the packet shall be sent to.	0x0
216	inputMirror	This is a result field used when this entry is hit. If set, input mirroring is enabled for this rule. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the destination Input Mirror port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0
220:217	destInputMirror	This is a result field used when this entry is hit. Destination physical port for input mirroring.	0x0

Bits	Field Name	Description	Default Value
221	imPrio	This is a result field used when this entry is hit. If multiple input mirror are set and this prio bit is set then this input mirror will be selected.	0x0
222	noLearning	This is a result field used when this entry is hit. If set this packets MAC SA will not be learned.	0x0
223	updateCounter	This is a result field used when this entry is hit. When set the selected statistics counter will be updated.	0x0
228:224	counter	This is a result field used when this entry is hit. Which counter in **Ingress Configurable ACL Match Counter** to update.	0x0
229	updateCfiDei	This is a result field used when this entry is hit. The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
230	newCfiDeiValue	This is a result field used when this entry is hit. The value to update to.	0x0
231	updatePcp	This is a result field used when this entry is hit. The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
234:232	newPcpValue	This is a result field used when this entry is hit. The PCP value to update to.	0x0
235	updateVid	This is a result field used when this entry is hit. The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
247:236	newVidValue	This is a result field used when this entry is hit. The VID value to update to.	0x0
248	updateEType	This is a result field used when this entry is hit. The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
250:249	newEthType	This is a result field used when this entry is hit. Select 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
251	cfiDeiPrio	This is a result field used when this entry is hit. If multiple updateCfiDei are set and this prio bit is set then this updateCfiDei will be selected.	0x0
252	pcpPrio	This is a result field used when this entry is hit. If multiple updatePcp are set and this prio bit is set then this updatePcp will be selected.	0x0
253	vidPrio	This is a result field used when this entry is hit. If multiple updateVid are set and this prio bit is set then this updateVid will be selected.	0x0
254	ethPrio	This is a result field used when this entry is hit. If multiple updateEType are set and this prio bit is set then this updateEType will be selected.	0x0

Bits	Field Name	Description	Default Value
255	forceColor	This is a result field used when this entry is hit. If set, the packet shall have a forced color.	0x0
257:256	color	This is a result field used when this entry is hit. Initial color of the packet if the forceColor field is set.	0x0
258	forceColorPrio	This is a result field used when this entry is hit. If multiple forceColor are set and this prio bit is set then this forceVid value will be selected.	0x0
259	mmpValid	This is a result field used when this entry is hit. If set, this entry contains a valid MMP pointer	0x0
264:260	mmpPtr	This is a result field used when this entry is hit. Ingress MMP pointer.	0x0
266:265	mmpOrder	This is a result field used when this entry is hit. Ingress MMP pointer order.	0x0
267	forceQueue	This is a result field used when this entry is hit. If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 18.1	0x0
270:268	eQueue	This is a result field used when this entry is hit. The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
271	forceQueuePrio	This is a result field used when this entry is hit. If multiple forceQueue are set and this prio bit is set then this forceQueue value will be selected.	0x0
272	forceVidValid	This is a result field used when this entry is hit. Override the Ingress VID, see chapter VLAN Processing.	0x0
284:273	forceVid	This is a result field used when this entry is hit. The new Ingress VID.	0x0
285	forceVidPrio	This is a result field used when this entry is hit. If multiple forceVid are set and this prio bit is set then this forceVid value will be selected.	0x0

34.10.47 Ingress Configurable ACL 0 Pre Lookup

The pre ACL lookup allows the user to defined a specific rules for certain packet types in the ACL engine 0. Setting the valid bit and a new rule will override the default rule pointer from the source port table.

Number of Entries :	256
Type of Operation :	Read/Write
Addressing :	
Address Space :	1463 to 1718

Address bits	Meaning
Address bits [1:0]	Value from **preLookupAclBits**.
Address bits [2:2]	L2 Type Of Packet. 0 = Others - Not listed in this list. 1 = IEEE 1722/AVTP
Address bits [4:3]	L3 Type Of Packet. 0 = IPv4 1 = IPv6 2 = MPLS 3 = Not IPv4, IPv6 or MPLS
Address bits [7:5]	L4 Type Of Packet. 0 = Not known. 1 = Is IPv4 or IPv6 but type is not any L4 type in this list. 2 = UDP 3 = TCP 4 = IGMP 5 = ICMP 6 = ICMPv6 7 = MLD

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. If not then use default port rule.	0x0
3:1	rulePtr	If the valid is entry then this rule pointer will be used.	0x0

34.10.48 Ingress Configurable ACL 0 Rules Setup

The rules are setup by selecting which fields shall be used in the ACL search. Each rule has a fixed number of fields. The fieldSelectBitmask has one bit for each field. The first 6 fields (bits) which are set to one are selected. It is not allowed to set more than 6 bit in the bitmask. The fields are described in ACL Fields

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	ACL rule pointer
Address Space :	32094 to 32101

Field Description

Bits	Field Name	Description	Default Value
31:0	fieldSelectBitmask	Bitmask of which fields to select. Set a bit to one to select this specific field, set zero to not select field. At Maximum 6 bits should be set.	0x0

34.10.49 Ingress Configurable ACL 0 Search Mask

Before the hashing and searching is done in the **Ingress Configurable ACL 0 Large Table** and **Ingress Configurable ACL 0 Small Table**. The search data is AND:ed with this mask. If a bit in the mask is set to zero then this bit in the lookup will be viewed as do not care. Seperate masks exists for both small and large tables.

Number of Entries :	1
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Address Space :	33652

Field Description

Bits	Field Name	Description	Default Value
207:0	mask_small	Which bits to compare in the **Ingress Configurable ACL 0 Small Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	$2^{208} - 1$
415:208	mask_large	Which bits to compare in the **Ingress Configurable ACL 0 Large Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	$2^{208} - 1$

34.10.50 Ingress Configurable ACL 0 Selection

This register selects which result to use when there are multiple hits.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30549

Field Description

Bits	Field Name	Description	Default Value
0	selectTcamOrTable	If set to zero then TCAM answer is selected. If set to one then hash table answer is selected.	0x0
1	selectSmallOrLarge	If set to zero then small hash table is selected. If set to one then large hash table is selected.	0x0

34.10.51 Ingress Configurable ACL 0 Small Table

This table is used for the configurable ACL lookup. A hash is calculated on the selected fields from the packet header. The hash is then used as index into this table.. If multiple buckets match then the result from the highest entry is selected.

Number of Entries :	64
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Addressing :	address[3:0] : hash of {compareData } address[5:4] : bucket number
Address Space :	3767 to 4790

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
208:1	compareData	The data which shall be compared in this entry.	0x0
209	sendToCpu	This is a result field used when this entry is hit. If set, the packet shall be sent to the CPU port.	0x0
210	dropEnable	This is a result field used when this entry is hit. If set, the packet shall be dropped and the **Ingress Configurable ACL Drop** counter is incremented.	0x0
211	sendToPort	This is a result field used when this entry is hit. Send the packet to a specific port. 0 = Disabled. 1 = Send to port configured in destPort.	0x0
215:212	destPort	This is a result field used when this entry is hit. The port which the packet shall be sent to.	0x0
216	inputMirror	This is a result field used when this entry is hit. If set, input mirroring is enabled for this rule. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the destination Input Mirror port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0
220:217	destInputMirror	This is a result field used when this entry is hit. Destination physical port for input mirroring.	0x0
221	imPrio	This is a result field used when this entry is hit. If multiple input mirror are set and this prio bit is set then this input mirror will be selected.	0x0
222	noLearning	This is a result field used when this entry is hit. If set this packets MAC SA will not be learned.	0x0
223	updateCounter	This is a result field used when this entry is hit. When set the selected statistics counter will be updated.	0x0
228:224	counter	This is a result field used when this entry is hit. Which counter in **Ingress Configurable ACL Match Counter** to update.	0x0
229	updateCfiDei	This is a result field used when this entry is hit. The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
230	newCfiDeiValue	This is a result field used when this entry is hit. The value to update to.	0x0
231	updatePcp	This is a result field used when this entry is hit. The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
234:232	newPcpValue	This is a result field used when this entry is hit. The PCP value to update to.	0x0

Bits	Field Name	Description	Default Value
235	updateVid	This is a result field used when this entry is hit. The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
247:236	newVidValue	This is a result field used when this entry is hit. The VID value to update to.	0x0
248	updateEType	This is a result field used when this entry is hit. The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
250:249	newEthType	This is a result field used when this entry is hit. Select 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
251	cfiDeiPrio	This is a result field used when this entry is hit. If multiple updateCfiDei are set and this prio bit is set then this updateCfiDei will be selected.	0x0
252	pcpPrio	This is a result field used when this entry is hit. If multiple updatePcp are set and this prio bit is set then this updatePcp will be selected.	0x0
253	vidPrio	This is a result field used when this entry is hit. If multiple updateVid are set and this prio bit is set then this updateVid will be selected.	0x0
254	ethPrio	This is a result field used when this entry is hit. If multiple updateEType are set and this prio bit is set then this updateEType will be selected.	0x0
255	forceColor	This is a result field used when this entry is hit. If set, the packet shall have a forced color.	0x0
257:256	color	This is a result field used when this entry is hit. Initial color of the packet if the forceColor field is set.	0x0
258	forceColorPrio	This is a result field used when this entry is hit. If multiple forceColor are set and this prio bit is set then this forceVid value will be selected.	0x0
259	mmpValid	This is a result field used when this entry is hit. If set, this entry contains a valid MMP pointer	0x0
264:260	mmpPtr	This is a result field used when this entry is hit. Ingress MMP pointer.	0x0
266:265	mmpOrder	This is a result field used when this entry is hit. Ingress MMP pointer order.	0x0
267	forceQueue	This is a result field used when this entry is hit. If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 18.1	0x0
270:268	eQueue	This is a result field used when this entry is hit. The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
271	forceQueuePrio	This is a result field used when this entry is hit. If multiple forceQueue are set and this prio bit is set then this forceQueue value will be selected.	0x0
272	forceVidValid	This is a result field used when this entry is hit. Override the Ingress VID, see chapter VLAN Processing.	0x0

Bits	Field Name	Description	Default Value
284:273	forceVid	This is a result field used when this entry is hit. The new Ingress VID.	0x0
285	forceVidPrio	This is a result field used when this entry is hit. If multiple forceVid are set and this prio bit is set then this forceVid value will be selected.	0x0

34.10.52 Ingress Configurable ACL 0 TCAM

This table is used for the configurable ACL lookup. A hash is calculated on the selected fields from the packet header. The hash is then used as index into this table.

Number of Entries :	16
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	34244 to 34499

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
208:1	mask	Which bits to compare in this entry.	$2^{208} - 1$
416:209	compareData	The data which shall be compared in this entry. Observe that this compare data must be AND:ed by software before the entry is searched. The hardware does not do the AND between mask and compareData (In order to save area).	0x0

34.10.53 Ingress Configurable ACL 0 TCAM Answer

This is the table holding the answer for the **Ingress Configurable ACL 0 TCAM**.

Number of Entries :	16
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	**Ingress Configurable ACL 0 TCAM** hit index
Address Space :	4791 to 4854

Field Description

Bits	Field Name	Description	Default Value
0	sendToCpu	If set, the packet shall be sent to the CPU port.	0x0
1	dropEnable	If set, the packet shall be dropped and the **Ingress Configurable ACL Drop** counter is incremented.	0x0
2	sendToPort	Send the packet to a specific port. 0 = Disabled. 1 = Send to port configured in destPort.	0x0
6:3	destPort	The port which the packet shall be sent to.	0x0

Bits	Field Name	Description	Default Value
7	inputMirror	If set, input mirroring is enabled for this rule. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the destination Input Mirror port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0
11:8	destInputMirror	Destination physical port for input mirroring.	0x0
12	imPrio	If multiple input mirror are set and this prio bit is set then this input mirror will be selected.	0x0
13	noLearning	If set this packets MAC SA will not be learned.	0x0
14	updateCounter	When set the selected statistics counter will be updated.	0x0
19:15	counter	Which counter in **Ingress Configurable ACL Match Counter** to update.	0x0
20	updateCfiDei	The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
21	newCfiDeiValue	The value to update to.	0x0
22	updatePcp	The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
25:23	newPcpValue	The PCP value to update to.	0x0
26	updateVid	The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
38:27	newVidValue	The VID value to update to.	0x0
39	updateEType	The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
41:40	newEthType	Select 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
42	cfiDeiPrio	If multiple updateCfiDei are set and this prio bit is set then this updateCfiDei will be selected.	0x0
43	pcpPrio	If multiple updatePcp are set and this prio bit is set then this updatePcp will be selected.	0x0
44	vidPrio	If multiple updateVid are set and this prio bit is set then this updateVid will be selected.	0x0
45	ethPrio	If multiple updateEType are set and this prio bit is set then this updateEType will be selected.	0x0
46	forceColor	If set, the packet shall have a forced color.	0x0
48:47	color	Initial color of the packet if the forceColor field is set.	0x0
49	forceColorPrio	If multiple forceColor are set and this prio bit is set then this forceVid value will be selected.	0x0
50	mmpValid	If set, this entry contains a valid MMP pointer	0x0
55:51	mmpPtr	Ingress MMP pointer.	0x0
57:56	mmpOrder	Ingress MMP pointer order.	0x0

Bits	Field Name	Description	Default Value
58	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 18.1	0x0
61:59	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
62	forceQueuePrio	If multiple forceQueue are set and this prio bit is set then this forceQueue value will be selected.	0x0
63	forceVidValid	Override the Ingress VID, see chapter VLAN Processing.	0x0
75:64	forceVid	The new Ingress VID.	0x0
76	forceVidPrio	If multiple forceVid are set and this prio bit is set then this forceVid value will be selected.	0x0

34.10.54 Ingress Configurable ACL 1 Large Table

This table is used for the configurable ACL lookup. A hash is calculated on the selected fields from the packet header. The hash is then used as index into this table.. If multiple buckets match then the result from the highest entry is selected.

Number of Entries :	256
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Addressing :	address[6:0] : hash of {compareData } address[7:7] : bucket number
Address Space :	4855 to 8950

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
372:1	compareData	The data which shall be compared in this entry.	0x0
373	sendToCpu	This is a result field used when this entry is hit. If set, the packet shall be sent to the CPU port.	0x0
374	dropEnable	This is a result field used when this entry is hit. If set, the packet shall be dropped and the **Ingress Configurable ACL Drop** counter is incremented.	0x0
375	sendToPort	This is a result field used when this entry is hit. Send the packet to a specific port. 0 = Disabled. 1 = Send to port configured in destPort.	0x0
379:376	destPort	This is a result field used when this entry is hit. The port which the packet shall be sent to.	0x0
380	inputMirror	This is a result field used when this entry is hit. If set, input mirroring is enabled for this rule. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the destination Input Mirror port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0

Bits	Field Name	Description	Default Value
384:381	destInputMirror	This is a result field used when this entry is hit. Destination physical port for input mirroring.	0x0
385	imPrio	This is a result field used when this entry is hit. If multiple input mirror are set and this prio bit is set then this input mirror will be selected.	0x0
386	noLearning	This is a result field used when this entry is hit. If set this packets MAC SA will not be learned.	0x0
387	streamValid	This is a result field used when this entry is hit. If set, this entry contains a valid stream handle	0x0
393:388	streamHandle	This is a result field used when this entry is hit. Stream handle.	0x0
394	updateCounter	This is a result field used when this entry is hit. When set the selected statistics counter will be updated.	0x0
399:395	counter	This is a result field used when this entry is hit. Which counter in **Ingress Configurable ACL Match Counter** to update.	0x0
400	updateCfiDei	This is a result field used when this entry is hit. The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
401	newCfiDeiValue	This is a result field used when this entry is hit. The value to update to.	0x0
402	updatePcp	This is a result field used when this entry is hit. The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
405:403	newPcpValue	This is a result field used when this entry is hit. The PCP value to update to.	0x0
406	updateVid	This is a result field used when this entry is hit. The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
418:407	newVidValue	This is a result field used when this entry is hit. The VID value to update to.	0x0
419	updateEType	This is a result field used when this entry is hit. The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
421:420	newEthType	This is a result field used when this entry is hit. Select 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
422	cfiDeiPrio	This is a result field used when this entry is hit. If multiple updateCfiDei are set and this prio bit is set then this updateCfiDei will be selected.	0x0
423	pcpPrio	This is a result field used when this entry is hit. If multiple updatePcp are set and this prio bit is set then this updatePcp will be selected.	0x0

Bits	Field Name	Description	Default Value
424	vidPrio	This is a result field used when this entry is hit. If multiple updateVid are set and this prio bit is set then this updateVid will be selected.	0x0
425	ethPrio	This is a result field used when this entry is hit. If multiple updateEType are set and this prio bit is set then this updateEType will be selected.	0x0
426	forceColor	This is a result field used when this entry is hit. If set, the packet shall have a forced color.	0x0
428:427	color	This is a result field used when this entry is hit. Initial color of the packet if the forceColor field is set.	0x0
429	forceColorPrio	This is a result field used when this entry is hit. If multiple forceColor are set and this prio bit is set then this forceVid value will be selected.	0x0
430	mmpValid	This is a result field used when this entry is hit. If set, this entry contains a valid MMP pointer	0x0
435:431	mmpPtr	This is a result field used when this entry is hit. Ingress MMP pointer.	0x0
437:436	mmpOrder	This is a result field used when this entry is hit. Ingress MMP pointer order.	0x0
438	forceQueue	This is a result field used when this entry is hit. If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 18.1	0x0
441:439	eQueue	This is a result field used when this entry is hit. The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
442	forceQueuePrio	This is a result field used when this entry is hit. If multiple forceQueue are set and this prio bit is set then this forceQueue value will be selected.	0x0
443	forceVidValid	This is a result field used when this entry is hit. Override the Ingress VID, see chapter VLAN Processing.	0x0
455:444	forceVid	This is a result field used when this entry is hit. The new Ingress VID.	0x0
456	forceVidPrio	This is a result field used when this entry is hit. If multiple forceVid are set and this prio bit is set then this forceVid value will be selected.	0x0

34.10.55 Ingress Configurable ACL 1 Pre Lookup

The pre ACL lookup allows the user to defined a specific rules for certain packet types in the ACL engine 1. Setting the valid bit and a new rule will override the default rule pointer from the source port table.

Number of Entries :	256
Type of Operation :	Read/Write
Addressing :	
Address Space :	31838 to 32093

Address bits	Description
Address bits [1:0]	Value from **preLookupAclBits**.
Address bits [2:2]	L2 Type Of Packet. 0 = Others - Not listed in this list. 1 = IEEE 1722/AVTP
Address bits [4:3]	L3 Type Of Packet. 0 = IPv4 1 = IPv6 2 = MPLS 3 = Not IPv4, IPv6 or MPLS
Address bits [7:5]	L4 Type Of Packet. 0 = Not known. 1 = Is IPv4 or IPv6 but type is not any L4 type in this list. 2 = UDP 3 = TCP 4 = IGMP 5 = ICMP 6 = ICMPv6 7 = MLD

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. If not then use default port rule.	0x0
3:1	rulePtr	If the valid is entry then this rule pointer will be used.	0x0

34.10.56 Ingress Configurable ACL 1 Rules Setup

The rules are setup by selecting which fields shall be used in the ACL search. Each rule has a fixed number of fields. The fieldSelectBitmask has one bit for each field. The first 10 fields (bits) which are set to one are selected. It is not allowed to set more than 10 bit in the bitmask. The fields are described in ACL Fields

Number of Entries :	8
Type of Operation :	Read/Write
Addressing :	ACL rule pointer
Address Space :	31830 to 31837

Field Description

Bits	Field Name	Description	Default Value
31:0	fieldSelectBitmask	Bitmask of which fields to select. Set a bit to one to select this specific field, set zero to not select field. At Maximum 10 bits should be set.	0x0

34.10.57 Ingress Configurable ACL 1 Search Mask

Before the hashing and searching is done in the **Ingress Configurable ACL 1 Large Table** and **Ingress Configurable ACL 1 Small Table**. The search data is AND:ed with this mask. If a bit in the mask is set to zero then this bit in the lookup will be viewed as do not care. Seperate masks exists for both small and large tables.

Number of Entries :	1
Number of Addresses per Entry :	32
Type of Operation :	Read/Write
Address Space :	33108

Field Description

Bits	Field Name	Description	Default Value
371:0	mask_small	Which bits to compare in the **Ingress Configurable ACL 1 Small Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	$2^{372} - 1$
743:372	mask_large	Which bits to compare in the **Ingress Configurable ACL 1 Large Table** lookup. A bit set to 1 means the corresponding bit in the search data is compared and 0 means the bit is ignored.	$2^{372} - 1$

34.10.58 Ingress Configurable ACL 1 Selection

This register selects which result to use when there are multiple hits.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30550

Field Description

Bits	Field Name	Description	Default Value
0	selectTcamOrTable	If set to zero then TCAM answer is selected. If set to one then hash table answer is selected.	0x0
1	selectSmallOrLarge	If set to zero then small hash table is selected. If set to one then large hash table is selected.	0x0

34.10.59 Ingress Configurable ACL 1 Small Table

This table is used for the configurable ACL lookup. A hash is calculated on the selected fields from the packet header. The hash is then used as index into this table.. If multiple buckets match then the result from the highest entry is selected.

Number of Entries :	16
Number of Addresses per Entry :	16
Type of Operation :	Read/Write
Addressing :	address[2:0] : hash of {compareData }
	address[3:3] : bucket number
Address Space :	8951 to 9206

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
372:1	compareData	The data which shall be compared in this entry.	0x0
373	sendToCpu	This is a result field used when this entry is hit. If set, the packet shall be sent to the CPU port.	0x0
374	dropEnable	This is a result field used when this entry is hit. If set, the packet shall be dropped and the **Ingress Configurable ACL Drop** counter is incremented.	0x0
375	sendToPort	This is a result field used when this entry is hit. Send the packet to a specific port. 0 = Disabled. 1 = Send to port configured in destPort.	0x0
379:376	destPort	This is a result field used when this entry is hit. The port which the packet shall be sent to.	0x0
380	inputMirror	This is a result field used when this entry is hit. If set, input mirroring is enabled for this rule. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the destination Input Mirror port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0
384:381	destInputMirror	This is a result field used when this entry is hit. Destination physical port for input mirroring.	0x0
385	imPrio	This is a result field used when this entry is hit. If multiple input mirror are set and this prio bit is set then this input mirror will be selected.	0x0
386	noLearning	This is a result field used when this entry is hit. If set this packets MAC SA will not be learned.	0x0
387	streamValid	This is a result field used when this entry is hit. If set, this entry contains a valid stream handle	0x0
393:388	streamHandle	This is a result field used when this entry is hit. Stream handle.	0x0
394	updateCounter	This is a result field used when this entry is hit. When set the selected statistics counter will be updated.	0x0
399:395	counter	This is a result field used when this entry is hit. Which counter in **Ingress Configurable ACL Match Counter** to update.	0x0
400	updateCfiDei	This is a result field used when this entry is hit. The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
401	newCfiDeiValue	This is a result field used when this entry is hit. The value to update to.	0x0
402	updatePcp	This is a result field used when this entry is hit. The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0

Bits	Field Name	Description	Default Value
405:403	newPcpValue	This is a result field used when this entry is hit. The PCP value to update to.	0x0
406	updateVid	This is a result field used when this entry is hit. The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
418:407	newVidValue	This is a result field used when this entry is hit. The VID value to update to.	0x0
419	updateEType	This is a result field used when this entry is hit. The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
421:420	newEthType	This is a result field used when this entry is hit. Select 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
422	cfiDeiPrio	This is a result field used when this entry is hit. If multiple updateCfiDei are set and this prio bit is set then this updateCfiDei will be selected.	0x0
423	pcpPrio	This is a result field used when this entry is hit. If multiple updatePcp are set and this prio bit is set then this updatePcp will be selected.	0x0
424	vidPrio	This is a result field used when this entry is hit. If multiple updateVid are set and this prio bit is set then this updateVid will be selected.	0x0
425	ethPrio	This is a result field used when this entry is hit. If multiple updateEType are set and this prio bit is set then this updateEType will be selected.	0x0
426	forceColor	This is a result field used when this entry is hit. If set, the packet shall have a forced color.	0x0
428:427	color	This is a result field used when this entry is hit. Initial color of the packet if the forceColor field is set.	0x0
429	forceColorPrio	This is a result field used when this entry is hit. If multiple forceColor are set and this prio bit is set then this forceVid value will be selected.	0x0
430	mmpValid	This is a result field used when this entry is hit. If set, this entry contains a valid MMP pointer	0x0
435:431	mmpPtr	This is a result field used when this entry is hit. Ingress MMP pointer.	0x0
437:436	mmpOrder	This is a result field used when this entry is hit. Ingress MMP pointer order.	0x0
438	forceQueue	This is a result field used when this entry is hit. If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 18.1	0x0
441:439	eQueue	This is a result field used when this entry is hit. The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
442	forceQueuePrio	This is a result field used when this entry is hit. If multiple forceQueue are set and this prio bit is set then this forceQueue value will be selected.	0x0

Bits	Field Name	Description	Default Value
443	forceVidValid	This is a result field used when this entry is hit. Override the Ingress VID, see chapter VLAN Processing.	0x0
455:444	forceVid	This is a result field used when this entry is hit. The new Ingress VID.	0x0
456	forceVidPrio	This is a result field used when this entry is hit. If multiple forceVid are set and this prio bit is set then this forceVid value will be selected.	0x0

34.10.60 Ingress Configurable ACL 1 TCAM

This table is used for the configurable ACL lookup. A hash is calculated on the selected fields from the packet header. The hash is then used as index into this table.

Number of Entries :	16
Number of Addresses per Entry :	32
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	33140 to 33651

Field Description

Bits	Field Name	Description	Default Value
0	valid	Is this entry valid. 0 = No 1 = Yes	0x0
372:1	mask	Which bits to compare in this entry.	$2^{372}-1$
744:373	compareData	The data which shall be compared in this entry. Observe that this compare data must be AND:ed by software before the entry is searched. The hardware does not do the AND between mask and compareData (In order to save area).	0x0

34.10.61 Ingress Configurable ACL 1 TCAM Answer

This is the table holding the answer for the **Ingress Configurable ACL 1 TCAM**.

Number of Entries :	16
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	**Ingress Configurable ACL 1 TCAM** hit index
Address Space :	32738 to 32801

Field Description

Bits	Field Name	Description	Default Value
0	sendToCpu	If set, the packet shall be sent to the CPU port.	0x0
1	dropEnable	If set, the packet shall be dropped and the **Ingress Configurable ACL Drop** counter is incremented.	0x0
2	sendToPort	Send the packet to a specific port. 0 = Disabled. 1 = Send to port configured in destPort.	0x0

Bits	Field Name	Description	Default Value
6:3	destPort	The port which the packet shall be sent to.	0x0
7	inputMirror	If set, input mirroring is enabled for this rule. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the destination Input Mirror port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0
11:8	destInputMirror	Destination physical port for input mirroring.	0x0
12	imPrio	If multiple input mirror are set and this prio bit is set then this input mirror will be selected.	0x0
13	noLearning	If set this packets MAC SA will not be learned.	0x0
14	streamValid	If set, this entry contains a valid stream handle	0x0
20:15	streamHandle	Stream handle.	0x0
21	updateCounter	When set the selected statistics counter will be updated.	0x0
26:22	counter	Which counter in **Ingress Configurable ACL Match Counter** to update.	0x0
27	updateCfiDei	The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
28	newCfiDeiValue	The value to update to.	0x0
29	updatePcp	The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
32:30	newPcpValue	The PCP value to update to.	0x0
33	updateVid	The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
45:34	newVidValue	The VID value to update to.	0x0
46	updateEType	The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
48:47	newEthType	Select 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
49	cfiDeiPrio	If multiple updateCfiDei are set and this prio bit is set then this updateCfiDei will be selected.	0x0
50	pcpPrio	If multiple updatePcp are set and this prio bit is set then this updatePcp will be selected.	0x0
51	vidPrio	If multiple updateVid are set and this prio bit is set then this updateVid will be selected.	0x0
52	ethPrio	If multiple updateEType are set and this prio bit is set then this updateEType will be selected.	0x0
53	forceColor	If set, the packet shall have a forced color.	0x0
55:54	color	Initial color of the packet if the forceColor field is set.	0x0
56	forceColorPrio	If multiple forceColor are set and this prio bit is set then this forceVid value will be selected.	0x0
57	mmpValid	If set, this entry contains a valid MMP pointer	0x0

Bits	Field Name	Description	Default Value
62:58	mmpPtr	Ingress MMP pointer.	0x0
64:63	mmpOrder	Ingress MMP pointer order.	0x0
65	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 18.1	0x0
68:66	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0
69	forceQueuePrio	If multiple forceQueue are set and this prio bit is set then this forceQueue value will be selected.	0x0
70	forceVidValid	Override the Ingress VID, see chapter VLAN Processing.	0x0
82:71	forceVid	The new Ingress VID.	0x0
83	forceVidPrio	If multiple forceVid are set and this prio bit is set then this forceVid value will be selected.	0x0

34.10.62 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 :	34500

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
3	learnL2HairpinDrop	Allow learning when the packet is dropped due to hairpin configurations.	0x0

34.10.63 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 :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	30989 to 30997

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

Bits	Field Name	Description	Default Value
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
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

Bits	Field Name	Description	Default Value
26:18	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

34.10.64 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 :	30544

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
26:18	ignoreStag	If set, type value 0x88A8 is not parsed as Service VLAN type.	0x0

34.10.65 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 :	30559

Field Description

Bits	Field Name	Description	Default Value
8:0	dropMask	Each bit in this mask refers to if ingress MMP drop is allowed on the corresponding egress port.	0x1ff

34.10.66 Ingress Multiple Spanning Tree State

Table of ingress Multiple Spanning Tree Protocol Instances. The field **msptPtr** in the **VLAN Table** is used to address this 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 :	**mspt Ptr** from **VLAN Table**
Address Space :	31798 to 31813

Field Description

Bits	Field Name	Description	Default Value
17:0	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

34.10.67 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 :	9
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	32102 to 32110

Field Description

Bits	Field Name	Description	Default Value
0	dropMacDaLocal	If bit 47 in the DA MAC address is set to zero then packet will be dropped. This is sometimes referred to as the Local / Globally Administered bit.	0x0
1	dropMacDaGlobal	If bit 47 in the DA MAC is set to one then packet will be dropped. This is sometimes referred to as the Local / Globally Administered bit.	0x0
2	dropMacDaUnicast	If bit 48 in the DA MAC is set to zero then packet will be dropped. This is sometimes referred to as the Multicast/Unicast bit, 0 being a unicast DA Address.	0x0
3	dropMacSaLocal	If bit 47 in the SA MAC address is set to zero then packet will be dropped. This is sometimes referred to as the Local / Globally Administered bit.	0x0
4	dropMacSaGlobal	If bit 47 in the SA MAC is set to one then packet will be dropped. This is sometimes referred to as the Local / Globally Administered bit.	0x0
5	dropMacSaNotSourceRouted	If bit 48 in the SA MAC address is set to zero then packet will be dropped. This is sometimes referred to as the Routing Information Indicator bit.	0x0

Bits	Field Name	Description	Default Value
6	dropMacSaSourceRouted	If bit 48 in the SA MAC is set to one then packet will be dropped. This is sometimes referred to as the Routing Information Indicator bit.	0x0
7	dropDaMac0	Drop or allow DA MAC 00:00:00:00:00:00. 0 = Allow 1 = Drop	0x0
8	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
9	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
10	moreThanOneVlans	When filtering with dropCtaggedVlans or drop-StaggedVlans then this field must be set to 1.	0x0
11	dropUntaggedVlans	Drop or Allow packets that are VLAN untagged on this ingress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
12	dropSingleTaggedVlans	Drop or Allow packets that are VLAN untagged on this ingress port. 0 = Allow untagged packets. 1 = Drop untagged packets.	0x0
13	dropMacDaEqSa	Drop or allow MAC packets which has a DA==SA on this ingress port. 0 = Allow MAC DA == MAC SA packets. 1 = Drop MAC DA == MAC SA packets.	0x0
14	dropIPv4DaEqSa	Drop or allow IPv4 packets which has a DA IP==SA IP on this ingress port. 0 = Allow IPv4 DA == IPv4 SA packets. 1 = Drop IPv4 DA == IPv4 SA packets.	0x0
15	dropIPv6DaEqSa	Drop or allow IPv6 packets which has a DA IP==SA IP on this ingress port. 0 = Allow IPv6 DA == IPv6 SA packets. 1 = Drop IPv6 DA == IPv6 SA packets.	0x0
16	dropIPv4Packets	Drop or allow IPv4 packets on this ingress port. 0 = Allow IPv4 packets. 1 = Drop IPv4 packets.	0x0
17	dropIPv6Packets	Drop or allow IPv6 packets on this ingress port. 0 = Allow IPv6 packets. 1 = Drop IPv6 packets.	0x0
18	dropMPLSPackets	Drop or allow MPLS packets on this ingress port. 0 = Allow MPLS packets. 1 = Drop MPLS packets.	0x0

Bits	Field Name	Description	Default Value
19	dropIPv4MulticastPackets	Drop or allow IPv4 multicast packets on this ingress port. 0 = Allow IPv4 MC packets. 1 = Drop IPv4 MC packets.	0x0
20	dropIPv6MulticastPackets	Drop or allow IPv6 multicast packets on this ingress port. 0 = Allow IPv6 MC packets. 1 = Drop IPv6 MC packets.	0x0
21	dropL2BroadcastFrames	Drop or allow L2 broadcast packets on this ingress port. 0 = Drop L2 broadcast packets. 1 = Allow L2 broadcast packets.	0x0
22	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
23	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
24	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
25	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
26	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
27	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

34.10.68 Ingress Rate Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for Ingress Rate Control

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Physcial Ingress Ports + Ingress Queue
Address Space :	34637 to 34708

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	Index Value 0-15 0x15ea 16-71 0x744

34.10.69 Ingress Rate Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for Ingress Rate Control

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Physcial Ingress Ports + Ingress Queue
Address Space :	34709 to 34780

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.	Index Value 0-15 0x74e 16-71 0x26c

34.10.70 Ingress Rate Control Current Size

Number of tokens currently in the token bucket.

Number of Entries :	72
Type of Operation :	Read Only
Addressing :	Physcial Ingress Ports + Ingress Queue
Address Space :	34781 to 34852

Field Description

Bits	Field Name	Description	Default Value
15:0	currentVal	Number of tokens currently in the token bucket for this Physcial Ingress Ports + Ingress Queue	Index Value 0-15 0x74e 16-71 0x26c

34.10.71 Ingress Rate Control Enable

Bitmask to turn Ingress Rate Control ON/OFF (1/0) for Physcial Ingress Ports + Ingress Queue

Number of Entries :	1
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Address Space :	34853

Field Description

Bits	Field Name	Description	Default Value
71:0	enable	Bitmask where the index is the Physcial Ingress Ports + Ingress Queue	0x0

34.10.72 Ingress Rate Control Rate Configuration

Token Bucket rate Configuration for Ingress Rate Control

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Physcial Ingress Ports + Ingress Queue
Address Space :	34565 to 34636

Field Description

Bits	Field Name	Description	Default Value
0	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x0
12:1	tokens	The number of tokens added each tick	Index / Value: 0-15 0xbb; 16-71 0x3e
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	0x0
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

34.10.73 Ingress Rate Control Type

For each queue of the ingress port, define the packet type that shall be metered.

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	address[2:0] : Ingress queue; address[6:3] : Physical ingress port
Address Space :	30471 to 30542

Field Description

Bits	Field Name	Description	Default Value
0	ucHit	Packet with DA=unicast MAC address, and hit a unicast entry in the forwarding process.	0x0
1	ucMiss	Packet with DA=unicast MAC address, and miss a unicast entry in the forwarding process.	0x0
2	mcHit	Packet with DA=multicast MAC address, and hit a multicast entry in the forwarding process.	0x0
3	mcMiss	Packet with DA=multicast MAC address, and miss a multicast entry in the forwarding process.	0x0
4	bc	Packet with DA=FF:FF:FF:FF:FF:FF.	0x0
5	reservedDa	Packet with DA hit reserved MAC addresses. Reserved MAC addresses are configured in: **LLDP Configuration** **Reserved Destination MAC Address Range**	0x0

34.10.74 Ingress Transmission Gate Base Tick

Select one of the 5 available PTP ticks. The tick frequencies are configured globaly in the **PTP Tick Configuration** register. The selected tick is used for counting the current time and the gate cycle time. The frequency shall not be changed when the transmission gate is enabled.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	34920

Field Description

Bits	Field Name	Description	Default Value
2:0	baseTick	PTP tick number. The master tick is number 0.	0x0

34.10.75 Ingress Transmission Gate Configuration

Setup configurations for ingress transmission gates. Hardware is not aware of the configuration updates unless an update request is triggered by writing 1 to **Ingress Transmission Gate Update**. The transmission gate execution will start using the new configuration when the current time meets the **adminBaseTime**.

Number of Entries :	32
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	Gate ID
Address Space :	34956 to 35083

Field Description

Bits	Field Name	Description	Default Value
56:0	adminBaseTime	Determine the start time of the updated gate cycle. The value needs to be larger than **Ingress Transmission Gate Current Time** when an update request is issued.	0x0
59:57	adminTick	Select one of the 5 available PTP ticks. The tick frequencies are configured globaly in the **PTP Tick Configuration** register. The selected tick is used for counting time intervals between gate list entries.	0x0
86:60	adminCycleTime	Time for one gate cycle based on the **Ingress Transmission Gate Base Tick**. Once a gate list starts executing, it will be restarted from the start address again when the elapsed time equals this field.	0x0
113:87	adminCycleTimeExtension	Extra time for retaining the current gate status when the time for a pending update in the future to occur is less than this field. The time extension is based on the **Ingress Transmission Gate Base Tick**.	0x0
118:114	adminControlListLength	Number of execution entries in the **Ingress Transmission Gate List** for one gate cycle. If the gate cycle time is not enough to execute all the entries, the execution will abort the remaining entries.	0x0
123:119	adminStartAddr	Point to the first entry to execute in one gate cycle.	0x0

34.10.76 Ingress Transmission Gate Current Status

Show current gate status.

Number of Entries :	32
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Stream Gate ID
Address Space :	32868 to 32931

Field Description

Bits	Field Name	Description	Default Value
0	gateClosed	0 = Allow the packet to pass. 1 = Drop the packet.	0x0
1	updateQueue	If set, eQueue field in this entry will replace the current egress queue.	0x0
4:2	eQueue	Egress queue	0x0
36:5	maxMSDU	Maximum number of MSDU (L2 payload) octets that are allowed to pass when the current entry is open and activated. 0 means no limit.	0x0

34.10.77 Ingress Transmission Gate Current Time

Counting the current time based on the **Ingress Transmission Gate Base Tick**.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read Only
Address Space :	34954

Field Description

Bits	Field Name	Description	Default Value
56:0	currentTime	Number of counted base ticks since the reset.	0x0

34.10.78 Ingress Transmission Gate Enabled

All Ingress Transmission Gate operations require this register set to 1.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Gate ID
Address Space :	34922 to 34953

Field Description

Bits	Field Name	Description	Default Value
0	enabled	If set, ingress transmission gate is enabled.	0x0

34.10.79 Ingress Transmission Gate List

Gate control list. Each entry gives gate status for the current time window, as well as a time interval for counting the execution time before jumping to the next entry. When the total number of executed entries reaches the configured list length but a new gate cycle is not started, the gate status for the last entry will be kept till the restart.

Number of Entries :	32
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Ingress Transmission Gate Address
Address Space :	35084 to 35147

Field Description

Bits	Field Name	Description	Default Value
0	blocked	0 = Allow the packet to pass. 1 = Drop the packet.	0x0
1	updateQueue	If set, eQueue field in this entry will replace the current egress queue.	0x0
4:2	eQueue	Egress Queue.	0x0
36:5	maxMSDU	Maximum number of MSDU (L2 payload) octets that are allowed to pass when the current entry is open and activated.	0x0
63:37	timeInterval	Number of ticks before jumping to the next entry in the gate control list. The tick frequency is based on the loaded **adminTick**.	0x0

34.10.80 Ingress Transmission Gate Update

When set to one a configuration update request is issued to the hardware. The set bit is always cleared after one cycle and **Ingress Transmission Gate Update Status** will be pulled high till the update process is done.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Gate ID
Address Space :	35148 to 35179

Field Description

Bits	Field Name	Description	Default Value
0	start	Issue an update request to load **Ingress Transmission Gate Configuration** to the hardware.	0x0

34.10.81 Ingress Transmission Gate Update Status

For each gate id, showing if a new configuration is pending to be updated. **Ingress Transmission Gate Configuration** shall not be modified while an update process is pending.

Number of Entries :	1
Type of Operation :	Read Only
Address Space :	34921

Field Description

Bits	Field Name	Description	Default Value
31:0	pending	1 means **Ingress Transmission Gate Update** has been issued and 0 means the update operation is done.	0x0

34.10.82 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 :	31814 to 31817

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

34.10.83 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 :	33052 to 33059

Field Description

Bits	Field Name	Description	Default Value
8:0	ports	Ports that this range search is activated on.	0x0
24:9	start	Start of Ethernet type range.	0x0
40:25	end	End of Ethernet type range.	0x0

34.10.84 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 :	31818 to 31821

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

34.10.85 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 :	33060 to 33067

Field Description

Bits	Field Name	Description	Default Value
8:0	ports	Ports that this range search is activated on.	0x0
9	vtype	Shall this entry match S-Type or C-Type VLAN. 0 = C-Type 1 = S-Type	0x0
21:10	start	Start of VID range.	0x0
33:22	end	End of VID range.	0x0

34.10.86 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 :	31826 to 31829

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

34.10.87 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 :	32686 to 32701

Field Description

Bits	Field Name	Description	Default Value
8:0	ports	Ports that this range search is activated on.	0x0
9	saOrDa	Is this rule for source or destination MAC address. 0 = Source MAC 1 = Destination MAC	0x0
57:10	start	Start of MAC address range.	0x0
105:58	end	End of MAC address range.	0x0

34.10.88 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 :	31822 to 31825

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

34.10.89 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 :	33068 to 33075

Field Description

Bits	Field Name	Description	Default Value
8:0	ports	Ports that this range search is activated on.	0x0
9	vtype	Shall this entry match S-Type or C-Type VLAN. 0 = C-Type 1 = S-Type	0x0
21:10	start	Start of VID range.	0x0
33:22	end	End of VID range.	0x0

34.10.90 L2 Action Table

The L2 action table can be used to limit what type of traffic shall be able to enter a port depending on which port its coming from and going to. There are three table results which can be taken into consideration, the l2 destination MAC lookup, the l2 source MAC lookup and finally the ingress ACL lookup. The **L2 Action Table Egress Port State** defines the highest bit in the address. This table is looked up for each of the destiantion ports which the packet is going to. If a packet is dropped then it is recorded in the drop counter **L2 Action Table Drop**.

Number of Entries :	128
Type of Operation :	Read/Write
Addressing :	Address Bit 0: Source Port State Bit from **Source Port Table** field **l2ActionTablePortState**. Address Bit 1: L2 SA Table was a hit. 0 = Miss. 1 = Hit. Address Bit 2: L2 SA Table - L2 Action Table Status bit. If this table was a miss then this bit will be zero. Address Bit 3: L2 DA Table - L2 Action Table Status bit. If this table was a miss then this bit will be zero. Address Bit [5:4]: L2 Packet Type. 0 = L2 Dest Table was a Unicast. 1 = L2 Dest Table was Multicast. 2 = L2 DA table was a miss and packet is being flooded. 3 = Packet was a Broadcast packet and L2 Dest Table did not hit. If both flooded and L2 Broadcast packet then this option will be selected. Address Bit 6: Destiantion Port State Bit comes from the **L2 Action Table Egress Port State**.
Address Space :	30215 to 30342

Address Bit	Description
Address Bit 0:	Source Port State Bit from **Source Port Table** field **l2ActionTablePortState**.
Address Bit 1:	L2 SA Table was a hit. 0 = Miss. 1 = Hit.
Address Bit 2:	L2 SA Table - L2 Action Table Status bit. If this table was a miss then this bit will be zero.
Address Bit 3:	L2 DA Table - L2 Action Table Status bit. If this table was a miss then this bit will be zero.
Address Bit [5:4]:	L2 Packet Type. 0 = L2 Dest Table was a Unicast. 1 = L2 Dest Table was Multicast. 2 = L2 DA table was a miss and packet is being flooded. 3 = Packet was a Broadcast packet and L2 Dest Table did not hit. If both flooded and L2 Broadcast packet then this option will be selected.
Address Bit 6:	Destiantion Port State Bit comes from the **L2 Action Table Egress Port State**.

Field Description

Bits	Field Name	Description	Default Value
0	noLearningUc	The packet shall not be learned. This is applied to L2 DA MAC unicast packets.	0x0
1	noLearningMc	If the packet is a L2 Multicast then the packet shall not be learned. If a packet is a L2 Multicast depends on if the SA MAC MC bit is set.	0x0

Bits	Field Name	Description	Default Value
2	dropAll	The packet shall drop all instances and update counter **L2 Action Table Drop**. However special packets which are allowed will still be allowed into the switch (using the field **useSpecialAllow** set to one and register **Allow Special Frame Check For L2 Action Table**)	0x0
3	drop	The packet shall only drop on the ports which hits this action.	0x0
4	dropPortMove	The packet shall be dropped if the result from the learning lookup is port-move.	0x0
5	sendToCpu	The packet shall be send to the CPU.	0x0
6	noPortMove	No port move is allowed for this packet.	0x0
7	useSpecialAllow	Use the special frame checks on this port. 0 = No. 1 = Yes.	0x0
9:8	allowPtr	Pointer to allow special packets defined in **Allow Special Frame Check For L2 Action Table**.	0x0
10	mmpValid	If set, this entry contains a valid MMP pointer	0x0
15:11	mmpPtr	Ingress MMP pointer.	0x0
17:16	mmpOrder	Ingress MMP pointer order.	0x0

34.10.91 L2 Action Table Egress Port State

The egress port state for the L2 Action Table Lookup.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30557

Field Description

Bits	Field Name	Description	Default Value
8:0	state	What is the egress port status bits in the L2 Action Table for the egress port. Bit [0] are used for port 0, Bits [1] are used for port 1 and so on.	0x0

34.10.92 L2 Action Table Source Port

The L2 action table for source port is looked up at the same time as the **L2 Action Table** and its result is merged with the lookup from the **L2 Action Table** table, this lookup is active when enabled in the **Source Port Table** field **enableL2ActionTable** is set to one. The **L2 Action Table** is enabled for each of the destination ports the packet is going to, this table is looked up based on the source port and even if the packet is going to no destination ports this lookup is still carried out. Another difference between **L2 Action Table** and this table is that the highest address bit (bit 6) which uses the status from the L2 SA Lookup and if the packet is going to do a port move then this address bit is high.

Number of Entries :	128
Type of Operation :	Read/Write
Addressing :	Address Bit 0: Source Port State Bit from **Source Port Table** field **l2ActionTablePortState**. Address Bit 1: L2 SA Table was a hit. 0 = Miss. 1 = Hit. Address Bit 2: L2 SA Table - L2 Action Table Status bit. Address Bit 3: L2 DA Table - L2 Action Table Status bit. If this table was a miss then this bit will be zero. Address Bit [5:4]: L2 Packet Type. 0 = L2 Dest Table was a Unicast. 1 = L2 Dest Table was Multicast. 2 = L2 DA table was a miss and packet is being flooded. 3 = Packet was a Broadcast packet and L2 Dest Table did not hit. If both flooded and L2 Broadcast packet then this option will be selected. Address Bit [6]: Port Move. Result bit from L2 SA lookup if the packet shall do a port move or not.
Address Space :	30343 to 30470

Field Description

Bits	Field Name	Description	Default Value
0	noLearningUc	The packet shall not be learned. This is applied to L2 DA MAC unicast packets.	0x0
1	noLearningMc	If the packet is a L2 Multicast then the packet shall not be learned. If a packet is a L2 Multicast depends on if the SA MAC MC bit is set.	0x0
2	dropAll	The packet shall drop all instances and update counter **L2 Action Table Drop**. However special packets which are allowed will still be allowed into the switch (using the field **useSpecialAllow** set to one and register **Allow Special Frame Check For L2 Action Table**)	0x0
3	drop	The packet shall only drop on the ports which hits this action.	0x0
4	dropPortMove	The packet shall be dropped if the result from the learning lookup is port-move.	0x0
5	sendToCpu	The packet shall be send to the CPU.	0x0
6	noPortMove	No port move is allowed for this packet.	0x0
7	useSpecialAllow	Use the special frame checks on this port. 0 = No. 1 = Yes.	0x0
9:8	allowPtr	Pointer to allow special packets defined in **Allow Special Frame Check For L2 Action Table**.	0x0
10	mmpValid	If set, this entry contains a valid MMP pointer	0x0
15:11	mmpPtr	Ingress MMP pointer.	0x0
17:16	mmpOrder	Ingress MMP pointer order.	0x0

34.10.93 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 :	31091 to 31106

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

34.10.94 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 :	292 to 307

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

34.10.95 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**.

Number of Entries :	1024	
Type of Operation :	Read/Write	
Addressing :	address[0:7] :	hash of {GID, destination MAC}
	address[8:9] :	bucket number
Address Space :	26103 to 27126	

Field Description

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

34.10.96 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**.

Number of Entries :	1024	
Type of Operation :	Read/Write	
Addressing :	address[0:7] :	hash of {GID, destination MAC}
	address[8:9] :	bucket number
Address Space :	317 to 1340	

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

34.10.97 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. .

Number of Entries :	1024	
Number of Addresses per Entry :	2	
Type of Operation :	Read/Write	
Addressing :	address[0:7] :	hash of {GID, destination MAC}
	address[8:9] :	bucket number
Address Space :	27127 to 29174	

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddr	MAC address.	0x0
56:48	gid	Global identifier from the VLAN Table.	0x0

34.10.98 L2 Destination Table

This table contains either a destination port or a pointer to the L2 multicast table..

Number of Entries :	1040
Type of Operation :	Read/Write
Addressing :	address 0 to 1023 : **L2 DA Hash Lookup Table** address address 1024 to 1039 : **L2 Lookup Collision Table** address
Address Space :	29175 to 30214

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
8	pktDropSa	If set, the packet will be dropped if this packet was hit with the SA search and the **L2 Destination Table SA Lookup Drop** incremented.	0x0
9	l2ActionTableDaStatus	The status DA bit to be used in the addressing for the table **L2 Action Table** Lookup.	0x0
10	l2ActionTableSaStatus	The status SA bit to be used in the addressing for the table **L2 Action Table** Lookup.	0x0

34.10.99 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 :	32948 to 32979

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddr	MAC address	0x0
56:48	gid	Global identifier for learning	0x0

34.10.100 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 :	32940 to 32947

Field Description

Bits	Field Name	Description	Default Value
47:0	macAddr	MAC address mask	$2^{48} - 1$
56:48	gid	Global identifier for learning mask	0x1ff

34.10.101 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 :	30558

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

34.10.102 L2 Multicast Table

L2 multicast table.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	mcAddr field from **L2 Destination Table**
Address Space :	31027 to 31090

Field Description

Bits	Field Name	Description	Default Value
8:0	mcPortMask	L2 portmask entry members. If set, the port is part of multicast group and shall be transmitted to.	0x1ff

34.10.103 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 :	32241 to 32496

Field Description

Bits	Field Name	Description	Default Value
8:0	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
17:9	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
26:18	sendToPortMask	Send the packet to a specific port. 0 = Do not sent to a port. 1 = Send to port.	0x0
30:27	destPort	The port which the packet shall be sent to.	0x0

34.10.104 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 :	32850

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

34.10.105 L4 Port Range to Queue Assignment

This register allows each egress port to determine the egress queue based on a L4 (TCP or UDP) source or destination port range. IPv4 as well as IPv6 protocol will match. The highest numbered index will determine the result if there are multiple matching ranges.

Number of Entries :	36
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	egress port * 4 + entry number
Address Space :	32980 to 33051

Field Description

Bits	Field Name	Description	Default Value
15:0	start_port	The start of an L4 port range.	0x0
31:16	end_port	The end of an L4 port range.	0x0
32	spOrDp	Select if a Source or Destination Port should be matched. 0 = Source Port 1 = Destination Port	0x0
33	udpTcp	Select if TCP or UDP packets should be matched. 0 = UDP 1 = TCP	0x0
34	force	Force the egress queue if the port matches.	0x0
37:35	queue	The egress queue to assign for this port	0x0

34.10.106 L4 Protocol to Queue Assignment

This register allows each egress port determine the queue based on a L4 protocol type for IPv4 or IPv6 packets.

Number of Entries :	36
Type of Operation :	Read/Write
Addressing :	egress port * 4 + entry number
Address Space :	31679 to 31714

Field Description

Bits	Field Name	Description	Default Value
7:0	proto	The L4 protocol type to match.	0x0
8	force	Force the queue if the L4 protocol is a match.	0x0
11:9	queue	The queue to assign for this port.	0x0

34.10.107 LACP Packet Decoder Options

This is the MAC address used to determine that a packet is a LACP packet. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Address Space :	32838

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
48:1	mac	The value to be used to find this packet type.	0x180c2000002
57:49	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
66:58	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.108 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 :	32842

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
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
169:161	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.	0xff

34.10.109 Latent Error Detection Configuration

Configurations for the latent error detection function.

Number of Entries :	32
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	35951 to 36078

Field Description

Bits	Field Name	Description	Default Value
31:0	latentErrorTestPeriod	Number of ticks (**Latent Error Detection Tick**) for the latent error test period. Set to zero to disable latent error detection. Only valid when the sequenceRecovery field is set to one.	0x14
63:32	latentErrorDifference	An error flag is raised when the offset counter between two latent error test period exceed this value.	0x64
67:64	latentErrorPaths	Number of member streams in the compound stream used by the latent error detection.	0x2
99:68	latentResetPeriod	Number of ticks to reset the latent error test and reload the current offset as the base offset for future comparisons.	0x64

34.10.110 Latent Error Detection Tick

Latent error detection period is counting based on this tick.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	35758

Field Description

Bits	Field Name	Description	Default Value
2:0	periodTick	Select one of the 5 available ticks. The tick frequencies are configured globaly in the Core Tick Configuration register.	0x0

34.10.111 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 :	280

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

34.10.112 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 :	2
Type of Operation :	Read/Write
Address Space :	276

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
57:49	gid	Global identifier from the VLAN Table.	0x0
61:58	port	Port number.	0x0

34.10.113 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 :	2
Type of Operation :	Read/Write
Address Space :	278

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
57:49	gid	Global identifier from the VLAN Table.	0x0
61:58	port	Port number.	0x0

34.10.114 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 :	30733 to 30988

Field Description

Bits	Field Name	Description	Default Value
8:0	ports	One bit per physical port.	0x0

34.10.115 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 :	30543

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	useVlanIdInHash	The packets VLAN Identifier tag shall be part of the hash key when calculating the link aggregate hash value.	0x0

34.10.116 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 :	9
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	32497 to 32505

Field Description

Bits	Field Name	Description	Default Value
3:0	la	The Link aggregation which this port is a member of	0x0

34.10.117 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 :	9
Type of Operation :	Read/Write
Addressing :	The link aggregate number.
Address Space :	30724 to 30732

Field Description

Bits	Field Name	Description	Default Value
8:0	members	Physical ports that are members of this link aggregate. One bit per port.	0x0

34.10.118 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 :	31786 to 31793

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueuePort_0	Egress queue for port 0	0x1
5:3	pQueuePort_1	Egress queue for port 1	0x1
8:6	pQueuePort_2	Egress queue for port 2	0x1
11:9	pQueuePort_3	Egress queue for port 3	0x1
14:12	pQueuePort_4	Egress queue for port 4	0x1
17:15	pQueuePort_5	Egress queue for port 5	0x1
20:18	pQueuePort_6	Egress queue for port 6	0x1
23:21	pQueuePort_7	Egress queue for port 7	0x1
26:24	pQueuePort_8	Egress queue for port 8	0x1

34.10.119 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 :	31107 to 31114

Field Description

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

34.10.120 Max SDU Filter

Packet length check based on the stream filter ID. Packet failed to pass the filter will be dropped and can optionally block the further stream with the same stream filter ID.

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35180 to 35195

Field Description

Bits	Field Name	Description	Default Value
14:0	maxSDU	Maximum number of bytes for a single packet. A packet is dropped if the packet length exceeds this value. Value 0 means no check.	0x7fff
15	blockingEn	If this field is set to 1, A packet failed to pass the max SDU filter will set **Max SDU Filter Blocking** to 1 and then block all traffic with the same stream filter ID. The blocking status can be cleared by writing 0 to this field or the **Max SDU Filter Blocking** register.	0x0

34.10.121 Max SDU Filter Blocking

Blocking status of the stream filter.

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35196 to 35211

Field Description

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

34.10.122 Port Move Options

Determine if port move is allowed on static entries.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30556

Field Description

Bits	Field Name	Description	Default Value
8:0	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.	0x1ff

34.10.123 RARP Packet Decoder Options

The Ethernet type used to determine if a packet is a RARP packet.. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Address Space :	32854

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
16:1	eth	The value to be used to find this packet type.	0x8035
25:17	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
34:26	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.124 Recovery Tick

Recovery timeout is counting based on this tick.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	35757

Field Description

Bits	Field Name	Description	Default Value
2:0	timeoutTick	Select one of the 5 available ticks. The tick frequencies are configured globaly in the Core Tick Configuration register.	0x0

34.10.125 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 :	32818 to 32833

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 18.1	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
110:106	mmpPtr	Ingress MMP pointer.	0x0
112:111	mmpOrder	Ingress MMP pointer order.	0x0

Bits	Field Name	Description	Default Value
121:113	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

34.10.126 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 :	32802 to 32817

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
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 18.1	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
110:106	mmpPtr	Ingress MMP pointer.	0x0
112:111	mmpOrder	Ingress MMP pointer order.	0x0
121:113	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

34.10.127 SCTP Packet Decoder Options

The L4 protocol number which is used to detemine if the packet has a SCTP header. If both the send to cpu option and drop packet option is selected on same source port then the packet will be dropped.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	30546

Field Description

Bits	Field Name	Description	Default Value
0	enabled	Is this decoding enabled. 0 = No 1 = Yes	0x1
8:1	l4Proto	The value to be used to find this packet type.	0x84
17:9	drop	If a packet comes in on this source port then drop the packet. 0 = Do not drop this packet. 1 = Drop this packet and update the drop counter.	0x0
26:18	toCpu	If a packet comes in on this source port then send the packet to the CPU port. 0 = Do not sent to CPU. Normal Processing of packet. 1 = Send to CPU , bypass normal packet processing.	0x0

34.10.128 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 :	32111 to 32112

Field Description

Bits	Field Name	Description	Default Value
3:0	srcPort	Source port	0x0
15:4	vid	VLAN ID	0x0

34.10.129 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 :	32834

Field Description

Bits	Field Name	Description	Default Value
8:0	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.	0x1ff
17:9	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.	0x1ff
26:18	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
74:27	cpuMacAddr	Packets with this destination MAC address will be sent to the CPU. Only valid if **uniqueCpuMac** on the source port is set.	0x0

34.10.130 Sequence Recovery Config

Configurations for the sequence recovery function.

Number of Entries :	32
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	35887 to 35950

Field Description

Bits	Field Name	Description	Default Value
0	sequenceRecovery	Apply sequence recovery	0x1
1	algo	Sequence recovery algorithm. 0 = Vector 1 = Match	0x0
5:2	historyLen	Specify the valid number of bits in the sequence history. Only valid for the vector recovery algorighm and the minimum value is 2.	0x8
37:6	timeoutCnt	Number of ticks (see Chapter Tick) for the timeout period. The timeout is only valid for non zero values.	0x0
38	takeNoSequence	0 = Drop packets without sequence number. 1 = Accept packets without sequence number.	0x0

34.10.131 Sequence Recovery Reset

Reset the sequence history of the sequence recovery function and allow any sequence number for the next packet.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	35725 to 35756

Field Description

Bits	Field Name	Description	Default Value
0	reset	Set to one to reset the sequence recovery function. Hardware clears the reset after one clock cycle.	0x0

34.10.132 Source Port Default ACL Action

The default ACL action which will be taken on a source port if the **enableDefaultPortAcl** is set and the ACL lookup misses. The action will also be taken if the **forcePortAclAction** is set and then it will override the result from the ACL even if the ACL was hit or not.

Number of Entries :	9
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	Source Port
Address Space :	32702 to 32737

Field Description

Bits	Field Name	Description	Default Value
0	inputMirror	If set, input mirroring is enabled for this rule. In addition to the normal processing of the packet a copy of the unmodified input packet will be send to the destination Input Mirror port and exit on that port. The copy will be subject to the normal resource limitations in the switch.	0x0
4:1	destInputMirror	Destination physical port for input mirroring.	0x0
5	noLearning	If set this packets MAC SA will not be learned.	0x0
6	streamValid	If set, this entry contains a valid stream handle	0x0
12:7	streamHandle	Stream handle.	0x0
13	updateCounter	When set the selected statistics counter will be updated.	0x0
18:14	counter	Which counter in **Ingress Configurable ACL Match Counter** to update.	0x0
19	forceVidValid	Override the Ingress VID, see chapter VLAN Processing.	0x0
31:20	forceVid	The new Ingress VID.	0x0
32	updateCfiDei	The CFI/DEI value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
33	newCfiDeiValue	The value to update to.	0x0

Bits	Field Name	Description	Default Value
34	updatePcp	The PCP value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
37:35	newPcpValue	The PCP value to update to.	0x0
38	updateVid	The VID value of the packets outermost VLAN should be updated. 0 = Do not update the value. 1 = Update the value.	0x0
50:39	newVidValue	The VID value to update to.	0x0
51	updateEType	The VLANs TPID type should be updated. 0 = Do not update the TPID. 1 = Update the TPID.	0x0
53:52	newEthType	Select 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
54	dropEnable	If set, the packet shall be dropped and the **Ingress Configurable ACL Drop** counter is incremented.	0x0
55	sendToCpu	If set, the packet shall be sent to the CPU port.	0x0
56	sendToPort	Send the packet to a specific port. 0 = Disabled. 1 = Send to port configured in destPort.	0x0
60:57	destPort	The port which the packet shall be sent to.	0x0
61	forceColor	If set, the packet shall have a forced color.	0x0
63:62	color	Initial color of the packet if the forceColor field is set.	0x0
64	mmpValid	If set, this entry contains a valid MMP pointer	0x0
69:65	mmpPtr	Ingress MMP pointer.	0x0
71:70	mmpOrder	Ingress MMP pointer order.	0x0
72	forceQueue	If set, the packet shall have a forced egress queue. Please see Egress Queue Selection Diagram in Figure 18.1	0x0
75:73	eQueue	The egress queue to be assigned if the forceQueue field in this entry is set to 1.	0x0

34.10.133 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 :	9
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	32506 to 32541

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
1	dropUnknownDa	If set to one packets with unknown destination MAC address from this port will be dropped.	0x0
2	colorFromL3	If the packet is IP/MPLS and this bit is set the packet initial color will be selected from Layer 3 decoding.	0x0
3	useAcl0	Use ACL on this source port. 0 = No. No ACL lookup is done 1 = Yes. The aclRule0 pointer selects which fields that are part of the lookup .	0x0
6:4	aclRule0	Pointer into the **Ingress Configurable ACL 0 Rules Setup** table selecting which ACL fields to select to do the ACL lookup with.	0x0
7	useAcl1	Use ACL on this source port. 0 = No. No ACL lookup is done 1 = Yes. The aclRule1 pointer selects which fields that are part of the lookup .	0x0
10:8	aclRule1	Pointer into the **Ingress Configurable ACL 1 Rules Setup** table selecting which ACL fields to select to do the ACL lookup with.	0x0
13:11	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
15:14	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
17:16	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

Bits	Field Name	Description	Default Value
19:18	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
21:20	nrVlansVidOperationIf	This alternative VID operation for port VLAN operation is selected if the following operation is true. 0 = Nr of VLANS in incoming packet is zero. 1 = Nr of VLANS in incoming packet is one. 2 = Nr of VLANS in incoming packet is two. 3 = Reserved and Disabled	0x3
24:22	vlanSingleOpIf	If the field **nrVlansVidOperationIf** is true then this operation will override the default port vid operation **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
26:25	vidSelIf	If the field **nrVlansVidOperationIf** is true then this operation will override the default port vid operation **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 **defaultVidIf** 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
28:27	cfiDeiSelIf	If the field **nrVlansVidOperationIf** is true then this operation will override the default port vid operation **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 **defaultCfiDeiIf** 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

Bits	Field Name	Description	Default Value
30:29	pcpSelIf	If the field **nrVlansVidOperationIf** is true then this operation will override the default port vid operation **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 **defaultPcpIf** 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
32:31	typeSelIf	If the field **nrVlansVidOperationIf** is true then this operation will override the default port vid operation **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
44:33	defaultVidIf	The default VID if **nrVlansVidOperationIf** is true. 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
45	defaultCfiDeiIf	The default CFI / DEI bit if **nrVlansVidOperationIf** is true. 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
48:46	defaultPcpIf	The default PCP bits if **nrVlansVidOperationIf** is true. 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
50:49	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
52:51	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
64:53	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
65	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
68:66	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
70:69	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
72:71	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
74:73	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
75	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

Bits	Field Name	Description	Default Value
76	learnMulticastSaMac	If set, the learning engine allows Ethernet multicast source MAC addresses to be learned.	0x0
77	learnMacDaEqSa	Set to zero to ignore the hardware learning request when MAC DA equals SA.	0x1
78	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
79	imUnderVlanMembership	If set, input mirroring to a destination that not a member of the VLAN will be ignored.	0x0
80	imUnderPortIsolation	If set, input mirroring to a destination that isolated the source port in the **srcPortFilter** will be ignored.	0x0
84:81	destInputMirror	Destination physical port for input mirroring. Only valid if **inputMirrorEnabled** is set.	0x0
87:85	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
88	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
100:89	priorityVid	The VID used in the outer VLAN tag comparison, see **enablePriorityTag**.	0x0
101	enableFromCpuTag	This option can validate the from CPU tag decoding on packets from non-CPU ports. The CPU port is not affected by this field and always decode the from CPU tag.	0x0
102	enableL2ActionTable	On packets coming in on this port should be checked with the **L2 Action Table** and **L2 Action Table Source Port**. 0 = No, Do not lookup on the **L2 Action Table** and **L2 Action Table Source Port**. 1 = Yes. Do Lookup in the **L2 Action Table** and **L2 Action Table Source Port**	0x0
103	l2ActionTablePortState	What is the source port status bit. Used in table **L2 Action Table** and **L2 Action Table Source Port**.	0x0

Bits	Field Name	Description	Default Value
104	enableDefaultPortAcl	If enabled then the default acl for this port will be done if the ACL misses in its lookup. 0 = Disabled. No default action taken. 1 = Enabled. If ACL lookup misses then this ACL actil will be carried out instead.	0x0
105	forcePortAclAction	If enabled then the default acl for this port will always be done, if the ACL is hit then the port ACL will overwrite the ACL result. 0 = Disabled. Not action forced. 1 = Enabled. The port ACL overwrites and result from the ingress ACL.	0x0
107:106	preLookupAclBits	Pre lookup bits which is used by this port in the pre-lookup tables in the ingress ACLS. Same value is used for all pre ACL lookups which has the source port bits in it.	0x0

34.10.134 Stream Filter Lookup Table

This table optionally takes the streamHandle and priority of the packet to select a stream filter. If a packet hits multiple entries, the earliest one will be returned as the hit index. Reference: 8.6.5.1 of IEEE Std 802.1Qci-2017

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream filter ID
Address Space :	30690 to 30705

Field Description

Bits	Field Name	Description	Default Value
0	enable	If set, this is a valid entry for comparison	0x0
1	compareStreamHandle	Determines if the streamHandle field in this entry shall be compared. 0 = Do not compare 1 = Include the streamHandle comparison in the entry comparison	0x0
7:2	streamHandle	Stream ID assigned from ACL hits.	0x0
8	comparePriority	Determines if the priority filed in this entry shall be compared. 0 = Do not compare 1 = Include the priority comparison in the entry comparison	0x0
11:9	priority	L2 Priority of the packet.	0x0
16:12	gateId	Gate ID for the stream gate control	0x0
17	mmpValid	If set, this entry contains a valid MMP pointer.	0x0
22:18	mmpPtr	Ingress MMP pointer.	0x0
24:23	mmpOrder	Ingress MMP pointer order.	0x0

34.10.135 Stream Gate Blocking Enable

Enable blocking stream gate when a closed gate receives a packet, or the total number of L2 payload bytes exceed a certain limit in the current time interval for a open gate.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Gate ID
Address Space :	35212 to 35243

Field Description

Bits	Field Name	Description	Default Value
0	invalidRxBlockingEn	After setting this field to 1, any packet hits a closed gate will put the gate under a blocking status and drop all traffic to the gate regardless of open or closed.	0x0
1	maxMsduBlockingEn	After setting this field to 1, if the total number of L2 payload bytes within the current gate time interval is more than the corresponding **`maxMSDU`**, the gate will be turned into a blocking status and drop all traffic to the gate regardless of open or closed.	0x0

34.10.136 Stream Gate Invalid RX Blocking

Blocking status of the stream gate due to invalid RX.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Gate ID
Address Space :	35244 to 35275

Field Description

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

34.10.137 Stream Gate Max MSDU Blocking

Blocking status of the stream gate due to exceeding maximum MSDU.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Gate ID
Address Space :	35276 to 35307

Field Description

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

34.10.138 Stream Handle To FRER Mapping Table

Assign a FRER ID to the stream. When a single FRER ID is used for multiple stream handles, usually that FRER ID is in recovery mode and each stream handle represents a member stream.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	Stream handle
Address Space :	30626 to 30689

Field Description

Bits	Field Name	Description	Default Value
4:0	frerId	Streams belong to the same FRER ID will be subjected to the same FRER instance.	0x0

34.10.139 TCP/UDP Flag Rules

IPv4/IPv6 TCP/UDP packets will be compared to all entries in this table. The TCP/UDP flags values can be compared by enabling some of the comparisons. The packets flags will be compared with the values in the entries for all flags that have comparison enabled. If comparison is disabled the flags values will be ignored. In addition the packets IP source and destination addresses are compared and if they are equal this status can also be used in the rules. The TCP source and destination ports are also compared if equal and this status can also be used in the rules. If a packet matches any of these rules the packet will be dropped and the **Attack Prevention Drop** will be incremented.

Number of Entries :	4
Type of Operation :	Read/Write
Addressing :	All entries are read out in parallel
Address Space :	31794 to 31797

Field Description

Bits	Field Name	Description	Default Value
0	urg	TCP flag URG compare value.	0x0
1	ack	TCP flag ACK compare value.	0x0
2	psh	TCP flag PSH compare value.	0x0
3	rst	TCP flag RST compare value.	0x0
4	syn	TCP flag SYN compare value.	0x0
5	fin	TCP flag FIN compare value.	0x0
6	DaSa	Value of IP address comparison.	0x0

Bits	Field Name	Description	Default Value
7	SpDpTcp	Value of TCP port comparison.	0x0
8	SpDpUdp	Value of UDP port comparison.	0x0
9	cmpUrg	Enable comparison of URG.	0x0
10	cmpAck	Enable comparison of ACK.	0x0
11	cmpPsh	Enable comparison of PSH.	0x0
12	cmpRst	Enable comparison of RST.	0x0
13	cmpSyn	Enable comparison of SYN.	0x0
14	cmpFin	Enable comparison of FIN.	0x0
15	cmpDaSa	Enable comparison of IP DA equal to SA.	0x0
16	cmpSpDpTcp	Enable comparison of TCP source port equal to destination port.	0x0
17	cmpSpDpUdp	Enable comparison of UDP source port equal to destination port.	0x0
18	enable	Enable this rule.	0x0

34.10.140 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 :	290

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

34.10.141 VID to Queue Assignment

This register allows each egress port to determine the queue based on an inner or outer VID. The VID is compared with the incoming packet header before any VLAN operations.

Number of Entries :	36
Type of Operation :	Read/Write
Addressing :	egress port * 4 + entry number
Address Space :	31643 to 31678

Field Description

Bits	Field Name	Description	Default Value
11:0	vid	The Packets VID to match. Bits that are masked must be 0 in this field.	0x0
23:12	mask	The VID mask. Setting a bit to 0 means this bit will not be compared. The corresponding bit in the vid field must be 0.	0xfff
24	innerOuter	Select if this entry shall compare VID in an inner or outer VLAN Tag. 0 = Outer 1 = Inner	0x0
25	cstag	Select if this entry shall compare VID in a Customer VLAN Tag or a Service VLAN Tag. 0 = C-VID 1 = S-VID	0x0
26	force	Force the queue if the VID is a match.	0x0
29:27	queue	The queue to assign for this port	0x0

34.10.142 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 :	31627 to 31642

Field Description

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

34.10.143 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 :	31778 to 31785

Field Description

Bits	Field Name	Description	Default Value
2:0	pQueuePort_0	Egress queue for egress port 0.	0x1
5:3	pQueuePort_1	Egress queue for egress port 1.	0x1
8:6	pQueuePort_2	Egress queue for egress port 2.	0x1
11:9	pQueuePort_3	Egress queue for egress port 3.	0x1
14:12	pQueuePort_4	Egress queue for egress port 4.	0x1

Bits	Field Name	Description	Default Value
17:15	pQueuePort_5	Egress queue for egress port 5.	0x1
20:18	pQueuePort_6	Egress queue for egress port 6.	0x1
23:21	pQueuePort_7	Egress queue for egress port 7.	0x1
26:24	pQueuePort_8	Egress queue for egress port 8.	0x1

34.10.144 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 :	4
Type of Operation :	Read/Write
Addressing :	The packet's Ingress VID plus offset as defined in **Source Port Table**.
Address Space :	9207 to 25590

Field Description

Bits	Field Name	Description	Default Value
8:0	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.	0x1ff
17:9	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
18	mmpValid	If set, this entry contains a valid MMP pointer	0x0
23:19	mmpPtr	Ingress MMP pointer.	0x0
25:24	mmpOrder	Ingress MMP pointer order.	0x0
29:26	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
32:30	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

Bits	Field Name	Description	Default Value
34:33	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
36:35	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
38:37	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
50:39	vid	The VID used in VLAN push or swap operation if selected by **vidSel**.	0x0
53:51	pcp	The PCP used in VLAN push or swap operation if selected by **pcpSel**.	0x0
54	cfiDei	The CFI/DEI used in VLAN push or swap operation if selected by **cfiDeiSel**	0x0
56:55	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
74:57	nrVlansVidOperationIf	A per source port setting. Port 0 uses bits [1:0], port 2 uses bits [3:2] and so on. If the packet coming in on the source port has this amount of VLANs then this operation will override the VLAN Tables VID operation and all associated data. This operation does take into account what operation the source port VID operation performed on the packet. If a already has 2 VLANs and a push operation is done it will still be counted as a packet with two vlans. If a packet has zero vlans and a pop operation is carried out it will still have zero VLANs. Swap operations does not change the number of VLANs on the packet. 0 = Incoming packet after source port VID op has zero VLANs 1 = Incoming packet after source port VID op has one VLAN 2 = Incoming packet after source port VID op has Two VLANs 3 = Reserved and Disabled	$2^{18}-1$
77:75	vlanSingleOpIf	This operation depends on if the **nrVlansVidOperationIf** is done on this port. Then the default operation is overriden with this value. 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
79:78	vidSelIf	This operation depends on if the **nrVlansVidOperationIf** is done on this port. Then the default operation is overriden with this value. Selects which VID to use when building a new VLAN header in a push or swap operation. 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
81:80	cfiDeiSelIf	This operation depends on if the **nrVlansVidOperationIf** is done on this port. 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
83:82	pcpSelIf	This operation depends on if the **nrVlansVidOperationIf** is done on this port. 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
85:84	typeSelIf	This operation depends on if the **nrVlansVidOperationIf** is done on this port. Then the default operation is overriden with this value. 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
97:86	vidIf	If this data is used depends on if the **nrVlansVidOperationIf** is done on this port. Then the default operation is overriden with this value. The VID used in VLAN push or swap operation if selected by **vidSel**.	0x0
100:98	pcpIf	If this data is used depends on if the **nrVlansVidOperationIf** is done on this port. Then the default operation is overriden with this value. The PCP used in VLAN push or swap operation if selected by **pcpSel**.	0x0
101	cfiDeiIf	If this data is used depends on if the **nrVlansVidOperationIf** is done on this port. Then the default operation is overriden with this value. The CFI/DEI used in VLAN push or swap operation if selected by **cfiDeiSel**	0x0

34.11 MBSC

34.11.1 L2 Broadcast Storm Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for L2 Broadcast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	137 to 145

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	Index / Value: 0-1 0x1b8; 2-8 0x5c8

34.11.2 L2 Broadcast Storm Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for L2 Broadcast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	146 to 154

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.	Index / Value: 0-1 0xdc; 2-8 0x2e4

34.11.3 L2 Broadcast Storm Control Current Size

Number of tokens currently in the token bucket.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Egress Ports
Address Space :	155 to 163

Field Description

Bits	Field Name	Description	Default Value
15:0	currentVal	Number of tokens currently in the token bucket for this Egress Ports	Index / Value: 0-1 0xdc; 2-8 0x2e4

34.11.4 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 :	164

Field Description

Bits	Field Name	Description	Default Value
8:0	enable	Bitmask where the index is the Egress Ports	0x0

34.11.5 L2 Broadcast Storm Control Rate Configuration

Token Bucket rate Configuration for L2 Broadcast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	128 to 136

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	Index Value 0-1 0x16 2-8 0x4a
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	Index Value 0-1 0x2 2-8 0x3
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

34.11.6 L2 Multicast Storm Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for L2 Multicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	174 to 182

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	Index Value 0-1 0x1b8 2-8 0x5c8

34.11.7 L2 Multicast Storm Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for L2 Multicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	183 to 191

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.	Index Value 0-1 0xdc 2-8 0x2e4

34.11.8 L2 Multicast Storm Control Current Size

Number of tokens currently in the token bucket.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Egress Ports
Address Space :	192 to 200

Field Description

Bits	Field Name	Description	Default Value
15:0	currentVal	Number of tokens currently in the token bucket for this Egress Ports	Index Value 0-1 0xdc 2-8 0x2e4

34.11.9 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 :	201

Field Description

Bits	Field Name	Description	Default Value
8:0	enable	Bitmask where the index is the Egress Ports	0x0

34.11.10 L2 Multicast Storm Control Rate Configuration

Token Bucket rate Configuration for L2 Multicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	165 to 173

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	Index Value 0-1 0x16 2-8 0x4a
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	Index Value 0-1 0x2 2-8 0x3
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

34.11.11 L2 Unknown Multicast Storm Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for L2 Unknown Multicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	248 to 256

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	Index Value 0-1 0x1b8 2-8 0x5c8

34.11.12 L2 Unknown Multicast Storm Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for L2 Unknown Multicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	257 to 265

Field Description

Bits	Field Name	Description	Default Value	
			Index	Value
15:0	threshold	Minimum number of tokens in bucket for the status to be set to accept.	0-1	0xdc
			2-8	0x2e4

34.11.13 L2 Unknown Multicast Storm Control Current Size

Number of tokens currently in the token bucket.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Egress Ports
Address Space :	266 to 274

Field Description

Bits	Field Name	Description	Default Value	
			Index	Value
15:0	currentVal	Number of tokens currently in the token bucket for this Egress Ports	0-1	0xdc
			2-8	0x2e4

34.11.14 L2 Unknown Multicast Storm Control Enable

Bitmask to turn L2 Unknown Multicast Storm Control ON/OFF (1/0) for Egress Ports

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	275

Field Description

Bits	Field Name	Description	Default Value
8:0	enable	Bitmask where the index is the Egress Ports	0x0

34.11.15 L2 Unknown Multicast Storm Control Rate Configuration

Token Bucket rate Configuration for L2 Unknown Multicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	239 to 247

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	Index Value 0-1 0x16 2-8 0x4a
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	Index Value 0-1 0x2 2-8 0x3
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

34.11.16 L2 Unknown Unicast Storm Control Bucket Capacity Configuration

Token Bucket Capacity Configuration for L2 Unknown Unicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	211 to 219

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	Index Value 0-1 0x1b8 2-8 0x5c8

34.11.17 L2 Unknown Unicast Storm Control Bucket Threshold Configuration

Token Bucket Threshold Configuration for L2 Unknown Unicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	220 to 228

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.	Index Value 0-1 0xdc 2-8 0x2e4

34.11.18 L2 Unknown Unicast Storm Control Current Size

Number of tokens currently in the token bucket.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Egress Ports
Address Space :	229 to 237

Field Description

Bits	Field Name	Description	Default Value
15:0	currentVal	Number of tokens currently in the token bucket for this Egress Ports	Index Value 0-1 0xdc 2-8 0x2e4

34.11.19 L2 Unknown Unicast Storm Control Enable

Bitmask to turn L2 Unknown Unicast Storm Control ON/OFF (1/0) for Egress Ports

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	238

Field Description

Bits	Field Name	Description	Default Value
8:0	enable	Bitmask where the index is the Egress Ports	0x0

34.11.20 L2 Unknown Unicast Storm Control Rate Configuration

Token Bucket rate Configuration for L2 Unknown Unicast Storm Control

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	202 to 210

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	Index Value 0-1 0x16 2-8 0x4a

Bits	Field Name	Description	Default Value
15:13	tick	Select one of the five available core ticks. The tick frequencies are configured globaly in the core Tick Configuration register.	Index / Value: 0-1 0x2; 2-8 0x3
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18

34.12 Scheduling

34.12.1 DWRR Bucket Capacity Configuration

Token Bucket Capacity Configuration for DWRR

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	37394 to 37402

Field Description

Bits	Field Name	Description	Default Value
17:0	bucketCapacity	Capacity of the byte bucket	$2^{18} - 1$

34.12.2 DWRR Bucket Misc Configuration

Bucket Configurations for DWRR

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Ports
Address Space :	37403 to 37411

Field Description

Bits	Field Name	Description	Default Value
4:0	threshold	When the number of bytes in any bucket goes below 2**thr, all buckets mapped to the same prio will be replenished.	0xe
5	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x0
13:6	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode.	0x14

34.12.3 DWRR Current Size

Number of bytes currently in the bucket.

Number of Entries :	72
Type of Operation :	Read Only
Addressing :	Egress Ports
Address Space :	37484 to 37555

Field Description

Bits	Field Name	Description	Default Value
17:0	currentVal	Number of bytes currently in the bucket for this Egress Ports	0x3fff

34.12.4 DWRR Rank

DWRR current queue ranks. The rank of queue N is defined as the number of queues on the same port that have a lower bucket level than queue N.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Egress Port
Address Space :	37556 to 37564

Field Description

Bits	Field Name	Description	Default Value
2:0	rank0	DWRR rank for queue 0	0x0
5:3	rank1	DWRR rank for queue 1	0x0
8:6	rank2	DWRR rank for queue 2	0x0
11:9	rank3	DWRR rank for queue 3	0x0
14:12	rank4	DWRR rank for queue 4	0x0
17:15	rank5	DWRR rank for queue 5	0x0
20:18	rank6	DWRR rank for queue 6	0x0
23:21	rank7	DWRR rank for queue 7	0x0

34.12.5 DWRR Weight Configuration

Weight Configuration for DWRR

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Egress port * 8 + queue
Address Space :	37412 to 37483

Field Description

Bits	Field Name	Description	Default Value
7:0	weight	The relative weight of the queue. A queue with weight 0 is not part of the round robin scheduling but will always be selected last.	0x1

34.12.6 Egress Transmission Gate Base Tick

Select one of the 5 available PTP ticks. The tick frequencies are configured globaly in the **PTP Tick Configuration** register. The selected tick is used for counting the current time and the gate cycle time. The frequency shall not be changed when the transmission gate is enabled.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	37565

Field Description

Bits	Field Name	Description	Default Value
2:0	baseTick	PTP tick number. The master tick is number 0.	0x0

34.12.7 Egress Transmission Gate Configuration

Setup configurations for egress transmission gates. Hardware is not aware of the configuration updates unless an update request is triggered by writing 1 to **Egress Transmission Gate Update**. The transmission gate execution will start using the new configuration when the current time meets the **adminBaseTime**.

Number of Entries :	9
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	37578 to 37613

Field Description

Bits	Field Name	Description	Default Value
56:0	adminBaseTime	Determine the start time of the updated gate cycle. The value needs to be larger than **Egress Transmission Gate Current Time** when an update request is issued.	0x0
59:57	adminTick	Select one of the 5 available PTP ticks. The tick frequencies are configured globaly in the **PTP Tick Configuration** register. The selected tick is used for counting time intervals between gate list entries.	0x0

Bits	Field Name	Description	Default Value
86:60	adminCycleTime	Time for one gate cycle based on the **Egress Transmission Gate Base Tick**. Once a gate list starts executing, it will be restarted from the start address again when the elapsed time equals this field.	0x0
113:87	adminCycleTimeExtension	Extra time for retaining the current gate status when the time for a pending update in the future to occur is less than this field. The time extension is based on the **Egress Transmission Gate Base Tick**.	0x0
118:114	adminControlListLength	Number of execution entries in the **Egress Transmission Gate List** for one gate cycle. If the gate cycle time is not enough to execute all the entries, the execution will abort the remaining entries.	0x0
123:119	adminStartAddr	Point to the first entry to execute in one gate cycle.	0x0

34.12.8 Egress Transmission Gate Current Time

Counting the current time based on the **Egress Transmission Gate Base Tick**.

Number of Entries :	1
Number of Addresses per Entry :	2
Type of Operation :	Read Only
Address Space :	37576

Field Description

Bits	Field Name	Description	Default Value
56:0	currentTime	Number of counted base ticks since the reset.	0x0

34.12.9 Egress Transmission Gate Enabled

All Egress Transmission Gate operations require this register set to 1.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	37567 to 37575

Field Description

Bits	Field Name	Description	Default Value
0	enabled	If set, egress transmission gate is enabled.	0x0

34.12.10 Egress Transmission Gate List

Gate control list. Each entry gives gate status for the current time window, as well as a time interval for counting the execution time before jumping to the next entry. When the total number of executed entries reaches the configured list length but a new gate cycle is not started, the gate status for the last entry will be kept till the restart.

Number of Entries :	32
Number of Addresses per Entry :	2
Type of Operation :	Read/Write
Addressing :	Egress Transmission Gate Address
Address Space :	37614 to 37677

Field Description

Bits	Field Name	Description	Default Value
7:0	disableQueueMask	Each bit refers to a gate status for the corresponding egress queue. 0 = Packet transmission is allowed. 1 = Packet transmission is not allowed.	0x0
34:8	timeInterval	Number of ticks before jumping to the next entry in the gate control list. The tick frequency is based on the loaded **adminTick**.	0x0

34.12.11 Egress Transmission Gate Update

When set to one a configuration update request is issued to the hardware. The set bit is always cleared after one cycle and **Egress Transmission Gate Update Status** will be pulled high till the update process is done.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	37678 to 37686

Field Description

Bits	Field Name	Description	Default Value
0	start	Issue an update request to load **Egress Transmission Gate Configuration** to the hardware.	0x0

34.12.12 Egress Transmission Gate Update Status

For each egress port, showing if a new configuration is pending to be updated. **Egress Transmission Gate Configuration** shall not be modified while an update process is pending.

Number of Entries :	1
Type of Operation :	Read Only
Address Space :	37566

Field Description

Bits	Field Name	Description	Default Value
8:0	pending	1 means **Egress Transmission Gate Update** has been issued and 0 means the update operation is done.	0x0

34.12.13 Map Queue to Priority

Map from egress queue to egress priority. Note that this setting must not be changed for any queue with packets queued.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	36752 to 36760

Field Description

Bits	Field Name	Description	Default Value
2:0	prio0	The priority for queue 0	0x0
5:3	prio1	The priority for queue 1	0x1
8:6	prio2	The priority for queue 2	0x2
11:9	prio3	The priority for queue 3	0x3
14:12	prio4	The priority for queue 4	0x4
17:15	prio5	The priority for queue 5	0x5
20:18	prio6	The priority for queue 6	0x6
23:21	prio7	The priority for queue 7	0x7

34.12.14 Output Disable

Bitmask for disabling the egress queues on egress ports.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	37385 to 37393

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

Bits	Field Name	Description	Default Value
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

34.13 Shapers

34.13.1 Port Shaper Bucket Capacity Configuration

Token Bucket Capacity Configuration for Port Shaper

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	38280 to 38288

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	Index / Value 0-1 / 0x2ae4 2-8 / 0xe4c

34.13.2 Port Shaper Bucket Threshold Configuration

Token Bucket Threshold Configuration for Port Shaper

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	38289 to 38297

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.	Index / Value 0-1 / 0xe4c 2-8 / 0x4c4

34.13.3 Port Shaper Current Size

Number of tokens currently in the token bucket.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Egress Port
Address Space :	38298 to 38306

Field Description

Bits	Field Name	Description	Default Value
15:0	currentVal	Number of tokens currently in the token bucket for this Egress Port	Index Value 0-1 0xe4c 2-8 0x4c4

34.13.4 Port Shaper Enable

Bitmask to turn Port Shaper ON/OFF (1/0) for Egress Port

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38307

Field Description

Bits	Field Name	Description	Default Value
8:0	enable	Bitmask where the index is the Egress Port	0x0

34.13.5 Port Shaper Rate Configuration

Token Bucket rate Configuration for Port Shaper

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	38271 to 38279

Field Description

Bits	Field Name	Description	Default Value
0	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x0
12:1	tokens	The number of tokens added each tick	Index Value 0-1 0x16e 2-8 0x7a

Bits	Field Name	Description	Default Value
15:13	tick	Select one of the five available PTP ticks. The tick frequencies are configured globaly in the PTP Tick Configuration register.	0x0
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18
24	avb	If set the bucket will work in AVB-mode. That is, the bucket will be set to the threshold level when there are no packets queued.	0x0

34.13.6 Prio Shaper Bucket Capacity Configuration

Token Bucket Capacity Configuration for Prio Shaper

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Egress Port * 8 + Egress Prio
Address Space :	38051 to 38122

Field Description

Bits	Field Name	Description	Default Value	
			Index	Value
15:0	bucketCapacity	Capacity of the token bucket	0-15	0x2bf2
			16-71	0xea6

34.13.7 Prio Shaper Bucket Threshold Configuration

Token Bucket Threshold Configuration for Prio Shaper

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Egress Port * 8 + Egress Prio
Address Space :	38123 to 38194

Field Description

Bits	Field Name	Description	Default Value	
			Index	Value
15:0	threshold	Minimum number of tokens in bucket for the status to be set to accept.	0-15	0xea6
			16-71	0x4e2

34.13.8 Prio Shaper Current Size

Number of tokens currently in the token bucket.

Number of Entries :	72
Type of Operation :	Read Only
Addressing :	Egress Port * 8 + Egress Prio
Address Space :	38195 to 38266

Field Description

Bits	Field Name	Description	Default Value
15:0	currentVal	Number of tokens currently in the token bucket for this Egress Port * 8 + Egress Prio	Index / Value: 0-15 0xea6; 16-71 0x4e2

34.13.9 Prio Shaper Enable

Bitmask to turn Prio Shaper ON/OFF (1/0) for Egress Port * 8 + Egress Prio

Number of Entries :	1
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Address Space :	38267

Field Description

Bits	Field Name	Description	Default Value
71:0	enable	Bitmask where the index is the Egress Port * 8 + Egress Prio	0x0

34.13.10 Prio Shaper Rate Configuration

Token Bucket rate Configuration for Prio Shaper

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Egress Port * 8 + Egress Prio
Address Space :	37979 to 38050

Field Description

Bits	Field Name	Description	Default Value
0	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x0
12:1	tokens	The number of tokens added each tick	Index / Value: 0-15 0x177; 16-71 0x7d

Bits	Field Name	Description	Default Value
15:13	tick	Select one of the five available PTP ticks. The tick frequencies are configured globaly in the PTP Tick Configuration register.	0x0
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18
24	avb	If set the bucket will work in AVB-mode. That is, the bucket will be set to the threshold level when there are no packets queued.	0x0

34.13.11 Queue Shaper Bucket Capacity Configuration

Token Bucket Capacity Configuration for Queue Shaper

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Egress Port * 8 + Egress Queue
Address Space :	37759 to 37830

Field Description

Bits	Field Name	Description	Default Value
15:0	bucketCapacity	Capacity of the token bucket	Index: 0-15 Value: 0x2bf2; Index: 16-71 Value: 0xea6

34.13.12 Queue Shaper Bucket Threshold Configuration

Token Bucket Threshold Configuration for Queue Shaper

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Egress Port * 8 + Egress Queue
Address Space :	37831 to 37902

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.	Index: 0-15 Value: 0xea6; Index: 16-71 Value: 0x4e2

34.13.13 Queue Shaper Current Size

Number of tokens currently in the token bucket.

Number of Entries :	72
Type of Operation :	Read Only
Addressing :	Egress Port * 8 + Egress Queue
Address Space :	37903 to 37974

Field Description

Bits	Field Name	Description	Default Value	
			Index	Value
15:0	currentVal	Number of tokens currently in the token bucket for this Egress Port * 8 + Egress Queue	0-15	0xea6
			16-71	0x4e2

34.13.14 Queue Shaper Enable

Bitmask to turn Queue Shaper ON/OFF (1/0) for Egress Port * 8 + Egress Queue

Number of Entries :	1
Number of Addresses per Entry :	4
Type of Operation :	Read/Write
Address Space :	37975

Field Description

Bits	Field Name	Description	Default Value
71:0	enable	Bitmask where the index is the Egress Port * 8 + Egress Queue	0x0

34.13.15 Queue Shaper Rate Configuration

Token Bucket rate Configuration for Queue Shaper

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Egress Port * 8 + Egress Queue
Address Space :	37687 to 37758

Field Description

Bits	Field Name	Description	Default Value	
0	packetsNotBytes	If set the bucket will count packets, if cleared bytes	0x0	
			Index	Value
12:1	tokens	The number of tokens added each tick	0-15	0x177
			16-71	0x7d

Bits	Field Name	Description	Default Value
15:13	tick	Select one of the five available PTP ticks. The tick frequencies are configured globaly in the PTP Tick Configuration register.	0x0
23:16	ifgCorrection	Extra bytes per packet to correct for IFG in byte mode. Default is 4 byte FCS plus 20 byte IFG.	0x18
24	avb	If set the bucket will work in AVB-mode. That is, the bucket will be set to the threshold level when there are no packets queuedand the bucket will not receive any tokens while the output is gated by the egress transmission gate..	0x0

34.14 Shared Buffer Memory

34.14.1 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
10:0	cells	Number of free cells.	0x600

34.14.2 Egress Port Depth

Number of packets available in the buffer memory for each egress port.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Egress Port
Address Space :	37303 to 37311

Field Description

Bits	Field Name	Description	Default Value
10:0	packets	Number of packet currently queued.	0x0

34.14.3 Egress Queue Depth

Number of packets available in the buffer memory for each egress queue.

Number of Entries :	72
Type of Operation :	Read Only
Addressing :	Global queue number
Address Space :	37312 to 37383

Field Description

Bits	Field Name	Description	Default Value
10:0	packets	Number of packets currently queued.	0x0

34.14.4 Minimum Buffer Free

Minimum number of cells available in the buffer memory

Number of Entries :	1
Type of Operation :	Read Only
Address Space :	37384

Field Description

Bits	Field Name	Description	Default Value
10:0	cells	Number of cells.	0x600

34.14.5 Packet Buffer Status

Queue status of the packet buffer

Number of Entries :	1
Type of Operation :	Read Only
Address Space :	36749

Field Description

Bits	Field Name	Description	Default Value
8:0	empty	Empty flags for the egress ports	0x1ff

34.15 Statistics: ACL

34.15.1 Ingress Configurable ACL Match Counter

Number of packets hit in entries from Ingress configurable ACL lookup.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Index from result of Ingress configurable ACL.
Address Space :	34533 to 34564

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.16 Statistics: Debug

34.16.1 EPP PM Drop

Number of drops due to FIFO overflows in EPP PM.
In Figure 29.1, **epmOverflow** with process sequence **22** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38391

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.16.2 IPP PM Drop

Number of drops due to FIFO overflows in IPP PM.
In Figure 29.1, **ipmOverflow** with process sequence **12** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1424

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.16.3 PS Error Counter

Number of errors occured in the PS-converter.
In Figure 29.1, **psError** with process sequence **25** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38898 to 38906

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

34.16.4 SP Overflow Drop

Number of packets dropped due to: FIFO overflow in the SP-converter.
In Figure 29.1, **spOverflow** with process sequence **5** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Ingress port
Address Space :	1376 to 1384

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets on this ingress port.	0x0

34.17 Statistics: EPP Egress Port Drop

34.17.1 Egress Port Disabled Drop

Number of packets dropped due to egress port disabled.
In Figure 29.1, **epppDrop** with process sequence **19** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38373 to 38381

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.17.2 Egress Port Filtering Drop

Number of packets dropped due to egress port filtering.
In Figure 29.1, **epppDrop** with process sequence **19** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38382 to 38390

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.17.3 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 29.1, **epppDrop** with process sequence **19** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38364 to 38372

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.18 Statistics: Enqueued and Dequeued

34.18.1 Dequeued Bytes

Number of bytes received after egress packet process.
In Figure 29.1, **dequeued** with process sequence **20** represents the internal location of this counter.

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Slice Global Egress Queue
Address Space :	38464 to 38535

Field Description

Bits	Field Name	Description	Default Value
23:0	bytes	Number of bytes.	0x0

34.18.2 Dequeued Packets

Number of packets received after egress packet process.
In Figure 29.1, **dequeued** with process sequence **20** represents the internal location of this counter.

Number of Entries :	72
Type of Operation :	Read/Write
Addressing :	Slice Global Egress Queue
Address Space :	38392 to 38463

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19 Statistics: FRER

34.19.1 Individual Recovery Discarded Counter

Number of packets dropped by the recovery function due to a duplicated sequence number.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	36143 to 36206

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.2 Individual Recovery Lost Counter

Number of packets with sequence numbers moved out of the sequence history window under the vector recovery algorithm.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	36335 to 36398

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.3 Individual Recovery Out Of Order Counter

Number of packets accepted by the recovery function but with sequence number not equals previous sequence number plus one.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	36207 to 36270

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.4 Individual Recovery Passed Counter

Number of packets accepted by the recovery function.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	36079 to 36142

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.5 Individual Recovery Rogue Counter

Number of packets dropped by the vector recovery algorithm due to its sequnece number out of the sequence history window.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	36271 to 36334

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.6 Individual Recovery Tagless Counter

Number of packets received without sequence number.

Number of Entries :	64
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	36399 to 36462

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.7 Sequence Recovery Discarded Counter

Number of packets dropped by the recovery function due to a duplicated sequence number.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Handle
Address Space :	36495 to 36526

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.8 Sequence Recovery Lost Counter

Number of packets with sequence numbers moved out of the sequence history window under the vector recovery algorithm.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Handle
Address Space :	36591 to 36622

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.9 Sequence Recovery Out Of Order Counter

Number of packets accepted by the recovery function but with sequence number not equals previous sequence number plus one.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Handle
Address Space :	36527 to 36558

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.10 Sequence Recovery Passed Counter

Number of packets accepted by the recovery function.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Handle
Address Space :	36463 to 36494

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.11 Sequence Recovery Rogue Counter

Number of packets dropped by the vector recovery algorithm due to its sequnece number out of the sequence history window.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Handle
Address Space :	36559 to 36590

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.19.12 Sequence Recovery Tagless Counter

Number of packets received without sequence number.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	Stream Handle
Address Space :	36623 to 36654

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.20 Statistics: IPP Egress Port Drop

34.20.1 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 29.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port (not aggregated)
Address Space :	34866 to 34874

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.20.2 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 29.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port (not aggregated)
Address Space :	34884 to 34892

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.20.3 L2 Action Table Per Port Drop

Number of packets dropped due to L2 Action Table per egress port drop configured in **L2 Action Table Drop**.
In Figure 29.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port (not aggregated)
Address Space :	34893 to 34901

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.20.4 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 29.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port (not aggregated)
Address Space :	34875 to 34883

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.20.5 Queue Off Drop

Number of packets dropped due to the queue being turned off.
In Figure 29.1, **preEppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port (not aggregated)
Address Space :	34857 to 34865

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21 Statistics: IPP Ingress Port Drop

34.21.1 AH Decoder Drop

Number of packets dropped due to setting in register **AH Header Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1450

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.2 ARP Decoder Drop

Number of packets dropped due to setting in register **ARP Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1443

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.3 Attack Prevention Drop

Number of packets dropped due to matching TCP/UDP flag rule.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1442

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.4 BOOTP and DHCP Decoder Drop

Number of packets dropped due to setting in register **BOOTP and DHCP Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1453

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.5 CAPWAP Decoder Drop

Number of packets dropped due to setting in register **CAPWAP Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1454

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.6 DNS Decoder Drop

Number of packets dropped due to setting in register **DNS Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1452

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.7 ESP Decoder Drop

Number of packets dropped due to setting in register **ESP Header Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1451

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.8 Empty Mask Drop

Number of packets dropped due to an empty destination port mask.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1427

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.9 GRE Decoder Drop

Number of packets dropped due to setting in register **GRE Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1455

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.10 IEEE 802.1X and EAPOL Decoder Drop

Number of packets dropped due to setting in register **IEEE 802.1X and EAPOL Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1447

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.11 IP Checksum Drop

Number of packets dropped due to incorrect IP checksum.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1439

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.12 Ingress Configurable ACL Drop

Number of packets dropped due to matching an Ingress Configurable ACL with drop.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1441

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.13 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1433

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.14 Ingress Rate Control Drop

Number of packets dropped due to ingress rate control.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1432

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.15 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1430

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.16 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1429

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.17 Ingress Spanning Tree Drop: Listen

Number of packets dropped due to that a port's ingress spanning tree protocol state was **Listening**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1428

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.18 L2 Action Table Drop

Number of packets dropped due to the **L2 Action Table** says drop all instances.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1457

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.19 L2 Action Table Port Move Drop

Number of packets dropped due to the **L2 Action Table** says drop due to port move packet.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1458

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.20 L2 Action Table Special Packet Type Drop

Number of packets dropped due to the **Allow Special Frame Check For L2 Action Table** dit not allow a certain packet/frame type.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1456

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.21 L2 Destination Table SA Lookup Drop

Number of packets dropped due to the table **L2 Destination Table** field

ieldL2 Destination TablepktDropSa says drop.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1459

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.22 L2 IEEE 1588 Decoder Drop

Number of packets dropped due to setting in register **IEEE 1588 L4 Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1445

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.23 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 or due to not allowing destination port being the same as the source port.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1431

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.24 L2 Reserved Multicast Address Drop

Number of packets dropped due to the L2 Reserved Multicast Addresses on counter 0
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1440

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.25 L4 IEEE 1588 Decoder Drop

Number of packets dropped due to setting in register **IEEE 1588 L4 Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1446

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.26 LACP Decoder Drop

Number of packets dropped due to setting in register **LACP Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1449

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.27 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1438

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.28 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1437

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.29 RARP Decoder Drop

Number of packets dropped due to setting in register **RARP Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1444

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.30 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1434

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.31 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1435

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.32 SCTP Decoder Drop

Number of packets dropped due to setting in register **SCTP Packet Decoder Options**.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1448

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.33 Source Port Default ACL Action Drop

Number of packets dropped due to the table **Source Port Default ACL Action** says drop.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1460

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.34 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 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1426

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.21.35 VLAN Member Drop

Number of packets dropped due to the packets source port notbeing part of the packets VLAN membership.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1436

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.22 Statistics: IPP Ingress Port Receive

34.22.1 Ingress MAC SA Change Counter

Number of broadcast packets received with MAC SA differed from the previous broadcast packet.
In Figure 29.1, **ippReception** with process sequence **11** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	34911 to 34919

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.22.2 Ingress Received and Dropped Counter

Number of packets received without errors on an ingress port but dropped during the ingress packet processing.
In Figure 29.1, **ipppDrop** with process sequence **11** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Ingress port
Address Space :	34902 to 34910

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.23 Statistics: Misc

34.23.1 Buffer Overflow Drop

Counter for the number of packets dropped due to the shared buffer memory being full.
In Figure 29.1, **bmOverflow** with process sequence **16** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	36750

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.23.2 Drain Port Drop

Number of packets dropped due to the port is drained.
In Figure 29.1, **drain** with process sequence **21** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress port
Address Space :	38355 to 38363

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.23.3 Egress Resource Manager Drop

Number of packets dropped by the egress resource manager.
In Figure 29.1, **erm** with process sequence **15** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read/Write
Addressing :	Egress Port
Address Space :	36740 to 36748

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.23.4 FRER Drop

Number of packets dropped due to FRER individual recovery.
In Figure 29.1, **frer** with process sequence **14** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	36687

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.23.5 Flow Classification And Metering Drop

Number of packets dropped due to flow classification and metering.
In Figure 29.1, **mmp** with process sequence **14** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	35308

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.23.6 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 29.1, **eopDrop** with process sequence **14** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1425

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.23.7 Ingress Resource Manager Drop

Counter for the number of packets dropped due to exeeding thresholds set up in the ingress resource manager.
In Figure 29.1, **irm** with process sequence **16** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	36751

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets.	0x0

34.23.8 Latent Error Detection Status

An error condition is reported to this register when latent error detection raise an error.

Number of Entries :	32
Type of Operation :	Read/Write
Addressing :	FRER ID
Address Space :	36655 to 36686

Field Description

Bits	Field Name	Description	Default Value
0	error	If the value is 1, the offset of FRER packet drops between two test periods is larger than expected.	0x0

34.23.9 MAC RX Broken Packets

Number of broken packets dropped (packets with last=1 and valid_bytes=0).
In Figure 29.1, **macBrokenPkt** with process sequence **3** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Ingress Port
Address Space :	57 to 65

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.23.10 MAC RX Long Packet Drop

Number of packets dropped due to length above **MAC RX Maximum Packet Length**.
In Figure 29.1, **macRxMax** with process sequence **4** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Ingress Port
Address Space :	75 to 83

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.23.11 MAC RX Short Packet Drop

Number of packets dropped due to length below 60 bytes.
In Figure 29.1, **macRxMin** with process sequence **4** represents the internal location of this counter.

Number of Entries :	9
Type of Operation :	Read Only
Addressing :	Ingress Port
Address Space :	66 to 74

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.23.12 Re-queue Overflow Drop

Counter for the number of packets dropped due to a FIFO overflow in re-queue.
In Figure 29.1, **rqOverflow** with process sequence **24** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	36761

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of dropped packets	0x0

34.24 Statistics: PSFP

34.24.1 PSFP Matching Frame Counter

Number of packets that hit stream filter entries

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35565 to 35580

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.24.2 PSFP Not Passing Frame Counter

Number of packets dropped by the stream gate

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35629 to 35644

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.24.3 PSFP Not Passing SDU Counter

Number of packets dropped by the max SDU filter

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35597 to 35612

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.24.4 PSFP Passing Frame Counter

Number of packets passing the stream gate

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35613 to 35628

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.24.5 PSFP Passing SDU Counter

Number of packets passing the max SDU filter

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35581 to 35596

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.24.6 PSFP Red Frames Counter

Number of packets dropped by the flow meter

Number of Entries :	16
Type of Operation :	Read/Write
Addressing :	Stream Filter ID
Address Space :	35645 to 35660

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.25 Statistics: Packet Datapath

34.25.1 EPP Packet Head Counter

Number of packet first cells through the Egress Packet Process module.
In Figure 29.1, **eppTxPkt** with process sequence **24** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38536

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

34.25.2 EPP Packet Tail Counter

Number of packet last cells through the Egress Packet Process module.
In Figure 29.1, **eppTxPkt** with process sequence **24** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38537

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

34.25.3 IPP Packet Head Counter

Number of packet first cells through the Ingress Packet Process module.
In Figure 29.1, **ippTxPkt** with process sequence **13** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1461

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

34.25.4 IPP Packet Tail Counter

Number of packet last cells through the Ingress Packet Process module.
In Figure 29.1, **ippTxPkt** with process sequence **13** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	1462

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

34.25.5 PB Packet Head Counter

Number of packet first cells through the Shared Buffer Memory module.
In Figure 29.1, **pbTxPkt** with process sequence **18** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38352

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

34.25.6 PB Packet Tail Counter

Number of packet last cells through the Shared Buffer Memory module.
In Figure 29.1, **pbTxPkt** with process sequence **18** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38353

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

34.25.7 PS Packet Head Counter

Number of packet first cells through the Parallel to Serial module.
In Figure 29.1, **psTxPkt** with process sequence **25** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38896

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet headers.	0x0

34.25.8 PS Packet Tail Counter

Number of packet last cells through the Parallel to Serial module.
In Figure 29.1, **psTxPkt** with process sequence **25** represents the internal location of this counter.

Number of Entries :	1
Type of Operation :	Read/Write
Address Space :	38897

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packet tails.	0x0

34.26 Statistics: SMON

34.26.1 SMON Set 0 Byte Counter

Number of bytes counted in SMON Set 0.
In Figure 29.1, **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 :	34517 to 34524

Field Description

Bits	Field Name	Description	Default Value
23:0	bytes	Number of bytes.	0x0

34.26.2 SMON Set 0 Packet Counter

Number of packets counted in SMON Set 0.
In Figure 29.1, **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 :	34501 to 34508

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0

34.26.3 SMON Set 1 Byte Counter

Number of bytes counted in SMON Set 1.
In Figure 29.1, **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 :	34525 to 34532

Field Description

Bits	Field Name	Description	Default Value
23:0	bytes	Number of bytes.	0x0

34.26.4 SMON Set 1 Packet Counter

Number of packets counted in SMON Set 1.
In Figure 29.1, **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 :	34509 to 34516

Field Description

Bits	Field Name	Description	Default Value
23:0	packets	Number of packets.	0x0