High Scale Ethernet MDA Capabilities

In This Section

This section provides information to configure HSMDA QoS policies using the command line interface.

Topics in this section include:

HSMDA QoS Model

This section describes QoS capabilities of the High Scale Ethernet MDA (HSMDA). The HSMDA extends subscriber and service density of the first and second generation IOMs by adding an MDA level of ingress and egress queues, shapers and schedulers.

The HSMDA replaces the ingress and egress service queuing function performed by the ingress and egress forwarding plane on the IOM, providing up to 160K service or subscriber-based queues in each direction. The queues are configured in groups of eight. The addition of the ingress packet classification and queues allows QoS classification and service-level queuing to be performed on the MDA instead of the directly-attached IOM forwarding plane. The egress side of the HSMDA relies on the existing egress forwarding plane to map egress packets to the HSMDA egress queues.

The HSMDA moves service and subscriber-level ingress and egress QoS functions off the IOM forwarding plane to the MDA. The HSMDA QoS model provides the following features:

- Expanded queue scale for ingress and egress
- Hardware implemented provider style port based scheduling
- Elimination of ingress dual pass queuing at the ingress hardware
- Egress intermediate destination (such as DSLAM) shaping using secondary shapers
- Expanded counters at ingress and egress
- CIR Bypass for color aware policing
- Ingress queue policing mode
- RED queue congestion control
- Egress dot1p remarking per packet based on egress queue scheduling rate
- Per queue packet byte offset for queue stats, queue PIR, queue CIR and queue group PIR accounting

Queue Scaling

The HSMDA supports 160,000 queues in the ingress and egress directions. Each queue supports two leaky buckets that perform a PIR shaping function and a CIR marking function based on scheduled rate out of the queue. Each queue also supports two RED slopes that may be used for managing queue congestion.

The queues are grouped into sets of eight queues each. 3840 queues are reserved for the system. A set of eight queues is called a queue group and is internally identified by a queue-group-id. Each queue within a group is numbered from 1 to 8 represented internally as the queue-id. The queue-id has an implicit scheduling class association based on the number of the queue (for example, queue 3 is a member of scheduling class 3). Queue groups are dynamically mapped to egress ports on an as needed basis.

Individual queues are also mapped to 1 of 64 secondary shapers, each representing an egress intermediate destination (such as a DSLAM). One is pre-allocated per egress port, so only 54 are user-definable in 10x1G HSMDAs and 63 are user-definable in 1x10G HSMDAs.

Port-Based Scheduling

Forwarding for each egress port on the HSMDA is managed by a port-based scheduler. Each port-based scheduler maintains a maximum of eight strict forwarding levels (Figure 28). Strict level 8 is the highest priority while strict level 1 is the lowest. There are also eight scheduling classes that contain each of the queues assigned to the port scheduler. A queues membership in a scheduler class is controlled by the queue's identifier. For example, all queues with a queue-id equal to 1 are in scheduler class 1 while all queues with queue-id equal to 2 are in scheduler class 2. By default, each scheduler class is directly mapped to its corresponding strict scheduling level.

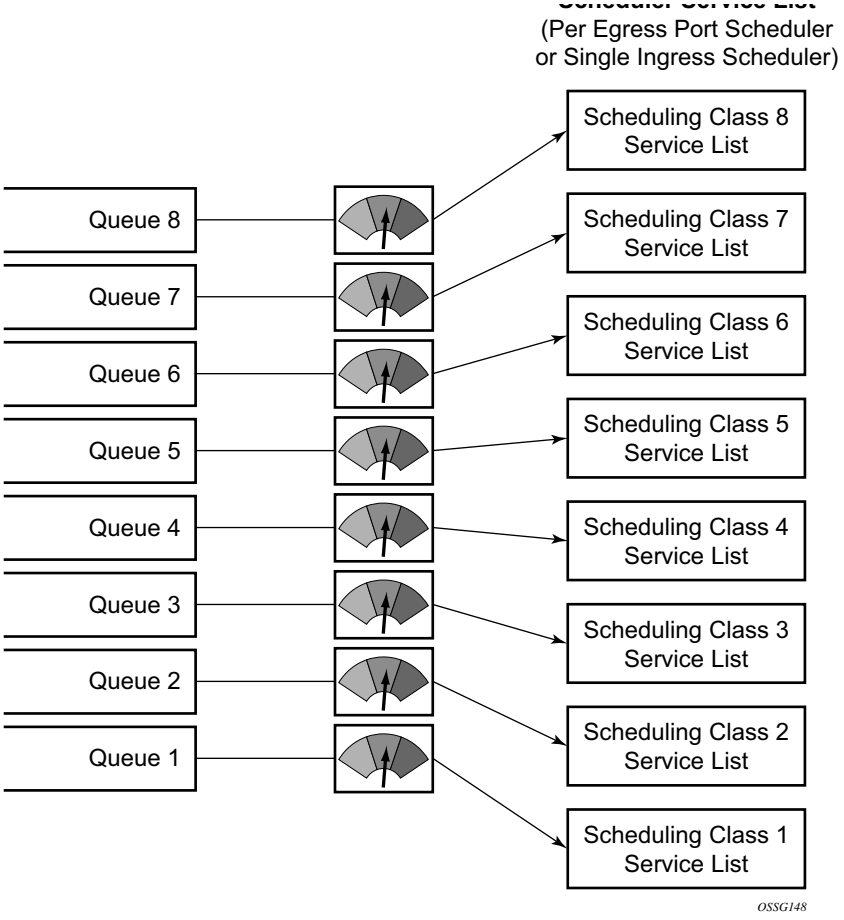

Figure 28: HSMDA Queue Mapping to Scheduler Class Service Lists

To allow for weighted servicing of selected scheduling classes, the port scheduler allows for two weighted groups to be optionally created (weighted-group-1 and weighted-group-2) and each may be populated with up to three consecutive scheduling classes (Figure 29). The group itself maps to the highest inherent strict scheduling level of its member scheduling classes. Each scheduling class in a scheduling group are individually weighted which allows for all queues represented by the class to be service according to the ratio of weights based on the active classes in the group (Figure 30).

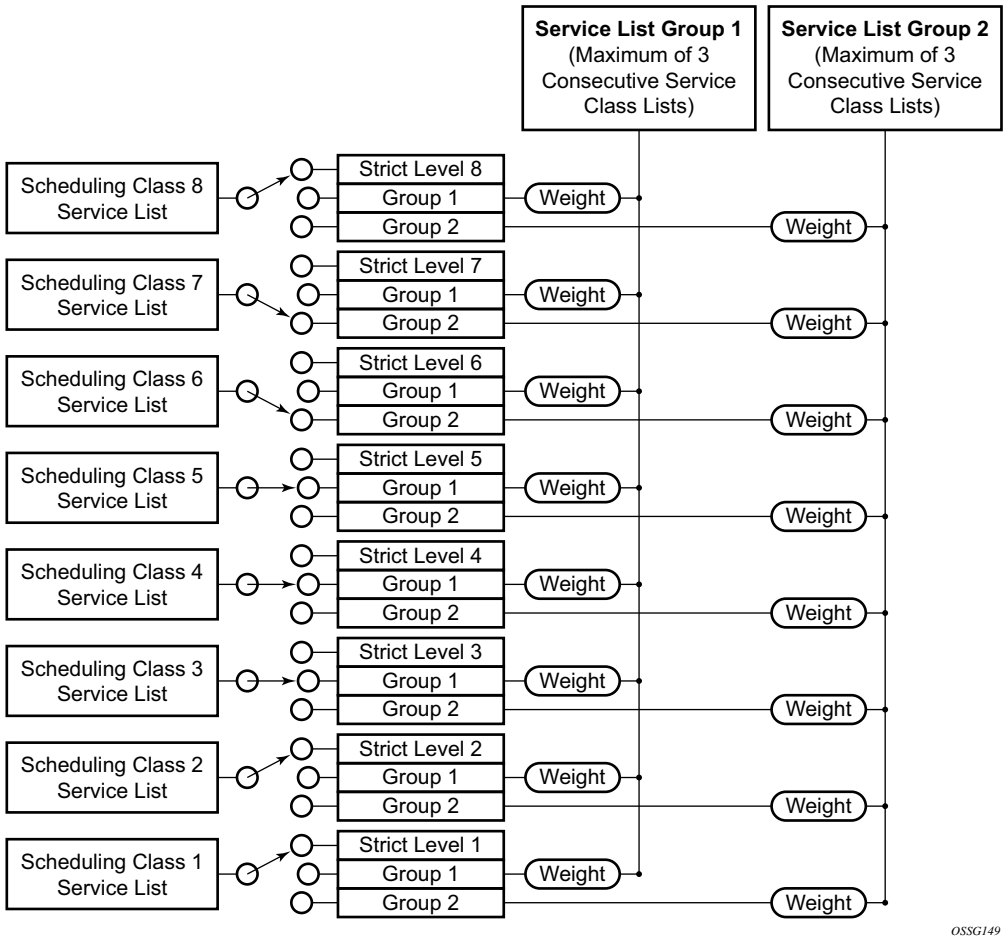

Figure 29: Scheduler Class Mapping to Strict Level or Weighted Group Example

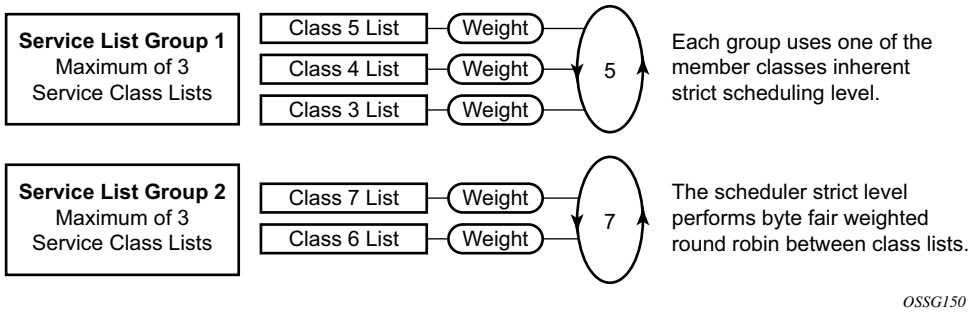

Figure 30: Scheduler Weighted Group Configuration Example

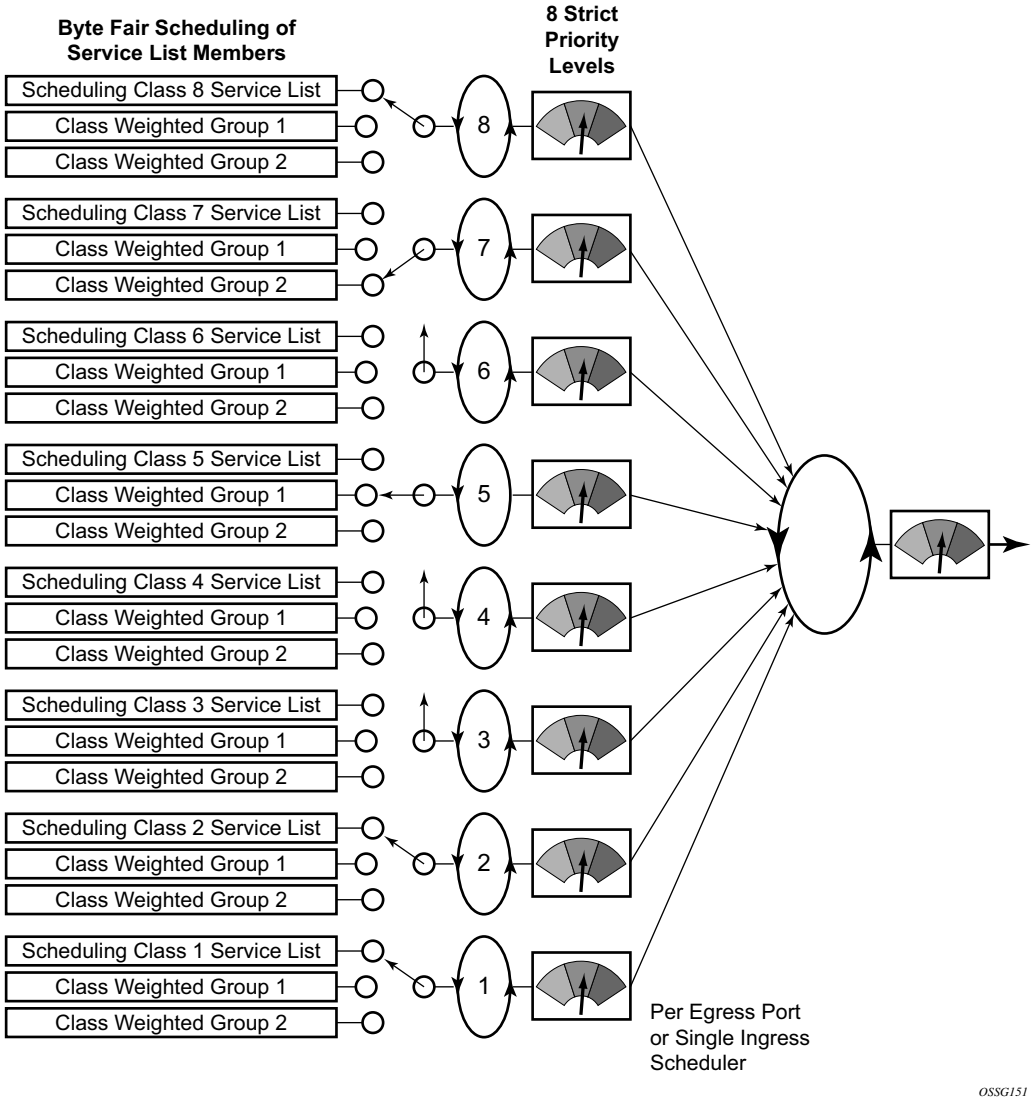

Figure 31: Scheduler Class and Weighed Group Scheduling Priority Mapping Example

The port based scheduler supports a port based shaper used for creating a sub-rate condition on the egress port. Each individual strict scheduling level may also be configured with a shaping rate used to limit the amount of bandwidth allowed for that strict level. In all, shaping PIRs can be defined at the following points in the queuing and scheduling architecture:

- Per port shaper
- Per strict level shaper
- Intermediate destination shaper (egress)
- Per queue-group shaper
- Per queue shaper

Dual Pass Queuing

In the standard queuing model, the ingress hardware performs both per service ingress SLA enforcement as well as per switch fabric destination based virtual output queuing. Due to the requirement that each queue on ingress be mapped to a single switch fabric destination, when a SAP for a service type that forwards to multiple switch fabric destinations (such as VPLS, IES and VPRN services), a single ingress service queue is created as multiple hardware virtual output queues in the hardware. In environments where the SAP or subscriber density requires more service queues than can be created with the available ingress hardware queues, two passes through the hardware are performed. This dual pass mechanism allows a single hardware queue to represent a service queue on the first pass through the hardware while the second pass allows the hardware to use shared virtual output queues into the switch fabric.

HSMDA removes the need to perform two passes through ingress since the service or subscriber-based queuing is being performed on the MDA. This frees the ingress hardware to perform the virtual output queuing function using shared queues mapped to switch fabric destinations.

Egress Intermediate Destination Secondary Shapers

The HSMDA supports placement at the apex of an aggregation network servicing multiple DSLAMs or other subscriber last-mile aggregation devices such as DSLAMs or gigabit passive optical network optical network terminals (GPON ONT). When operating in this fashion, multiple subscriber aggregators will be reached over an HSMDA 10GE port. The egress port scheduler combines all subscriber queues of the same scheduling class and services the queues in a byte fair round robin fashion. An effect of this behavior results in more packets being forwarded into the aggregation network towards a DSLAM than the DSLAM can accept. If the HSMDA egress port is congested, the egress bandwidth represented by the downstream discarded packets to the DSLAM may have been allocated to packets destined to other DSLAMs. The HSMDA supports egress secondary shapers that are used to provide a control mechanism to prevent downstream overruns without affecting the class based scheduling behavior on the port. When used in this manner, the secondary shaper is called an intermediate destination shaper. Egress secondary shapers are configured on a per port basis.

In other systems, the downstream destination is represented as a scheduler in a tiered hierarchical scheduler. This means that bandwidth is allocated on a per DSLAM basis without regard to the class of packets being forwarded to that DSLAM. In this model, a set of subscribers on one DSLAM receiving packets for a premium service are treated equally to subscribers on other DLSAMs receiving packets associated with best-effort type services.

The intermediate destination secondary shaper solves the downstream overrun issue without sacrificing class preference at the HSMDA egress port scheduler. All subscribers destined to the same DSLAM have their queue groups mapped to the same egress secondary shaper. As the scheduler services the queues within the groups according to their scheduler class, the intermediate

destination shaper is updated. Once the shapers rate threshold is exceeded, scheduling for all queues associated with the shaper is stopped. Once the dynamic rate drops below the threshold, the queues are allowed to be placed back on to the scheduler service lists. By removing the queues from their scheduling context for a downstream congested DSLAM, the port scheduler is allowed to fill the egress port with packets destined to other DSLAMs without sacrificing class behavior on the port.

64 egress secondary shapers are supported which may be dynamically associated with an egress physical port. Each HSMDA queue must be associated with a secondary shaper. By default, a secondary shaper is created per physical port and all queues within a queue group on the port are associated with the default secondary shaper. All default secondary shapers will operate at the maximum rate and have no impact on queue scheduling. The hardware supports mapping individual queues within a queue group to separate secondary shapers. Based on the intended use case for secondary shapers, the provisioning model does not allow queues within the same queue group to map to separate secondary shapers. Queues are mapped to a secondary shaper on a queue group basis.

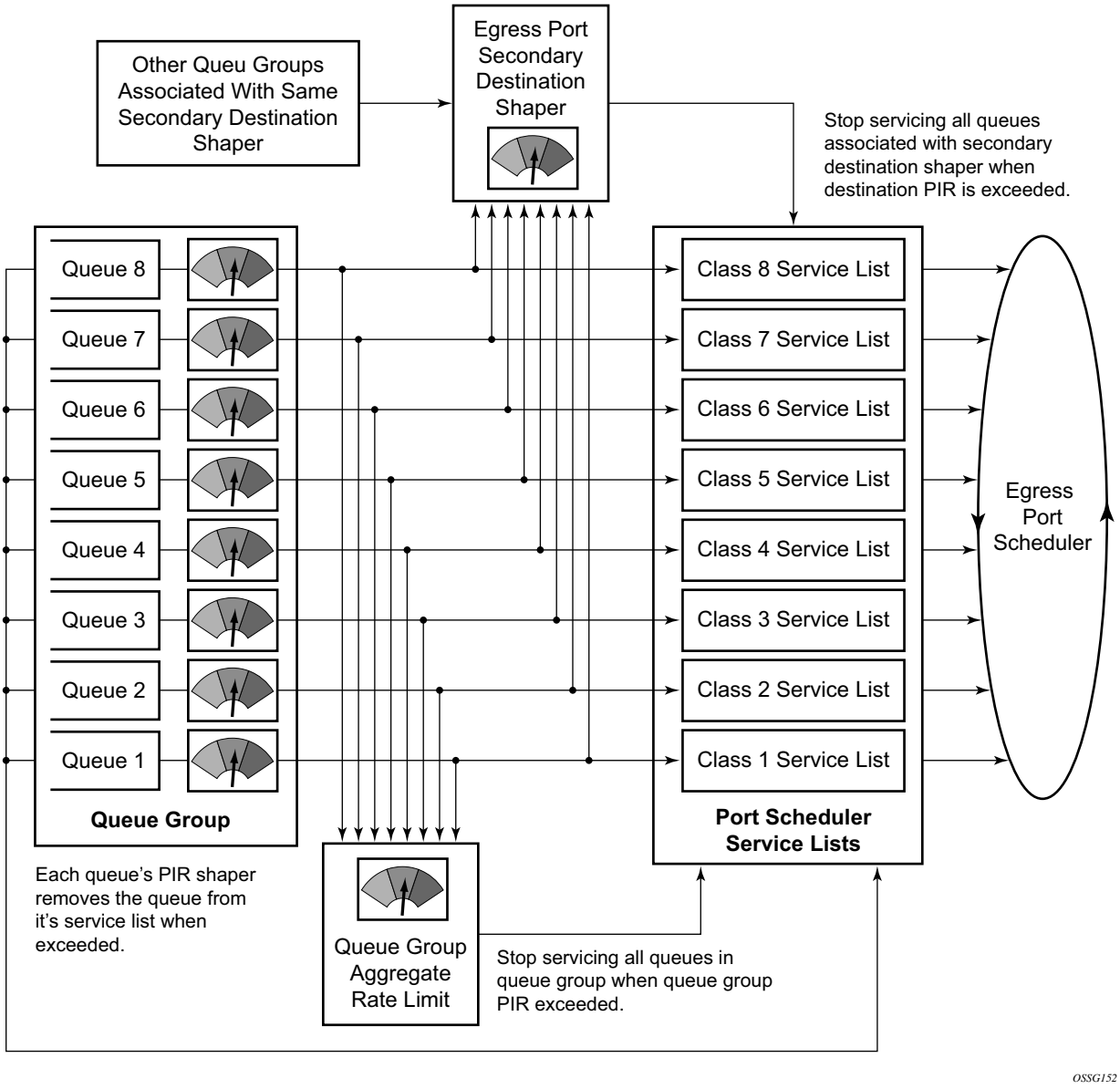

Figure 32: HSMDA Egress Queue Group and Secondary Destination Shaper Behavior

Packet and Octet Counting

Each queue group supports a set of 16 counters. Each set of counters is identified by a counter-id and contains individual counters for:

- Discarded out-of-profile (low priority) packet
- Discarded out-of-profile (low priority) octet
- Discarded in-profile (high priority) packet
- Discarded in-profile (high priority) octet
- Forwarded out-of-profile packet
- Forwarded out-of-profile octet
- Forwarded in-profile packet
- Forwarded in-profile octet

The discard counters are incremented during “en-queuing” discarded events and the forward counters are incremented during scheduled “de-queuing” events.

For standard ingress queues, the offered stats-per-queue are counted by the ingress forwarding plane. For standard egress queues, the offered stats are derived by adding the hardware per-queue discard and forward stats. This means packets waiting to be scheduled in a standard egress queue will not be counted until it is forwarded. The HSMDA queue-offered stats operate the same as the standard egress queues where the discard and forwarded stats are combined to derive the offered stats for an HSMDA queue.

The decision on the counter to use is made per packet by HSMDA ingress hardware or by the egress forwarding plane hardware. The default behavior is to use the counter-id that corresponds to the queue-id to which the packet is mapped. This means the packets destined to queue 2 will be accounted for by counter set 2 within the same queue group. This sets aside the first eight counter sets as the default counters for the queue group. The remaining eight are available as counter override decisions.

A counter override can be performed within the ingress QoS classification rules wherever an HSMDA is installed.

The eight counter sets used as exception counters are identified as counter 1 through counter 8. While the discard and forwarding statistics can be overridden based on exception criteria, the offered statistics are maintained per queue. This means that the offered statistics for a queue includes all packets offered to the queue, but the discard and forwarding statistics only reflect packets handled by the queue that have not been associated with exception counters. It is possible to estimate the number of packets not represented by the queue statistics by subtracting the discard and forwarding statistics from the queue-offered statistics. The resulting number may be off slightly if packets are still in the queue when the statistics where collected, but this error is minimized when calculated over an appropriate amount of time and can be completely eliminated

if the queue is allowed to drain prior to performing the calculation. When the queue statistics and the exception statistics are considered as a whole, all packets handled by the queue group are accurately represented by the counters.

Since the HSMDA only updates a single counter set per packet, overriding the counter for a packet causes that packet not to be represented within the default counter-id for the queue. If the queue counter-id is being used to determine CIR or PIR accuracy or basic throughput for a queue, any packets forwarded through the queue using a counter override is considered.

Above CIR Discard with PIR Bypass

HSMDA Ingress Queue Policing Mode

Since HSMDA queue-based CIR and PIR leaky bucket behavior is driven by scheduling events from the queue, a true policing function where the ingress offered rate determines the color of the packet (green, yellow or red) is not available, by default. While enabling a PIR-based shaping rate for the queue will perform a similar function, the shaping function is simply stopping the queue from scheduling. When the queue is stopped, packets are allowed to be buffered within the queue up to the RED slope or MBS-configured limits. The jitter for a shaping queue is based on how full the queue can get and how fast the queue is scheduled (influenced by the queue PIR, queue group PIR and ingress secondary shapers). While the MBS for the queue may be configured to minimize jitter by preventing an excessive amount of data to accumulate in the queue, a policing mode can be enabled on the queue that uses the CIR leaky bucket to dynamically discard packet above the CIR threshold.

The HSMDA ingress queue policing mode behaves normally while the CIR leaky bucket is below its configured threshold. If the CIR fill depth rises above the threshold, the packet is discarded without updating any of the ingress schedulers PIR leaky buckets. The result is that while the queue is operating within its CIR, scheduled packets will be forwarded to the ingress forwarding plane for further processing and each packet will update all applicable PIRs. But when the queue scheduled rate rises above the CIR, scheduled packets are discarded. By discarding out-of-profile packets, the policing rate is enforced without unnecessary jitter based on queue congestion.

In order for ingress policing mode to function properly, the following configuration guidelines should be observed:

- The queue should be scheduled at the highest appropriate strict scheduling priority. If the queue is not scheduled at a high enough priority, scheduling from the queue may momentarily stall. The scope of this issue is limited since the ingress port bandwidth is less than the available scheduling bandwidth to the ingress forwarding plane.
- Ensure that the policing queue is not stalled by the queue groups configured aggregate rate limit. If the queue is not the highest scheduling priority, the sum of the allowed scheduling for the queues with higher scheduling priority may cause the queue groups PIR to be exceeded and thus scheduling for the policing queue will stall. If the queue is at a high enough priority, lower priority queues will only be allowed to consume the group PIR while the higher priority queues are inactive (empty). In the event that lower priority queues cause the PIR to suspend scheduling for the queue group, higher priority queues will have first access to ingress scheduling once the group PIR decrements below the threshold.

- Care should also be taken with ingress secondary shapers. If the queue is assigned to an ingress secondary shaper, the queue may be stalled when the aggregate rate of the queues associated with the shaper exceeds the shapers PIR.

Packets sent to an HSMDA ingress queue configured for policing cannot have the ignore-CIR flag set. Color-aware profiling and ingress queue policing should not be mixed. In the event that a packet is classified as in-profile or out-of-profile while ingress queue policing is configured, the ignore-CIR bit will automatically be reset to zero (the queue CIR will be updated by the packet).

Two implementation options exist to account for out-of-profile scheduling discards:

1. When a packet is discarded due to out-of-profile scheduling, the out-of-profile packet and octet forward counters within the counter ID associated with the packet are updated. If the packet is forwarded, the in-profile packet and octet forward counters within counter ID' are incremented. Software must add the out-of-profile forwarded counter to the low priority discard counter to determine the total discards based on out-of-profile for the queue.
2. Alternatively, the discard event may be hard-coded to increment the discard counters directly. When discarding, the congestion-priority bit is used to determine whether the high or low discard counter is incremented.

For packets discarded at enqueuing time, the high priority or low priority packet and octet discard counters are updated. The decision to use the high or low priority discard counter is driven by the congestion-priority' bit associated with the packet. This bit is set based on both color aware profiling and ingress priority of the packet. If the packet is explicitly classified as in-profile, the bit is set to high. If the packet is explicitly classified as out-of-profile, the bit is set to low. If the profile of the packet is undetermined (not explicitly in-profile or out-of-profile), the bit is set to high or low based on the classified ingress priority of the packet.

HSMDA Buffer Utilization Controls

The HSMDA has 1 million ingress and 1 million egress 168-byte buffers available for packet queuing purposes. The average of approximately six buffers per queue when all 163,480 are active. Certain queues need more than 6 buffers while other queues require very shallow buffering based on the type of traffic the queue is servicing and the scheduling priority of the queue. To facilitate management of the available buffer space, the HSMDA supports a hierarchical buffer pool scheme and a per queue set of RED slopes. The buffer pools allow proper sharing of the buffer space while the slopes within each queue set limits on how many buffers each queue may consume.

HSMDA Buffer Pools

Two types of queues are created on the HSMDA; provisioned service or subscriber queues and system created queues. System queues are transparent to the user and perform functions like discard bypass. Since system queues are critical to the operation of the system, normal service or subscriber queue activity will not cause buffer starvation on the system queues. Buffer utilization is separated based on the scheduling class, ensuring that activity on one set of class queues does not impact buffer availability for other class queues.

The pooled buffer management capabilities in the HSMDA include:

- Identifying which queue groups consume provisioning buffers and which consume system reserved buffers.
- Setting the total provisioning buffers available per port for each scheduling class.
- Setting the total system reserved buffers available per port for each scheduling class.
- 32 aggregation buffer pools used for managing buffers available per class and per type (provisioned/system).

Identifying Queue Groups as Provisioned or System

All queues are contained in a set of eight queues called a queue group. Two sets of 20,480 queue groups exist on the HSMDA, one set for ingress and the other for egress. The queue groups are defined as either provisioned (service or subscriber) or reserved for system use. The HSMDA uses a two 20,480 bit-wide tables to allow the system to define each queue group as either provisioned or system reserved separately for ingress and egress. The queue group id mapping table represented in Figure 33 places queue groups in pool group P (provisioned) or pool group S (system). For ingress and egress, the first 20,000 table entries (0..19,999) is set to group P and the remaining 480 (20,000..20,479) is set to group S.

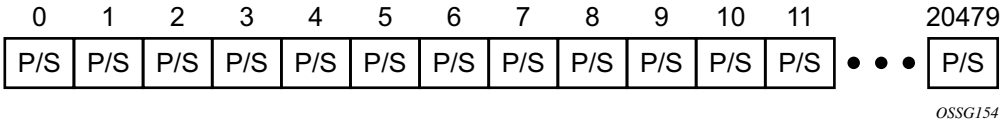

Figure 33: Queue Group ID Mapping Table

Provisioned and System Port Class Pools

The HSMDA uses a second table called the port class buffer pools table (Figure 34) that represents a set of 180 buffer pools. Pools 0 through 79 are used by queues within group P and 80 through 179 are used by queues within group S. Each set of eighty pools is divided into 10 subsets of 8 pools each. Each subset is dedicated to a physical port on the HSMDA. For group P, pools 0 through 7 are for port 1, 8 through 15 are for port 2 and pools 72 through 79 are for port 10. A queue is mapped to a pool based on the queue-group-id mapping to P or S and then the port the queue is associated with is used to pick the subset. Within the subset, the internal queue-id is used as an offset to pick an actual pool. Queue-id 0 (provisioned as 1) on port 3 in group P is mapped to pool 16. Each pool in the table also has two aggregate pool pointers used to provide further control on buffer allocation. Agg-Pool-Ptr-1 and Agg-Pool-Ptr-2 arbitrarily tie the port class pool to two aggregate pools from a third table of 32 buffer pools.

Figure 34: Port Class Buffer Pools Table

Aggregate Pools for Type and Class Separation

The third table is called the aggregate control buffer pools table (Figure 35). The table consists of 32 buffer pools that may be used arbitrarily by the port class pools. While the association from port class pool to aggregate control pool is arbitrary based on the 5-bit pointers, it is expected that the control pools will be divided into two groups of 16 pools, each group having two sub-groups of eight pools each (32 pools total). The first aggregate control group (pools 0 through 15) will be for provisioned buffer management and will be used by group P port class pools. The second aggregate control group (pools 16 through 31) will be for system level buffer management and used by group S port class pools.

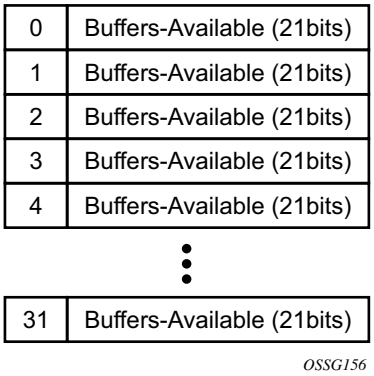

Figure 35: Aggregate Control Buffer Pools Table

Use of Aggregate Control Buffer Pools

The aggregate control buffer pools are separated into two sets. The first 16 pools (0 through 15) are used by the provisioned group (group P) port class pools. The second 16 pools (16 through 31) are used by the system group (group S) port class pools.

The first 8 (0 through 7) are used as class based pools. Pool 0 is used by scheduling class 1 (0 internally) and pool 7 is used by scheduling class 8. Agg-Pool-Ptr-1 for port class pools 0, 8, 16, 24, 32, 40, 48, 56, 64 and 72 (all port call pools associated with queues with queue-id 1) is set to aggregate control pool 0. This ensures that all provisioned queues in scheduling class 1 are limited based on the amount of buffering for class 1 on their port and also the total buffers used by the class is limited for the MDA. If either of the pools is exhausted, the queue will not receive a buffer. In like manner, Agg-Pool-Ptr-1 for all port class pools 1, 9, 17, 25, 33, 41, 49, 57, 65 and 73 is set to pool 1. This is true up to port class pools 7, 15, 23, 31, 39, 47, 55, 63, 71 and 79 having Agg-Pool-Ptr-1 set to pool 7.

The second 8 aggregate control buffer pools (8 through 15) are used as provisioned root pools. The purpose of the root pools is to allow the class pools to be oversubscribed without the possibility of the provisioned buffer usage stealing buffers from the system reserved buffers. Before sizing the provisioned root pools, a portion of the total buffer space is set aside for system purposes. The remaining buffers are divided between the provisioned root pools based on a weight parameter in each root pool. The weights may be set between 0 and 100. A value of zero indicates that a specific provisioned root pool will not receive buffers (pool size will be 0). Because root pools cannot be oversubscribed, they provide a protection mechanism for higher level pools. The number of root pools in use is dependant on the HSMDA pool policy applied to the MDA (ingress and egress are controlled by independent policies). The aggregate control class pools are associated with the root pools through the policy as well. Figure 36 represents the buffer pool hierarchy.

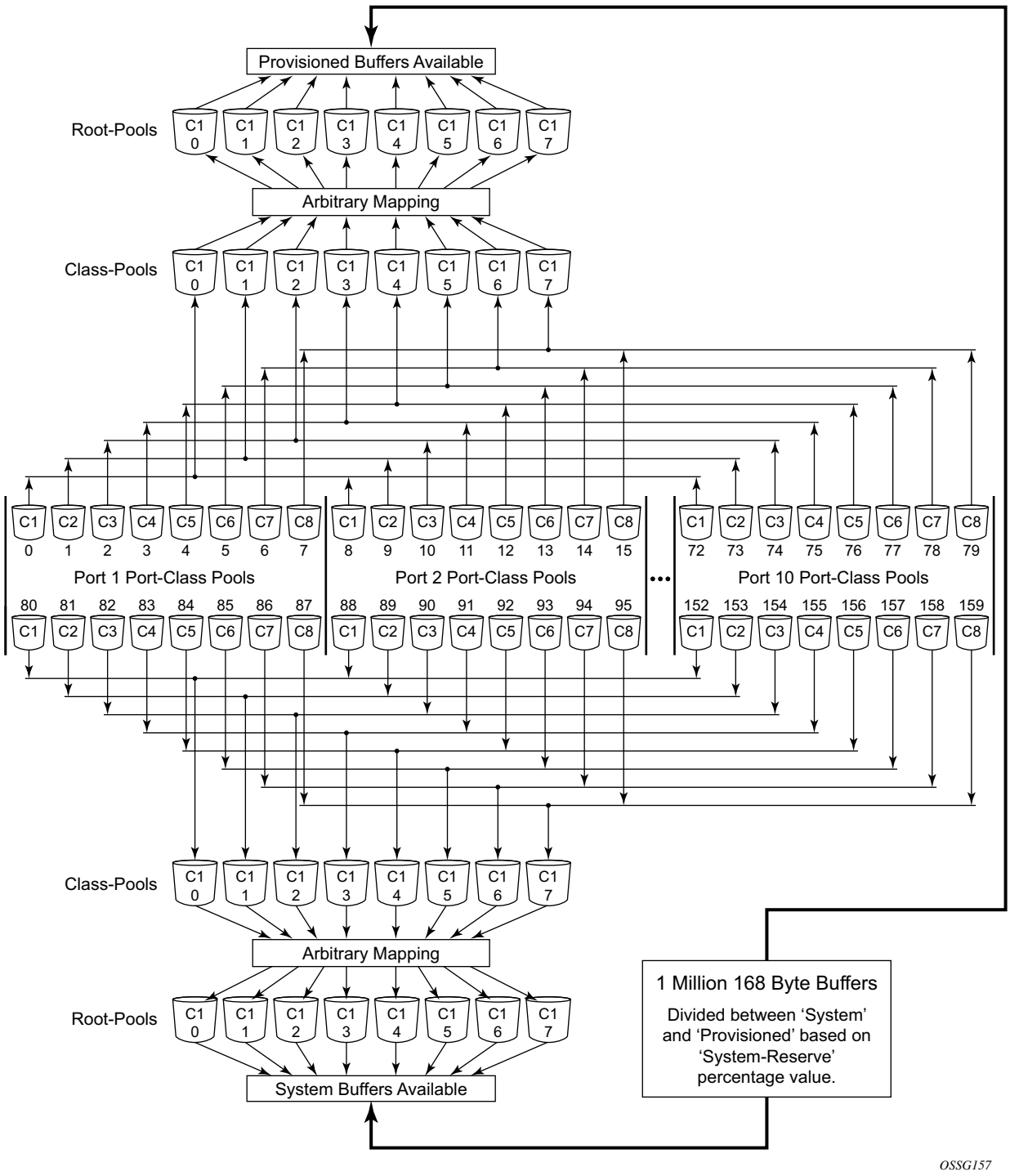

Figure 36: Buffer Pool Hierarchy

HSMDA Buffer Pool Policy

HSMDA buffer pool policies contain information the system uses to configure the individual pool sizes and the root pools used by the class pools. The division between total provisioned and total system buffers is based on the system-reserve parameter that defines the percentage of buffers reserved for system use. The remaining buffers are placed in the provisioned root buffer pools based on the weight associated with each root pool.

The eight class pools within the policy are defined with a root-pool parent association (1 through 8) and a percentage value. The parent associated with the class-pool defines the Agg-Pool-Ptr-2 setting in the port class pools table for each port class pool using the class-pool. For example, if class-pool 1 (aggregate control pool 0) is associated with root-pool 3 (aggregate control pool 10) within the policy, port class pools 0, 8, 16, 24, 32, 40, 48, 56, 64 and 72 will contain a value of 10 in Agg-Pool-Ptr-2. The percentage used to size the class pool is applied to the class pool's root-pool size to derive the class pool size. The sum of the sizes of the class pools parented by a root-pool may exceed (oversubscribe) the root-pool size.

System pools are managed similarly. The actual system port class pools and system aggregate control pools (16 through 31) are not user configurable.

Default HSMDA Buffer Pool Policy

An HSMDA buffer pool policy named **default** always exists on the system and cannot be deleted or edited. The default policy is used for ingress and egress on all HSMDAs until an explicitly created HSDMA policy is defined on the HSMDA.

The default policy contains the following parameters:

Table 51: Default Policy Parameters

System Reserve	
Percentage:	10%
Root Pools:	
Root-Pool 1 Weight:	75
Root-Pool 2 Weight:	25
Root-Pool 3 Weight:	0
Root-Pool 4 Weight:	0
Root-Pool 5 Weight:	0

Table 51: Default Policy Parameters

System Reserve	
Root-Pool 6 Weight:	0
Root-Pool 7 Weight:	0
Root-Pool 8 Weight:	0

Table 52: Class Pool Parameters

Class Pools		
Class-Pool 1	Parent:	Root-Pool 1
	Percentage:	40%
Class-Pool 2	Parent:	Root-Pool 1
	Percentage:	35%
Class-Pool 3	Parent:	Root-Pool 1
	Percentage:	30%
Class-Pool 4	Parent:	Root-Pool 1
	Percentage:	25%
Class-Pool 5	Parent:	Root-Pool 1
	Percentage:	20%
Class-Pool 6	Parent:	Root-Pool 2
	Percentage:	50%
Class-Pool 7	Parent:	Root-Pool 2
	Percentage:	40%
Class-Pool 8	Parent:	Root-Pool 2
	Percentage:	30%

Port Class Pool Sizing

The port class pools are sized based on the port's active bandwidth, provisioned use of the port and the port bandwidth rate modifier percentages defined on the port. For the HSMDA, a port's active bandwidth is simply the current speed provisioned for the port. For ingress, this is the current line rate of the port. For egress, it is the lesser of the port's line rate and the port scheduler's current maximum rate. The active bandwidth of the port is considered to be zero when a SAP or subscriber has not been provisioned on the port. Once a queue group is associated with the port, the actual active bandwidth of the port is used for port class buffer pool sizing.

The active bandwidth can be modified by the **max-rate** commands on the port. The parameters are used to artificially increase or decrease the amount of buffers that may be used by the port. These commands have no effect on the actual bandwidth used by the port.

The system uses the active bandwidth of each port to decide on how much each class pool the port should receive. The ports active bandwidth is divided into the sum of all ports active bandwidth to derive the port's pool factor. This factor is multiplied by the size of each class pool and the result is applied to the port's class pool. Thus, the sum of the sizes of a given port class pool over all ports should equal the size of the actual aggregate control class pool.

HSMDA Available Buffer Register Operation

The HSMDA considers a buffer pool empty when the buffer-available register for the pool drops below 64. As buffers are allocated to a queue, the HSMDA evaluates whether the queue's port class pool buffer-available register is less than 64. If not, the HSMDA then looks at the aggregate control pools pointed to by the port class pool's Agg-Pool-Ptr-1 (class-pool) and Agg-Pool-Ptr-2 (root-pool) to determine whether either of the pool's buffer-available registers are less than 64. If not, then the buffers required to enqueue the packet on the queue are given to the queue and the buffer-available registers are decremented based on the number of buffers given. As packets are scheduled out of the queue, the buffers are returned to the free list and the buffer-available registers are incremented by the correct amount.

The HSMDA stops allocating buffers at 63 or less remaining to allow packet size fairness on the pools. If it simply stopped at 0, large packets would be at a disadvantage as the buffer pool neared the zero mark. If the register read 20 buffers remaining and a packet arrives needing 21 buffers, the pool would deny the buffer request (it would decrement below 0). But if a smaller packet arrives needing less buffers than 20, it would be allowed. And the buffer pool would continue to allow the smaller packets until the pool was depleted. By stopping at 63 or less, every packet has access to the remaining pool since the maximum size packet requires less than 64 buffers. While this scheme allows for fair access to the buffer pool, some buffers will not be used by the HSMDA. This is considered inconsequential due to the limited number of buffer pools. The absolute worst

case in buffer inefficiency is 12,096 buffers (63 buffers * 192 pools) out of 1 million buffers (about 1.2%).

When the pools are first initialized, each pool's buffer-available register is set to the number of buffers available on the pool. When the HSMDA buffer pool policy managing a pool is changed so that the number of buffers managed by the pool increases or decreases, the system will increment or decrement the current value of the buffer-available register based on the change in the pool size. The register supports a negative value for the case where the buffer pool size is decreased and the buffer-available register is currently less than the size of the decrease. If the buffer pool goes negative (or below 64 available buffers), no buffers are allocated by the pool until buffers associated with the pool are returned to the free list and the register increments to a value of 64 or higher.

HSMDA Queue Congestion and Buffer Utilization Controls

Each queue supports a 10 bit index into an HSMDA slope policy table. Each policy in the table consists of two RED slopes (high priority and low priority) for the purpose of managing queue congestion. Due to the large number of queues supported on the MDA, the ability of a queue to effectively mange a weighted sliding window of queue utilization is not practical. HSMDA RED slopes operate as on the instantaneous depth of the queue.

Each slope policy within the HSMDA consists of two slope definitions, each represented by a 14 bit start slope value, a 14 bit end slope value and an 8 bit fixed point inverse slope value. The fixed point value is represented as a 4 bit whole value (values from 0 to 15) and a 4 bit fraction (from 0 to 0.9375 in 0.0625 increments).

In operation, a packet attempting to enter a queue triggers a check to see if the packet should be allowed based on queue congestion conditions. The packet contains a congestion-priority flag in the shim header telling the HSMDA whether to use the high or low slope. The slope policy containing the slope is derived from the policy index in the queue configuration parameters on the HSMDA. The MDA retrieves the current queue depth in buffers and the slope's three configuration values (start buffer, end buffer and inverse slope value). Logically, the following algorithm is used to determine whether the packet should be allowed in the queue based on discard probability:

- If the queue depth is greater than or equal to the slope's end-buffer value, the packet is discarded.
- A random number in the range of 0 to 127 is generated.
- The random number is multiplied by the inverse slope value and then added to the slope Start-Buffer to derive the random fill depth which is the number of buffers that need to be full in order for the slope discard threshold to cross the random number.
- If the slope fill depth is equal to or greater than the random fill depth, the packet is discarded.

- The packet still may be discarded if a buffer pool associated with the queue has decremented to the discard point or if the free buffer list is exhausted.

Maximum HSMDA Queue Depth

Since RED slope discards are done based on the current queue depth before allowing a packet into the queue, a queue may consume buffers beyond the configured MBS value based on the size of the packet. The maximum number of buffers is based on the high slopes end-buffer parameter plus the maximum packet size less one buffer. Once the slopes end-buffer is reached or exceeded, all other packets reaching the queue and associated with the slope will be discarded. When scheduling removes packets from the queue, the queue depth will decrease, eventually lowering the depth below the end-buffer threshold.

Control Plane HSMDA RED Slope Policy Management

RED slope configuration is managed by defining up to 1,024 named HSMDA slope policies on the chassis and mapping the queue to a specific policy name. Each slope policy contains configuration information for the high-priority slope and the Low-Priority slope. HSMDA Slope policies differ from standard slope policies in that they do not support the time-average-factor feature used to manage the weighting utilization of the buffer space the standard slopes are managing. HSMDA queue slopes operate based on instantaneous queue utilization and do not maintain a weighted utilization value. Figure 37demonstrates the high and low priority RED slopes used to derive the discard probability based on the current depth of the queue.

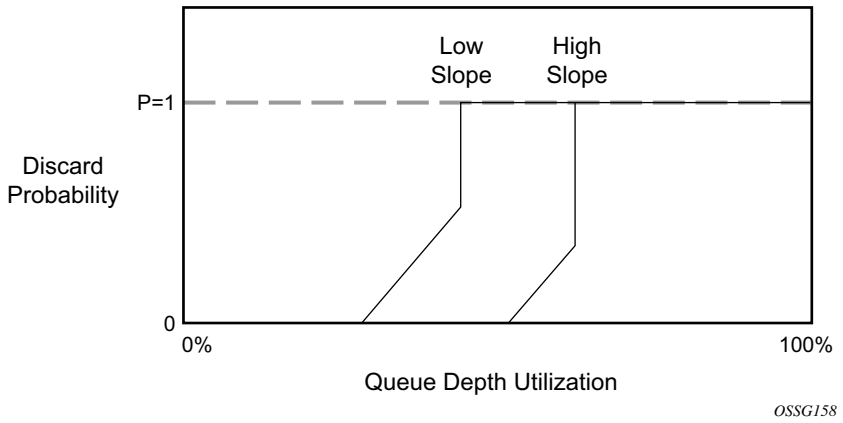

Figure 37: High and Low RED Slopes

HSMDA Slope Policy MBS Parameter

Unlike standard queues, HSMDA queues do not have a configured maximum buffer size (MBS) parameter. Instead the high and low RED slopes are used for all queue congestion control functions by the queue. The system uses an MBS value to define how much buffering an HSMDA queue may use, but it is contained in the HSMDA slope policy. The slope policy uses the configured MBS value to provide context for the slope parameters which are defined as percentages of MBS.

HSMDA Slope Policy Slope Parameters

Each slope in the slope policy is defined by three points; the starting-depth, slope maximum-depth and the slope maximum-discard-probability. Only three points are required since the starting-depth value implies a starting-discard-probability of zero percent. The starting-depth defines at which queue depth the slope starts to rise from zero. The maximum-depth point defines the queue depth where the slope ends and goes straight to 100%. The maximum-discard-probability defines the how high the slope rises from zero before going straight to 100% at the maximum-depth point.

If the starting-depth and maximum-depth percentages are equal, the system performs a simple drop tail; the discard probability slope is essentially non-existent in this case.

The system takes the configured slope parameters and uses them to calculate the HSMDA internal slope definitions:

- The MBS value is multiplied by the starting-depth percentage to derive the number of bytes at which the slope will start. This value is then converted to buffers by dividing the resulting byte value by the buffer size (168 bytes) and rounding to the nearest integer value. This is the HSMDA slope start-buffer parameter (14 bit value).
- The same process is performed for the maximum-depth percentage to derive the HSMDA slope end-buffer parameter (14 bit value).
- The system calculates the Inverse-slope value using the following formula:

  → Inverse-slope = (end-buffer — start-buffer) / maximum-discard-probability * 1.27.
- The Inverse-Slope value is converted to an 8 bit fixed point notation where:

  → Bits 7 through 4 represent the value above zero and bits 3 through 0 represent the fraction below zero.
- If the inverse slope is less than 0.0625, the system uses a value of 0.0625 (0000.0001) as the 8 bit value.
- If the inverse slope is greater than 15.9375, the system uses 15.9375 (1111.1111) as the 8 bit value.
- Table 53 can be used to derive the 8-bit value for inverse slopes within the range 0.0625 and 15.9375.

Table 53: HSMDA Inverse Slope Fixed Point Binary Values

Inverse Slope Fixed Point Binary Format (23 22 21 20 . 2-1 2-2 2-3 2-4)	
Whole Number Decimal Value	**Corresponding Binary Value (23 22 21 20)**
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011

Table 53: HSMDA Inverse Slope Fixed Point Binary Values

Inverse Slope Fixed Point Binary Format (23 22 21 20 . 2-1 2-2 2-3 2-4)	
12	1100
13	1101
14	1110
Whole Number Decimal Value	**Corresponding Binary Value (23 22 21 20)**
15	1111
0.0000	0000
0.0625	0001
0.1250	0010
0.1875	0011
0.2500	0100
0.3125	0101
0.3750	0110
0.4375	0111
0.5000	1000
0.5625	1001
0.6250	1010
0.6875	1011
0.7500	1100
0.8125	1101
0.8750	1110
0.9375	1111

For instance, if the MBS value is defined as 16,800 bytes and the low slope was configures with a starting depth set to 75 percent, a maximum depth set to 100 percent and a maximum discard probability set to 80 percent:

- The system takes 75 percent of 16,800 bytes and derives a starting slope at a queue depth of 12,600 bytes
- The system takes 100 percent of 16,800 bytes and derives an ending slope of 16,800 bytes.

- The system converts the starting, ending and slope duration values to number of buffers and then calculates the slope duration (end — start).

  → 12,600 / 168 = 75 start-buffer

  → 16,800 / 168 = 100 end-buffer

- The actual step of the slope may be calculated by dividing the maximum discard probability value with the slope run:

  → 80 / (100 ñ 75) = slope step is 3.2 (for every buffer beyond the start of slope, the probability rises 3.2 percent, after 25 buffers the slope reaches 80%).

- But the HSMDA uses the inverse slope within its algorithm and a drop probability random number between 0 and 128. The system calculates the inverse slope by dividing the slope run by the maximum discard probability multiplied by 1.27 (conversion from 0..100 to 0..127):

  → (100 — 75) / 80 * 1.27 = 0.396875 inverse-slope

- The system takes the inverse slope step and converts it to an internal HSMDA fixed point binary notation with the most significant 4 bits representing the whole portion of the inverse slope (above 0) and least significant 4 bits representing the fractional portion of the slope (below 0).

  → The inverse slope is less than 0 so the most significant 4 bits is 0000.

  → The fractional portion of the inverse slope is 0.396875 and the closest result in four bits based on Table 53 is a least significant bit value of 0110 (0.375 decimal).

  → The system concatenates the two results into an 8 bit number resulting in 0000110 as the inverse slope binary value.

The system uses the starting buffer value, the ending buffer value and the inverse slope eight bit value to populate a slope definition into the HSMDA. Two slopes are populated per slope policy. Each slope policy is given an HSMDA slope index between 0 and 1023. Since every packet received on an HSMDA queue is associated with either the high or low slope, the provisioned MBS value is not required and is not a managed parameter for HSMDA queues.

HSMDA Slope Shutdown Behavior

HSMDA slope policies allow the high or low RED slope to be shutdown. This effectively configures the HSDMA internal slope to have a Start-buffer and end-buffer equal to the buffer value closest to the configured policy MBS value. Packets matching the slope will still be discarded based on the slope, but it will appear that the MBS value is being enforced within the queue. The maximum-discard-probability is set to 100% when the slope is shutdown. The inverse-slope value sent to the MDA is set to all zeros 00000000 but has no effect since the starting and ending buffers are the same.

Ingress Packet Mapping to HSMDA RED Slope

At ingress, the following is used to determine packet mapping to a RED slope:

High slope

- Explicit profile classified as in-profile
- Undefined profile, but classified as high priority

Low slope

- Explicit profile classified as out-of-profile
- Undefined profile, but classified as low priority

Egress Packet Mapping to HSMDA RED Slope

At egress, the following is used to determine packet mapping to a RED slope:

High slope

- Egress reclassified high priority
- Non-reclassified in-profile

Low slope

- Egress reclassified low priority
- Non-reclassified out-of-profile

A default HSMDA slope policy for HSMDA queues with default parameters for the low and high slopes will be used to mimic the non-HSMDA queue drop tail behavior:

MBS	16,800 bytes (100 buffers)
High Slope	Start-depth 100% Max-depth 100% Max-probability 100% Shutdown
Low Slope	Start-depth 90% Max-depth 90% Max-probability 100% No shutdown

HSMDA Queue Congestion or Pool Congestion Discard Stats

When a packet is discarded, the high priority or low priority packet and octet discard counter is incremented. Which counter is incremented is based on the counter ID and the value of the congestion-priority flag within the packets HSMDA header. The classification rules within the SAP ingress QoS policy use the following logic to determine the discard counter associated with a packet:

Explicit in-profile (color aware profiling)	high priority
Explicit out-of-profile (color aware profiling)	low priority
Non-profiled, high priority	high priority
Non-profiled, low priority	low priority

The SAP egress QoS policy uses the following logic to determine the discard counter associated with a packet:

In-profile at ingress	high priority
Out-of-profile at ingress	low priority

Note that the discard counters and the RED slope determination are both driven by the same classification results. When a packet is defined as explicitly in-profile or out-of-profile, the high or low priority of a packet is ignored at ingress.

Egress Queue CIR Based Dot1P Remarking

The HSMDA adds the capability to perform remarking of one dot1p value within a dot1q or qinq-tagged packet based on the dynamic CIR state of the egress queue at the time the packet is scheduled out the egress port. This allows downstream aggregation Layer 2 aggregation devices to manage congestion based on the dot1p field (including the DEI bit). This feature is not supported on IOM-1.

SAP Ingress and SAP Egress QoS Policies

The queue definition and scheduling behavior for HSMDA queues require different provisioning behavior from the standard QChip based service level queuing.

SAP Ingress QoS Policy

The SAP Ingress QoS policy performs three distinct functions:

- Service queue definitions
- FC and sub-class queue mappings and ingress attributes
- Definition of hierarchical packet classification rules

For standard queuing, the application of the SAP ingress QoS policy to a SAP results in a set of hardware queues being dynamically assigned to the SAP representing the service queues defined in the policy.

The remaining sections of the policy affect resources within the hardware forwarding plane. Each ingress policy within the ingress forwarding plane consists of a set of classification rules consisting of dot1p, IP Precedence and IP DSCP tables used to match packets that ingress the SAP. Each table entry maps the packet to a forwarding class (each sub-class is contained with a forwarding class, so sub-classes indirectly map to a forwarding class). The section of the policy that represents the forwarding class to queue mappings is also represented by a table which allows the forwarding plane to take the forwarding class and determine which service queue (and the resulting hardware queue) that will handle the packet.

The SAP ingress policy has been expanded so a single ingress policy contains queue definitions for both standard service queues and for HSMDA service queues. This provides a policy assignment model that does not need to know the difference between a SAP using standard service queues and another SAP using HSMDA service queues.

The standard ingress service queues are separated into two types, point-to-point and multipoint. point-to-point queues are used either for VLL services such as Epipe or for unicast traffic within a VPLS, VPRN or IES service. Multipoint queues are used by VPLS, VPRN and IES services for packets that must be replicated by the switch fabric. Within the IES and VPRN services, only IP multicast routed packets are forwarded through the multipoint queues. Within VPLS services, broadcast (MAC DA = ff:ff:ff:ff:ff:ff), multicast (non-broadcast destination with multicast bit set) and packets with an unknown destination MAC address are forwarded through the multipoint queues. The standard service queues within the SAP Ingress QoS policy are numbered 1 through 32 of which 8 may be point-to-point queues and 24 multipoint queues. Since the HSMDA queues are not required to handle multipoint forwarding into the switch fabric, a distinction between point-to-point and multipoint is not present for HSMDA queues. Also, HSMDA queues are not created or dynamically assigned to a SAP or subscriber context. Instead eight queues (numbered 1

through 8) always exist for each queue group on the MDA. Since the queue group is assigned to the ingress SAP, all eight queues within the group automatically are available for the SAP. This means that although a SAP ingress QoS policy does not reference a particular queue ID, that queue is available for forwarding class mappings.

Another difference between standard service queues and HSMDA service queues is that they cannot be associated with a port or service level virtual scheduler and so the port-parent and parent commands are not available. Instead, each queue is implicitly mapped directly to the HSMDA port or ingress scheduler based on the queue ID. Queue 1 is mapped to scheduling class 1 and queue 8 is mapped to scheduling class 8. Scheduling class eight has the highest priority and one has the lowest unless a scheduling class is grouped with another scheduling class in which case the group itself inherits the scheduling priority of the highest class within the group (the classes within the group are handled based on the weight assigned to each class).

The last major difference between standard service queues and HSMDA service queues are the support for RED slopes within the HSMDA queues. The HSMDA uses the Slope policy the queue is associated with to configure the contour of the high and low slope within the queue.

The SAP ingress QoS policy classification rule actions are also modified to allow for a counter override capability. The counter override function allows for one of the eight extra sets of counters per queue group to be used instead of the per queue counters. This is intended for diagnostic or exception based accounting purposes.

SAP Egress QoS Policy

The SAP egress QoS policy requires the same type of modifications as the SAP ingress policy. The policy supports queue definitions for both standard service queues and HSMDA service queues and the forwarding class mappings to the individual queue IDs.

Another modification is the ability to define egress HSMDA counter override criteria which relies on an egress TCAM lookup based on IP flow criteria match entries. Egress IP flow based HSMDA counter overrides are ignored when applied to a SAP not on an HSMDA. Egress counter overrides are ignored when a SAP is a member of an efficient multicast group.

Subscriber Queuing Differences

Standard service queues are instantiated on a SAP or subscriber sla-profile basis. Each SAP or subscriber sla-profile instance within a SAP has its own set of queues which are managed into an aggregate SLA by a software based virtual scheduler. Due to its internal architecture, subscriber queues on the HSMDA are handled differently. The subscriber aggregate rate limit is represented by the queue group shaper. Since the queue group shaper only manages the eight queues within the group, all subscriber hosts within a subscriber context must share the same eight queues. Each subscriber SLA-profile (the object that allows different groups hosts for a single subscriber) must classify packets to the subscriber level queues and not to queues specific to the SLA-profile instance. While each SLA-profile instance does not maintain its own set of queues on the HSMDA, the packet classification rules are still maintained per sla-profile instance and not restricted to the subscriber level.

To allow provisioning of the sla-profile QoS classification rules and also provide the ability to specify the queuing behavior at the subscriber profile level, the SAP ingress and SAP egress QoS policies are reused in each case. At the sla-profile level, the SAP QoS policies packet forwarding class classification rules and forwarding class to queue ID mappings are in-effect (the queue definitions are ignored on the sla-profile instances for HSMDA subscribers). At the subscriber profile level, the queue-id definitions are in-effect while the packet classification rules and mappings are ignored.

HSMDA Features

HSMDA LAG

The addition of the HSMDA affects Ethernet Link Aggregation Groups (LAGs). Due to the different behavior between a SAP created on a standard Ethernet MDA and a SAP created on an HSMDA, it is important to know the expected behavior at the time the SAP is created. A SAP can be created on a LAG that has no port members which is clear, that at the time of creation, the type of SAP may be unknown.

To negate the issue between SAP type and LAG port membership, the **config>lag>port-type** {**standard** | **hsmda-ports**} command has must be executed prior to adding any ports to the LAG. This command allows the type ports that will be added to the LAG to be pre-defined. Without executing this command, HSMDA ports cannot be added to the LAG and after execution, the LAG may only be populated with HSMDA ports.

The LAG port type restriction can only be changed prior to adding SAPs or binding network IP interfaces to the LAG. If the port type for the LAG must be changed, all ports, SAPs and IP interfaces must be removed from the LAG.

A LAG with HSMDA member ports cannot be configured as mode network.

A LAG with HSMDA member ports can only operate in link-level SLA distribution. Since hierarchical QoS is only supported for queues within a single queue group and a queue group is limited to a single egress port, it cannot spread a SAP's ingress or egress queue CIR and PIR parameters over multiple LAG links. Each SAP created on an HSMDA LAG is assigned a queue group for each link within the LAG. The CIR and PIR defined in the SAP ingress or SAP egress QoS policy is replicated for the queue ID in each queue group.

Billing

The HSMDA SAP and subscriber queue billing statistics collection process supports the same information as non-HSMDA objects with the addition of the exception counter information. Statistics from queues within a queue group that are not currently mapped to forwarding classes are removed. Since the queues always exist, the counter information for the unused queues will be presented by the underlying collection mechanisms. Because of a large amount of data that could potentially exist, longer statistics collection intervals can occur.

Resource Management

The HSMDA presents a different set of resource management features to the system than with non-HSMDAs. Since the HSMDA does not manage a pool of queues that are individually mapped to SAPs or subscribers, it cannot run out of queues on the MDA. Instead, a pool of available queue groups are managed and allocated on a per SAP or subscriber basis.

Note that each SAP and subscriber created is managed against a system wide maximum. The maximum for SAPs and subscribers are each 64K within a single chassis.

HSMDA Queue Groups

A fundamental concept on an HSMDA is the queue group. Queue groups are not directly managed by the provisioner, they are indirectly assigned when creating SAPs or subscribers on the MDA. A queue group has eight queue members. The queues within the group are numbered from 1 through 8. When creating a SAP or subscriber associated with a port on the MDA, an ingress and egress queue group is allocated to the object. Every SAP and subscriber using the MDA has 8 ingress and 8 egress queues (whether they are in use or not). In the SAP ingress and egress QoS policies, the HSMDA queues within the group are represented by queue-id 1 through 8.

Each queue within the group has two RED slopes (managed by associating a slope policy to the queue), an MBS defined in bytes, a byte offset parameter used to add or subtract bytes to each packet handled by the queue for accounting purposes, a PIR and a CIR leaky bucket.

The queue group supports an aggregate shaper used to manage an aggregate rate limit for all queues within the group. Scheduling for queues within the group is stopped and started based on the rate set on the shaper.

Each queue group also supports 16 counter sets. Eight of the counters are the default counters used by packets assigned to each queue respectively. The remaining eight are exception counters and are named Counter 1 through Counter 8.

The number of queue groups available is dependent on the HSMDA variant. The ESS variant supports up to 8K ingress and egress queue groups. The SR variant supports up to 20K ingress and egress queue groups. The ability to utilize all available queue groups is dependant on the type of IOM that is hosting the HSMDA.

Scheduling Classes

The HSMDA supports eight scheduler classes that are directly mapped to the queue-id (1 through 8) for each SAP and subscriber queue. The scheduler class is not an internal QoS policy driven forwarding class. Forwarding classes within the system are used between the ingress and egress forwarding complexes and help the system to manage packet marking or remarking decisions and also are used to map each packet to an ingress and egress queue. It is possible to have two different QoS policies, the first that maps forwarding class AF (for example) to queue number 3 and the second may map AF to queue number 5. While the system will make certain common decisions based on the AF forwarding class, the fact that it is being mapped to different scheduler classes within the HSMDA will dictate that the scheduling of AF will be different for the two QoS policies based on the scheduler behavior.

Scheduling Class Weighted Groups

As indicated above, an HSMDA scheduler handles groups of queues based on each queues identifier. All queues numbered 1 are automatically placed in scheduler class 1, queues numbered 2 are placed in scheduler class 2 through queues numbered 8 being placed in scheduler class 8. Each scheduling class may be directly associated with a strict priority level or may be placed in one of two weighted groups. Each weighted group is used to map up to three scheduling classes into a single strict priority level and provides a weight for each member class of the group. Using the weighted groups allows for a mixture of strict and weighted scheduling between the scheduling classes. The scheduling classes mapped to a group must be consecutive in class order. If class 3 is placed into group 1, then the next class that may be placed into the group could be 2 or 4. If 4 were added to the group, then the next (and last) class that can be added to the group would be 5. If 2 had been added to the group instead of 4, then the next class would be limited to class 1.

Scheduler Strict Priority Levels

The scheduler maintains 8 strict levels with strict level 8 being the highest priority and strict level 1 being the lowest. For each strict priority level, either the scheduler class with the same ID (1 through 8) may be mapped to the strict level, or one of the two weighted groups may be mapped to the strict level. If a scheduling class is not mapped to a weighted group, the class is instead mapped to its relative strict scheduling level. Once a scheduling class is mapped to a weighted group, it is removed from the strict level and shares the strict level assigned to the weighted group. The weighted group itself is mapped to the inherent strict scheduling level of the highest member scheduling class. It should be apparent that when weighted scheduling class groups are used, fewer strict levels are active on the scheduler.

Strict Priority Level PIR

The scheduler supports a strict scheduling level PIR that limits the amount of bandwidth allowed for the level. The rate is defined in increments of megabits per second and may be set to max (the default setting) which disables the shaping function. The shaping rate is not defined on the strict priority level, but is inherited from the scheduling class or weighted group that is mapped to the strict level. The scheduler includes the full Ethernet frame encapsulation overhead when updating the priority level PIR, including the 12 byte inter-frame gap and the 8 byte preamble.

Scheduler Maximum Rate

A maximum scheduling rate may be defined for the scheduler. The rate is specified in megabits per second and the default rate is max which allows the scheduler to operate without a set limit. When the HSMDA scheduling policy is applied to an egress port, the maximum scheduling rate may be used to define a rate less then the available line rate of the port. When the HSMDA scheduler policy is applied to the ingress path of an HSMDA, it sets the maximum ingress bandwidth for the MDA (not usually a desirable action). The scheduler includes the full Ethernet frame encapsulation overhead when updating the scheduler level PIR, including the 12 byte inter-frame gap and the 8 byte preamble.

HSMDA Scheduler Policy Overrides

Once an HSMDA scheduler is applied to an Egress port or to an ingress HSMDA, the various parameters may be overridden. This allows an HSMDA scheduler policy to be adapted to changing needs on a port or HSMDA basis without requiring a new policy to be created.

Orphan Queues

Unlike port or service based virtual scheduling behavior, the HSMDA schedulers do not need to deal with orphaned queues (queues without an explicit scheduler parent defined). Every queue on an HSMDA is implicitly mapped to the scheduler based on the queues identifier.

Default HSMDA Scheduling Policy

An HSMDA scheduling policy with the name default always exists on the system and does not need to be created. The default policy cannot be modified or deleted. Attempting to delete the default policy using the no hsmda-scheduler-policy default command will return an error without changing the default policy.

The default policy contains the following parameters:

Table 54: HSMDA Scheduling Policy Default Values

Command	Default
description	no description
max-rate	no max-rate
group	group 1 rate max
	group 2 rate max
scheduling-class	scheduling-class 1 rate max
	scheduling-class 2 rate max
	scheduling-class 3 rate max
	scheduling-class 4 rate max
	scheduling-class 5 rate max
	scheduling-class 6 rate max
	scheduling-class 7 rate max
	scheduling-class 8 rate max

Basic HSMDA Configurations

HSMDA Pool Policies

The following displays details of the **default** HSMDA pool policy configuration.

```
A:ALA-48>config>qos>hsmda-pool-policy# info detail
----------------------------------------------
            no description
            system-reserve 10.00
            root-tier
                root-pool 1 allocation-weight 75
                root-pool 2 allocation-weight 25
                root-pool 3 allocation-weight 0
                root-pool 4 allocation-weight 0
                root-pool 5 allocation-weight 0
                root-pool 6 allocation-weight 0
                root-pool 7 allocation-weight 0
                root-pool 8 allocation-weight 0
            exit
            class-tier
                class-pool 1 root-parent 1 allocation-percent 40.00
                class-pool 2 root-parent 1 allocation-percent 35.00
                class-pool 3 root-parent 1 allocation-percent 30.00
                class-pool 4 root-parent 1 allocation-percent 25.00
                class-pool 5 root-parent 1 allocation-percent 20.00
                class-pool 6 root-parent 2 allocation-percent 50.00
                class-pool 7 root-parent 2 allocation-percent 40.00
                class-pool 8 root-parent 2 allocation-percent 30.00
            exit
----------------------------------------------
A:ALA-48>config>qos>hsmda-pool-policy#
```

HSMDA Scheduler Policies

HSMDA scheduler policies can be assigned to an egress HSMDA port or as the ingress control scheduler between the HSMDA and the ingress forwarding plane. The policy contains the needed commands to provision the scheduling behavior of a set of HSMDA scheduler classes. When assigned to an HSMDA egress port, the policy is used to define the scheduling behavior for all queues associated with the egress port. When assigned to the ingress path of an ESDMA, the policy is used to define the scheduling behavior for all ingress queues on the HSMDA (regardless of ingress port).

The following displays details of the **default** HSMDA scheduler policy configuration:

```
*A:ALA-48>config>qos# info detail
#------------------------------------------------
echo "HSMDA Scheduler Policies Configuration"
#------------------------------------------------
        hsmda-scheduler-policy "default" create
            description "Default hsmda scheduler QoS policy"
            no max-rate
            group 1 rate max
            group 2 rate max
            scheduling-class 1 rate max
            scheduling-class 2 rate max
            scheduling-class 3 rate max
            scheduling-class 4 rate max
            scheduling-class 5 rate max
            scheduling-class 6 rate max
            scheduling-class 7 rate max
            scheduling-class 8 rate max
        exit
        hsmda-scheduler-policy "HSMDA-test" create
            description "HSMDA policy"
            no max-rate
            group 1 rate max
            group 2 rate 50000
            scheduling-class 1 rate max
            scheduling-class 2 group 2 weight 1
            scheduling-class 3 rate max
            scheduling-class 4 rate max
            scheduling-class 5 rate max
            scheduling-class 6 rate max
            scheduling-class 7 rate max
            scheduling-class 8 rate max
        exit
#------------------------------------------------
...
*A:ALA-48>config>qos#
```

HSMDA Slope Policies

The following displays details of the **default** HSMDA slope policy configuration.

```
A:ALA-48>config>qos#
------------------------------------------------
...
        hsmda-slope-policy "default" create
            description "Default hsmda slope policy."
            queue-mbs 16800
            high-slope
                start-depth 100.00
                max-depth 100.00
                max-prob 100.00
                no shutdown
            exit
            low-slope
                start-depth 90.00
                max-depth 90.00
                max-prob 100.00
                no shutdown
            exit
        exit
...
------------------------------------------------
A:ALA-48>config>qos#
```

Egress Queue Group

The following displays details of the **default** HSMDA egress queue group configuration.

```
config
   qos
     queue-group-templates
      egress
         queue-group group-name
            hsmda-queues
               low-burst-max-class class
               packet-byte-offset {add add-bytes|subtract sub-bytes}
               wrr-policy wrr-policy-name
               queue queue-id [create]
                  adaptation-rule pir adaptation-rule [cir adaptation-rule]
                  burst-limit size [bytes|kilobytes]
                  mbs mbs
                  rate pir-rate [cir cir-rate]
                  slope-policy hsmda-slope-policy-name
                  wrr-weight weight
                  exit
```

Configuring HSMDA Queue Group Overrides

When a queue group template is instantiated at the port, queue group queues can be overridden at the instance level using **hsmda-queue-overrides**.

The following display output displays a hsmda-queue-group and overrides example configuration at the port level.

```
*A:Dut-A>config>port>ethernet# /configure port 4/1/1
*A:Dut-A>config>port# info
----------------------------------------------
        shutdown
        ethernet
            mode access
            egress
                exp-secondary-shaper "Exp_Shaper_1" create
                exit
            exit
            access
                egress
                    queue-group "100" instance 1 create
                        hsmda-queue-override
                            secondary-shaper "Exp_Shaper_1"
                            wrr-policy "wrrRes"
                            packet-byte-offset add 3
                            queue 2 create
                                rate 2222
                            exit
                        exit
                    exit
                exit
            exit
        exit
----------------------------------------------
```

Applying HSMDA Policies

HSMDA policies and values are associated in the following entities. Refer to the router OS Interface Guide for command syntax and usage.

config
- **card**
  - **mda**
    - **ingress**
      - **hsmda-pool-policy** *policy-name*
      - **hsmda-scheduler-overrides**
        - **group** *group-id* **rate** *rate*
        - **no group** *group-id*
        - **max-rate** *rate*
        - **no max-rate**
        - **scheduling-class** *class* **rate** *rate*
        - **scheduling-class** *class* [**weight** *weight-in-group*]
        - **no scheduling-class** *class*
- **lag**
  - **port-type** {**standard**|**hsmda-ports**}
  - **no port-type**
- **port**
  - **ethernet**
    - **acess**
      - **egress**
        - **queue-group** *queue-group-name* [**create**] [**instance** *instance-id*]
          - [**no**] **hsmda-queue-override**
            - **packet-byte-offset** {**add add-bytes**|**subtract sub-bytes**}
            - **no packet-byte-offset**
            - **queue** *queue-id* [**create**]
            - **no queue** *queue-id*
              - **mbs** { [0..2625][**kilobytes**] | [0..2688000] **bytes** | **default** }
              - **no mbs**
              - **rate** *pir-rate*
              - **no rate**
              - **slope-policy** *hsmda-slope-policy-name*
              - **no slope-policy**
              - **wrr-weight** *weight*
              - **no wrr-weight**
            - **secondary-shaper** *secondary-shaper-name*
            - **no secondary-shaper**
            - **wrr-policy** *wrr-policy-name*
      - **hsmda-scheduler-overrides**
        - **group** *group-id* **rate** *rate*
        - **no groupv**
        - **max-rate** *rate*
        - **no max-rate**
        - **scheduling-class** *class* **rate** *rate*
        - **scheduling-class** *class* [**weight** *weight-in-group*]
        - **no scheduling-class** *class*