## What is Traffic Shaping?

Traffic shaping (also known as packet shaping) is bandwidth management technique that delays the flow of certain types of network packets in order to ensure network performance for higher priority applications. Traffic shaping essentially limits the amount of bandwidth that can be consumed by certain types of applications. It is primarily used to ensure a high quality of service for business-related network traffic.

The most common type of traffic shaping is application-based traffic shaping. Fingerprinting tools are first used to identify the application associated with a data packet. Based on this, specific traffic shaping policies are applied. For example, you might want to use application-based traffic-shaping to throttle peer-to-peer file sharing, while giving maximum bandwidth to a business-critical application such as Voice-over-IP (VoIP), which is especially sensitive to latency.

### **Why Traffic Shaping is Important**

Limited network resources make bandwidth prioritization a necessity. Traffic shaping is the one of the most important techniques to ensure a high quality of service for business applications and data. It is an essential requirement for a network firewall.

Other info:

https://www.techtarget.com/searchnetworking/definition/traffic-shaping

Traffic Shaping: -

The bandwidth for each channel is controlled by the priorities and settings of the Traffic Scheduler.

## Credit-Based Traffic Shaping Algorithm

<https://docs.amd.com/r/en-US/pg051-tri-mode-eth-mac/Credit-Based-Traffic-Shaping-Algorithm>

A diagram of a computer network

Description automatically generated

As shown in above figure, data for transmission over an AVB network can be obtained from three types of sources.

The transmitter (TX) arbiter selects from these three sources in the following manner:

* If there is AV data available and the programmed AV bandwidth limitation is not exceeded, then the AV packet is transmitted
* otherwise, the TX arbiter checks to see if there are any PTP packets to be transmitted
* otherwise, if there is an available legacy packet, this is transmitted.

The Ethernet AVB Endpoint uses configuration registers to set up the percentage of available Ethernet bandwidth reserved for AV traffic. To comply with the IEEE802.1Q specification these should not be configured to exceed 75%. The arbiter then polices this bandwidth restriction for the AV traffic and ensures that on average, it is never exceeded. Consequently, despite the AV traffic having a higher priority than the legacy traffic, there is always remaining bandwidth available to schedule legacy traffic.

The relevant configuration registers for programming the bandwidth percentage dedicated to AV traffic are defined in [Configuration and Status](https://docs.amd.com/r/aHIVLD3sFpYD~dvBT3HAYg/QesZINEPkT68pJrXdGy9Ug) and are:

* [TX Arbiter Send Slope Control Register](https://docs.amd.com/r/aHIVLD3sFpYD~dvBT3HAYg/M2sCmEbtQ8axk10QCYTbIA)
* [TX Arbiter Idle Slope Control Register](https://docs.amd.com/r/aHIVLD3sFpYD~dvBT3HAYg/4nd_FStvaWZ0w~zx~LsxcQ)

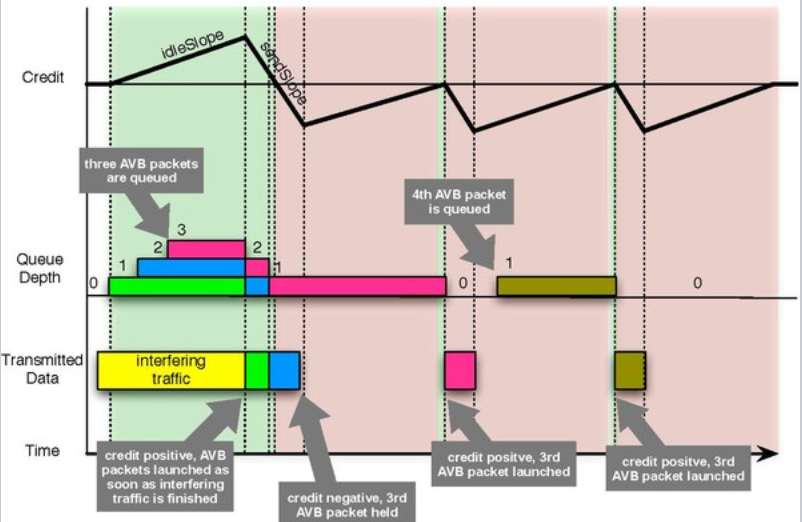
To enforce the bandwidth policing of the AV Traffic, a credit-based shaper algorithm is implemented in the TX Arbiter. The following figure illustrates the basic operation of the algorithm and indicates how the TX Arbiter decides which Ethernet frame to transmit.

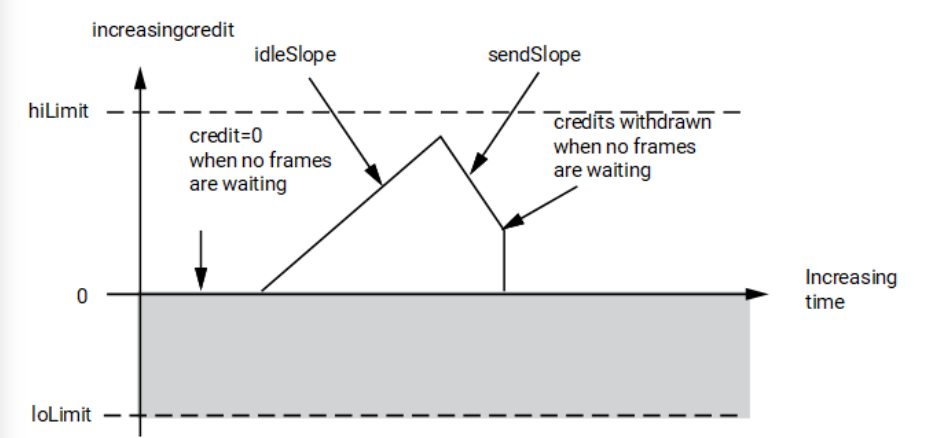
## Credit-based fair queuing

**Credit-based fair queuing** is a computationally efficient alternative to [fair queueing](https://en.wikipedia.org/wiki/Fair_queueing).

* Credit is accumulated to queues as they wait for service.
* Credit is spent by queues while they are being serviced.

**Queues with positive credit are eligible for service.** The rate of credit accumulation and release can be adjusted on a queue-by-queue basis to produce a weighted queuing behavior.





A diagram of a legacy frame

Description automatically generated

The previous figure illustrates the key features of the credit-based algorithm, which are:

* If the algorithm is in credit (≥ 0), the TX Arbiter schedules queued transmission from the [TX AV Traffic Interface](https://docs.amd.com/r/aHIVLD3sFpYD~dvBT3HAYg/Me_VmQ83Ef48RhNUqlvxTA). If there is less than 0 credit (not shown in the previous figure , but the credit can sink below 0), the TX Arbiter does not allow AV traffic to be transmitted; legacy traffic is scheduled instead when queued.
* When no AV traffic is queued, any positive credit is lost and the credit is reset to 0.
* When AV traffic is queued, and until the time at which the TX Arbiter is able to schedule it (while waiting for an in-progress legacy frame to complete transmission), credit can be gained at a rate defined by the idleSlope.
* During AV traffic transmission, credit is removed at a rate defined by the sendSlope.
* The hiLimit and loLimit settings impose a fixed range on the possible values of credit. If the available credit hits one of these limits, it does not exceed but saturates at the magnitude of that limit. These limits are fixed at the Ethernet MAC core level to ensure that the interface is not used incorrectly.

The overall intention of the two settings idleSlope and sendSlope is to spread out the AV traffic transmission as evenly as possible over time, preventing periods of bursty AV transmission surrounded by idle AV transmission periods. No further background information is provided in this document with regard to the credit-based algorithm. The remainder of this section describes the idleSlope and sendSlope variables from the perspective of the TX Arbiter.

##### **TX Arbiter Bandwidth Control**

The configuration register settings used for setting the cores local definitions of idleSlope and sendSlope, are described in general, and then from the point of view of a single example which describes the calculations made to set the register default values. This example dedicates up to 75% of the overall bandwidth to be reserved for the AV traffic (leaving at least 25% for the Legacy Traffic).

The calculations described are independent of Ethernet operating speed (no re-calculation is required when changing between Ethernet speeds of 1 Gbps and 100 Mbps).

###### **idleSlope**

The general equation is:

idleSlopeValue=(AV percentage / 100) x 8192

In this example, dedicating up to 75% of the total bandwidth to the AV traffic:

idleSlopeValue=(75 / 100) x 8192 = 6144

The calculated value for the idleSlopeValue should be written directly to the [TX Arbiter Send Slope Control Register](https://docs.amd.com/r/aHIVLD3sFpYD~dvBT3HAYg/M2sCmEbtQ8axk10QCYTbIA). This provides a per-byte increment value when relating this to Legacy Ethernet frame transmission.

###### **sendSlope**

The general equation is:

sendSlopeValue=((100 - AV percentage) / 100) x 8192

In this example, dedicating up to 75% of the total bandwidth to the AV traffic, obtain:

sendSlopeValue=((100 - 75) / 100) x 8192 = 2048

The calculated value for the sendSlopeValue should be written directly to the [TX Arbiter Idle Slope Control Register](https://docs.amd.com/r/aHIVLD3sFpYD~dvBT3HAYg/4nd_FStvaWZ0w~zx~LsxcQ). This provides a per-byte decrement value when relating this to AV Ethernet frame transmission.

## Traffic Shaping by EMAC Configuration

* Traffic shaping can also be made by configuring the EMAC hardware of the Samsung SOC.
* The V920 offers up to 16 DMA channels that can be assigned to different parts of the SOC via the Hypervisor (eg. node0, Android VM, DSP).
* Each DMA channel is assigned to a queue and different scheduling algorithms can be configured to process each queue.

A diagram of a network

Description automatically generated

## EMAC Scheduling Algorithms

**Weighted Round Robin**

* In Weighted Round Robin (WRR) algorithm, each traffic class is assigned a weight based on the percentage of configured bandwidth.
* All traffic classes are serviced in the round-robin order according to the programmed weights.

**Strict Priority**

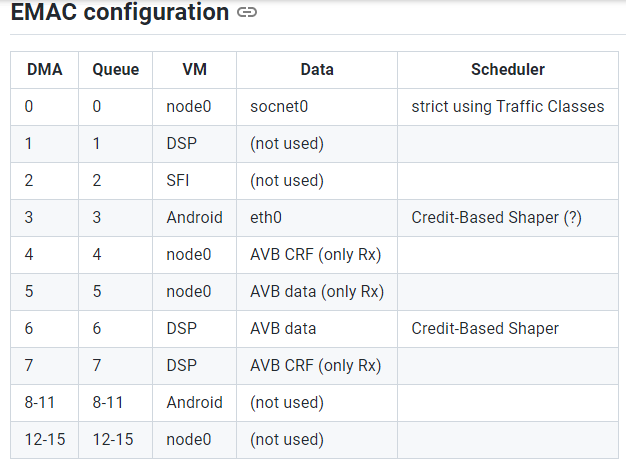
* In the strict priority algorithm, Traffic Class 0 has the lowest priority, and the priority increases with the Traffic Class number.

**Data Center Bridging**

* Priority-based flow control uses up to eight traffic classes which can be mapped to one of the eight priorities.
* When this feature is enabled, additional scheduling algorithms Deficit Weighted Round Robin and Weighted Fair Queuing can be used.

**Audio Video Bridging**

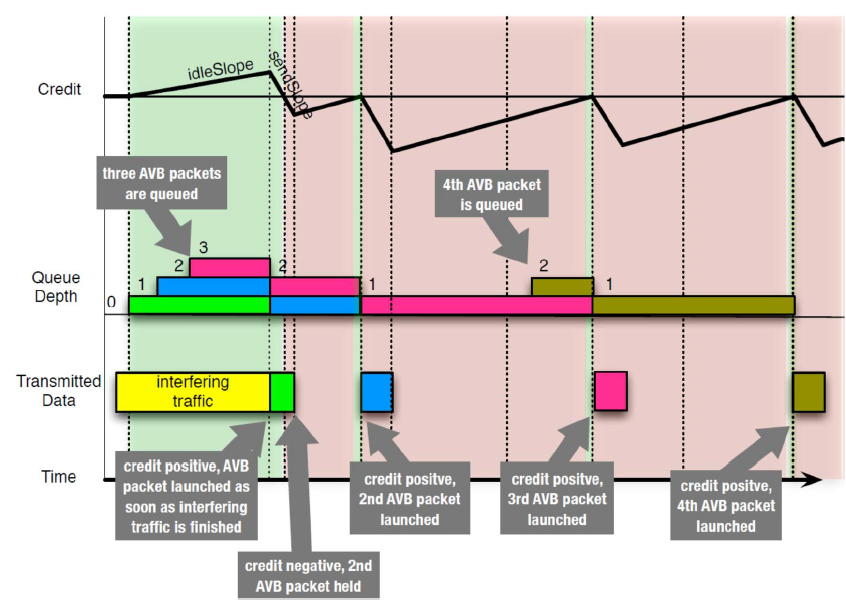
* Additionally, the EMAC supports prioritization of time-sensitive traffic like AVB.
* A strict scheduling algorithm can be used on queue0 for best-effort traffic while a Credit-Based Shaper Algorithm can be used to arbitrate AV traffic in all queues. **The credit-, idleSlopeCredit-, and sendSlopeCredit-** values can be configured together with the bandwidth status.



A screenshot of a computer

Description automatically generated

## IEEE 802.1Qav: Forwarding and Queuing for TSS



### Credit based shape TS Algorithm

* At the Frames of a particular AVB traffic class are transmitted only if there is at least zero credit for that class.
* If the credit for a particular class is negative, AVB frames for that SR class will not be transmitted, despite AVB streams having higher priority than best-effort traffic.
* This mechanism enforces fairness by guaranteeing that a certain percentage of best-effort, non-AVB traffic is also transmitted.
* Credits are calculated on a per-class and per-port basis, and the operation of the algorithm is represented by two parameters: idleSlope and sendSlope.
* The idleSlope parameter determines the rate at which credits are accumulated, while sendSlope defines the rate at which they are consumed.

sendSlope = idleSlope – portTransmitRate

A screenshot of a computer program

Description automatically generated

## CBS formula as per DWC handbook: -

[ Document name – DesignWare Core XGMAC ]

MAC Transaction Layer (MTL)The MAC Transaction Layer provides FIFO memory to buffer and regulate the frames between the application system memory and the DWC\_xgmac IP. The native interface enables data transfer between the host and the MAC transaction layer through a simple handshake protocol.

## DWC\_xgmac Audio Video (AV) features

* Separate channels or queues for AV data transfer
* Up to eight queues on the Receive paths for AV traffic and seven queues on the Transmit path for AV traffic
* IEEE 802.1-Qav specified credit-based shaper (CBS) algorithm for Transmit channels
* Single Tx FIFO and Rx FIFO (MTL) for all selected queues (system-side interface [AXI, or native] remains the same)

## Credit-Based Shaper Algorithm

[Reference: - DesignWare XGMAC document, Section – 6.4.4.4, Page – 301]

The MTL Queue Scheduler uses the credit-based shaper algorithm to arbitrate the AV traffic in all queues and the legacy Ethernet traffic in Queue 0. You can program the additional queues to use the credit-based shaper algorithm.

### Credit Value

* The credit value is accumulated every transmit clock cycle.
* The credit to be added or subtracted per cycle can be fractional based on the required idleSlope and sendSlope values.
* The credit for a TC builds up only when the packet is available, but it cannot be transmitted because the MAC is sending a packet from another TC.
* The DWC\_xgmac supports a mode (CC field of ETS control register) in which the credit can build up in advance for a TC which has no outstanding packets. This enables sending a burst of high priority traffic in a TC as soon as the data is available.
* The credit does not accumulate when there is no packet waiting in a queue and other queues are transmitting.

idleSlopeCredit and sendSlopeCredit ValuesThe software must program the idleSlopeCredit and sendSlopeCredit values. The programmed values must be the credit accumulated or drained per clock cycle scaled by 1024, such as, 320 \* 1024 = 327680 and 480 \*1024 = 491520.

In addition, the software must program the hiCredit and loCredit values, scaled by 1024, to  
adjust for scaling of the idleSlopeCredit and sendSlopeCredit values.

**Programming example for idleSlope, sendSlope, hiCredit, and loCredit**Consider a case where 3 TCs are in use (TC0, TC1, and TC2) and of which:  
■ TC2 [AV enabled] has 45% bandwidth allocated  
■ TC1 [AV enabled] has 35% bandwidth allocated  
and the remaining bandwidth is assigned to TC0  
■ TC0 [Best Effort] gets the left over bandwidth of 20%  
Consider 1500 bits for hiCredit and 800 bits for loCredit programming

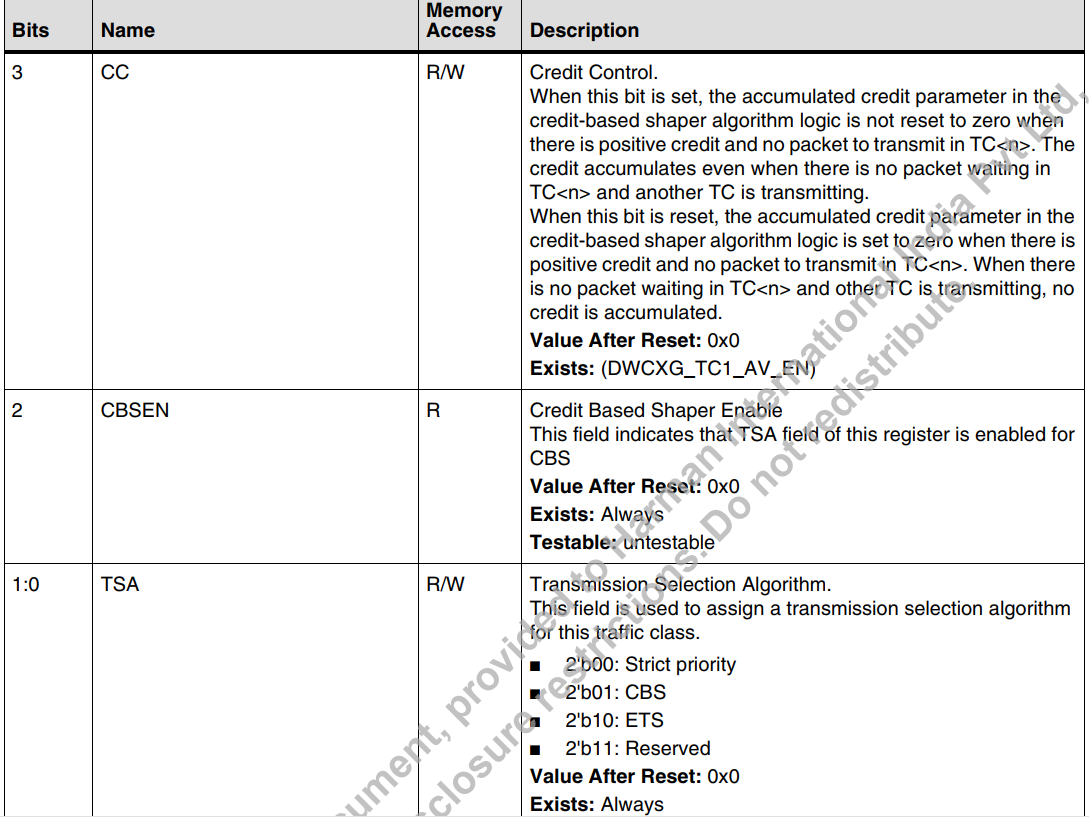
**TC2 Programming**idleSlope: (45/100) \* 32 \* 1024 = 14746 (45 is the bandwidth, 32 since MAC uses a 32-bit interface, and 1024 is the scaling factor)

sendSlope: ((100-45)/100) \* 32 \* 1024 = 18022 (100- bandwidth must be used to compute sendSlope)

hiCredit: ((100-45)/100) \* 1500 \* 1024 = 844800  
loCredit: ((100-45)/100) \* -800 \* 1024 = -446600

**TC1 Programming**idleSlope: (35/100) \* 32 \* 1024 = 11469  
sendSlope: ((100-35)/100) \* 32 \* 1024 = 21299  
hiCredit: ((100-35)/100) \* 1500 \* 1024 = 998400  
loCredit: ((100-35)/100) \* -800 \* 1024 = -532480  
The 2's complement value must be programmed for loCredit field because it is always a negative value.

### Hw Register: MTL\_TC(#i)\_ETS\_Control (for i = 0; i <= DWCXG\_NUM\_TC-1)



### Hw Register: MTL\_TC(#i)\_HiCredit (for i = 1; i <= DWCXG\_NUM\_TC-1)

A close-up of a memory card

Description automatically generated

### Hw Register: MTL\_TC(#i)\_LoCredit (for i = 1; i <= DWCXG\_NUM\_TC-1)

A close-up of a memory card

Description automatically generated

## Enabling the Audio Video Traffic Shaper

[Reference: - DWC XGMAC document, section – 14.27, Page - 1633]

### Initializing the DMA

This initialization sequence is used only for XGMAC-AXI configuration with AV feature. Complete the following steps to initialize the DMA:

1. Provide a software reset to reset all internal registers and logic (bit 0 in DMA\_Mode register).

2. Wait for the completion of the reset process. Poll bit 0 of the DMA\_Mode register), which is cleared only after the reset operation is completed.

3. Program the fields to initialize the DMA register by setting the values in DMA\_Mode register.

4. Create a proper descriptor list for transmit and receive. In addition, ensure that the DMA owns the Transmit and Receive descriptors. When OSF mode is used, at least two TX descriptors are required. For more information about descriptors, see “Descriptors” on page 1553.

5. Make sure that your software creates three or more different transmit or receive descriptors in the list before reusing any of the descriptors.

6. Program the Transmit and Receive Ring length registers DMA\_CH(#i)\_Tx\_Control2 register. The ring length programmed must be at least 4.

7. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor DMA\_CH(#i)\_TxDesc\_List\_Address register. In addition, you must program the Transmit and Receive tail pointer registers indicating to the DMA about the available descriptors (DMA\_CH(#i)\_Tx\_Desc\_Tail\_Pointer) register.

8. Program the following fields to initialize the mode of operation in the MTL\_TxQ0\_Operation\_Mode register:  
a. Transmit Store And Forward (TSF)  
b. Transmit Threshold Control (TTC)  
c. Transmit Queue Enable (TXQEN) to value 2‘b10 to enable Transmit Queue0  
d. Transmit Queue Size (TQS)

9. Enable the interrupts by programming the DMA\_CH(#i)\_Interrupt\_Enable register.

10. Repeat steps4 through 9 for all additional channels of AV feature.

11. Program the CBS control register, idleSlope, sendSlope, hiCredit, and loCredit registers of the AV Queues.

12. Start the Receive and Transmit DMA by setting bit 0 of the DMA\_CH(#i)\_TX\_Control register and bit 0 of the DMA\_CH(#i)\_RX\_Control register.

## DTS Mapping: -

File: - exynosautov720-evt1-idcevo-sp25-phy-b3-eth.dtsi

&mtl\_tx\_setup {

queue5 {

/delete-property/ snps,dcb-algorithm;

snps,avb-algorithm;

snps,send\_slope = <0x1000>;

snps,idle\_slope = <0x1000>;

snps,high\_credit = <0x3E800>;

snps,low\_credit = <0xFFC18000>;

};

queue6 {

/delete-property/ snps,dcb-algorithm;

snps,avb-algorithm;

snps,send\_slope = <0x1000>;

snps,idle\_slope = <0x1000>;

snps,high\_credit = <0x3E800>;

snps,low\_credit = <0xFFC18000>;

};

Send slope = 0x1000 => 4096

Idle\_slope = 0x1000 => 4096

Hi Jongho,

We need to program the CBS control register, idleSlope, sendSlope, hiCredit, and loCredit registers of the AV Queues ,which is routed/mapped with DSP via DMA ch6 for Audio specific packet communication.

Requirements ref:

Audio Tx by considering 3 Tx streams with each having 10 channels

Pkt size: 1324 bytes  , 750 pkt per cycle

1324  × 750 = 993,000 bytes

993000 × 3 Tx stream = 2,979,000 bytes

2979000 \* 8 bits = 23,832,000 bits, so around 22 Mbps

22 Mbps +3 margin =25Mbps

# For GMAC DWC doc: -

Reference: - DWC\_ether\_qos\_databook\_5.10a 3.pdf

## Credit value

The credit value is part of the credit-based sharper algorithm used by the MTL Queue Scheduler.

The credit value is accumulated every transmit clock cycle, that is, 40 ns for 100 Mbps and 8 ns for 1000

Mbps. The credit to be added or subtracted per cycle can be fractional based on the required idleSlope and sendSlope values.

idleSlope:

The rate of change of credit [bps]. when the value of credit is increasing. (i.e., while transmitting is FALSE).

sendSlope:

The rate of change of credit, in bits per second, when the value of credit is decreasing (i.e., while transmitting is TRUE.

## Forwarding rule in CBS algorithm: -

[ Reference: - chrome-extension://oemmndcbldboiebfnladdacbdfmadadm/https://www.duo.uio.no/bitstream/handle/10852/87168/5/Analysis\_of\_bandwidth\_reservation\_strategy\_for\_AVB\_classes\_in\_Ethernet\_TSN\_switch.pdf ]

1. If the transmission line is free, the scheduler transmits a frame of the highest priority class that satisfies the conditions: a) its queue is not empty; and b) it has a non-negative credit.

2. The credit of an AVB class is reduced linearly with rate send slope when the class transmits.

3. The credit of an AVB class increases linearly with rate idle slope when the following conditions hold simultaneously for that class: a) its queue is not empty; and b) other AVB or BE classes are transmitting.

4. Whenever an AVB class has a positive credit and its queue becomes empty, the credit is set to zero; this is called a credit reset.

5. If the credit is negative and the queue becomes empty, the credit increases with rate idle slope until the zero value.

Page - 1200

A screenshot of a computer

Description automatically generated

## idleSlopeCredit

When AV feature is enabled, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns for 100 Mbps; 8 ns for 1000 Mbps; 3.2 ns for 2500 Mbps) when the credit is increasing. The software

should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. Bits[20:14] must be written to zero.

{ idleSlopeCredit = rate of change of credit in bits per cycle }

Per cycle time: -

40 ns - - > 100Mbps

8 ns - - > 1000 Mbps

3.2 ns - - > 2500 Mbps

## sendSlopeCredit Value

When AV operation is enabled, this field contains the sendSlopeCredit value required for credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns, 8 ns and 3.2 ns for 100 Mbps, 1000 Mbps and 2500 Mbps respectively) when the credit is decreasing.

The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. This field should be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when Channel 1 is selected for transmission.

## hiCredit Value

When the AV feature is enabled, this field contains the hiCredit value required for the credit-based shaper algorithm. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024.

The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16,384 bytes or 131,072 bits). The value to be specified is 131,072 \* 1,024 = 134,217,728 or 0x0800\_0000.