



Switch Abstraction Interface (SAI) Specification for Credit-based Flow Control

v1.0

October 28, 2026

Copyright © 2026 Ultra Ethernet Consortium.

The Ultra Ethernet Consortium (UEC) is a consensus-based standards organization operating under the Linux Foundation. Its members collaborate openly to define and promote high-performance Ethernet technologies for modern computing environments. This specification ("Specification") contains Approved Deliverables of UEC as such term is defined in the UEC Charter.

This document is made available under the terms of the Creative Commons Attribution-NoDerivatives 4.0 International License (CC BY-ND 4.0). As stated in the license, you may copy and redistribute the document, provided that attribution is given to the Ultra Ethernet Consortium. However, if you create a derivative work from the material, you may not distribute that work. A copy of the license is available at <https://creativecommons.org/licenses/by-nd/4.0/>.

This Specification is provided on an "AS IS" basis. UEC and The Linux Foundation (the "Parties") make no warranties of any kind, either express or implied, including without limitation warranties of merchantability, fitness for a particular purpose or noninfringement of any third party intellectual property rights, or compliance with applicable laws. Use of the information in this Specification is at your own risk. The entire risk as to the results and performance of this Specification is assumed by the user. In no event will the Parties (as defined above) be liable to any other party for lost profits or any form of indirect, special, incidental, or consequential damages of any character from any causes of action of any kind with respect to this Specification or its governing document, whether based on breach of contract, tort (including negligence), or otherwise, and whether or not the recipient of this Specification has been advised of the possibility of such damage.

This Specification may include references to technologies for which patent rights might be claimed. UEC does not make any representation regarding the existence or status of any such rights. Implementers of this Specification are solely responsible for obtaining any necessary licenses or permissions from the appropriate rights holders.

UEC reserves the right to adopt any changes or alterations to this Specification as it deems necessary or appropriate.

Ultra Ethernet, Ultra Ethernet Consortium, and the Ultra Ethernet Consortium logo are the unregistered trademarks of Ultra Ethernet Consortium in the United States and other countries. All rights reserved. Other brands, names, or logos contained herein are the property of their respective owners.

For more information about the Ultra Ethernet Consortium, visit: <https://ultraethernet.org>.

Contents

1	Introduction	7
1.1	Terminology	7
1.2	Overview	7
2	SAI Specification	8
2.1	SAI Logical Pipeline	10
2.2	CBFC Credit Pool	10
2.3	CBFC Credit Profile	10
2.4	CBFC Virtual Channel	11
2.5	New Virtual Channel Statistics	11
2.6	QoS Map Updates	12
2.7	SAI Port Updates	12
2.8	New CBFC Port Statistics	13
2.9	SAI Queue Updates	13
2.10	Capability Query	13
2.10.1	Feature Capability	13
2.10.2	Statistics Capability Query	14
3	Normative References	15

Tables

Table 2-1 - Attributes of the New SAI CBFC Credit Pool Object Type	10
Table 2-2 - CBFC Credit Profile Attributes	10
Table 2-3 - CBFC Credit Profile Threshold Mode	11
Table 2-4 - Attributes of the New SAI Virtual Channel Object Type	11
Table 2-5 - Virtual Channel Statistics	11
Table 2-6 - New QoS Map Types	12
Table 2-7 - New Attributes of the existing SAI PORT object type	12
Table 2-8 - Port CBFC Statistics	13
Table 2-9 - New Attributes of the existing SAI Queue object type	13

Figures

Figure 1-1 - Overview of SAI attributes for CBFC	8
Figure 2-2 - Role of SAI objects in CBFC lossless operation at the sender.	9
Figure 2-3 - Role of SAI objects in CBFC lossless operation at the receiver.	9

Acknowledgments

This specification is developed under the Process, Procedures, and Guidelines of the Ultra Ethernet Consortium (UEC). The organization would like to thank all that are involved. A special acknowledgment goes out to the following individuals for their leadership, authoring, and technical contributions to this specification.

Steering Committee at Time of Publication

Role	Name	Affiliation
Chair	J Metz	AMD
Vice Chair	Abdul Kabbani	Microsoft
TAC Chair	Tom Emmons	Arista
TAC Vice Chair	Brad Matthews	Marvell

Working Group Leadership at Time of Publication

Working Group	Name	Affiliation
Physical Layer WG	Jannik Hammel Nielsen	Huawei
	Adee Ran	Cisco
Link Layer WG	Bob Alverson	HPE
	Yan Zhuang	Huawei
Transport Layer WG	Torsten Hoefler	Microsoft
	Karen Schramm	Broadcom
Software Layer WG	Rip Sohan	Broadcom
	Jianxin Xiong	Intel
Management WG	Ravindranath Kanakarajan	Marvell
	Hemal Shah	Broadcom
Compliance & Test WG	Dave Estes	Viavi Solutions
	Cliff Tavares	Keysight
Performance & Debug WG	Rajalaxmi Angadi	Intel
	Nathan Hanford	LLNL
Storage WG	Venkat Goli	Broadcom
	Claudio DeSanti	Dell

Specification Development

Role	Name	Affiliation
Author	Rupa Budhia	Marvell
Author	Ravindranath Kanakarajan	Marvell
Contributor	Jai Kumar	Broadcom
Editor	Paul Congdon	Linux Foundation
Program Manager	Rob Craig	Linux Foundation

1 Introduction

In high-performance Ethernet networks—especially those supporting AI and HPC workloads—network congestion can lead to buffer overflows, causing packet drops. These dropped packets trigger end-to-end retransmissions, which introduce latency and degrade the performance of latency-sensitive applications like distributed training or inference.

CBFC is a link-layer mechanism that ensures lossless frame delivery by preventing frame loss due to buffer congestion at the receiver.

1.1 Terminology

Term	Description
Credit	A token representing a unit of data storage at the receiver. Credits allow a sender to transmit data packets and are consumed at the sender as packets are transmitted. Credits are returned to the sender by the receiver when the receiver has released buffer resources that can be utilized for new packet arrivals.
Virtual channel (VC)	An entity that contains a subset of a port's traffic with similar traffic characteristics, dedicated buffering, and flow control management.
Control ordered set (CtlOS)	An 8-byte message format used by CBFC and LLR to pass information between link partners.
Credits consumed (CC)	Credits consumed so far (cyclical 20-bit counter)
Credits freed (CF)	Credits freed so far (cyclical 20-bit counter)
Credit limit (CL)	Credit limit for a given VC or port
Credits in use (CU)	Credits currently in use for a given VC or Port

1.2 Overview

In CBFC, the sender ensures that the receiver has enough buffer before transmitting a frame. The receiver advertises its buffer availability to the sender using credits. These credits represent the amount of buffer space available. The sender can transmit frames only if it has sufficient credits—ensuring that the receiver has space to accept the incoming data. Periodically, the receiver sends back to the sender the credits for the frames that have left the buffer. These are sent as control ordered sets (CtlOS). To handle credit leakage (e.g., corrupted packets, lost CBFC messages), cyclic counters are maintained and periodically synchronized using CC_Update messages.

CBFC supports up to 32 virtual channels (VCs) per port. Each VC can be configured as either lossless or best-effort, and this configuration can be changed dynamically.

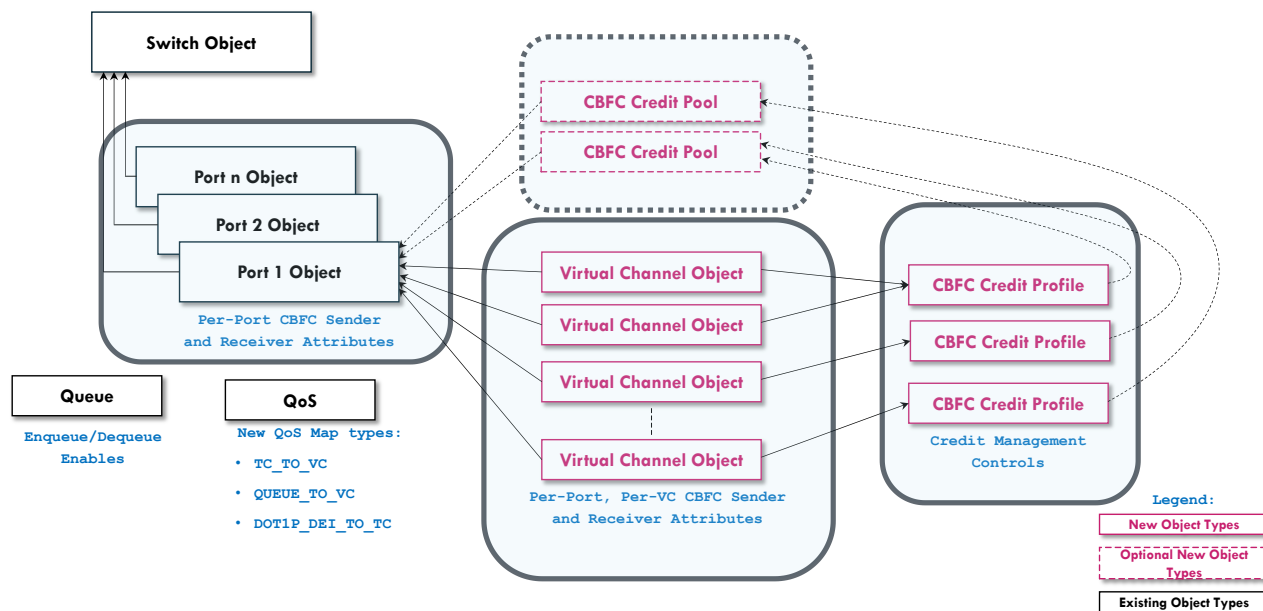


Figure 1-1 - Overview of SAI attributes for CBFC

2 SAI Specification

Credit-based flow control (CBFC) is a newly introduced feature in the Switch Abstraction Interface (SAI) that enhances link-level flow control without impacting the existing SAI logical pipeline.

- Three new objects are created:
 - Virtual channel: Contains per-VC attributes associated with CBFC sender and CBFC receiver functions on a port
 - CBFC credit profile: Attributes for management of receiver credits (i.e., receiver's buffer) on a per-VC basis
 - CBFC credit pool: Used to provide isolation of credits (i.e., receiver's buffer) between groups of VCs
- New attributes for CBFC are added to the port object.
 - CBFC sender attributes including negotiated parameters, receiver credit management controls, etc.
 - CBFC receiver attributes including negotiated parameters
 - Statistics and QoS mapping controls
 - Read-only attributes needed for LLDP parameter query
- New QoS map types are added for VC derivation from TC, queue, or a packet's dot1p+DEI.
- New queue attributes to enable/disable enqueue and dequeue operations. These attributes are needed for lossless VC initialization and deinitialization.

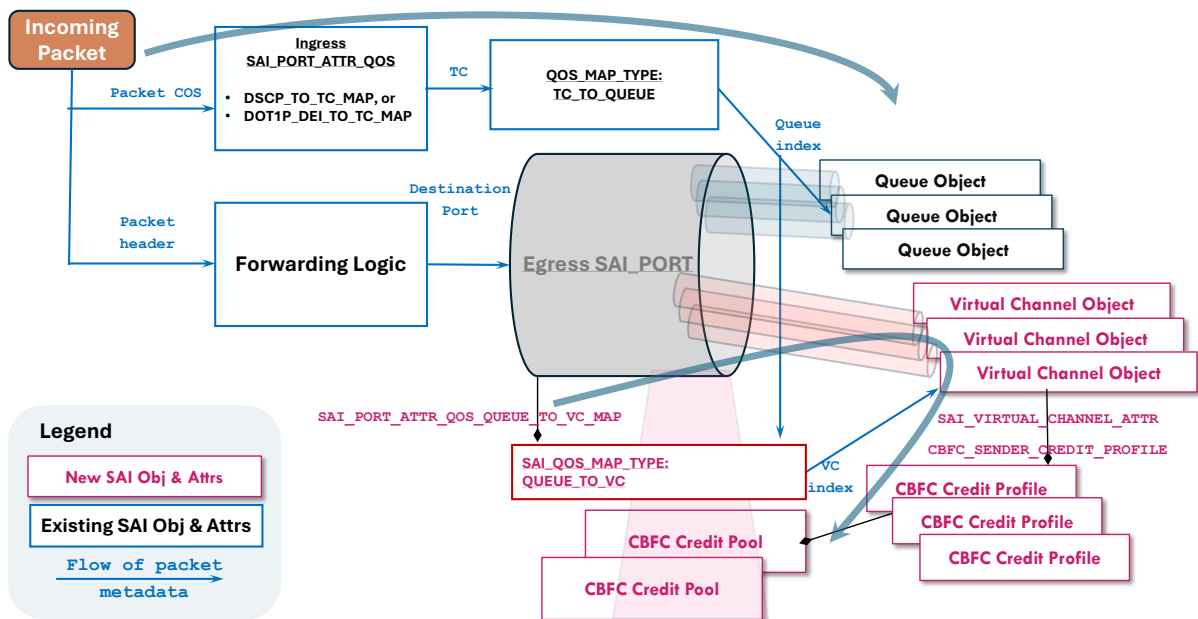


Figure 2-2 - Role of SAI objects in CBFC lossless operation at the sender.

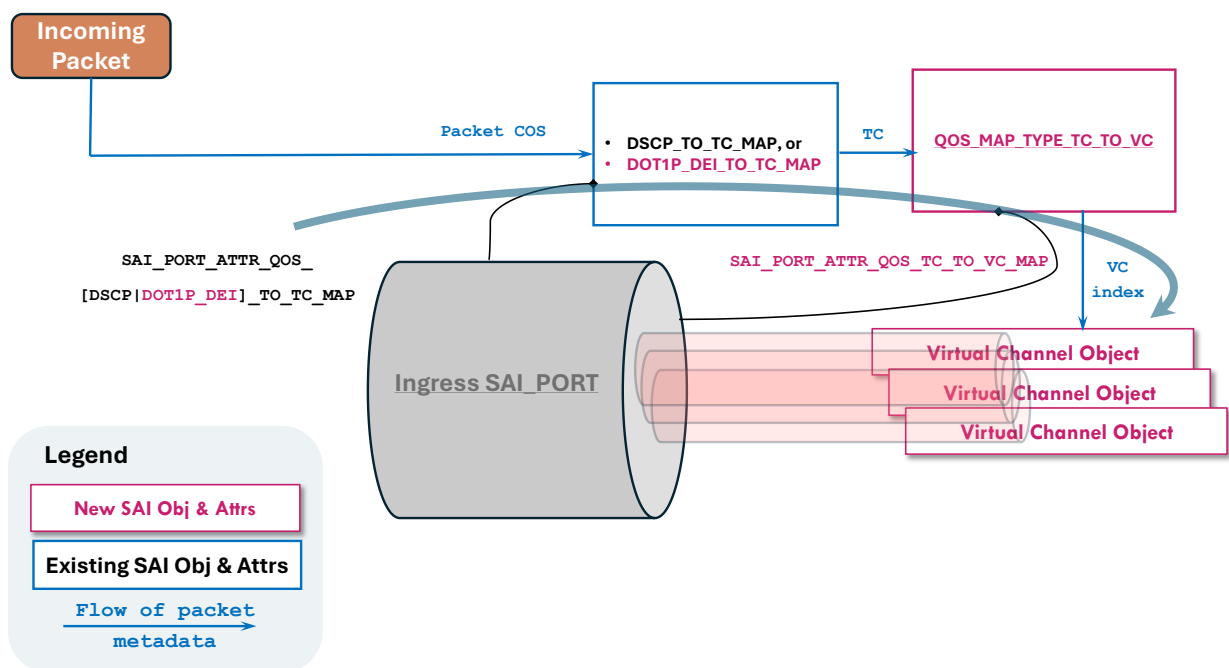


Figure 2-3 - Role of SAI objects in CBFC lossless operation at the receiver.

2.1 SAI Logical Pipeline

CBFC does not need any changes in the SAI logical pipeline.

2.2 CBFC Credit Pool

- CBFC credit pool is used at the sender to manage receiver credits.
- Credit pools are optional. They are created when:
 - Receiver sends total credits for the port and sender manages the credits across VCs, and
 - There is a need to provide isolation of credits between groups of VCs.
- When no credit pools are created at the sender, the total credits given by the receiver (configured via SAI_PORT_ATTR_CBFC_SENDER_CREDIT_LIMIT Table 2-7) are shared among all the VCs of the port. This sharing is based on the credit profiles attached to the VCs.

Credit pool attributes are listed in Table 2-1.

Table 2-1 - Attributes of the New SAI CBFC Credit Pool Object Type

SAI_CBFC_CREDIT_POOL_ATTR	Type	Flags	Default	Description
PORT	sai_object_id_t	MANDATORY_ON_CREATE CREATE_ONLY	NA	Port ID of the port to which the pool belongs.
SIZE	sai_uint64_t	MANDATORY_ON_CREATE CREATE_AND_SET	NA	CBFC credit pool size in credits.
SHARED_SIZE	sai_uint64_t	READ_ONLY	NA	Shared CBFC credit pool size in credits. This is derived from subtracting all reserved credits of VCs associated with this credit pool from the total pool size.

2.3 CBFC Credit Profile

The CBFC credit profile is used at the sender to manage receiver credits per VC. Credit profile attributes are listed in Table 2-2.

Table 2-2 - CBFC Credit Profile Attributes

SAI_CBFC_CREDIT_PROFILE_ATTR	Type	Flags	Default	Description
POOL_ID	sai_object_id_t	MANDATORY_ON_CREATE CREATE_ONLY; @allownull true	SAI_NULL_OBJECT_ID	Pointer to credit pool object ID. Can be NULL when credit pools are not used (e.g., per-VC credit limit).
RESERVED_CREDIT_SIZE	sai_uint64_t	MANDATORY_ON_CREATE CREATE_AND_SET	NA	Reserved credits for the VC. Represents dedicated buffer allocation.
THRESHOLD_MODE	sai_cbfc_credit_profile_threshold_mode_t	CREATE_ONLY	SAI_CBFC_CREDIT_PROFILE_THRESHOLD_MODE_NONE	Shared threshold mode for the credit profile. Determines how shared buffer usage is limited.
SHARED_DYNAMIC_TH	sai_int8_t	MANDATORY_ON_CREATE CREATE_AND_SET; @condition	NA	Dynamic threshold for shared usage. Set to 2^n of available credit. Valid only if THRESHOLD_MODE=DYNAMIC.

SAI_CBFC_CREDIT_PROFILE_ATTR	Type	Flags	Default	Description
		THRESHOLD_MODE == DYNAMIC		
SHARED_STATIC_TH	sai_uint64_t	MANDATORY_ON_CREATE CREATE_AND_SET; @condition THRESHOLD_MODE == STATIC	0	Static threshold for shared usage in credits. Zero means no limit. Valid only if THRESHOLD_MODE=STATIC.

Table 2-3 - CBFC Credit Profile Threshold Mode

typedef sai_cbfc_credit_profile_threshold_mode_t	Description
SAI_CBFC_CREDIT_PROFILE_THRESHOLD_MODE_NONE	No access to shared credits
SAI_CBFC_CREDIT_PROFILE_THRESHOLD_MODE_STATIC	Static maximum
SAI_CBFC_CREDIT_PROFILE_THRESHOLD_MODE_DYNAMIC	Dynamic maximum (relative)

2.4 CBFC Virtual Channel

The CBFC virtual channel object is a per-port object to enable/disable CBFC per VC, configure the credit management using credit profiles, and provide CBFC statistics. Virtual channel attributes are listed in Table 2-4.

Table 2-4 - Attributes of the New SAI Virtual Channel Object Type

SAI_VIRTUAL_CHANNEL_ATTR	Type	Flags	Default	Description
PORT	sai_object_id_t	MANDATORY_ON_CREATE CREATE_ONLY KEY	NA	Port ID of the port to which the VC belongs
INDEX	sai_uint8_t	MANDATORY_ON_CREATE CREATE_ONLY KEY	NA	Virtual channel index Range 0 to 31
CBFC_RECEIVER_NATIVE_CREDIT_LIMIT	sai_uint32_t	READ_ONLY	NA	Get receiver credit limit for this VC. If 0, receiver sets port credit limit. Up to $2^{20}-1$.
CBFC_SENDER_CREDIT_PROFILE	sai_object_id_t	CREATE_AND_SET	NULL	CBFC credit profile pointer. Default no profile; i.e., best-effort VC
CBFC_RECEIVER_ENABLE	bool	CREATE_AND_SET	False	Enable lossless CBFC receiver on this VC
CBFC_SENDER_ENABLE	bool	CREATE_AND_SET	False	Enable lossless CBFC sender on this VC

2.5 New Virtual Channel Statistics

Table 2-5 shows the new virtual channel statistics for CBFC.

Table 2-5 - Virtual Channel Statistics

SAI_VIRTUAL_CHANNEL_STAT	Description
SENDER_CREDITS_USED	Get the number of sender credits used.
SENDER_CREDITS_CONSUMED	Get the number of sender credits consumed.
SENDER_CREDITS_FREED	Get the number of sender credits freed.
SENDER_CREDITS_USED_WATERMARK	Get the watermark for the number of sender credits used.
RECEIVER_CREDITS_CONSUMED	Get the number of receiver credits consumed.
RECEIVER_CREDITS_FREED	Get receiver credits freed

2.6 QoS Map Updates

Mappings from {VLAN.PCP, VLAN.DEI} ->TC, TC->VC and queue->VC are added as described below.

Three new QoS map types added to *sai_qos_map_type_t* in *saiqosmap.h*

Table 2-6 - New QoS Map Types

QoS Map Type	Description
SAI_QOS_MAP_TYPE_DOT1P_DEI_TO_TC	QoS map to map packet {dot1p+DEI} to TC
SAI_QOS_MAP_TYPE_TC_TO_VC	QoS map to map traffic class to VC
SAI_QOS_MAP_TYPE_QUEUE_TO_VC	QoS map to set queue to VC

The existing QoS map parameter structure, *sai_qos_map_params_t* in *saitypes.h*, is extended to have the fields shown below to support the new QoS map types shown in Table 2-6.

```
/** DEI used in SAI_QOS_MAP_TYPE_DOT1P_DEI_TO_TC */
sai_uint8_t dei;
/** Virtual Channel */
sai_uint8_t vc;
```

2.7 SAI Port Updates

New attributes are added to the existing SAI port object to support CBFC.

Table 2-7 - New Attributes of the existing SAI PORT object type

SAI_PORT_ATTR	Type	Flags	Default	Description
CBFC_RECEIVER_NATIVE_CREDIT_SIZE	sai_uint16_t	READ_ONLY	NA	Get CBFC native credit size in bytes. Receiver's input buffer is typically divided into cells, and the native credit size is this cell size (in bytes).
CBFC_RECEIVER_NATIVE_PACKET_OVERHEAD	sai_int16_t	READ_ONLY	NA	Get CBFC receiver native packet overhead in bytes.
CBFC_RECEIVER_NATIVE_TOTAL_CREDITS	sai_uint32_t	READ_ONLY	NA	Get CBFC receiver total port credits in credits. If value is 0, receiver sets per-VC credit limits.
CBFC_RECEIVER_CREDIT_SIZE	sai_uint16_t	CREATE_AND_SET	0	Set CBFC receiver credit size in bytes. If value is 0, CBFC_RECEIVER_NATIVE_CREDIT_SIZE will be used.
CBFC_RECEIVER_PACKET_OVERHEAD	sai_int16_t	CREATE_AND_SET	128	Set CBFC receiver packet overhead in bytes. Range: -16 B to 127 B. If value is 128 B, CBFC_RECEIVER_NATIVE_PACKET_OVERHEAD will be used.
CBFC_SENDER_SUPPORTED_CREDIT_SIZE	sai_u16_list_t	READ_ONLY	NA	Get CBFC-supported credit size in bytes.
CBFC_SENDER_CREDIT_SIZE	sai_uint16_t	CREATE_AND_SET	128	Set CBFC sender credit size in bytes. To maximize utilization of the receiver's input buffer, set to largest value in CBFC_SENDER_SUPPORTED_CREDIT_SIZE that does not exceed CBFC_RECEIVER_NATIVE_CREDIT_SIZE.
CBFC_SENDER_PACKET_OVERHEAD	sai_int16_t	CREATE_AND_SET	0	Set CBFC sender packet overhead in bytes. Range: -16 B to 127 B.
CBFC_SENDER_CREDIT_LIMIT	sai_uint64_t	CREATE_AND_SET	0	Set CBFC sender port credit limit. Range: 0 to (2^20 - 1).
CBFC_CC_MESSAGE_TIMER	sai_uint32_t	CREATE_AND_SET	256	Set CBFC CC_Update message timer in microseconds. Range: 1 µs to 25,000 µs.

SAI_PORT_ATTR	Type	Flags	Default	Description
CBFC_CF_MIN_SPACING	sai_uint32_t	CREATE_AND_SET	6400	Set the minimum space between CBFC CF_Update messages in bytes. Must be >=800 B.
CTLOS_MIN_SPACING	sai_uint32_t	CREATE_AND_SET	6400	Set the minimum space in bytes between CtIOS messages like CBFC CF_Update, LLR ACK, etc. Must be >=800 B.
CBFC_CF_MAX_SPACING	sai_uint32_t	CREATE_AND_SET	262144	Set the maximum space between CBFC CF_Update messages in bytes. Range: 16 kB to 1 MB with minimum granularity of 16 kB.
QOS_QUEUE_TO_VC_MAP	sai_object_id_t	CREATE_AND_SET	SAI_NULL_OBJECT_ID	Enable queue to VC MAP on a port. QoS map of type SAI_QOS_MAP_TYPE_QUEUE_TO_VC.
QOS_TC_TO_VC_MAP	sai_object_id_t	CREATE_AND_SET	SAI_NULL_OBJECT_ID	Enable TC to VC MAP on port. QoS map of type SAI_QOS_MAP_TYPE_TC_TO_VC.
QOS_DOT1P_DEI_TO_TC_MAP	sai_object_id_t	CREATE_AND_SET	SAI_NULL_OBJECT_ID	Enable {dot1p+DEI} -> TC MAP on a port. Cannot co-exist with QOS_DOT1P_TO_TC_MAP.
QOS_VIRTUAL_CHANNEL_LIST	sai_object_list_t	READ_ONLY	NA	List of virtual channels for the port.
CBFC_CREDIT_POOL_LIST	sai_object_list_t	READ_ONLY	NA	List of CBFC credit pools for the port.

2.8 New CBFC Port Statistics

Table 2-8 show the new port statistics for CBFC.

Table 2-8 - Port CBFC Statistics

SAI_PORT_STAT	Description
CBFC_SENDER_CREDITS_USED	The number of sender credits used.
CBFC_SENDER_CREDITS_USED_WATERMARK	The watermark of the number of sender credits used.
CBFC_NUM_CC_UPDATE_MESSAGES_TX	The number of credits consumed update messages transmitted.
CBFC_NUM_CF_UPDATE_MESSAGES_TX	The number of credits freed update messages transmitted.
CBFC_NUM_CC_UPDATE_MESSAGES_RX	The number of credits consumed update messages received.
CBFC_NUM_CF_UPDATE_MESSAGES_RX	The number of credits freed update messages received.

2.9 SAI Queue Updates

Table 2-9 - New Attributes of the existing SAI Queue object type

SAI_QUEUE_ATTR	Type	Flags	Default	Description
PKT_DEQUEUE_ENABLE	bool	CREATE_AND_SET	True	Enable/disable packet transmission of a queue. When packet transmission is disabled on a queue, packets are still subject to regular ingress and egress admission control to determine the actions on a packet.
PKT_ENQUEUE_ENABLE	bool	CREATE_AND_SET	True	Enable/disable packet enqueue to a queue

2.10 Capability Query

2.10.1 Feature Capability

No new APIs are needed for a feature capability query. The existing attribute capability query API can be used to identify CBFC capability.

```
sai_attr_capability_t attr_cap[1];
sai_query_attribute_capability(switch_id, SAI_OBJECT_TYPE_VIRTUAL_CHANNEL,
SAI_VIRTUAL_CHANNEL_ATTR_PORT, attr_cap);
```

```

if (!attr_cap.create_implemented) {

    return -1; // CBFC is not supported
}

```

2.10.2 Statistics Capability Query

The existing statistics capability query can be used to identify supported port CBFC statistics.

```

sai_stat_capability_list_t  sai_port_stats_capability;
sai_port_stats_capability.count = SAI_PORT_STAT_END;
sai_stat_capability_t stats[SAI_PORT_STAT_END] = {};
sai_port_stats_capability.list = stats;

sai_query_stats_capability(switch_id, SAI_OBJECT_TYPE_PORT,
sai_port_stats_capability);

/* Fetching the supported CBFC port statistics */

sai_port_stat_t supported_stat_ids[SAI_PORT_STAT_END] = {0};

for (int i = 0; i < sai_port_stats_capability.count; i++) {
    switch (sai_port_stats_capability.list[i].stat_enum) {
        case SAI_PORT_STAT_CBFC_SENDER_CREDITS_USED:
            :
            case SAI_PORT_STAT_CBFC_NUM_CF_UPDATE_MESSAGES_RX:
                if (sai_port_stats_capability.list[i].stat_modes &
SAI_STAT_MODE_READ) {
                    supported_stat_ids[j++] =
sai_port_stats_capability.list[i].stat_enum;
                }
                break;
            default:
                break;
        }
    }
}

sai_get_port_attribute_fn(sai_port_id, switch_id, j, supported_stat_ids,
counters);

```

3 Normative References

- [1] <https://github.com/opencomputeproject/SAI/pull/2263>
- [2] Chapter 5.2 of <https://ultraethernet.org/wp-content/uploads/sites/20/2025/06/UE-Specification-6.11.25.pdf>