User’ s Manual

TSNA (ETHA)
(R-Switch-3 TSN Ethernet Agent)

Document number：0.50
Date of issue：2025/02/18

- This document and contents shall be strictly managed for all document users and used for specific/limited purpose only, issued under NDA between Renesas Electronics and the company of this document holders
- Under development
- Preliminary document
  Specification in this document are tentative and subject to change
- Renesas Electronics Confidential - issued under NDA

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published on Renesas Electronics home page.

ルネサス エレクトロニクス
www.renesas.com

Notice

1.Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.

2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.

3.Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of information of product data, diagrams, tables, programs, algorithms or application circuit example etc. described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

4.You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.

5.Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

“Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

“High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anticrime systems; and safety equipment etc.

Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.

6.You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.

7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.

8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.

10. Please note that Renesas Electronics shall not be responsible for any damage if products are not used as per the various conditions stated in this manual, due to resale by customer etc.

11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.

Note 1. “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

Note 2.“Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(2012.4)

General precautions for handling of product

The following notes are applicable to entire CBIC with CPU core. For detailed usage notes, refer to the relevant sections of the manual. If the description under General precautions and in the body of the manual differs from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

- The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flow internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

The state of the product is undefined at the moment when power is supplied.

- The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses Access to reserved addresses is prohibited.

- The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Regarding Clock

After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized. When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

Table of Contents

1. Overview

The TSN agent consists of an agent interface module to allowing communication within the Rswitch. It handles the data exchange between the Rswitch and an Ethernet PHY.

1.1 Features

TSN agent Features are described Table 1-1.

Table 1-1 TSN agent Feature List

Function		Details
Interfaces	Clock/Reset interface	Clock/Reset interface [TOP]
	Switch mode	Interface to receive the switch mode [FWD]
	Pause interface (**PAUSE**)	Interface to pause TX queues [COMA]
	xMII	MII, GMII, USXGMII 10Mbps, 100Mbps, 1Gbps, 2.5Gbps, 5Gbps,10Gbps [RMAC]
	SFR interface	Interface to access TSNA(ETHA) SFRs [COMA]
	Interrupts	TSNA interrupt to CPU
	Fabric interface	Interface to communicate with the data, TAG and pointer RAM [FAB].
	Descriptor bus	Interface to receive frame information to send them to CPU [FWD].
	L2/L3 update bus	Interface to fetch the frame routing information [FWD].
	BP Request	Interface to receive BPs to store frames in the data, TAG and pointer RAM [COMA].
	BP Release	Interface to release the BPs that has been used [COMA].
	TX Timestamp Capture interface	Interface to send TX timestamps to CPU [RMAC].
	RAM interfaces	Interface to communication with RAMs [TOP]
Data provision	Ethernet Frames	Corresponds to the IEEE 802.3, 802.1Q and 802.1CB standards [802.3] [802.1Q] [802.1CB].
	Descriptors	Local Ethernet Descriptor (TSNA to Forwarding engine) Forwarding descriptor (Forwarding engine to TSNA)

1.2 TSN agent block diagram

Fig 1.1 shows TSN agent block diagram.

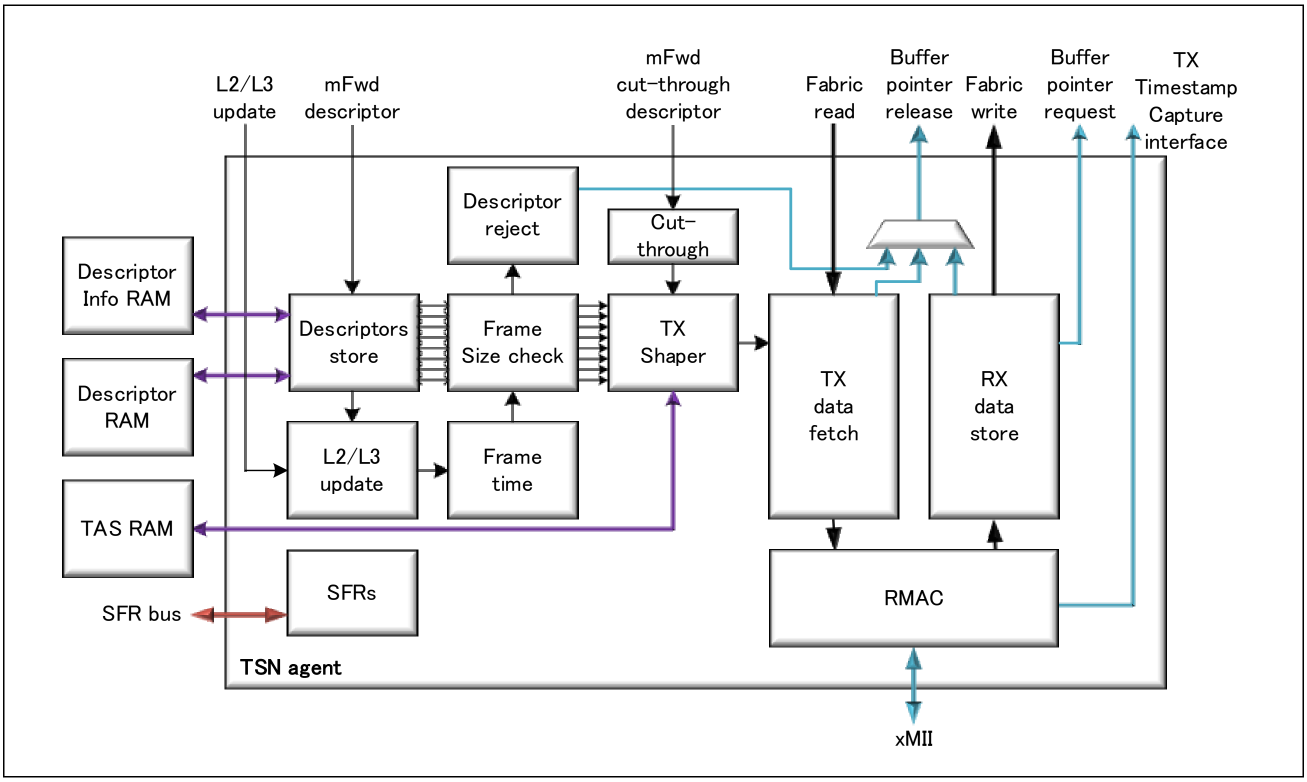


Fig 1.1: TSN agent Block diagram

Table 1-2 TSN agent Functional Blocks

Block name	Function
RX data store	Transmits frames from RMAC to the local RAM
TX data fetch	Transmits frame from local RAM to RMAC
Cut-through	Receives cut-through descriptors from the forwarding engine
Descriptor store	Receives the descriptors from forwarding engine and controls the priority between descriptors
L2/L3 update	Fetches L2/L3 update information
Frame time	Calculate the frame time for TAS scheduling
Frame size check	Check the frame size
Descriptor reject	Reject descriptors
TX shaper	Shapes the traffic on TX side
RMAC	Mac interface [RMAC]
SFRs	TSNA SFRs

2. Parameter list

TSN agent global parameter list is shown in Table 2-1.

TSN agent local parameter list is shown in Table 2-2

Table 2-1 Global parameter list

Parameter Name	**RSW3 Values**	**Explanation**
SFRs		
PADDR_NBR_TSNA	8192/4	Number of addresses used by TSNA SFRs
My port number		
MY_PORT_N	0-12 for TSNA0-12	My port number Refer to Fabric specification to know an agent port number [FAB].
Port Number		
PORT_TSNA_N	13	TSN Agent number [TSNA]
PORT_GWCA_N	2	CPU Agent number [GWCA]
Local RAM		
LCL_RAM_SZ	1024	Local RAM size in Kbytes
LCL_RAM_BSZ	128	Local RAM block size (A pointer will always link to a LOCAL_RAM_BSZ byte block size in the local RAM)
LCL_RAM_DW_FAB	512	Local RAM data width (on the mfab IF)
Frame		
FRM_TPL_W	16	Frame TPL (Total payload length) width
FRM_PRIO_N	8	Priority number handled by GWCA
FRM_VCTRL_W	3	Frame VLAN control width
Forwarding		
LTH_RRULE_N	2048	Layer 3 rule number
gPTP timer		
PTP_TN	2	gPTP timer number connected to the switch
Counter		
COUNT_LOW_W	32	Low size counter width
COUNT_MED_W	32	Medium size counter width
COUNT_HIGH_W	64	High sizecounter width
Flow Control		
PAS_LVL_N	2	Pause level
Cut-through		

Parameter Name	RSW3 Values	Explanation
CT_CRULE_N	8	Cut-through rule number
CT_DESCR_N	8	Cut-through descriptor number
Descriptor RAM		
DES_RAM_DP	2048	Descriptor RAM depth
TAS RAM		
TAS_RAM_DP	256	TAS RAM depth

Table 2-2 Local parameter list

Parameter Name	RSW3 Values	Explanation
SFRs		
PADDR_NBR_TSNA_W	11	Number of addresses used by TSNA SFRs bus width
Port Number		
PORT_N	15	Port number on the switch
PORT_W	4	Port number on the switch bus width
PORT_TIME_N	13	Number of time critical ports on the switch (plugged to the Fabric time arbiter [FAB])
PORT_TIME_W	2	Number of time critical ports on the switch bus width
Local RAM		
LCL_PTR_N	8192	Pointer number to address local RAM
LCL_PTR_W	13	Pointer width
LCL_RAM_AW	16	Local RAM address width
LCL_RAM_DW_FAB	512	Local RAM data width (on the mfab IF)
LCL_RAM_AW_FAB	14	Local RAM address width (on the mfab IF)
LCL_DATA_SIZE_FAB	6	Local RAM data size width (on the mfab IF)
Forwarding		
LTH_RRULE_W	11	Layer 3 table rule number bus width
Frame		
FRM_TIME_W	26	Frame time in ns bus width
FRM_PRIO_W	3	Priority width
FRM_MTN_W	5	Frame multicast number width
FRM_TPRIO_N	9	Number of priorities including cut-through
FRM_TPRIO_W	4	Number of priorities including cut-through bus width
gPTP timer		
PTP_TN_W	1	gPTP Timer number width
Cut-through		
CT_CRULE_W	3	Cut-through rule number bus width
CT_DESCR_W1	4	Cut-through descriptor number +1 bus width
CT_MTN_N	18	Cut-through multicast number
CT_MTN_W	4	Cut-through multicast number bus width

Parameter Name	**RSW3** **Values**	**Explanation**
Descriptor RAM		
DES_RAM_AW	11	Descriptor RAM address width
DES_RAM_AW1	12	Descriptor RAM address width
DES_RAM_DW	72	Descriptor RAM Data Width
TAS RAM		
TAS_RAM_AW	8	Timestamp RAM address width

3. Register

3.1 Register attributes

The register attribute defines what kind of access a register supports. Per one register, there are always two attributes, a register access attribute which define what kind of accesses a register supports and, a register security attribute which define what accesses can perform the unsecure APB [APB] in the register access attribute depending on the security setting in security registers.

“Representation of register access attributes “ describes register access attributes and “Representation of register security attributes” describes register security attributes. Attributes are given to a register field in Register detailed explanation section by specifying the attribute symbols in the R/W-P column.

Table 3-1: Register access attributes

Symbol	Meaning	Impact on accesses	
		Write access	Read access
RW	Read write	Write value is written	Written value is read
R!=W	Read different than write	Write access happens	Read value differs from written value
R	Read only	Write value is ignored	Read access happens
R0	Only Read 0	Write value is ignored	Always read '0'
R1	Only Read 1	Write value is ignored	Always read '1'
R0W	Read 0 write	Write access happens	Always read as '0'
R1W	Read 1 write	Write access happens	Always read as '1'
RC	Read clear	Write value is ignored	Read access happens Read access clears the register

Table 3-2: Register security attributes (For R-Car products only)

Symbol	Meaning	Impact on accesses	
		Write access	Read access
U	Unprotected	Write access happens for unsecure APB	Read access happens for unsecure APB
P	Protected	A security register should be set to authorize write access by the unsecure APB.	A security register should be set to authorize read access by the unsecure APB
RU	Read-Unprotected	Write value ignored for unsecure APB	Read access happens for unsecure APB
RP	Read protected	Write value ignored for unsecure APB	A security register should be set to authorize read access by the unsecure APB
D	Duplicated	Write access happens for unsecure APB to a duplicated and independent register	Read access happens for unsecure APB to a duplicated and independent register
F	Forbidden	Write value ignored for unsecure APB	Read value ignored for unsecure APB
S	Switch	A security register should be set to authorize write access by the unsecure APB. A security register should be set to unauthorize write access by the secure APB.	A security register should be set to authorize read access by the unsecure APB

3.2 Register list

The TSNA register list is described in Table 3-3. TARO (TSN Agent Register Offset) indicates base address of address space allocated to TSNA by the system. TSNA uses only the 13-lower address bits. All registers representations are done with the default values of the section 2. If the TSNA is not use with default parameters, it should be taken in account by the user while reading the SFR representation.

Access Mode:

- Any: Register can be accessed in any mode.
- D: Register can be accessed in DISABLE mode
- C: Register can be accessed in CONFIG mode
- O: Register can be accessed in OPERATION mode

Notes:

- All registers can be read in any mode.
- A register can have two addresses. The address preceded by “E:” correspond to an emulation address which allows to read a register without modifying its content.

Table 3-3: List of TSNA registers

Offset/Address	Register name	Abbreviation	Write Access Mode
TARO + 0000H	Ethernet Agent Mode Configuration	EAMC	Any
TARO + 0004H	Ethernet Agent Mode Status	EAMS	- -
TARO + 0008H	Ethernet Agent TX Descriptor RAM Configuration	EATDRC	C
TARO + 0010H	Ethernet Agent IPV Remapping Configuration	EAIRC	C
TARO + 0014H	Ethernet Agent TX Descriptor Queue Security Configuration	EATDQSC	C, O
TARO + 0018H	Ethernet Agent TX Descriptor Queue Configuration	EATDQC	O
TARO + 001CH	Ethernet Agent TX Descriptor Queue Arbitration Configuration	EATDQAC	C
TARO + 0020H	Ethernet Agent TX Pre-Emption Configuration	EATPEC	C
TARO + 0040H + 4H*q	Ethernet Agent Transmission Maximum Frame Size Configuration q (q=0.. FRM_PRIO_N-1)	EATMFSCq	C, O
TARO + 0060H + 4H*q	Ethernet Agent Transmission Descriptor Queue Depth Configuration q (q=0.. FRM_PRIO_N-1)	EATDQDCq	C
TARO + 0080H + 4H*q	Ethernet Agent Transmission Descriptor Queue q Monitoring (q=0.. FRM_PRIO_N-1)	EATDQMq	- -
TARO + 00A0H + 4H*q E: TARO + 00C0H + 4H*q	Ethernet Agent Transmission Descriptor Queue q Max Level Monitoring (q=0.. FRM_PRIO_N-1)	EATDQMLMq	- -
TARO + 0100H	Ethernet Agent Cut-Through Queue Configuration	EACTQC	C
TARO + 0104H	Ethernet Agent Cut-Through Descriptor Queue Depth Configuration	EACTDQDC	C
TARO + 0108H	Ethernet Agent Cut-Through Descriptor Queue Monitoring	EACTDQM	- -
TARO + 010CH E: TARO + 0110H	Ethernet Agent Cut-Through Descriptor Queue Max Level Monitoring	EACTDQMLM	- -
TARO + 0130H	Ethernet Agent VLAN Control Configuration	EAVCC	C
TARO + 0134H	Ethernet Agent VLAN TAG Configuration	EAVTC	C
TARO + 0138H	Ethernet Agent Reception TAG Filtering Configuration	EARTFC	C
TARO + 013CH	Ethernet Agent ChecKSum Configuration	EACKSC	C
TARO + 0200H	Ethernet Agent CBS Admin Enable Configuration	EACAEC	C, O
TARO + 0204H	Ethernet Agent CBS Configuration	EACC	C, O
TARO + 0220H + 4H*q	Ethernet Agent CBS Admin Increment Value Configuration q (q=0.. FRM_PRIO_N-1)	EACAIVCq	C, O
TARO + 0240H + 4H*q	Ethernet Agent CBS Admin Upper Limit Configuration q (q=0.. FRM_PRIO_N-1)	EACAULCq	C, O

Offset/Address	Register name	Abbreviation	Write Access Mode
TARO + 0260H	Ethernet Agent CBS Oper Enable Monitoring	EACOEM	- -
TARO + 0280H + 4H*q	Ethernet Agent CBS Oper Increment Value Monitoring q (q=0.. FRM_PRIO_N-1)	EACOIVMq	- -
TARO + 02A0H + 4H*q	Ethernet Agent CBS Oper Upper Limit Monitoring q (q=0.. FRM_PRIO_N-1)	EACOULMq	- -
TARO + 02C0H	Ethernet Agent CBS Gate State Monitoring	EACGSM	- -
TARO + 0300H	Ethernet Agent TAS Configuration	EATASC	C, O
TARO + 0304H	Ethernet Agent TAS Initial Gate State Configuration	EATASIGSC	C, O
TARO + 0320H + 4H*q	Ethernet Agent TAS Entry Number Configuration q (q=0..FRM_PRIO_N)	EATASENCq	C, O
TARO + 0340H	Ethernet Agent TAS Cut-Through Entry Number Configuration	EATASCTENC	C, O
TARO + 0360H + 4H*q	Ethernet Agent TAS Entry Number Monitoring q (q=0..FRM_PRIO_N)	EATASENMq	C, O
TARO + 0380H	Ethernet Agent TAS Cut-Through Entry Number Monitoring	EATASCTENM	C, O
TARO + 03A0H	Ethernet Agent TAS Cycle Start Time Configuration 0	EATASCSTC0	C, O
TARO + 03A4H	Ethernet Agent TAS Cycle Start Time Configuration 1	EATASCSTC1	C, O
TARO + 03A8H	Ethernet Agent TAS Cycle Start Time Monitoring 0	EATASCSTM0	C, O
TARO + 03ACH	Ethernet Agent TAS Cycle Start Time Monitoring 1	EATASCSTM1	C, O
TARO + 03B0H	Ethernet Agent TAS Cycle Time Configuration	EATASCTC	C, O
TARO + 03B4H	Ethernet Agent TAS Cycle Time Monitoring	EATASCTM	C, O
TARO + 03C0H	Ethernet Agent TAS Gate Learn 0	EATASGL0	C, O
TARO + 03C4H	Ethernet Agent TAS Gate Learn 1	EATASGL1	C, O
TARO + 03C8H	Ethernet Agent TAS Gate Learn Result	EATASGLR	- -
TARO + 03D0H	Ethernet Agent TAS Gate Read	EATASGR	C, O
TARO + 03D4H	Ethernet Agent TAS Gate Read Result	EATASGRR	- -
TARO + 03E0H	Ethernet Agent TAS Hardware Calibration Configuration	EATASHCC	C
TARO + 03E4H	Ethernet Agent TAS RAM Initialization Register Monitoring	EATASRIRM	C, O
TARO + 03E8H	Ethernet Agent TAS Status Monitoring	EATASSM	- -
TARO + 0400H E: TARO + 0480H	Ethernet Agent Switch Minimum Frame Size Error CouNter	EAUSMFSECN	- -
TARO + 0404H E: TARO + 0484H	Ethernet Agent TAG Filtering Error CouNter	EATFECN	- -
TARO + 0408H E: TARO + 0488H	Ethernet Agent Frame Size Error CouNter	EAFSECN	- -
TARO + 040CH E: TARO + 048CH	Ethernet Agent Descriptor Queue Overflow Error CouNter	EADQOECN	- -
TARO + 0410H E: TARO + 0490H	Ethernet Agent Descriptor Queue Security Error CouNter	EADQSECN	- -
TARO + 0414H E: TARO + 0494H	Ethernet Agent ChecKSum Error CouNter	EACKSECN	- -
TARO + 047CH E: TARO + 04FCH	Ethernet Agent Lost Descriptor CouNter	EALDCN	- -
TARO + 0500H	Ethernet Agent Error Interrupt Status 0	EAEIS0	C, O, D
TARO + 0504H	Ethernet Agent Error Interrupt Enable 0	EAEIE0	C, O, D
TARO + 0508H	Ethernet Agent Error Interrupt Disable 0	EAEID0	C, O, D
TARO + 0510H	Ethernet Agent Error Interrupt Status 1	EAEIS1	C, O, D
TARO + 0514H	Ethernet Agent Error Interrupt Enable 1	EAEIE1	C, O, D
TARO + 0518H	Ethernet Agent Error Interrupt Disable 1	EAEID1	C, O, D
TARO + 0520H	Ethernet Agent Error Interrupt Status 2	EAEIS2	C, O, D
TARO + 0524H	Ethernet Agent Error Interrupt Enable 2	EAEIE2	C, O, D

Offset/Address	Register name	Abbreviation	Write Access Mode
TARO + 0528H	Ethernet Agent Error Interrupt Disable 2	EAEID2	C, O, D
TARO + 0580H	Ethernet Agent Security Configuration Register	EASCR	C, O
TARO + 0600H	Ethernet Agent Ingress C-TAG DEI 0 Remapping Configuration	EAICD0RC	C, O, D
TARO + 0604H	Ethernet Agent Ingress C-TAG DEI 1 Remapping Configuration	EAICD1RC	C, O, D
TARO + 0608H	Ethernet Agent Ingress S-TAG DEI 0 Remapping Configuration	EAISD0RC	C, O, D
TARO + 060CH	Ethernet Agent Ingress S-TAG DEI 1 Remapping Configuration	EAISD1RC	C, O, D
TARO + 0610H	Ethernet Agent Egress C-TAG DEI 0 Remapping Configuration	EAECD0RC	C, O, D
TARO + 0614H	Ethernet Agent Egress C-TAG DEI 1 Remapping Configuration	EAECD1RC	C, O, D
TARO + 0618H	Ethernet Agent Egress S-TAG DEI 0 Remapping Configuration	EAESD0RC	C, O, D
TARO + 061CH	Ethernet Agent Egress S-TAG DEI 1 Remapping Configuration	EAESD1RC	C, O, D
TARO + 0700H E: TARO + 0780H	Ethernet Agent Received Frame CouNter E-frame per Octets 0	EARFCNEO0	- -
TARO + 0704H E: TARO + 0784H	Ethernet Agent Received Frame CouNter E-frame per Octets 1	EARFCNEO1	- -
TARO + 0708H E: TARO + 0788H	Ethernet Agent Received Frame CouNter E-frame per Octets 2	EARFCNEO2	- -
TARO + 070CH E: TARO + 078CH	Ethernet Agent Received Frame CouNter E-frame per Octets 3	EARFCNEO3	- -
TARO + 0710H E: TARO + 0790H	Ethernet Agent Received Frame CouNter E-frame per Octets 4	EARFCNEO4	- -
TARO + 0714H E: TARO + 0794H	Ethernet Agent Received Frame CouNter E-frame per Octets 5	EARFCNEO5	- -
TARO + 0718H E: TARO + 0798H	Ethernet Agent Received Frame CouNter E-frame per Octets 6	EARFCNEO6	- -
TARO + 071CH E: TARO + 079CH	Ethernet Agent Received Frame CouNter P-frame per Octets 0	EARFCNPO0	- -
TARO + 0720H E: TARO + 07A0H	Ethernet Agent Received Frame CouNter P-frame per Octets 1	EARFCNPO1	- -
TARO + 0724H E: TARO + 07A4H	Ethernet Agent Received Frame CouNter P-frame per Octets 2	EARFCNPO2	- -
TARO + 0728H E: TARO + 07A8H	Ethernet Agent Received Frame CouNter P-frame per Octets 3	EARFCNPO3	- -
TARO + 072CH E: TARO + 07ACH	Ethernet Agent Received Frame CouNter P-frame per Octets 4	EARFCNPO4	- -
TARO + 0730H E: TARO + 07B0H	Ethernet Agent Received Frame CouNter P-frame per Octets 5	EARFCNPO5	- -
TARO + 0734H E: TARO + 07B4H	Ethernet Agent Received Frame CouNter P-frame per Octets 6	EARFCNPO6	- -
TARO + 0740H + 4H*q E: TARO + 07C0H + 4H*q	Ethernet Agent Descriptor Queue Overflow Error CouNter Priority q (q=0..FRM_PRIO_N-1)	EADQOECNPq	- -
TARO + 0760H E: TARO + 07E0H	Ethernet Agent Descriptor Queue Overflow Error CouNter Cut Through	EADQOECNCT	- -
TARO + 1000H	RMAC registers. Address value corresponds to RMAC Register Offset RMRO [RMAC]	RMRO	- -

3.3 Register detailed explanation

This section describes SFR details.

3.3.1 TSNA Function registers

3.3.1.1 TSNA Mode function registers

(1) EAMC

Ethernet Agent Mode Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV														OPC[1:0]	

Bits	Bit name	RW-P	Initial value	Function description
31:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1:0	OPC	RW-P	01B	OPerating mode Command. Functions: - Used to set TSNA mode. Values: - 2’b00: Enter RESET mode - 2’b01: Enter DISABLE mode - 2’b10: Enter CONFIG mode - 2’b11: Enter OPERATION mode Restrictions : - HW: This register is not writable if its value is different than **EAMS.OPS**

(2) EAMS

Ethernet Agent Mode Status.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV														OPS[1:0]	

Bits	Bit name	RW-P	Initial value	Function description
31:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1:0	OPS	R-P	01B	OPerating mode Status. Functions: - Indicates the current TSNA mode. Values: - 2'b00: RESET mode - 2'b01: DISABLE mode - 2'b10: CONFIG mode - 2'b11: OPERATION mode Update conditions: - HW: Mode change completed. - HW: This register is changed at "Maximum frame communication time (exp: 64Kbyte / 100Mbps = 5.12ms)" from OPERATION to DISABLE. Notes: - In case there is no PHY TX/RX clock provided to [RMAC], the transition OPERATION to DISABLED may not be possible. Some products have a function to release this state. Writing 1'b1 MIOC.MIOC[0] [RMAC]. This is a debug (not supported) function for checking the cause.

3.3.1.2 Transmission function registers

(1) EATDRC

Ethernet Agent Tx Descriptor RAM Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV															TDRM

Bits	Bit name	RW-P	Initial value	Function description
31: 1	RSV	R0-P	0H	Reserved area. On read, 0 will be returned
0	TDRM	RW-P	0B	Tx Descriptor RAM Mode Values : - 1’b0: Tx Descriptor RAM Separation Mode - 1’b1: Tx Descriptor RAM Share Mode Functions: - "Separation Mode”: Tx Descriptor RAM is separated for each queues by values of EATDQDCq.DQDq. “Share Mode”: All area of Descriptor RAM is shared by all queues. - It isn’t applicable for the Cut-through descriptor queue depth. (Cut-through descriptor queue exist standalone.)

(2) EAIRC

Ethernet Agent IPV Remapping Configuration [802.1Q]

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV	IPVR7 [FRM_PRIO_W-1:0]			RSV	IPVR6 [FRM_PRIO_W-1:0]			RSV	IPVR5 [FRM_PRIO_W-1:0]			RSV	IPVR4 [FRM_PRIO_W-1:0]		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV	IPVR3 [FRM_PRIO_W-1:0]			RSV	IPVR2 [FRM_PRIO_W-1:0]			RSV	IPVR1 [FRM_PRIO_W-1:0]			RSV	IPVR0 [FRM_PRIO_W-1:0]		

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
4*(i+1)-1: FRM_PRIO_W+4*i i=0..7	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_PRIO_W+4*i-1: 4*i i=0..7	IPVRi	RW-P	i	IPV Remapping i Functions: - Configure to which descriptor queue descriptor received with IPV i will be stored (when a descriptor is received from forwarding engine with **FDESCR.IPV** [FWD] equal to i).

(3) EATDQSC

Ethernet Agent TX Descriptor Queue Security Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TDQDEL[15:0]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								TDQSL[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:16	TDQDEL	RW-RP	0H	TX Descriptor Queue DELay. Value: - 16'h0: No functions. - Any value: The setting **Minimum IFG time (ns)** will be guaranteed. **Minimum IFG time (ns)** = "(**EATDQSC.TDQDEL*** clk_period[ns] * 8) – **PHY_latency**" For example: If you want "min 100 PHY clock cycle IFG for MII 10Mbps, **Minimum IFG time (ns)** = 400(2.5MHz)*100ns = 40,000ns", **EATDQSC.TDQDEL** have to be "(40,000+**PHY_latency**)/(5*8)ns" (clk_period = 5ns) - There are **PHY_latency (ns)**. USXGMII 10Gbps = 25 agent clock cycle + 16 PHY clock cycle USXGMII 5Gbps = 60 agent clock cycle + 16 PHY clock cycle USXGMII 2.5Gbps = 131 agent clock cycle + 16 PHY clock cycle GMII 1Gbps = 347 agent clock cycle + 80 PHY clock cycle MII 100Mbps = 3,600 agent clock cycle + 162 PHY clock cycle MII 10Mbps = 36,000 agent clock cycle + 162 PHY clock cycle Restrictions: - SW: This function can be supported only E-Frame. (Not supported P-Frame). - SW: This function is not usable with Cut-through forwarding. - SW: To avoid violation of IFG setting, configure the TDQDEL within the allowed range (greater than 0 and less than the maximum IFG). - HW: This feature affects TAS. In other words, please extend the TAS OPEN period by the delay of this function. Otherwise, the TAS CLOSE period will be touched by the delay. Notes: - When increasing IFG, the upper limit is as follows. More configurations are possible, but not recommended. Max calculated IFG of XGMII_10G = 102.4ns Max calculated IFG of XGMII_5G = 204.8ns Max calculated IFG of XGMII_2.5G = 409.6ns Max calculated IFG of GMII = 1,024ns Max calculated IFG of MII_100mbps = 10,240ns Max calculated IFG of MII_10mbps = 102,400ns
31: FRM_PRIO_N	RSV	R0-U	0H	Reserved area. On read, 0 will be returned

i i=0.. FRM_PRIO_N-1	TDQSL	RW-RP	0H	TX Descriptor Queue Security Level i. Values: - 1’b0: Queue i unsecure - 1’b1: Queue i secure Functions: - When a queue is secured, an unsecure descriptor cannot enter it (when a descriptor is from forwarding engine with **FDESCR.SEC** [FWD] equal to 0).

(4) EATDQC

Ethernet Agent TX Descriptor Queue Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV								TDQP[FRM_PRIO_N-1:0]							
B15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RSV							TCTDQD	TDQD[FRM_PRIO_N-1:0]							

Bits	**Bit name**	**R/W-P**	**Initial value**	**Function description**
31: FRM_PRIO_N+16	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
i+16 i=0.. FRM_PRIO_N-1	TDQP	R!=W-P	0H	TX Descriptor Queue Pause i. Values: - 1’b0: Queue i active - 1’b1: Queue i paused Functions: - Avoid frames to be sent to the RMAC [RMAC] by stopping descriptor fetching from the descriptor RAM. Set conditions: - SW: Writing 1’b1 to this bit will set it. Clear conditions: - SW: Writing 1’b0 to this bit will clear it. - HW: Going out of OPERATION mode will clear this register (**EAMC.OPC** != 2’b11). Restrictions: - SW: Pausing a queue during a long time could result in switch overflow. In case of switch overflow [COMA] this register should be set to 1’b0. Cautions: - This register is used to stop a queue but forwarding engine [FWD] will not stop sending descriptor to the corresponding queue. In this case the queue might overflow. - When a queue is paused, the CBS module is still active, so the credit value continues to increase.
15: FRM_PRIO_N+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned

FRM_PRIO_N	TCTDQD	R!=W-F	0H	TX Cut-Through Descriptor Queue Disable. Values: - 1’b0: Cut-through queue enabled - 1’b1: Cut-through queue disabled Functions: - Avoid descriptor from forwarding engine [FWD] to enter the cut-through queue by de-asserting eha_ct_ready pin. The Forwarding Engine will reject the corresponding descriptor using the Common Agent reject bus [COMA]. Set conditions: - SW: Writing 1’b1 to this bit will set it. Clear conditions: - SW: Writing 1’b0 to this bit will clear it. - HW: Going out of OPERATION mode will clear this register (**EAMC.OPC** != 2’b11). Restriction: - HW: Since the frame stops midway, releasing the pointer may fail.
i i=0.. FRM_PRIO_N-1	TDQD	RW-P	0H	TX Descriptor Queue Disable i. Values: - 1’b0: Queue i enabled - 1’b1: Queue i disabled Functions: - Avoid descriptor from forwarding engine [FWD] to enter the corresponding queue. The Forwarding Engine will reject the corresponding descriptor. - If disabled queue data is not needed anymore, corresponding **EATMFSCq.MFSq** register can be set to 0 in order to avoid more frames to go to the RMAC. Set conditions: - SW: Writing 1’b1 to this bit will set it. Clear conditions: - SW: Writing 1’b0 to this bit will clear it. - HW: Going out of OPERATION mode will clear this register (**EAMC.OPC** != 2’b11).

(5) EATDQAC

Ethernet Agent TX Descriptor Queue Arbitration Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TDQA7[3:0]				TDQA6[3:0]				TDQA5[3:0]				TDQA4[3:0]			
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDQA3[3:0]				TDQA2[3:0]				TDQA1[3:0]				TDQA0[3:0]			

Bits	Bit name	RW-P	Initial value	Function description
[4*(i+1)-1:4*i] i=0..FRM_PRIO_N	TDQAi	RW-P	0H	TX Descriptor Queue Arbitration i. Values: - 4’b0: Queue i strict arbitration - Others: Queue i WRR arbitration Functions: - For more details, refer to GWCA specification document **GWRDQAC** register explanation [GWCA]. Restrictions: - SW: In hybrid arbitration mode, all the queues with an **EATDQAC.TDQAi** value different than 4’d0 should have a consecutive priority. - SW: In all arbitration mode except strict priority arbitration mode, at least two queues should have an **EATDQAC.TDQAi** different than 4’d0. - SW: WRR (and RR) arbitration is only possible between queues with the same type (e or p), refer to **EATPEC.TTQ** register explanation (i.e all queues with an **EATDQAC.TDQAi** value different than 4’d0 should have the same setting in **EATPEC.TTQ[i]** register). - SW: Only queues set to strict arbitration can use CBS functions. - SW: When using TAS, all queues not set to strict arbitration should be opened simultaneously.

(6) EATPEC

Ethernet Agent TX Pre-Emption Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV														AFS[1:0]	
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								TTQ[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
17:16	AFS	RW-P	0H	Additional Fragment Size Values: - 2’b00: 0 byte is added to minimum fragment size (minimum fragment size = 64 bytes). - 2’b01: 64 bytes are added to minimum fragment size (minimum fragment size = 128 bytes). - 2’b10: 128 bytes are added to minimum fragment size (minimum fragment size = 192 bytes). - 2’b11: 192 bytes are added to minimum fragment size (minimum fragment size = 256 bytes). Functions: - Used for preemption. It defines the minimum size of a fragment which is not the last fragment of a frame.
15: FRM_PRIO_N	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
i i=0.. FRM_PRIO_N-1	TTQ	RW-P	0H	Transmission Type Queue i. Values: - 1’b0: Queue i contains E-Frames - 1’b1: Queue i contains P-Frames Functions: - Refer to section 5.1.7.2 Restrictions: - If one of **EATPEC.TTQ** bits is set to 1’b1, RMAC is in XGMII mode (**MPIC.PIS** set to 3’b100 [RMAC]) and PCH is enabled (**MPCH.TXPCH_M** is set [RMAC]), PCH extension type should be set to 2’b10 (**MPCH.TXPCH_ETYPE** set to 2’b10)

(7) EATMFSCq (q=0.. FRM_PRIO_N-1)

Ethernet Agent Transmission Maximum Frame Size Configuration q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MFSq[FRM_TPL_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31: FRM_TPL_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_TPL_W-1:0	MFSq	RW-P	FFFFH	Maximum Frame Size q Functions: - Maximum frame size for descriptor queue q. All bigger frames will be rejected. Restrictions: - HW: The MFS size comparison takes the frame size on the PHY interface.

(8) EATDQDCq (q=0.. FRM_PRIO_N-1)

Ethernet Agent Transmission Descriptor Queue Depth Configuration q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV				DQDq[DES_RAM_AW1-1:0]											

Bits	Bit name	RW-P	Initial value	Function description
31: DES_RAM_AW1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
DES_RAM_AW1-1:0	DQDq	RW-P	DES_RAM_DP/FRM_PRIO_N	Descriptor Queue Depth q Functions : - Set the number of descriptors that can contain descriptor queue q. - For details, refer to GWCA specification document **GWRDQDCq** register explanation [GWCA]. Restrictions: - SW: When **EATDRC.TDRM** is 0, the sum of DQD fields should always be smaller or equal to DES_RAM_DP. When **EATDRC.TDRM** is 1, the maximum value for this field is DES_RAM_DP.

(9) EATDQMq (q=0.. FRM_PRIO_N-1)

Ethernet Agent Transmission Descriptor Queue q Monitoring.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV				DNQq[DES_RAM_AW1-1:0]											

Bits	Bit name	RW-P	Initial value	Function description
31: DES_RAM_AW1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
DES_RAM_AW1-1:0	DNQq	R-P	0H	Descriptor Number in Queue q. Functions: - These bits indicate the current number of descriptors stored in TX Descriptor Queue q. - All descriptor stored here will be rejected/dropped at the out of OPERATION. Increment conditions: - HW: Incremented by 1 when a descriptor is received from the forwarding engine [FWD] for the corresponding queue and, the queue is not full and there is no descriptor security error. Decrement conditions: - HW: Decremented by 1 when a descriptor is read by TSN Agent to send data to RMAC [RMAC] or to reject data.

(10) EATDQMLMq (q=0.. FRM_PRIO_N-1)

Ethernet Agent Transmission Descriptor Queue q Max Level Monitoring.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV				DMLQq[DES_RAM_AW1-1:0]											

Bits	Bit name	RW-P	Initial value	Function description
31: DES_RAM_AW1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
DES_RAM_AW1-1:0	DMLQq	RC-P	0H	Descriptor Max Level in Queue q. Functions: - These bits indicate the maximum number of descriptors that has been stored in TX Descriptor Queue q. Clear conditions: - HW: Being in RESET mode will clear this register (**EAMS.OPS** == 2'b00). - SW: By reading this register, it is cleared to **EATDQMq.DNQq.** Increment conditions: - HW: Increments to **EATDQMq.DNQq** value when smaller than **EATDQMq.DNQq.**

(11) EACTQC

Ethernet Agent Cut-Through Queue Configuration.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTQD[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:16	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
15:0	CTQD	RW-F	0H	Cut-Through Queue Delay (clk cycle number) Functions: - Set the number of clocks to wait for descriptor transmission. Restrictions: [RSW2.*] - SW: If PHY speed less than 2.5Gbps, this register should be set to 0. - SW: If PHY speed 2.5Gbps or more, this register should be set to “PHY clock 3 cycles time (for exp 2.5Gbps = 38.4ns)” or more. Restrictions: [RSW3.*] - SW: This register should be set follow value. CTQD[15:0] = (128[bit] / (PHY Speed[Mbps] / clk_frequency[MHz])) + PORT_TSNA_N Exp: (clk_frequency = 600MHz) PHY Speed 10Gbps : 0x7 + PORT_TSNA_N PHY Speed 5Gpbs : 0xF + PORT_TSNA_N PHY Speed 2.5Gbps : 0x1E + PORT_TSNA_N PHY Speed 1Gbps : 0x4C + PORT_TSNA_N PHY Speed 100Mbps: 0x300 + PORT_TSNA_N PHY Speed 10Mbps : 0x1E00 + PORT_TSNA_N

(12) EACTDQDC

Ethernet Agent Cut-Through Descriptor Queue Depth Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV												CTDQD[CT_DESCR_W1-1:0]			

Bits	Bit name	RW-P	Initial value	Function description
31: CT_DESCR_W1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
CT_DESCR_W1-1:0	CTDQD	RW-F	0H	Cut-Through Descriptor Queue Depth Functions: - Number of descriptors that can contain cut-through descriptor queue. Restrictions: - SW: This register maximum value is CT_DESCR_N.

(13) EACTDQM

Ethernet Agent Cut-Through Descriptor Queue Monitoring.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV												CTQDN[CT_DESCR_W1-1:0]			

Bits	Bit name	RW-P	Initial value	Function description
31: CT_DESCR_W1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
CT_DESCR_W1-1:0	CTQDN	R-F	0H	Cut-Though Queue Descriptor Number. Functions: - These bits indicate the current number of descriptors stored Cut-Through Descriptor Queue. - All cut-through descriptors stored here will not be rejected / dropped at the out-of-operation. But the frame being received by TSNA will be rejected / dropped, if TSNA transitioned to reset before the frame is completely received. - All Cut-Through descriptor stored here will not be rejected/dropped at the out of OPERATION. But will be rejected/dropped at the transitioned to RESET. Increment conditions: - HW: Incremented by 1 when a descriptor is received from the forwarding for the cut-through queue and the queue is not full. Decrement conditions: - HW: Decremented by 1 when a descriptor is read by TSN Agent to send data to RMAC [RMAC].

(14) EACTDQMLM

Ethernet Agent Cut-Through Descriptor Queue Max Level Monitoring.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV												CTDMLQ[CT_DESCR_W1-1:0]			

Bits	Bit name	RW-P	Initial value	Function description
31:CT_DESCR_W1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
CT_DESCR_W1-1:0	CTDMLQ	RC-F	0H	Cut-Through Descriptor Max Level in Queue. Functions: - These bits indicate the maximum number of descriptors that has been stored in Cut-Through Descriptor Queue. Clear conditions: - HW: Being in RESET mode will clear this register (**EAMS.OPS** == 2'b00). - SW: By reading this register, it is cleared to **EACTDQM.CTQDN.** Increment conditions: - HW: Increments to **EACTDQM.CTQDN** value when smaller than **EACTDQM.CTQDN.**

3.3.1.3 TAG function registers

(1) EAVCC

Ethernet Agent VLAN control configuration.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV													VEM[2:0]		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV		STDUM	STPUM	STVUM	CTDUM	CTPUM	CTVUM	RSV							VIM

Bits	Bit name	RW-P	Initial value	Function description
31:19	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
18:16	VEM	RW-P	0H	VLAN Egress Mode Values : - 3'b000: No VLAN mode - 3'b001: C-TAG VLAN mode - 3'b010: HW C-TAG VLAN mode - 3'b011: SC-TAG VLAN mode - 3'b100: HW SC-TAG VLAN mode
15:14	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
13	STDUM	RW-P	0H	S-TAG DEI Update Mask Values: - 1'b0: No function. - 1'b1: S-TAG DEI will not overwrite in port based VLAN mode. Additions are not suppurasse.
12	STPUM	RW-P	0H	S-TAG PCP Update Mask Values: - 1'b0: No function. - 1'b1: S-TAG PCP will not overwrite in port based VLAN mode.
11	STVUM	RW-P	0H	S-TAG VLAN Update Mask Values: - 1'b0: No function. - 1'b1: S-TAG VLAN will not overwrite in port based VLAN mode.
10	CTDUM	RW-P	0H	C-TAG DEI Update Mask Values: - 1'b0: No function. - 1'b1: C-TAG DEI will not overwrite in port based VLAN mode.
9	CTPUM	RW-P	0H	C-TAG PCP Update Mask Values: - 1'b0: No function. - 1'b1: C-TAG PCP will not overwrite in port based VLAN mode.
8	CTVUM	RW-P	0H	C-TAG VLAN Update Mask Values: - 1'b0: No function. - 1'b1: C-TAG VLAN will not overwrite in port based VLAN mode.
7:1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned

0	VIM	RW-P	0H	VLAN Ingress Mode - 1’b0: Incoming VLAN mode - 1’b1: Port based VLAN mode

(2) EAVTC

Ethernet Agent VLAN TAG configuration.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STD	STP[2:0]			STV[11:0]											
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTD	CTP[2:0]			CTV[11:0]											

Bits	Bit name	RW-P	Initial value	Function description
31	STD	RW-P	0H	S-TAG DEI Functions: - Set the S-TAG DEI values.
30:28	STP	RW-P	0H	S-TAG PCP Functions: - Set the S-TAG PCP values.
27:16	STV	RW-P	1H	S-TAG VLAN Functions: - Set the S-TAG VID values.
15	CTD	RW-P	0H	C-TAG DEI Functions: - Set the C-TAG DEI values.
14:12	CTP	RW-P	0H	C-TAG PCP Functions: - Set the C-TAG PCP values.
11:0	CTV	RW-P	1H	C-TAG VLAN Functions: - Set the C-TAG VID values.

(3) EARTFC

Ethernet Agent Reception TAG Filtering Configuration.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV							UT	SCRT	SCT	CRT	CT	CSRT	CST	RT	NT

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:9	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
8	UT	RW-P	1H	Unknown TAG Values: - 1’b0: Unknow Tag frame passed - 1’b1: Unknow Tag frame rejected
7	SCRT	RW-P	0H	SCR-TAG Values: - 1’b0: SCR-TAG frame passed - 1’b1: SCR-TAG frame rejected
6	SCT	RW-P	0H	SC-TAG Values: - 1’b0: SC-TAG frame passed - 1’b1: SC-TAG frame rejected
5	CRT	RW-P	0H	CR-TAG Values: - 1’b0: CR-TAG frame passed - 1’b1: CR-TAG frame rejected
4	CT	RW-P	0H	C-TAG Values: - 1’b0: C-TAG frame passed - 1’b1: C-TAG frame rejected
3	CSRT	RW-P	0H	CoSR-TAG Values: - 1’b0: CoSR-TAG frame passed - 1’b1: CoSR-TAG frame rejected
2	CST	RW-P	0H	CoS-TAG Values: - 1’b0: CoS-TAG frame passed - 1’b1: CoS-TAG frame rejected
1	RT	RW-P	0H	R-TAG Values: - 1’b0: R-TAG frame passed - 1’b1: R-TAG frame rejected
0	NT	RW-P	0H	No Tag Values: - 1’b0: No Tag frame passed - 1’b1: No Tag frame rejected

(4) EAICD0RC

Ether Agent Ingress C-TAG DEI 0 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ICD0DR7	ICD0PR7			ICD0DR6	ICD0PR6			ICD0DR5	ICD0PR5			ICD0DR4	ICD0PR4		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ICD0DR3	ICD0PR3			ICD0DR2	ICD0PR2			ICD0DR1	ICD0PR1			ICD0DR0	ICD0PR0		

Bits	Bit name	RW-P	Initial value	Function description
4*i+3 i=0..7	ICD0DRi	RW-P	0H	Ingress C-TAG DEI 0 DEI Remapping i Functions: - When ingress C-TAG (PCP==i and DEI==0), the incoming frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ICD0PRi	RW-P	i	Ingress C-TAG DEI 0 PCP Remapping i Functions: - When ingress C-TAG (PCP==i and DEI==0), the incoming frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

(5) EAICD1RC

Ether Agent Ingress C-TAG DEI 1 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ICD1DR7	ICD1PR7			ICD1DR6	ICD1PR6			ICD1DR5	ICD1PR5			ICD1DR4	ICD1PR4		
B15	14	13	12	11	10	9	8	B15	14	13	12	11	10	9	8
ICD1DR3	ICD1PR3			ICD1DR2	ICD1PR2			ICD1DR1	ICD1PR1			ICD1DR0	ICD1PR0		

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
4*i+3 i=0..7	ICD1DRi	RW-P	1H	Ingress C-TAG DEI 1 DEI Remapping i Functions: - When ingress C-TAG (PCP==i and DEI==1), the incoming frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ICD1PRi	RW-P	i	Ingress C-TAG DEI 1 PCP Remapping i Functions: - When ingress C-TAG (PCP==i and DEI==1), the incoming frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

(6) EAISD0RC

Ether Agent Ingress S-TAG DEI 0 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISD0DR7	ISD0PR7			ISD0DR6	ISD0PR6			ISD0DR5	ISD0PR5			ISD0DR4	ISD0PR4		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISD0DR3	ISD0PR3			ISD0DR2	ISD0PR2			ISD0DR1	ISD0PR1			ISD0DR0	ISD0PR0		

Bits	Bit name	RW-P	Initial value	Function description
4*i+3 i=0..7	ISD0DRi	RW-P	0H	Ingress S-TAG DEI 0 DEI Remapping i Functions: - When ingress S-TAG (PCP==i and DEI==0), the incoming frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ISD0PRi	RW-P	i	Ingress S-TAG DEI 0 PCP Remapping i Functions: - When ingress S-TAG (PCP==i and DEI==0), the incoming frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

(7) EAISD1RC

Ether Agent Ingress S-TAG DEI 1 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISD1DR7	ISD1PR7			ISD1DR6	ISD1PR6			ISD1DR5	ISD1PR5			ISD1DR4	ISD1PR4		
B15	14	13	12	11	10	9	8	B15	14	13	12	11	10	9	8
ISD1DR3	ISD1PR3			ISD1DR2	ISD1PR2			ISD1DR1	ISD1PR1			ISD1DR0	ISD1PR0		

Bits	Bit name	RW-P	Initial value	Function description
4*i+3 i=0..7	ISD1DRi	RW-P	1H	Ingress S-TAG DEI 1 DEI Remapping i Functions: - When ingress S-TAG (PCP==i and DEI==1), the incoming frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ISD1PRi	RW-P	i	Ingress S-TAG DEI 1 PCP Remapping i Functions: - When ingress S-TAG (PCP==i and DEI==1), the incoming frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

(8) EAECD0RC

Ether Agent Egress C-TAG DEI 0 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECD0DR7	ECD0PR7			ECD0DR6	ECD0PR6			ECD0DR5	ECD0PR5			ECD0DR4	ECD0PR4		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECD0DR3	ECD0PR3			ECD0DR2	ECD0PR2			ECD0DR1	ECD0PR1			ECD0DR0	ECD0PR0		

Bits	Bit name	RW-P	Initial value	Function description
4*i+3 i=0..7	ECD0DRi	RW-P	0H	Egress C-TAG DEI 0 DEI Remapping i Functions: - When egress C-TAG (PCP==i and DEI==0), the outgoing frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ECD0PRi	RW-P	i	Egress C-TAG DEI 0 PCP Remapping i Functions: - When egress C-TAG (PCP==i and DEI==0), the outgoing frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

(9) EAECD1RC

Ether Agent Egress C-TAG DEI 1 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECD1DR7	ECD1PR7			ECD1DR6	ECD1PR6			ECD1DR5	ECD1PR5			ECD1DR4	ECD1PR4		
B15	14	13	12	11	10	9	8	B15	14	13	12	11	10	9	8
ECD1DR3	ECD1PR3			ECD1DR2	ECD1PR2			ECD1DR1	ECD1PR1			ECD1DR0	ECD1PR0		

Bits	Bit name	RW-P	Initial value	Function description
4*i+3 i=0..7	ECD1DRi	RW-P	1H	Egress C-TAG DEI 1 DEI Remapping i Functions: - When egress C-TAG (PCP==i and DEI==1), the outgoing frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ECD1PRi	RW-P	i	Egress C-TAG DEI 1 PCP Remapping i Functions: - When egress C-TAG (PCP==i and DEI==1), the outgoing frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

(10) EAESD0RC

Ether Agent Egress S-TAG DEI 0 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ESD0DR7	ESD0PR7			ESD0DR6	ESD0PR6			ESD0DR5	ESD0PR5			ESD0DR4	ESD0PR4		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ESD0DR3	ESD0PR3			ESD0DR2	ESD0PR2			ESD0DR1	ESD0PR1			ESD0DR0	ESD0PR0		

Bits	Bit name	RW-P	Initial value	Function description
4*i+3 i=0..7	ESD0DRi	RW-P	0H	Egress S-TAG DEI 0 DEI Remapping i Functions: - When egress S-TAG (PCP==i and DEI==0), the outgoing frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ESD0PRi	RW-P	i	Egress S-TAG DEI 0 PCP Remapping i Functions: - When egress S-TAG (PCP==i and DEI==0), the outgoing frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

(11) EAESD1RC

Ether Agent Egress S-TAG DEI 1 Remapping Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ESD1DR7	ESD1PR7			ESD1DR6	ESD1PR6			ESD1DR5	ESD1PR5			ESD1DR4	ESD1PR4		
B15	14	13	12	11	10	9	8	B15	14	13	12	11	10	9	8
ESD1DR3	ESD1PR3			ESD1DR2	ESD1PR2			ESD1DR1	ESD1PR1			ESD1DR0	ESD1PR0		

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
4*i+3 i=0..7	ESD1DRi	RW-P	1H	Egress S-TAG DEI 1 DEI Remapping i Functions: - When egress S-TAG (PCP==i and DEI==1), the outgoing frame DEI will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.
4*i+2 : 4*i i=0..7	ESD1PRi	RW-P	i	Egress S-TAG DEI 1 PCP Remapping i Functions: - When egress S-TAG (PCP==i and DEI==1), the outgoing frame PCP will be remapped with this value. Restrictions: - SW: This function can change dynamically but not recommended.

3.3.1.4 Checksum function registers

(1) EACKSC

Ethernet Agent ChecKSum Configuration.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USMFSPE	RSV														
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV												ICMPCKSE	TCPCKSE	UDPCKSE	IP4CKSE

Bits	Bit name	RW-P	Initial value	Function description
31	USMFSPE	RW-P	0H	Under Switch Minimum Frame Size Padding Enable Values: - 1’b0: “Switch Minimum Frame Size” is 49 bytes **WITHOUT** TAGs. (Payload 35 bytes, if FCS is removed by RMAC). **Example of “Switch Error Frame”:** **- DMAC(6) + SMAC(6) + EtherType(2) + Payload(34)** **- DMAC(6) + SMAC(6) + EtherType(2) + Payload(30) + FCS(4) (not removed by RMAC)** **- DMAC(6) + SMAC(6) + TAGs(any) + EtherType(2) + Payload(34)** **- DMAC(6) + SMAC(6) + TAGs(any) + EtherType(2) + Payload(30) + FCS(4) (not removed by RMAC)** - 1’b1: “Switch Minimum Frame Size” is 32 bytes **WITH** including TAGs. And 32-59 bytes frame will be padded to 60 bytes frame (This 60 bytes does not include TAGs). **Example of “Switch Error Frame”:** **- DMAC(6) + SMAC(6) + EtherType(2) + Payload(17)** **- DMAC(6) + SMAC(6) + EtherType(2) + Payload(13) + FCS(4) (not removed by RMAC)** **- DMAC(6) + SMAC(6) + C-TAG(4) + EtherType(2) + Payload(13)** **- DMAC(6) + SMAC(6) + C-TAG(4) + EtherType(2) + Payload(9) + FCS(4) (not removed by RMAC)** Restrictions: - SW: If enabling this function, don’t enable “Receive CRC pass through” (**MRGC.RCPT** [RMAC]). Because padding bits are added at the end of the frame. If there is an FCS, padding bits will be given after FCS. - SW: If enabling this function, don’t enable “Cut-Through forwarding”. (Padded frame’s FCS will be invalid).
31:4	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
3	ICMPCKSE	RW-P	0H	ICMP ChecKSum check Enable Values: - 1’b0: IPv4/IPv6 ICMP incoming frame checksum is not checked - 1’b1: IPv4/IPv6 ICMP incoming frame checksum is checked
2	TCPCKSE	RW-P	0H	TCP ChecKSum check Enable Values: - 1’b0: IPv4/IPv6 TCP incoming frame checksum is not checked - 1’b1: IPv4/IPv6 TCP incoming frame checksum is checked

1	UDPCKSE	RW-P	0H	UDP ChecKSum check Enable Values: - 1’b0: IPv4/IPv6 UDP incoming frame checksum is not checked - 1’b1: IPv4/IPv6 UDP incoming frame checksum is checked
0	IP4CKSE	RW-P	0H	IPv4 ChecKSum check Enable Values: - 1’b0: IPv4 incoming frame checksum is not checked - 1’b1: IPv4 incoming frame checksum is checked

3.3.1.5 CBS function registers [802.1Qav]

(1) EACAEC

Ethernet Agent CBS Admin Enable Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								CE[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_PRIO_N-1:0	CE	RW-S	0H	CBS Enable Values: - Bit i set to1’b0: CBS for descriptor queue i disabled - Bit i set to 1’b1: CBS for descriptor queue i enabled Functions: - This register is only used to configure CBS and is not directly used by CBS modules.

(2) EACC

Ethernet Agent CBS Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								CC[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_PRIO_N-1:0	CC	R0W-S	0H	Config Change Functions: - By writing 1 to bit i, the Admin settings of CBS i are copied to its Oper settings (**EACAEC.CE[i]**, **EACAIVCi.CIVi** and **EACAULCi.CULi** are respectively copied to **EACOEM.CE[i]**, **EACOIVMi.CIVi** and **EACOULMi.CULi**).

(3) EACAIVCq (q=0.. FRM_PRIO_N-1)

Ethernet Agent CBS Admin Increment Value Configuration q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV												CIVq[19:16]			
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CIVq[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:20	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
19:0	CIVq	RW-S	0H	Credit Increment Value q Functions: - Configure CBS q throughput. - CIVq[19:16] is the credit increment in byte per clock and CIVq[15:0] is the credit increment in sub-byte per clock. - This register is only used to configure CBS and is not directly used by CBS modules. - For more details, refer to section 5.1.5 Cautions: - SW: Setting this register to 0 or a value close to 0 would set corresponding transmission queue to a very low throughput or could stuck the corresponding queue and lead to a queue overflow.

(4) EACAULCq (q=0.. FRM_PRIO_N-1)

Ethernet Agent CBS Admin Upper Limit Configuration q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV	CULq[30:16]														
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CULq[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
30:0	CULq	RW-S	7FFF_FFFFH	Credit Upper Limit q Functions: - Configure queue q interference time. - This register is only used to configure CBS and is not directly used by CBS modules. - For more details, refer to section 5.1.5

(5) EACOEM

Ethernet Agent CBS Oper Enable Monitoring

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								CE[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_PRIO_N-1:0	CE	R-S	0H	CBS Enable Values: - Bit i set to1'b0: CBS for descriptor queue i disabled - Bit i set to 1'b1: CBS for descriptor queue i enabled Update conditions: - HW: This register is updated to **EACACC.CE[i]** when 1'b1 is written to **EACC.CC[i]**.

(6) EACOIVMq (q=0.. FRM_PRIO_N-1)

Ethernet Agent CBS Oper Increment Value Monitoring q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV												CIVq[19:16]			
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CIVq[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:20	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
19:0	CIVq	R-S	0H	Credit Increment Value q Functions: - Configure CBS q throughput. - CIVq[19:16] is the credit increment in byte per clock and CIVq[15:0] is the credit increment in sub-byte per clock. - This register is only used to configure CBS and is not directly used by CBS modules. Update conditions: - HW: This register is updated to **EACAIVq.CIVq** when 1’b1 is written to **EACC.CC[q]**.

(7) EACOULMq (q=0.. FRM_PRIO_N-1)

Ethernet Agent CBS Oper Upper Limit Monitoring q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV	CULq[30:16]														
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CULq[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
30:0	CULq	R-S	7FFF_FFFFH	Credit Upper Limit q Functions: - Configure queue q interference time. - This register is only used to configure CBS and is not directly used by CBS modules. Update conditions: - HW: This register is updated to **EACAULq.CULq** when 1'b1 is written to **EACC.CC[q]**.

(8) EACGSM

Ethernet Agent CBS Gate State Monitoring

<table>
<tr><td>B31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="16">RSV</td></tr>
<tr><td>B15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">RSV</td><td colspan="8">CGS[FRM_PRIO_N-1:0]</td></tr>
</table>

Bits	Bit name	RW-P	Initial value	Function description
31:8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_PRIO_N-1:0	CGS	R-S	All1	CBS Gate State Values: - Bit i set to 1’b0: CBS doesn’t authorize queue i transmission (CBS credit negative) - Bit i set to 1’b1: CBS authorize queue i transmission (CBS credit positive)

3.3.1.6 TAS function registers [802.1Qbv]

(1) EATASC

Ethernet Agent TAS Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV								TASCA[TAS_RAM_AW-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV							TASTS	RSV					TASCI	TASCC	TASE

Bits	Bit name	RW-P	Initial value	Function description
31: TAS_RAM_AW+16	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
TAS_RAM_AW+15:16	TASCA	R-S	0H	TAS Configuration Address Functions: - Shows the address from which entry learning should happen in TAS RAM for the next configuration. Clear conditions: - HW: Being in RESET mode will clear this register.
15: PTP_TN_W+8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
PTP_TN_W+7:8	TASTS	RW-S	0H	TAS Timer Select Functions: - Selects the gPTP that will be used for TAS module.
7:3	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
2	TASCI	R-S	0H	TAS Config Impossible Values: - 1’b0: TAS configuration possible - 1’b1: TAS configuration is not possible Functions: - This bit helps the SW to known when TAS configuration is possible, it is equal to (**EATASC.TASE** & ~**EATASSM.TASSO**) | **EATASC.TASCC**. Clear conditions: - HW: Being in RESET mode will clear this register.
1	TASCC	R!=W-S	0H	TAS Config Change Values: - 1’b0: TAS is not changing configuration - 1’b1: TAS is changing configuration Set conditions: - SW: Writing 1 to this bit will set it Clear conditions: - HW: This bit will be cleared when TAS configuration change is done. - Refer to section 5.1.6.1 - HW: Being in RESET mode will clear this register.

0	TASE	R!=W-S	0H	TAS Enable Values: - 1’b0: TAS disabled - 1’b1: TAS enabled Functions: - Enables the TAS schedule - Refer to section 5.1.6.1 Set conditions: - SW: Writing 1 to this bit will set it Clear conditions: - SW: Writing 0 to this bit will clear it - HW: Being in RESET mode will clear this register.

(2) EATASIGSC

Ethernet Agent TAS Initial Gate State Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV							TASCTIGS	TASIGS[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: FRM_PRIO_N+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_PRIO_N	TASCTIGS	RW-S	0H	TAS Cut-Through Initial Gate State Values: - 1'b0: Initial gate state is closed - 1'b1: Initial gate state is opened Functions: - Gate state used for initialization of cut-through gate.
FRM_PRIO_N-1:0	TASIGS	RW-S	0H	TAS Initial Gate State Values: - Bit q set to 1'b0: Initial gate state is closed - Bit q set to 1'b1: Initial gate state is opened Functions: - Bit q is the gate state used for initialization of descriptor queue q gate.

(3) EATASENCq (q=0..FRM_PRIO_N)

Ethernet Agent TAS Entry Number Configuration q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								TASAENq[TAS_RAM_AW:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: TAS_RAM_AW+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
TAS_RAM_AW:0	TASAENq	RW-S	0H	TAS Admin Entry Number for gate q Functions: - This register is only used to configure TAS and is not directly used by TAS module. - Sets the number of TAS RAM entries used for gate i schedule. Restrictions: - SW: If TAS module is used in dynamic (TAS setting flow in Fig 4.10 is used when **EATASC.TASE** is already set), the sum of **EATASENCi.TASAENi** and **EATASCTENC.TASCTAEN** should be smaller or equal to TAS_RAM_DP/2 - FRM_TPRIO_N – 1. - SW: If TAS module is used in static (TAS setting flow in Fig 4.10 is never used when **EATASC.TASE** is already set), the sum of **EATASENCi.TASAENi** and **EATASCTENC.TASCTAEN** should be smaller or equal to TAS_RAM_DP – FRM_TPRIO_N – 1.

(4) EATASCTENC

Ethernet Agent TAS Cut-Through Entry Number Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV							TASCTAEN[TAS_RAM_AW:0]								

Bits	Bit name	RW-P	Initial value	Function description
31: TAS_RAM_AW+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
TAS_RAM_AW:0	TASCTAEN	RW-S	0H	TAS Admin Cut-Through Entry Number Functions: - This register is only used to configure TAS and is not directly used by TAS module. - Sets the number of TAS RAM entries used for cut-through gate schedule. Restrictions: - SW: If TAS module is used in dynamic (TAS setting flow in Fig 4.10 is used when **EATASC.TASE** is already set), the sum of **EATASENCi.TASAENi** and **EATASCTENC.TASCTAEN** should be smaller or equal to TAS_RAM_DP/2 - FRM_TPRIO_N – 1. - SW: If TAS module is used in static (TAS setting flow in Fig 4.10 is never used when **EATASC.TASE** is already set), the sum of **EATASENCi.TASAENi** and **EATASCTENC.TASCTAEN** should be smaller or equal to TAS_RAM_DP - FRM_TPRIO_N - 1.

(5) EATASENMi (q=0..FRM_PRIO_N)

Ethernet Agent TAS Entry Number Monitoring q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								TASOENq[TAS_RAM_AW:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: TAS_RAM_AW+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
TAS_RAM_AW:0	TASOENq	R-S	0H	TAS Oper Entry Number for gate q Update conditions: - HW: This register is updated to **EATASENCi.TASAENi** when configuration change occurs or when schedule start occurs. Refer to section 5.1.6.1. Clear conditions: - HW: Being in RESET mode will clear this register.

(6) EATASCTENM

Ethernet Agent TAS Cut-Through Entry Number Monitoring

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								TASCTOEN[TAS_RAM_AW:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: TAS_RAM_AW+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
TAS_RAM_AW:0	TASCTOENi	R-S	0H	TAS Cut-Through Oper Entry Number Update conditions: - HW: This register is updated to **EATASCTENC.TASCTAEN** when configuration change occurs or when schedule start occurs. Refer to section 5.1.6.1. Clear conditions: - HW: Being in RESET mode will clear this register.

(7) EATASCSTC0

Ethernet Agent TAS Cycle Start Time Configuration 0

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TASACSTP0[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASACSTP0[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	TASACSTP0	RW-S	0H	TAS Admin Cycle Start Time Part 0 Functions: - Time at which TAS scheduler should start/change configuration - This register is in ns.

(8) EATASCSTC1

Ethernet Agent TAS Cycle Start Time Configuration 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TASACSTP1[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASACSTP1[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	TASACSTP1	RW-S	0H	TAS Admin Cycle Start Time Part 1 Functions: - Time at which TAS scheduler should start/change configuration - This register is in 2^{32}*ns.

(9) EATASCSTM0

Ethernet Agent TAS Cycle Start Time Monitoring 0

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TASOCSTP0[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASOCSTP0[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	TASOCSTP0	R-S	0H	TAS Oper Cycle Start Time Part 0 Update conditions: - HW: **{EATASCSTM1,EATASCSTM0}** is updated to **{EATASCSTC1,EATASCSTC0}** + **EATASCTC** when configuration change occurs or when schedule start occurs. Refer to section 5.1.6.1. - HW: This register is updated to the next cycle start time every time a new cycle starts. Clear conditions: - HW: Being in RESET mode will clear this register.

(10) EATASCSTM1

Ethernet Agent TAS Cycle Start Time Monitoring 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TASOCSTP1[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASOCSTP1[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	TASOCSTP1	R-S	0H	TAS Oper Cycle Start Time Part 1 Update conditions: - HW: **{EATASCSTM1,EATASCSTM0}** is updated to the value of **{EATASCSTC1,EATASCSTC0}** + **EATASCTC** when configuration change occurs or when schedule start occurs. Refer to section 5.1.6.1. - HW: This register is updated to the next cycle start time every time a new cycle starts. Clear conditions: - HW: Being in RESET mode will clear this register.

(11) EATASCTC

Ethernet Agent TAS Cycle Time Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TASACT[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASACT[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	TASACT	RW-S	0H	TAS Admin Cycle Time Functions: - Configure the cycle time for TAS in ns Restrictions: - SW: This register should be set to a value greater than 100ns

(12) EATASCTM

Ethernet Agent TAS Cycle Time Monitoring

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TASOCT[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASOCT[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	TASOCT	R-S	0H	TAS Oper Cycle Time Update conditions: - HW: This register is updated to **EATASCTC.TASACT** when configuration change occurs or when schedule start occurs. Refer to section 5.1.6.1. Clear conditions: - HW: Being in RESET mode will clear this register.

(13) EATASGL0

Ethernet Agent TAS Gate Learn 0

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								TASGAL[TAS_RAM_AW-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: TAS_RAM_AW	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
TAS_RAM_AW-1: 0	TASGAL	RW-S	0H	TAS Gate Address Learn Functions: - Configures the address in which the TAS entry will be learnt

(14) EATASGL1

Ethernet Agent TAS Gate Learn 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV			TASGSL	TASGTL[27:16]											
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASGTL[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:29	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
28	TASGSL	RW-S	0H	TAS Gate State Learn Functions: - Configures gate state
27:0	TASGTL	RW-S	0H	TAS Gate Time Learn Functions: - Configures gate time

(15) EATASGLR

Ethernet Agent TAS Gate Learn Result

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GL	RSV														
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV															

Bits	Bit name	RW-P	Initial value	Function description
31	GL	R-S	0B	Gate Learn Set conditions: - HW: Writing **EATASGL1** register will set this bit. Clear conditions: - HW: This bit will be de-asserted when learning is completed.
30:0	RSV	R0-U	0H	Reserved area. On read, 0 will be returned

(16) EATASGR

Ethernet Agent TAS Gate Read

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								TASGAR[TAS_RAM_AW-1:0]							

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31: TAS_RAM_AW	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
TAS_RAM_AW-1: 0	TASGAR	RW-S	0H	TAS Gate Address Read Functions: - Configures the address in which the TAS entry will be read

(17) EATASGRR

Ethernet Agent TAS Gate Read Result

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GR	RSV	TASREF	TASGSR	TASGTR[27:16]											
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASGTR[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31	GR	R-S	0B	Gate Read Set conditions: - HW: Writing **EATASGR** register will set this bit. Clear conditions: - HW: This bit will be de-asserted when reading is completed.
30	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
29	TASREF	R-S	0H	TAS Read ECC Fail Update conditions: - **EATASGRR.GR** clear event Functions: - Set to 1’b1 when an ECC error happens during reading.
28	TASGSR	R-S	0H	TAS Gate State Read Update conditions: - **EATASGRR.GR** clear event Functions: - Displays gate state read value.
27:0	TASGTR	R-S	0H	TAS Gate Time Read Update conditions: - **EATASGRR.GR** clear event Functions: - Displays gate time read value.

(18) EATASHCC

Ethernet Agent TAS Hardware Calibration Configuration

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASJ[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:16	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
15:0	TASJ	RW-S	0H	TAS Jitter Functions: - Configure the jitter between the TAS data transmission decision and RMAC PHY interface [RMAC]. - Refer to section 5.1.6.3

(19) EATASRIRM

Ethernet Agent TAS RAM Initialization Register Monitoring.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV														TASRR	TASRIOG

Bits	Bit name	RW-P	Initial value	Function description
31:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
1	TASRR	R-S	0H	TAS RAM Ready Set conditions: - When **EATASRIRM.TASRIOG** is getting cleared. Clear conditions: - By writing 1 to **EATASRIRM.TASRIOG**.
0	TASRIOG	R!=W-S	0B	TAS RAM Initialization Ongoing. Set conditions: - SW: By writing 1 to this register. It starts TAS RAM initialization. Clear conditions: - HW: This bit is cleared when TAS RAM initialization is finished. Notes: - This process is required only once during initialization. This process cannot complete with TAS module already enabled (EATASC.TASE == 1’b1).

(20) EATASSM

Ethernet Agent TAS Status Monitoring.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV															TASSO
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV							TASCTGS	TASGS[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
16	TASSO	R-S	0H	TAS Scheduler Ongoing Values: - 1’b0: TAS scheduler is not ongoing - 1’b1: TAS scheduler is ongoing Clear conditions: - HW: Being in RESET mode will clear this register.
15: FRM_PRIO_N+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
FRM_PRIO_N	TASCTGS	R-S	1H	TAS Cut-Through Gate State Values: - 1’b0: Cut-through gate is closed - 1’b1: Cut-through gate is opened Restrictions: - HW: This register only exists if UCIAPRACE020_CT_ON is defined.
FRM_PRIO_N-1:0	TASGS	R-S	All1	TAS Gate State Values: - Bit i set to1’b0: Gate i is closed - Bit i set to 1’b1: Gate i is opened

3.3.2 TSNA Counter registers

(1) EAUSMFSECN

Ethernet Agent Switch Minimum Frame Size Error CouNter

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USMFSEN[COUNT_LOW_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	USMFSEN	RC-P	0H	Under Switch Minimum Frame Size Error Number Functions: - This register counts the number of received data lost because of under switch minimum size error. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a Under Switch Minimum Frame Size Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(2) EATFECN

Ethernet Agent TAG Filtering Error CouNter

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TFEN[COUNT_LOW_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	TFEN	RC-P	0H	TAG Filtering Error Number Functions: - This register counts the number of received data lost because of TAG filtering. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a TAG filter error happens and if this register has a value different than {{COUNT_LOW_W}{1’b1}}.

(3) EAFSECN

Ethernet Agent Frame Size Error CouNter

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSEN[COUNT_LOW_W-1:0]															

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	FSEN	RC-P	0H	Frame Size Error Number Functions: - This register counts the number of received data lost because of a Frame Size Error. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a Frame Size Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(4) EADQOECN

Ethernet Agent Descriptor Queue Overflow Error CouNter

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DQOEN[COUNT_LOW_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	DQOEN	RC-P	0H	Descriptor Queue Overflow Error Number Functions: - This register counts the number of transmit data lost because of a Descriptor Queue Overflow Error. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a Descriptor Queue Overflow Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(5) EADQSECN

Ethernet Agent Descriptor Queue Security Error CouNter

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DQSEN[COUNT_LOW_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	DQSEN	RC-P	0H	Descriptor Queue Security Error Number Functions: - This register counts the number of transmit data lost because of a Descriptor Queue Security Error. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a Descriptor Queue Security Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(6) EACKSECN

Ethernet Agent ChecKSum Error CouNter

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKSEN[COUNT_LOW_W-1:0]															

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	CKSEN	RC-P	0H	ChecKSum Error Number Functions: - This register counts the number of received data lost because of checksum error. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a checksum error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(7) EALDCN

Ethernet Agent Lost Descriptor CouNter

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LDN[COUNT_LOW_W-1:0]															

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	LDN	RC-P	0H	Lost Descriptor Number Functions: - When **EATDRC.TDRM**==1, this register counts the number of Lost Descriptor because of ECC Error at Descriptor Info RAM. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by “the number of stored Descriptor in the queue” when ECC Error at Descriptor Info RAM happened in the queue and if this register has a value different than {{COUNT_LOW_W}{1’b1}}.

(8) EARFCNEO0

Ethernet Agent Received Frame CouNter E-frame per Octets 0

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROEN0[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROEN0	RC-P	0H	Received 64 Octets E-frames Number Functions: - This register counts when the number of octets in the received E-frame is 64. Frame sizes are from "MAC address" to "FCS". Even if frame's FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. (Count by the frame size before being processed by **EAVCC.VIM**.) Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received a E-frames is 64 octets and if this register has a value different than {{COUNT_MED_W}{1'b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(9) EARFCNEO1

Ethernet Agent Received Frame CouNter E-frame per Octets 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROEN1[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROEN1	RC-P	0H	Received 65 to 127 Octets E-frames Number Functions: - This register counts when the number of octets in the received E-frame is 65 to 127. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received a E-frame is 65 to 127 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(10) EARFCNEO2

Ethernet Agent Received Frame CouNter E-frame per Octets 2

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROEN2[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROEN2	RC-P	0H	Received 128 to 255 Octets E-frames Number Functions: - This register counts when the number of octets in the received E-frame is 128 to 255. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received E-frames are 128 to 255 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(11) EARFCNEO3

Ethernet Agent Received Frame CouNter E-frame per Octets 3

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROEN3[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROEN3	RC-P	0H	Received 256 to 511 Octets E-frames Number Functions: - This register counts when the number of octets in the received E-frame is 256 to 511. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received E-frames are 256 to 511 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(12) EARFCNEO4

Ethernet Agent Received Frame CouNter E-frame per Octets 4

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROEN4[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROEN4	RC-P	0H	Received 512 to 1023 Octets E-frames Number Functions: - This register counts when the number of octets in the received E-frame is 512 to 1023. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received E-frames are 512 to 1023 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(13) EARFCNEO5

Ethernet Agent Received Frame CouNter E-frame per Octets 5

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROEN5[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROEN5	RC-P	0H	Received 1024 to 1518 Octets E-frames Number Functions: - This register counts when the number of octets in the received E-frame is 1024 to 1518. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received E-frames are 1024 to 1518 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(14) EARFCNEO6

Ethernet Agent Received Frame CouNter E-frame per Octets 6

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROEN6[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROEN6	RC-P	0H	Received 1519 or more Octets E-frames Number Functions: - This register counts when the number of octets in the received E-frame is 1519 or more. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received E-frames are 1519 or more octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(15) EARFCNPO0

Ethernet Agent Received Frame CouNter P-frame per Octets 0

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROPN0[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROPN0	RC-P	0H	Received 64 Octets P-frames Number Functions: - This register counts when the number of octets in the received P-frame is 64. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received P-frames are 64 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(16) EARFCNPO1

Ethernet Agent Received Frame CouNter P-frame per Octets 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROPN1[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROPN1	RC-P	0H	Received 65 to 127 Octets P-frames Number Functions: - This register counts when the number of octets in the received P-frame is 65 to 127. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received P-frames are 65 to 127 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(17) EARFCNPO2

Ethernet Agent Received Frame CouNter P-frame per Octets 2

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROPN2[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROPN2	RC-P	0H	Received 128 to 255 Octets P-frames Number Functions: - This register counts when the number of octets in the received P-frame is 128 to 255. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received P-frames are 128 to 255 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(18) EARFCNPO3

Ethernet Agent Received Frame CouNter P-frame per Octets 3

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROPN3[COUNT_MED_W-1:0]															

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROPN3	RC-P	0H	Received 256 to 511 Octets P-frames Number Functions: - This register counts when the number of octets in the received P-frame is 256 to 511. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received P-frames are 256 to 511 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(19) EARFCNPO4

Ethernet Agent Received Frame CouNter P-frame per Octets 4

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROPN4[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROPN4	RC-P	0H	Received 512 to 1023 Octets P-frames Number Functions: - This register counts when the number of octets in the received P-frame is 512 to 1023. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received P-frames are 512 to 1023 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(20) EARFCNPO5

Ethernet Agent Received Frame CouNter P-frame per Octets 5

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROPN5[COUNT_MED_W-1:0]															

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROPN5	RC-P	0H	Received 1024 to 1518 Octets P-frames Number Functions: - This register counts when the number of octets in the received P-frame is 1024 to 1518. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received P-frames are 1024 to 1518 octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(21) EARFCNPO6

Ethernet Agent Received Frame CouNter P-frame per Octets 6

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROPN6[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_MED_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_MED_W-1:0	ROPN6	RC-P	0H	Received 1519 or more Octets P-frames Number Functions: - This register counts when the number of octets in the received P-frame is 1519 or more. Frame sizes are from “MAC address” to “FCS”. Even if frame’s FCS is removed by [RMAC], this will be counted with the size of FCS. If frame is tagged(S/C-TAG) by **EAVCC.VIM**, this will not be included. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when received P-frames are 1519 or more octets and if this register has a value different than {{COUNT_MED_W}{1’b1}}. It will be counted regardless of the error frame (MAC error or buffer overflow).

(22) EADQOECNPq (q=0..FRM_PRIO_N-1)

Ethernet Agent Descriptor Queue Overflow Error CouNter Priority q

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DQOENPq[COUNT_LOW_W-1:0]															

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	DQOENPq	RC-P	0H	Descriptor Queue Overflow Error Number Priority q Functions: - This register counts the number of transmit data lost because of a Descriptor Queue Overflow Error Priority q. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a Descriptor Queue Overflow Error happens on Priority q and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(23) EADQOECNCT

Ethernet Agent Descriptor Queue Overflow Error CouNter Cut Through

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DQOENCT[COUNT_LOW_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:COUNT_LOW_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned
COUNT_LOW_W-1:0	DQOENCT	RC-P	0H	Descriptor Queue Overflow Error Number Cut Through Functions: - This register counts the number of transmit data lost because of a Descriptor Queue Overflow Error Cut Through Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it. Increment conditions: - HW: Incremented by 1 when a Descriptor Queue Overflow Error Cut Through happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

3.3.3 TSNA Interrupt registers

3.3.3.1 Error interrupt registers

(1) EAEIS0

Ethernet Agent Error Interrupt Status 0.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV							TASCTGEES	TASGEES[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSES[FRM_PRIO_N-1:0]								CKSES	TFES	USMFSES	L23UECCES	DSECCES	PECCES	TECCES	DECCES

Bits	Bit name	RW-P	Initial value	Function description
31:FRM_PRIO_N+17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N+16	TASCTGEES	R!=W-F	0H	TAS Cut-Through Gate ECC Error Status Set conditions: - HW: This bit is set when an ECC error is detected while reading the TAS RAM to control cut-through gate. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: Cut-through gate go back to initial gate state. - SW: Nothing if it is acceptable to operate with cut-through gate in initial gate state else TAS disable -> TAS configuration or TAS disable or switch reset.
FRM_PRIO_N+15:16	TASGEES	R!=W-P	0H	TAS Gate ECC Error Status Set conditions: - HW: Bit q of this register is set when an ECC error is detected while reading the TAS RAM to control gate for descriptor queue q. (This flag is not set when SW reads the TAS RAM and an ECC error happens. In this case, ECC error detection should happen using **EATASGRR.TASREF** register). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: Gate q will go back to initial gate state. - SW: Nothing if it is acceptable to operate with corresponding gate in initial gate state else TAS disable -> TAS configuration or TAS disable or switch reset.

FRM_PRIO_N+7:8	FSES	R!=W-P	0H	Frame Size Error Status Set conditions: - HW: Bit q of this register is set when a frame bigger than **EATMFSC.MFSq** has been received for descriptor queue q. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: The current frame will be lost. Following data will be processed normally. - SW: System dependent, cannot de defined here.
7	CKSES	R!=W-P		ChecKSum Error Status Set conditions: - HW: A checksum error has been detected. (Refer to section 5.2.1). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: Frame is sent to Forwarding Engine [FWD] with local descriptor **DESCR.CKSE** bit set (Refer to section 5.2.1). - SW: System dependent, cannot de defined here.
6	TFES	R!=W-P	0H	TAG Filtering Error Status Set conditions: - HW: An unauthorized TAG format has been detected. (Refer to section 5.2.1). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: Frame is sent to Forwarding Engine [FWD] with local descriptor **DESCR.TFE** bit set (Refer to section 5.2.1). - SW: System dependent, cannot de defined here.
5	USMFSES	R!=W-P	0H	Under Switch Minimum Frame Size Error Status Set conditions: - HW: A frame smaller than “Switch Minimum Frame Size” has been transmitted data lost from RMAC. - HW: This error will happen with [RMAC] MEIS.RPOES or MEIS.REOES. - HW: This frame will not be counted by “TSNA Counter registers”. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: The frame is discarded. - SW: System dependent, cannot de defined here. Restrictions: - If a frame is scraped by MEIS.FOES [RMAC], it will be filtered by this function.

4	L23UECCES	R!=W-P	0H	Layer 2/3 Update ECC Error Status Set conditions: - HW: When an ECC error has been detected while reading Layer 2/3 update information from the Layer 2/3 update RAM. This RAM is in the forwarding Engine [FWD] and is read by TSNA using L2/L3 update bus. The error is flag to TSNA using **L23U.ERR** [FWD]. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Layer 2/3 update information corresponding frame is discarded. - SW: Refer to forwarding engine specification “Layer2/Layer3 Update ECC Error” [FWD]
3	DSECCES	R!=W-P	0H	Descriptor ECC Error Status. Set conditions: - HW: When an ECC error has been detected while reading a descriptor from the descriptor RAM. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Loss of the descriptor. One or several pointers will be lost so the switch will continue operating but with a reduced Local RAM. - SW: Nothing if it is acceptable to operate with a reduced Local RAM else Switch reset.
2	PECCES	R!=W-P	0H	Pointer ECC Error Status. Set conditions: - HW: When an ECC error has been detected in a pointer from the fabric read interface. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Data fetching for the current frame will happen always in the same pointer. One or several pointers will be lost so the switch will continue operating but with a reduced Local RAM. - HW: Data will be sent to the RMAC with an error. - SW: Nothing if it is acceptable to operate with a reduced Local RAM else Switch reset.

1	TECCES	R!=W-P	0H	TAG ECC Error Status Set conditions: - HW: When an ECC error has been detected in a TAG from the fabric read interface. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Received invalid TAG because of ECC error will be set to 0 but tagging from Layer2/Layer3 routing will still happen. - HW: Data will be sent to the RMAC with an error. - SW: Nothing.
0	DECCES	R!=W-P	0H	Data ECC Error Status Set conditions: - HW: When an ECC error has been detected in a data from the fabric read interface. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Received invalid data because of ECC error will be set to 0. - HW: Data will be sent to the RMAC with an error. - SW: Nothing.

(2) EAEIE0

Ethernet Agent Error Interrupt Enable 0.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV							TASCTGEEE	TASGEEE[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSEE[FRM_PRIO_N-1:0]								CKSEE	TFEE	USMFSEE	L23UECCEE	DSECCEE	PECCEE	TECCEE	DECCEE

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:FRM_PRIO_N+17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N+16	TASCTGEEE	R!=W-F	0H	TAS Cut-Through Gate ECC Error Enable Values: - 1’b0: Interrupt disabled. - 1’b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.TASCTGEED** register will clear this bit.
FRM_PRIO_N+15:16	TASGEEE	R!=W-P	0H	TAS Gate ECC Error Enable Values: - 1’b0 for bit i: Interrupt disabled for descriptor queue i. - 1’b1 for bit i: Interrupt enabled for descriptor queue i. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to one of **EAEID0.TASGEED** bits will clear the corresponding bit in this register.
FRM_PRIO_N+7:8	FSEE	R!=W-P	0H	Frame Size Error Enable Values: - 1’b0 for bit i: Interrupt disabled for descriptor queue i. - 1’b1 for bit i: Interrupt enabled for descriptor queue i. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to one of **EAEID0.FSED** bits will clear the corresponding bit in this register.

7	CKSEE	R!=W-P		ChecKSum Error Enable Values: - 1’b0: Interrupt disabled. - 1’b1: Interrupt enabled. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.CKSED** register will clear this bit.
6	TFEE	R!=W-P	0H	TAG Filtering Error Enable Values: - 1’b0: Interrupt disabled. - 1’b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.TFED** register will clear this bit.
5	USMFSEE	R!=W-P	0H	Under Switch Minimum Frame Size Error Enable Values: - 1’b0: Interrupt disabled. - 1’b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.USMFSED** register will clear this bit.
4	L23UECCEE	R!=W-P	0H	Layer 2/3 Update ECC Error Enable Values: - 1’b0: Interrupt disabled. - 1’b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.L23UECCED** register will clear this bit.
3	DSECCEE	R!=W-P	0H	Descriptor ECC Error Enable Values: - 1’b0: Interrupt disabled. - 1’b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.DSECCED** register will clear this bit.

2	PECCEE	R!=W-P	0H	Pointer ECC Error Enable Values: - 1'b0: Interrupt disabled. - 1'b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.PECCED** register will clear this bit.
1	TECCEE	R!=W-P	0H	TAG ECC Error Enable Values: - 1'b0: Interrupt disabled. - 1'b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.TECCED** register will clear this bit.
0	DECCEE	R!=W-P	0H	Data ECC Error Enable Values: - 1'b0: Interrupt disabled. - 1'b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID0.DECCED** register will clear this bit.

(3) EAEID0

Ethernet Agent Error Interrupt Disable 0.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV							TASCTGEED	TASGEED[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSED[FRM_PRIO_N-1:0]								CKSED	TFED	USMFSED	L23UECCED	DSECCED	PECCED	TECCED	DECCED

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:FRM_PRIO_N+17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N+16	TASCTGEED	R0W-F	0H	TAS Cut-Through Gate ECC Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **EAEIE0.TASCTGEEE** register.
FRM_PRIO_N+15:16	TASGEED	R0W-P	0H	TAS Gate ECC Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **EAEIE0.TASGEEE** register.
FRM_PRIO_N+7:8	FSED	R0W-P	0H	Frame Size Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **EAEIE0.FSEE** register.
7	CKSED	R0W-P	0H	ChecKSum Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.CHSEE** register.
6	TFED	R0W-P	0H	TAG Filtering Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.TFEE** register.
5	USMFSED	R0W-P	0H	Under Switch Minimum Frame Size Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.USMFSEE** register.
4	L23UECCED	R0W-P	0H	Layer 2/3 Update ECC Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.L23UECCEE** register.
3	DSECCED	R0W-P	0H	Descriptor ECC Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.DSECCEE** register.
2	PECCED	R0W-P	0H	Pointer ECC Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.PECCEE** register.
1	TECCED	R0W-P	0H	TAG ECC Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.TECCEE** register.

0	DECCED	R0W-P	0H	Data ECC Error Disable Functions: - Writing 1 to this bit will clear **EAEIE0.DECCEE** register.

(4) EAEIS1

Ethernet Agent Error Interrupt Status 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV							TASCT GES	TASGES[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								CULES[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:FRM_PRIO_N+17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N+16	TASCTGES	R!=W-F	0H	TAS Cut-Through Gate Error Status Set conditions: - HW: This register is set when cut-through corresponding TAS gate has no time to fetch the next gate state value until it starts. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: Gate with restart during next cycle. - SW: If this error happens regularly, TAS should be reconfigured with a different schedule (software issue because the minimum gate time is not respected). During reconfiguration, this error will continue to happen until the new start time is reached and so, it should be ignored. Cautions: - This error can happen for the following reasons: TAS start time **EATASCSTC0/1** is set in the past, gPTP timer had an offset correction[gPTP], the minimum gate time is not respected.
FRM_PRIO_N+15:16	TASGES	R!=W-P	0H	TAS Gate Error Status Set conditions: - HW: Bit q of this register is set when descriptor queue q corresponding TAS gate had no time to fetch the next gate state value until it starts. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: Gate with restart during next cycle. - SW: If this error happens regularly, TAS should be reconfigured with a different schedule (software issue because the minimum gate time is not respected). During reconfiguration, this error will continue to happen until the new start time is reached and so, it should be ignored. Cautions: - This error can happen for the following reasons: TAS start time **EATASCSTC0/1** is set in the past, gPTP timer had an offset correction[gPTP], the minimum gate time is not respected (a gate time is smaller than 50ns + **EATASHCC.TASJ**).

15:FRM_PRIO_N-1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N-1	CULES	R!=W-P	0H	CBS Upper Limit Error Status Set conditions: - HW: Bit q of this register is set when descriptor queue q corresponding CBS credits has reached its upper limit **EACOULMq.CULq**. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Error recovery: - HW: Credit calculation stops at **EACOULMq.CULq** and no further credit are added until frame transmission for the corresponding queue. As a consequence, the queue throughput will be lower than expected. - SW: This error shouldn't happen. It is a software error (Setting TAS with gates which are partially covering each other can set this error).

(5) EAEIE1

Ethernet Agent Error Interrupt Enable 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV							TASCT GEE	TASGEE[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								CULEE[FRM_PRIO_N-1:0]							

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:FRM_PRIO_N+17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N+16	TASCTGEE	R!=W-F	0H	TAS Cut-through Gate Error Enable Values: - 1'b0: Interrupt disabled. - 1'b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID1.TASCTGED** register will clear this bit.
FRM_PRIO_N+15:16	TASGEE	R!=W-P	0H	TAS Gate Error Enable Values: - 1'b0 for bit i: Interrupt disabled for descriptor queue i. - 1'b1 for bit i: Interrupt enabled for descriptor queue i. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to one of **EAEID1.TASGED** bits will clear the corresponding bit in this register.
15:FRM_PRIO_N-1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N-1	CULEE	R!=W-P	0H	CBS Upper Limit Error Enable Values: - 1'b0 for bit i: Interrupt disabled for descriptor queue i. - 1'b1 for bit i: Interrupt enabled for descriptor queue i. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to one of **EAEID1.CULED** bits will clear the corresponding bit in this register.

(6) EAEID1

Ethernet Agent Error Interrupt Disable 1

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV							TASCTGED	TASGED[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								CULED[FRM_PRIO_N-1:0]							

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31:FRM_PRIO_N+17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N+16	TASCTGED	R0W-F	0H	TAS Cut-through Gate Error Disable Functions: - Writing 1 to this bit will clear **EAEIE1.TASCTGEE** register.
FRM_PRIO_N+15:16	TASGED	R0W-P	0H	TAS Gate Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **EAEIE1.TASGEE** register.
15:FRM_PRIO_N-1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N-1	CULED	R0W-P	0H	CBS Upper Limit Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **EAEIE1.CULEE** register.

(7) EAEIS2

Ethernet Agent Error Interrupt Status 2.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIECCES[FRM_PRIO_N-1:0]								DQSES[FRM_PRIO_N-1:0]							
B15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RSV							CTDQOES	DQOES[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
FRM_PRIO_N+23:24	DIECCES	R!=W-P	0H	Descriptor Info ECC Error Status. Set conditions: - HW: When **EATDRC.TDRM**==1, **EATDQMq.DNQq**>1 and an ECC error has been detected while reading Descriptor Info RAM for the queue[q]. The number of frames/descriptors lost due to this error is equal to the number of descriptors stored (**EATDQMq.DNQq**) in the queue. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Loss of the descriptor. One or several pointers will be lost so the switch will continue operating but with a reduced Local RAM. - SW: Nothing if it is acceptable to operate with a reduced Local RAM else Switch reset.
FRM_PRIO_N+15:16	DQSES	R!=W-F	0H	Descriptor Queue Security Error Status Set conditions: - HW: Bit q of this register is set when a non-Secure descriptor is received (**FDESCR.SEC** is not set [FWD]) and queue q is a secure queue (**EATDQSC.TDQSL[q]** is set) Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Any non-Secure descriptor received for secure Descriptor queue will not be accepted by TSNA and will not be forwarded to CPU. - SW: System dependent, cannot de defined here.
15: FRM_PRIO_N+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

FRM_PRIO_N	CTDQOES	R!=W-F	0H	Cut-Through Descriptor Queue Overflow Error Status Set conditions: - HW: This register has exclusive two set conditions. 1) [TSNA is ongoing] (**EAMC.OPC** == 2'b11 and **EAMS.OPS** == 2'b11) Descriptor queue is "full (**EACTDQDCq.CTDQD** == **EACTDQM.CTQDN**)" and "not disabled (**EATDQC.TCTDQD** is not set)" when descriptor is received for cut-through descriptor. OR 2) [TSNA is (going) out of OPERATION] (**EAMC.OPC** != 2'b11) When descriptor is received for cut-through descriptor. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Any descriptor received for a full Descriptor queue will not be accepted by TSNA and will not be forwarded to RMAC [RMAC]. - SW: Initialize the state with Emergency reset flow [TOP] because pointers can disappear. So too many descriptors are received for the corresponding queue. **EACTDQDC.CTDQD** setting should be reviewed.
FRM_PRIO_N-1:0	DQOES	R!=W-P	0H	Descriptor Queue Overflow Error Status [Cond1] [TSNA is ongoing] (**EAMC.OPC** == 2'b11 and **EAMS.OPS** == 2'b11) and a descriptor is received for queue q. [Cond2] Descriptor queue is "full (**EATDQDCq.DQDq** == **EATDQMq.DNQq**)" and "not disabled (**EATDQC.TDQD[q]** is not set). [Cond3] "**EATDRC.TDRM** == 1", (Descriptor queue is "full (**EATDQDCq.DQDq** == **EATDQMq.DNQq**)" or "descriptor RAM is full (Sum of **EATDQMq.DNQq** **[q=0..FRM_PRIO_N-1]** == (**DES_RAM_DP** **- FRM_PRIO_N** ***2**))") and "not disabled (**EATDQC.TDQD[q]** is not set)". Set conditions: - HW: Bit q of this register is set because… [Cond1] and [Cond2] OR [Cond1] and [Cond3] Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: Any descriptor received for a full Descriptor queue will not be accepted by TSNA and will not be forwarded to RMAC [RMAC]. - SW: Too many descriptors are received for the corresponding queue. **EATDQDCq.DQDq** setting should be reviewed.

(8) EAEIE2

Ethernet Agent Error Interrupt Enable 2.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIECCEE[FRM_PRIO_N-1:0]								DQSEE[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV							CTDQOEE	DQOEE[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
FRM_PRIO_N+23:24	DIECCEE	R!=W-P	0H	Descriptor Info ECC Error Enable Values: - 1’b0 for bit q: Interrupt disabled for error q. - 1’b1 for bit q: Interrupt enabled for error q. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to bit q in **EAEID2.DIECCED** register will clear the bit q in this register.
FRM_PRIO_N+15:16	DQSEE	R!=W-F	0H	Descriptor Queue Security Error Enable Values: - 1’b0 for bit i: Interrupt disabled for descriptor queue i. - 1’b1 for bit i: Interrupt enabled for descriptor queue i. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to one of **EAEID2.DQSED** bits will clear the corresponding bit in this register.
15: FRM_PRIO_N+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N	CTDQOEE	R!=W-F	0H	Cut-Through Descriptor Queue Overflow Error Enable Values: - 1’b0: Interrupt disabled. - 1’b1: Interrupt Enabled. Set conditions: - Writing 1 to this bit will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to **EAEID2.CTDQOED** register will clear this bit.

FRM_PRIO_N-1:0	DQOEE	R!=W-P	0H	Descriptor Queue Overflow Error Enable Values: - 1’b0 for bit i: Interrupt disabled for descriptor queue i. - 1’b1 for bit i: Interrupt enabled for descriptor queue i. Set conditions: - Writing 1 to one of these bits will set it. Clear conditions: - HW: Being in RESET mode will clear this register. - Writing 1 to one of **EAEID2.DQOED** bits will clear the corresponding bit in this register.

(9) EAEID2

Ethernet Agent Error Interrupt Disable 2.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIECCED[FRM_PRIO_N-1:0]								DQSED[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV							CTDQOED	DQOED[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
FRM_PRIO_N+23:24	DIECCED	R0W-P	0H	Descriptor Info ECC Error Disable Functions: - Writing 1 to bit q in this register will clear bit q in **EAEIE2.DIECCEE** register.
FRM_PRIO_N+15:16	DQSED	R0W-F	0H	Descriptor Queue Security Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **EAEIE2.DQSEE** register.
15: FRM_PRIO_N+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N	CTDQOED	R0W-F	0H	Cut-Through Descriptor Queue Overflow Error Disable Functions: - Writing 1 to this bit will clear **EAEIE2.CTDQOEE** register.
FRM_PRIO_N-1:0	DQOED	R0W-P	0H	Descriptor Queue Overflow Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **EAEIE2.DQOEE** register.

3.3.4 TSNA Security registers

(1) EASCR

Ethernet Agent Security Configuration Register.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV								DQRSL[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV									CRSL	EIRSL	TASRSL	TGRSL	MCRSL	TRSL	MRSL

Bits	Bit name	RW-P	Initial value	Function description
31: FRM_PRIO_N+16	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

FRM_PRIO_N+15:16	DQRSL	RW-F	0H	Descriptor queue Register Security Level. Values: - 1’b0 for bit q: Descriptor queue q registers can only be accessed by the APB secure interface - 1’b1 for bit q: Descriptor queue q registers can be accessed by both APBs Descriptor queue q registers include the following registers: - **EATDQC.TDQD[q]** - **EATDQC.TDQP[q]** - **EATMFSCq** - **EATDQDCq** - **EATDQMq** - **EATDQMLMq** - **EACAEC.CE[q]** - **EACC.CC[q]** - **EACAIVCq** - **EACAULCq** - **EACOEM.CE[q]** - **EACOIVMq** - **EACOULMq** - **EACGSM.CGS[q]** - **EAEIS0.TASGEES[q]** - **EAEIS0.FSES[q]** - **EAEIE0.TASGEEE[q]** - **EAEIE0.FSEE[q]** - **EAEID0.TASGEED[q]** - **EAEID0.FSED[q]** - **EAEIS1.TASGES[q]** - **EAEIS1.CULES[q]** - **EAEIE1.TASGEE[q]** - **EAEIE1.CULEE[q]** - **EAEID1.TASGED[q]** - **EAEID1.CULED[q]** - **EAEIS2.DQOES[q]** - **EAEIE2.DQOEE[q]** - **EAEID2.DQOED[q]** - **EAEIS2.DIECCES[q]** - **EAEIS2.DIECCEE[q]** - **EAEIE2.DIECCED[q]**
15: 7	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

6	CRSL	RW-F	0H	Counter Register Security Level. Values: - 1’b0: Counter registers can only be accessed by the APB secure interface - 1’b1: Counter registers can be accessed by both APBs Counter registers include the following registers: - **EAUSMFSECN** - **EATFECN** - **EAFSECN** - **EADQOECN** - **EADQSECN** - **EACKSECN** - **EALDCN** - **EARFCNEO0** - **EARFCNEO1** - **EARFCNEO2** - **EARFCNEO3** - **EARFCNEO4** - **EARFCNEO5** - **EARFCNEO6** - **EARFCNPO0** - **EARFCNPO1** - **EARFCNPO2** - **EARFCNPO3** - **EARFCNPO4** - **EARFCNPO5** - **EARFCNPO6** - **EADQOECNPq** - **EADQOECNCT**

5	EIRSL	RW-F	0H	Error Interrupt Register Security Level Values: - 1’b0: Error Interrupt registers can only be accessed by the APB secure interface - 1’b1: Error Interrupt registers can be accessed by both APBs Error Interrupt registers include the following registers: - **EAEIS0.DECCES** - **EAEIS0.TECCES** - **EAEIS0.PECCES** - **EAEIS0.DSECCES** - **EAEIS0.L23UECCES** - **EAEIS0.USMFSES** - **EAEIS0.TFES** - **EAEIS0.CKSES** - **EAEIE0.DECCEE** - **EAEIE0.TECCEE** - **EAEIE0.PECCEE** - **EAEIE0.DSECCEE** - **EAEIE0.L23UECCEE** - **EAEIE0.USMFSEE** - **EAEIE0.TFEE** - **EAEIE0.CKSEE** - **EAEID0.DECCED** - **EAEID0.TECCED** - **EAEID0.PECCED** - **EAEID0.DSECCED** - **EAEID0.L23UECCED** - **EAEID0.USMFSED** - **EAEID0.TFED** - **EAEID0.CKSED**

4	TASRSL	RW-F	0H	TAS Register Security Level. Values: - 1’b0: TAS registers can only be accessed by the APB secure interface - 1’b1: TAS registers can only be accessed by the APB unsecure interface TAS registers include the following registers: - **EATASC** - **EATASIGSC** - **EATASENCi** - **EATASCTENC** - **EATASENMi** - **EATASCTENM** - **EATASCSTC0** - **EATASCSTC1** - **EATASCSTM0** - **EATASCSTM1** - **EATASCTC** - **EATASCTM** - **EATASGL0** - **EATASGL1** - **EATASGLR** - **EATASGR** - **EATASGRR** - **EATASHCC** - **EATASRIRM** - **EATASSM**
3	TGRSL	RW-F	0H	TAG Register Security Level. Values: - 1’b0: TAG (checksum also) registers can only be accessed by the APB secure interface - 1’b1: TAG (checksum also) registers can be accessed by both APBs TAG registers include the following registers: - **EAVCC** - **EAVTC** - **EARTFC** - **EACKSC** - **EAICD0RC** - **EAICD1RC** - **EAISD0RC** - **EAISD1RC** - **EAECD0RC** - **EAECD1RC** - **EAESD0RC** - **EAESD1RC**
2	MCRSL	RW-F	0H	MAC Register Security Level. Values: - 1’b0: MAC registers can only be accessed by the APB secure interface - 1’b1: MAC registers can be accessed by both APBs MAC registers include the following registers: - All registers included in this Ethernet agent RMAC [RMAC]. All register security attribute in RMAC is Protected (P).

1	TRSL	RW-F	0H	Transmission Register Security Level. Values: - 1’b0: Transmission registers can only be accessed by the APB secure interface - 1’b1: Transmission registers can be accessed by both APBs Reception registers include the following registers: - **EATDRC** - **EAIRC** - **EATDQSC** - **EATDQAC** - **EATPEC**
0	MRSL	RW-F	0H	Mode Register Security Level. Values: - 1’b0: Mode registers can only be accessed by the APB secure interface - 1’b1: Mode registers can be accessed by both APBs Mode registers include the following registers: - **EAMC** - **EAMS**

4. Register utilization

4.1 Operation Modes

Table 4-1 describes ETHA operation modes.

Table 4-1: ETHA Operation Modes

Operation mode	GWMS.OPS value	Description
DISABLE	2'd1	- No transaction is ongoing. - Only status registers are accessible for writing when the agent clock is enabled.
RESET	2'd0	- No transaction is ongoing. - An internal Reset is asserted to reset ETHA logic with status registers (When a register is reset in RESET mode, it is mentioned in its description). - No register is accessible for writing. - RAM values are held.
CONFIG	2'd2	- No transaction is ongoing. - Static and some dynamic registers are accessible for writing.
OPERATION	2'd3	- Transactions are ongoing. - Dynamic registers are accessible for writing.

4.1.1 Operation mode transitions

Fig 4.1 shows the operating mode transitions.

A mode transition can be trigger by:

- Hardware reset.
- Software reset.
- Configuration of **EAMC.OPC**. In that case, mode transition will be confirmed by **EAMS.OPS**.

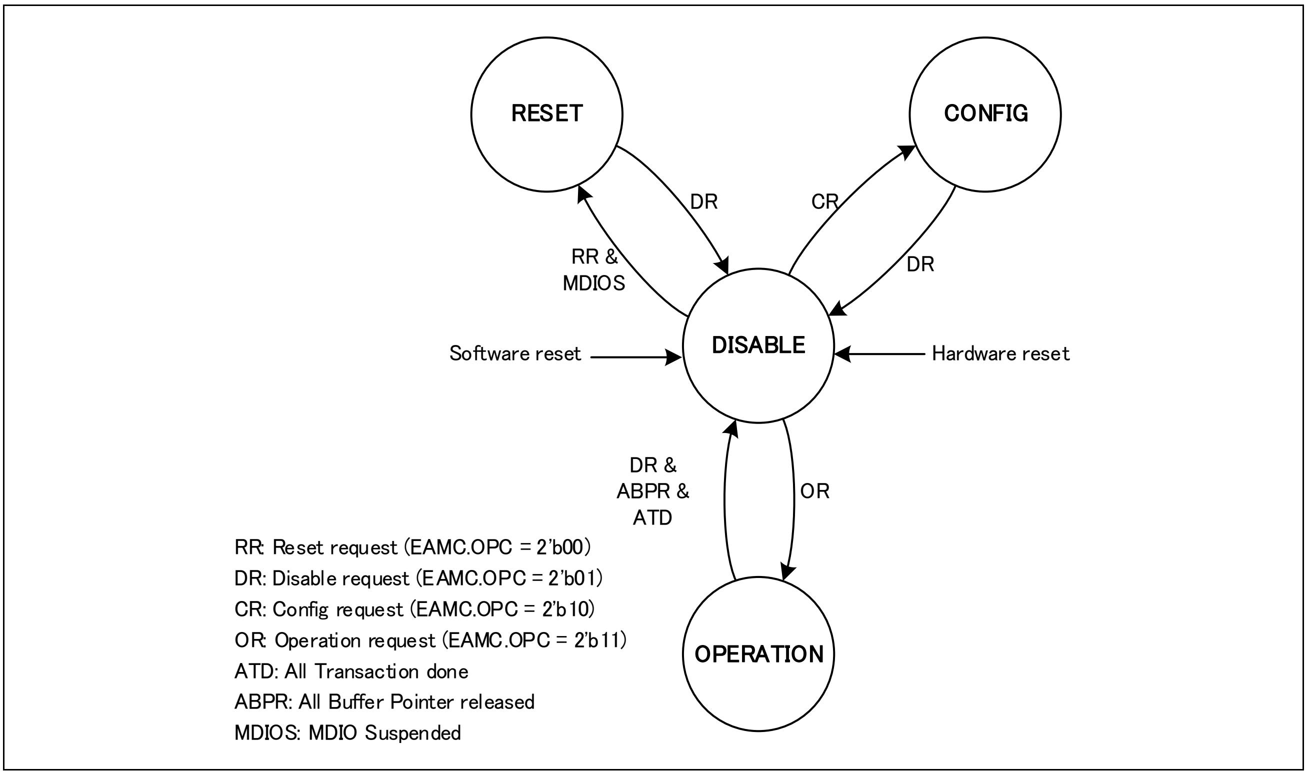

RESET
CONFIG
DR
CR
DR
RR &
MDIOS
DISABLE
Software reset
Hardware reset
DR &
ABPR &
ATD
OR
OPERATION
RR: Reset request (EAMC.OPC = 2'b00)
DR: Disable request (EAMC.OPC = 2'b01)
CR: Config request (EAMC.OPC = 2'b10)
OR: Operation request (EAMC.OPC = 2'b11)
ATD: All Transaction done
ABPR: All Buffer Pointer released
MDIOS: MDIO Suspended

Fig 4.1: Operating mode transitions

Complementary information on transition conditions

ABPR:

- All the buffer pointers contained in the TSNA are release to the forwarding engine [FWD].

ATD:

- All the descriptors already received are processed and corresponding frames are transmitted to the Ethernet PHY.
- All ongoing reception frames are fully sent to the forwarding engine [FWD].

MDIOS:

- MDIO has no ongoing access.

[Restrictions]

- Software shall only trigger transitions shown to Fig 4.1.
  Exp : Don't directory transition from OPERATION to CONFIG.

[Notes]

In case there is no PHY TX/RX clock provided to [RMAC], the transition OPERATION to DISABLED may not be possible. Some products have a function to release this state. Writing 1'b1

MIOC.MIOC[0] [RMAC]. This is a debug (not supported) function for checking the cause.

4.2 Software flows

Restrictions:

SW: Please follow to the flow in this section.

4.2.1 Software flow legend

Software flow legend is described in Fig 4.2.

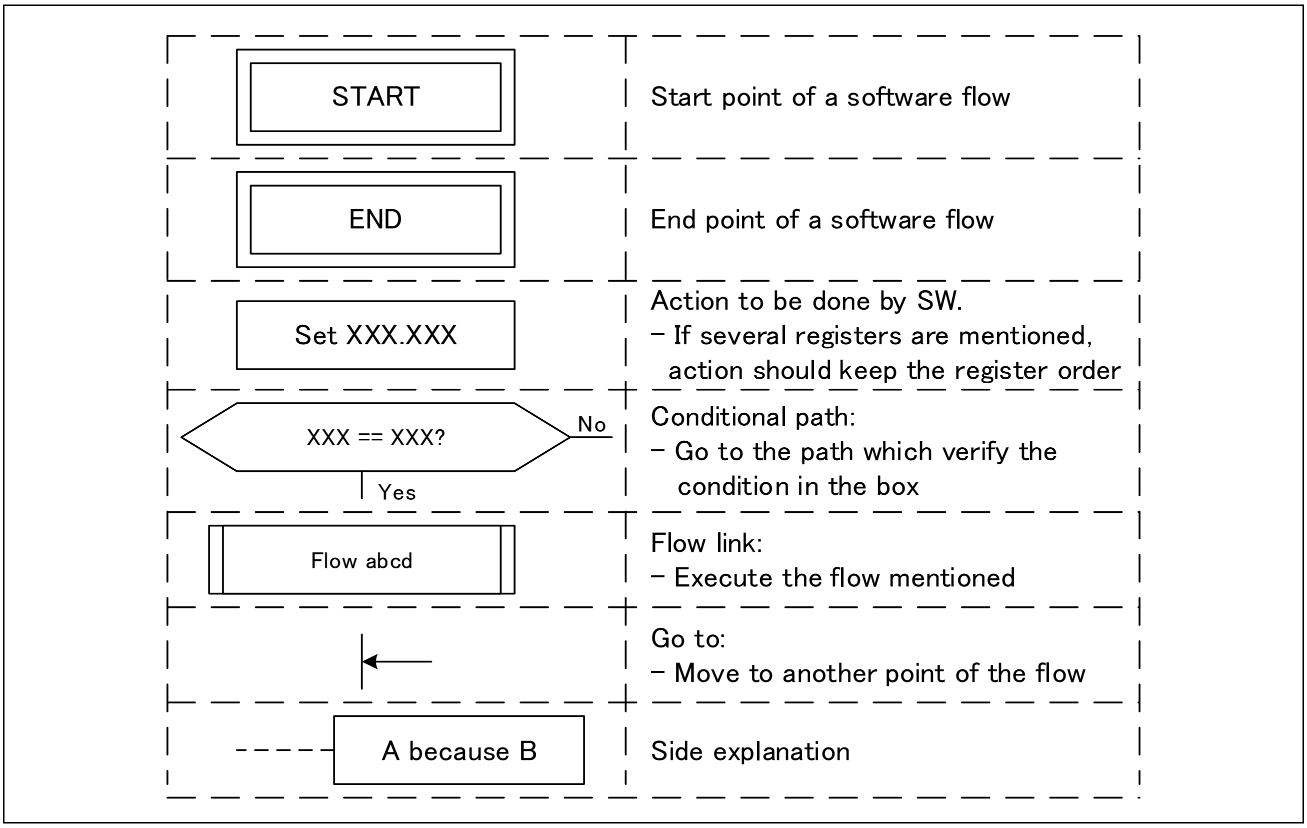

START
Start point of a software flow
END
End point of a software flow
Set XXX.XXX
Action to be done by SW.
– If several registers are mentioned, action should keep the register order
XXX == XXX?
No
Yes
Conditional path:
– Go to the path which verify the condition in the box
Flow abcd
Flow link:
– Execute the flow mentioned
Go to:
– Move to another point of the flow
A because B
Side explanation

Fig 4.2: Software flow legend

4.2.2 Mode transition flow

The mode transition flow is described in Fig 4.3.

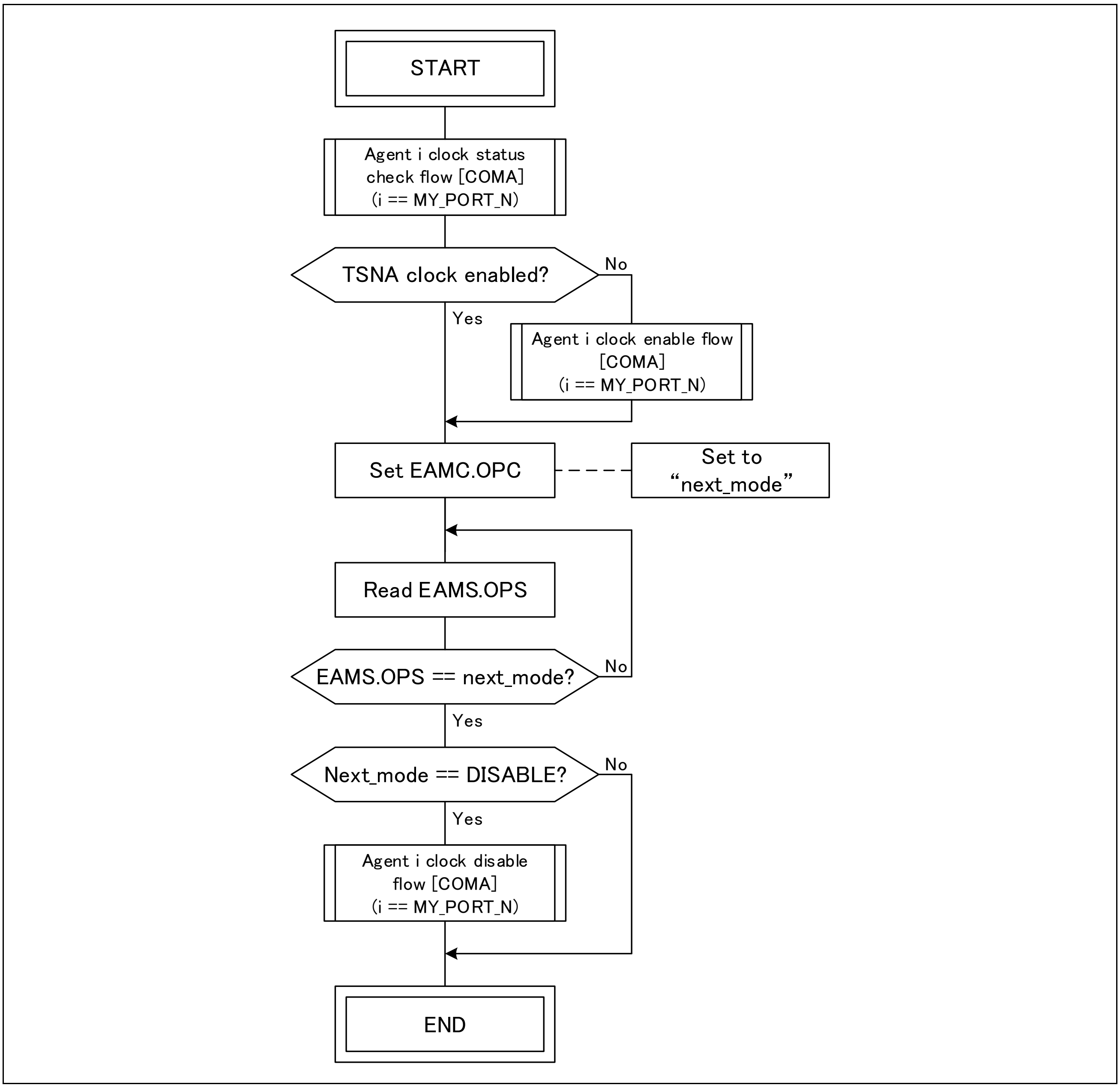

START
Agent i clock status check flow [COMA] (i == MY_PORT_N)
TSNA clock enabled?
No
Yes
Agent i clock enable flow [COMA] (i == MY_PORT_N)
Set EAMC.OPC
Set to "next_mode"
Read EAMS.OPS
EAMS.OPS == next_mode?
No
Yes
Next_mode == DISABLE?
No
Yes
Agent i clock disable flow [COMA] (i == MY_PORT_N)
END

Fig 4.3: Mode transition flow

Notes:

- If SW already know that the clock is enabled, clock status check step can be skipped.
- If SW doesn't need to disable the agent clock in DISABLE mode (because it will go directly to another mode or because the clock never needs to be disabled), clock disable step can be skipped.

4.2.3 Reset flow

The reset flow is described in Fig 4.4.

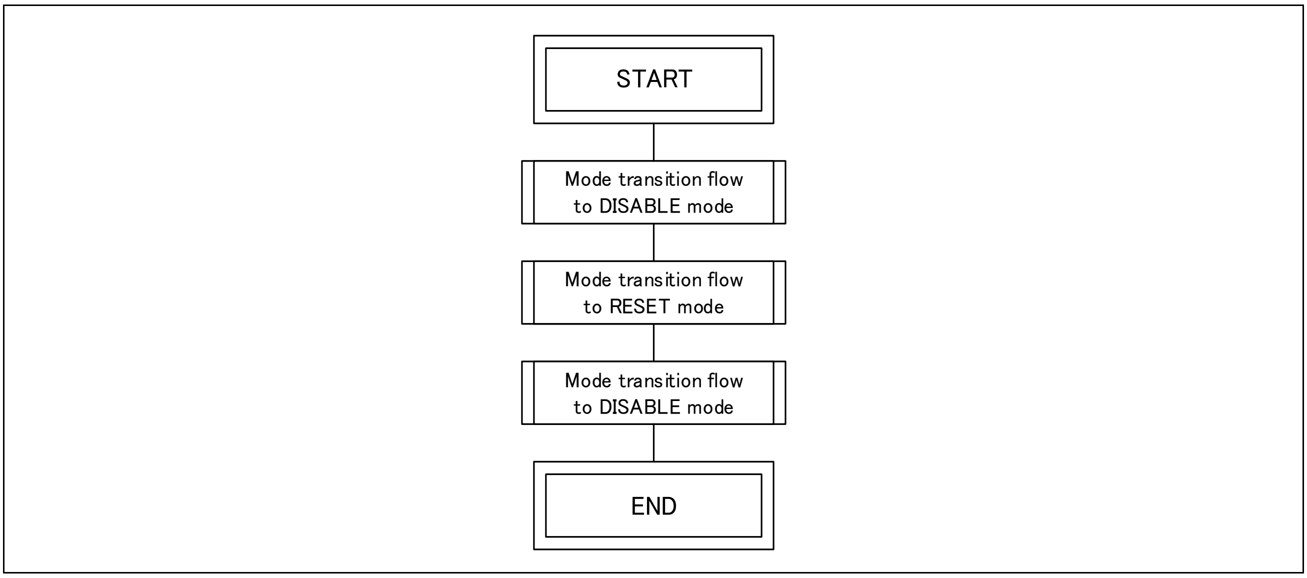

START
Mode transition flow to DISABLE mode
Mode transition flow to RESET mode
Mode transition flow to DISABLE mode
END

Fig 4.4: Reset flow

4.2.4 Initialization flow

The initialization flow is described in Fig 4.5.

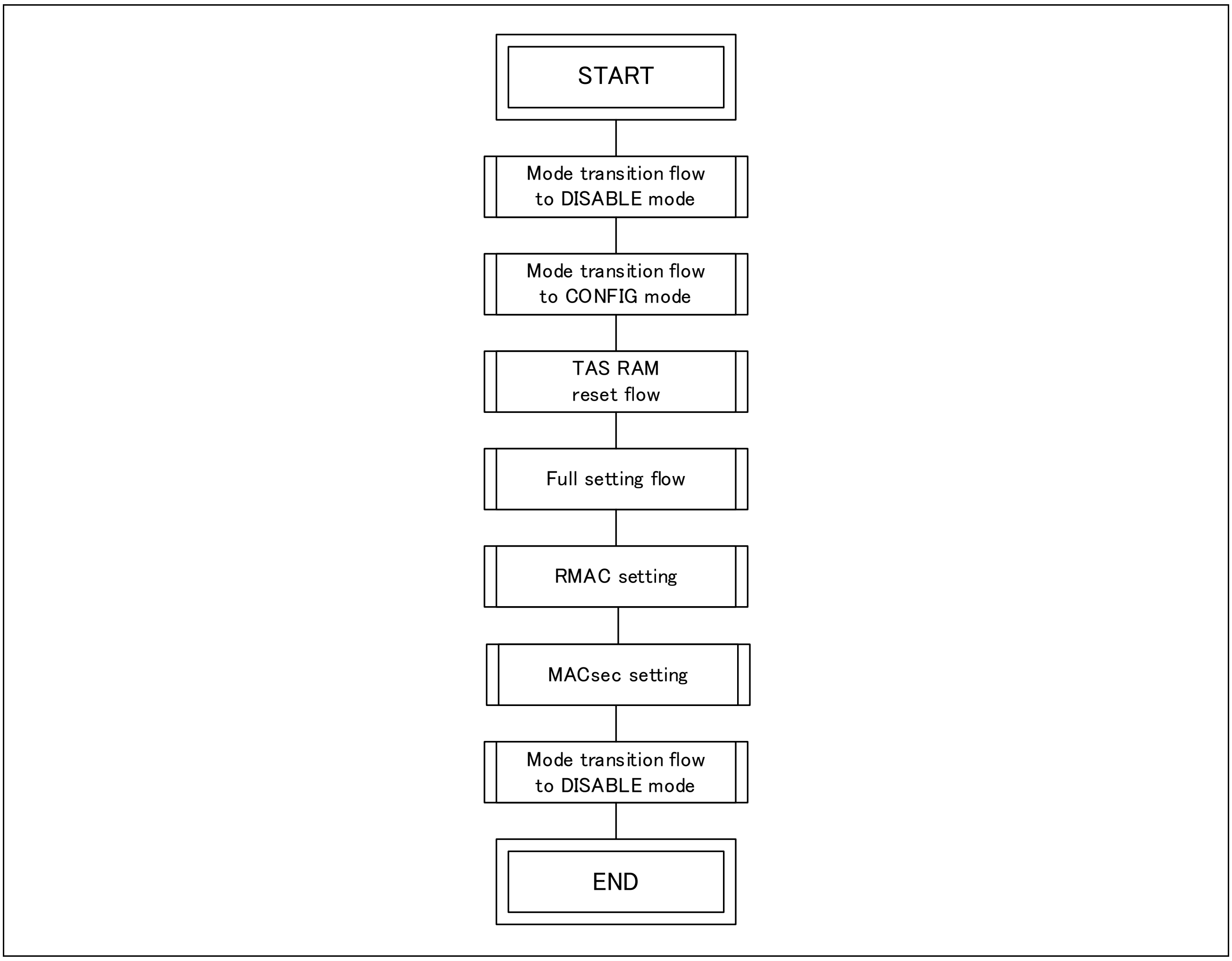

START
Mode transition flow to DISABLE mode
Mode transition flow to CONFIG mode
TAS RAM reset flow
Full setting flow
RMAC setting
MACsec setting
Mode transition flow to DISABLE mode
END

Fig 4.5: Initialization flow

Notes:

- For MACsec setting refer to “Initialization flow” of MACsec specification document [MACsec]
- For RMAC setting refer to RMAC specification document [RMAC]

4.2.5 Reinitialization flow

The reinitialization flow is described in Fig 4.6.

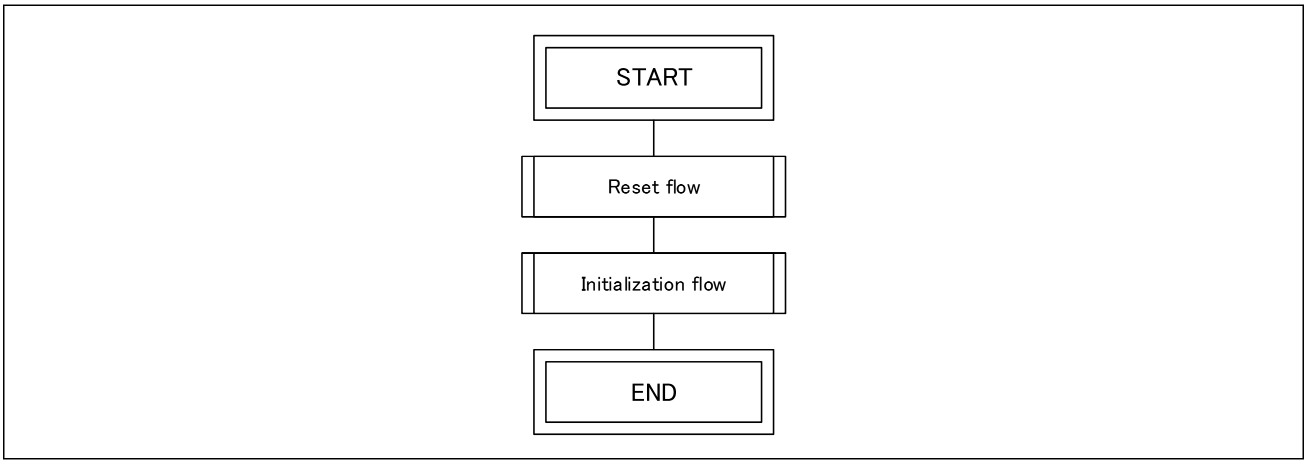


Fig 4.6: Reinitialization flow

4.2.6 TAS RAM reset flow

The TAS RAM reset flow is described in Fig 4.7.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.

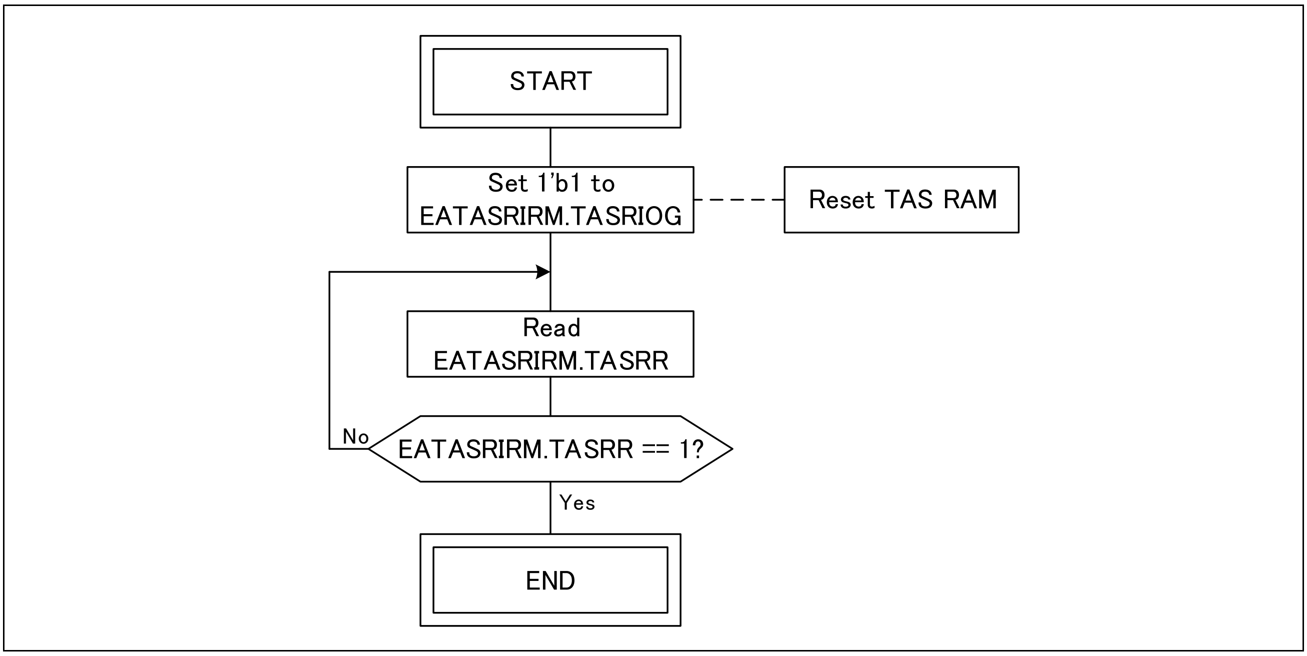


Fig 4.7: TAS RAM reset flow

4.2.7 CBS q setting flow (q=0.. FRM_PRIO_N-1)

The CBS q setting flow is described in Fig 4.8.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.

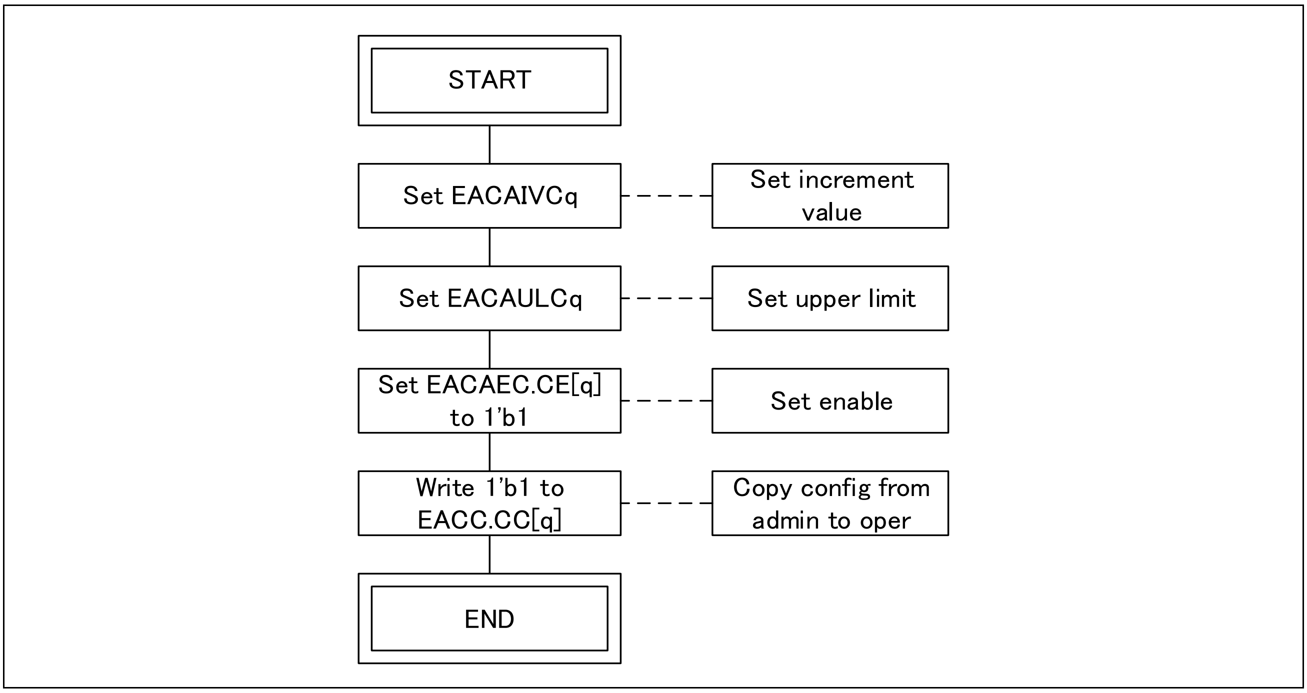


Fig 4.8: CBS q setting flow

4.2.8 CBS q disabling flow (q=0.. FRM_PRIO_N-1)

The CBS q disabling flow is described in Fig 4.9.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.

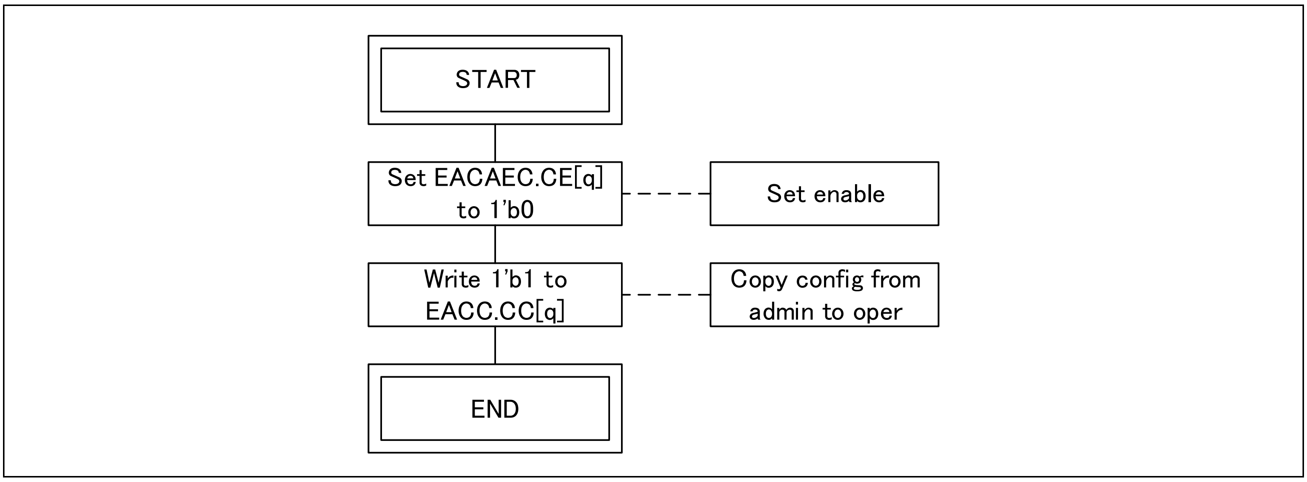


Fig 4.9: CBS q disabling flow

4.2.9 TAS setting flow

The TAS setting (and re-config) flow is described in Fig 4.10.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.

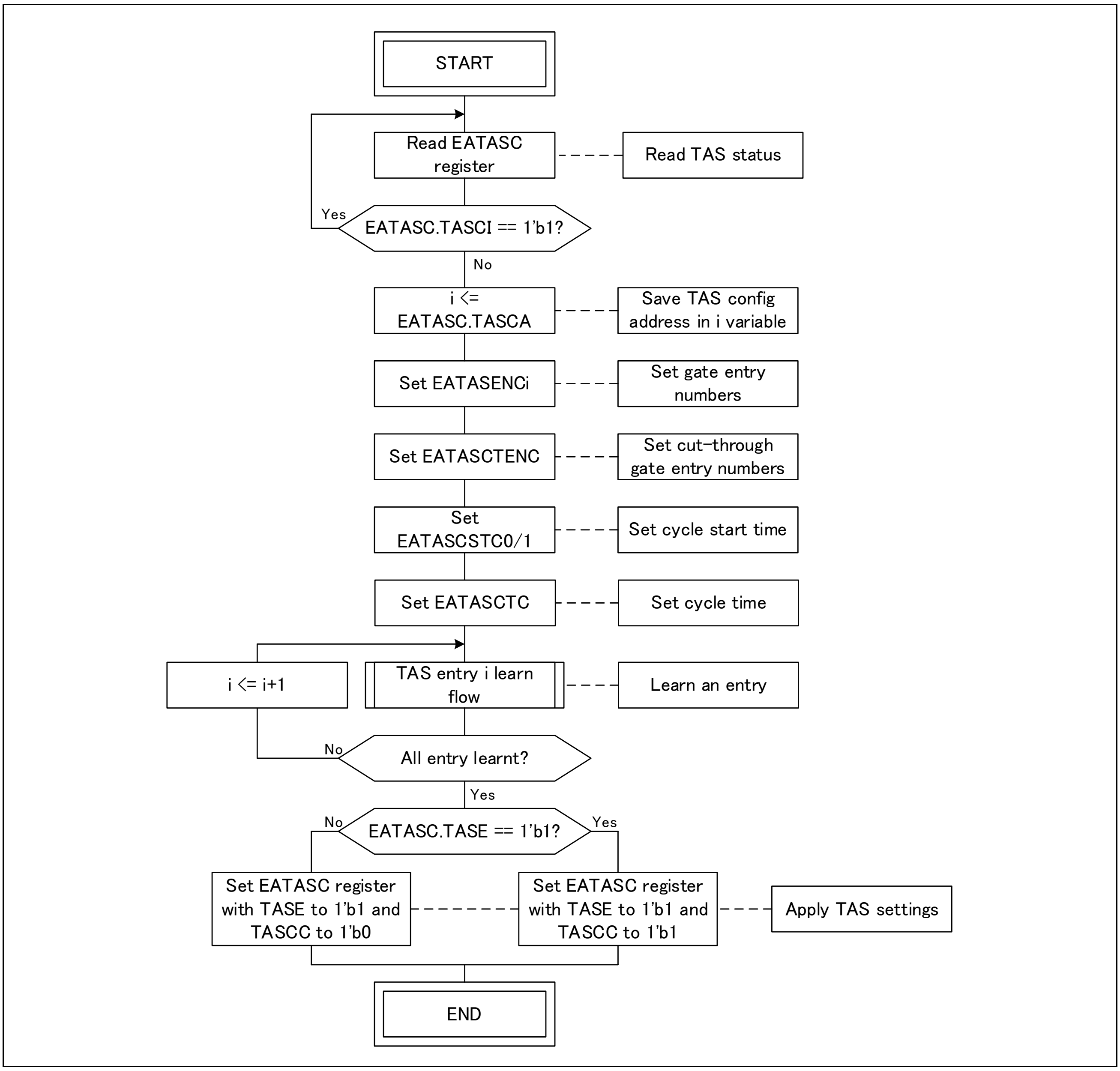


Fig 4.10: TAS setting flow

4.2.10 TAS disabling flow

TAS disabling flow is described in Fig 4.11.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.

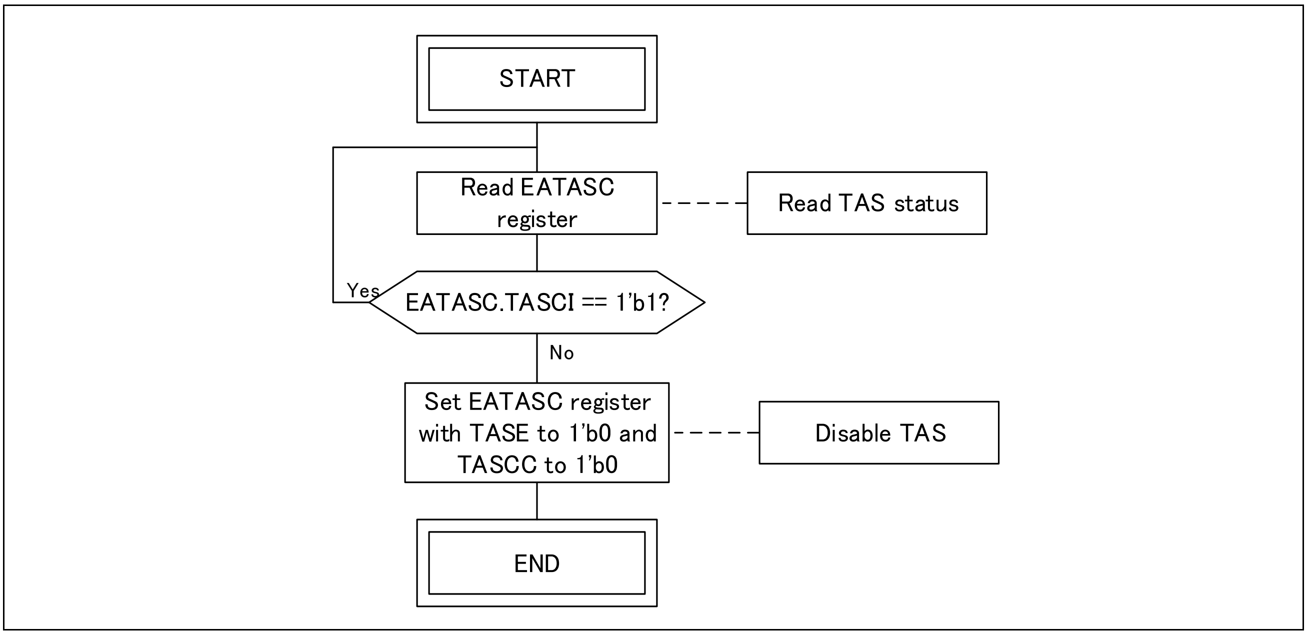


Fig 4.11: TAS disabling flow

4.2.11 TAS enabling flow

TAS enabling flow is described in Fig 4.12.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.
- TAS enabling flow can only be applied after TAS setting flow has been done at least one. By using TAS enabling flow, TAS will start will the schedule previously set with TAS setting flow.

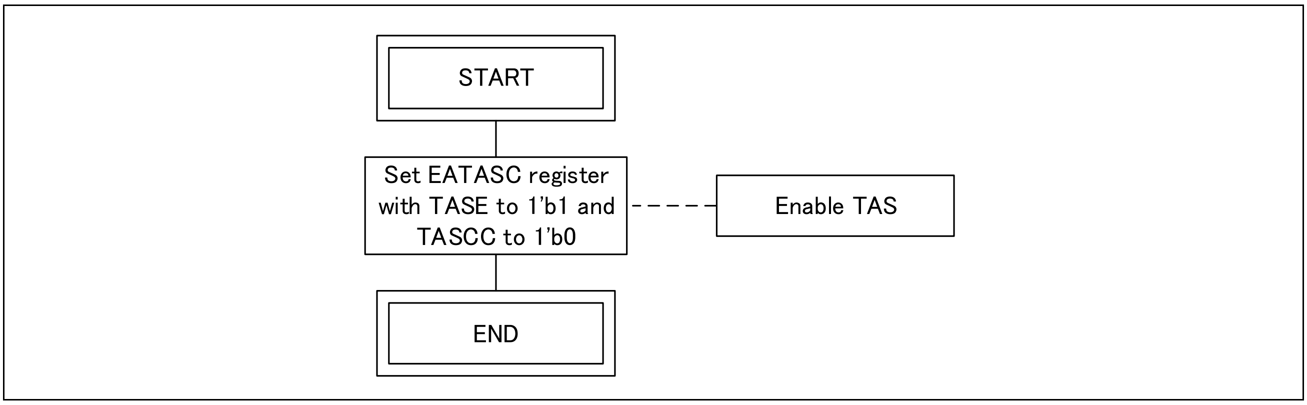


Fig 4.12: TAS enabling flow

4.2.12 TAS entry i learn flow

The TAS entry i learn flow is described in Fig 4.13.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.

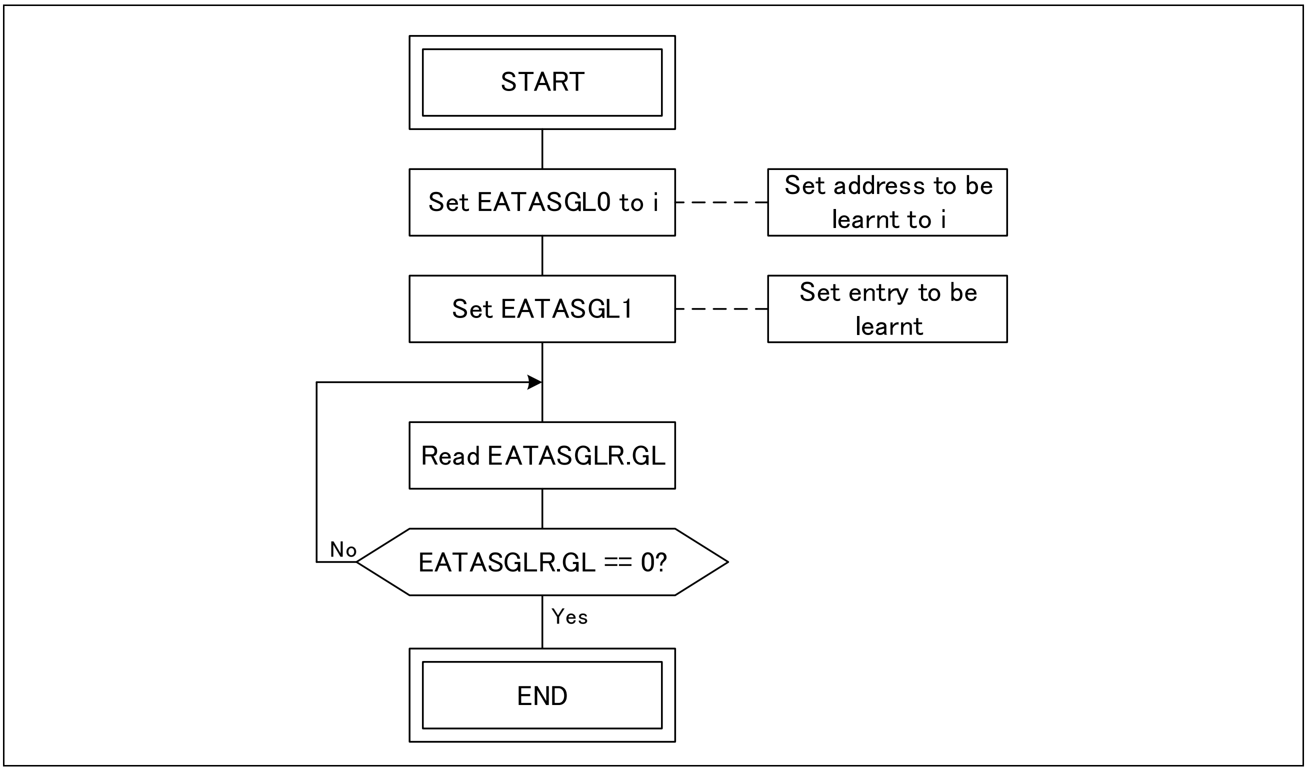


Fig 4.13: TAS entry i learn flow

4.2.13 TAS entry i read flow

The TAS entry i read flow is described in Fig 4.14.

Restrictions:

- This flow is not usable in RESET and DISABLE modes.

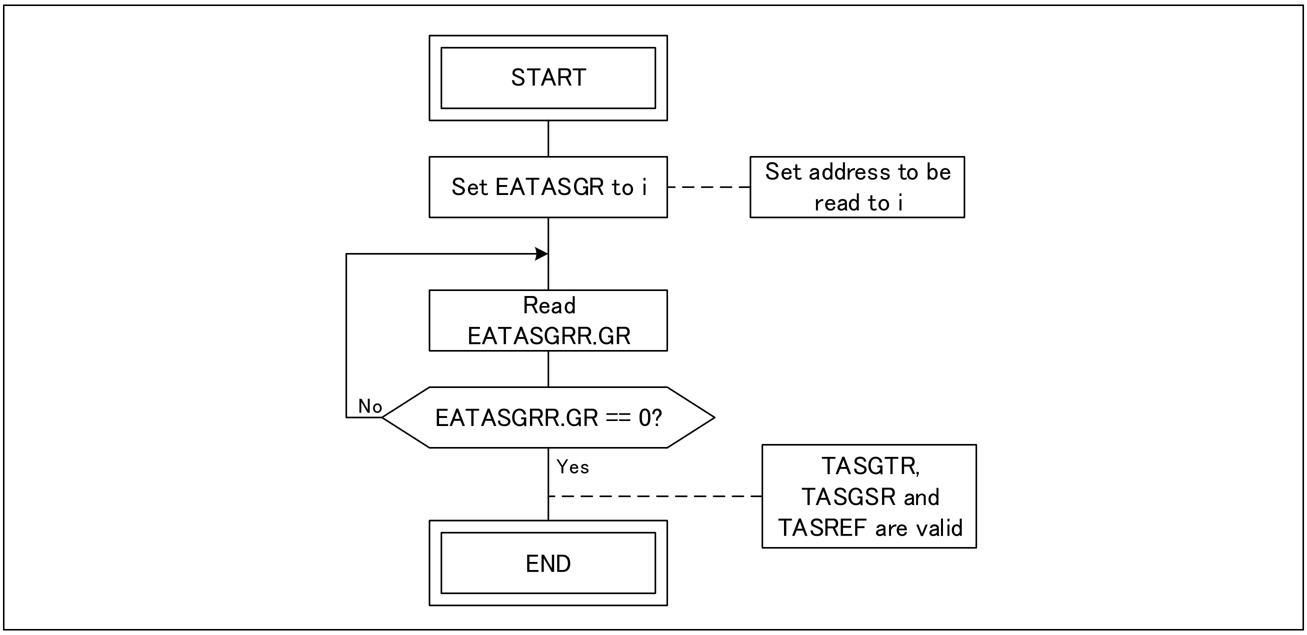

START
Set EATASGR to i
Set address to be read to i
Read EATASGRR.GR
EATASGRR.GR == 0?
No
Yes
TASGTR, TASGSR and TASREF are valid
END

Fig 4.14: TAS entry i read flow

4.2.14 Interrupt handling flow

The interrupt handling flow is described in Fig 4.15.

Restrictions:

- This flow is not usable in RESET mode.

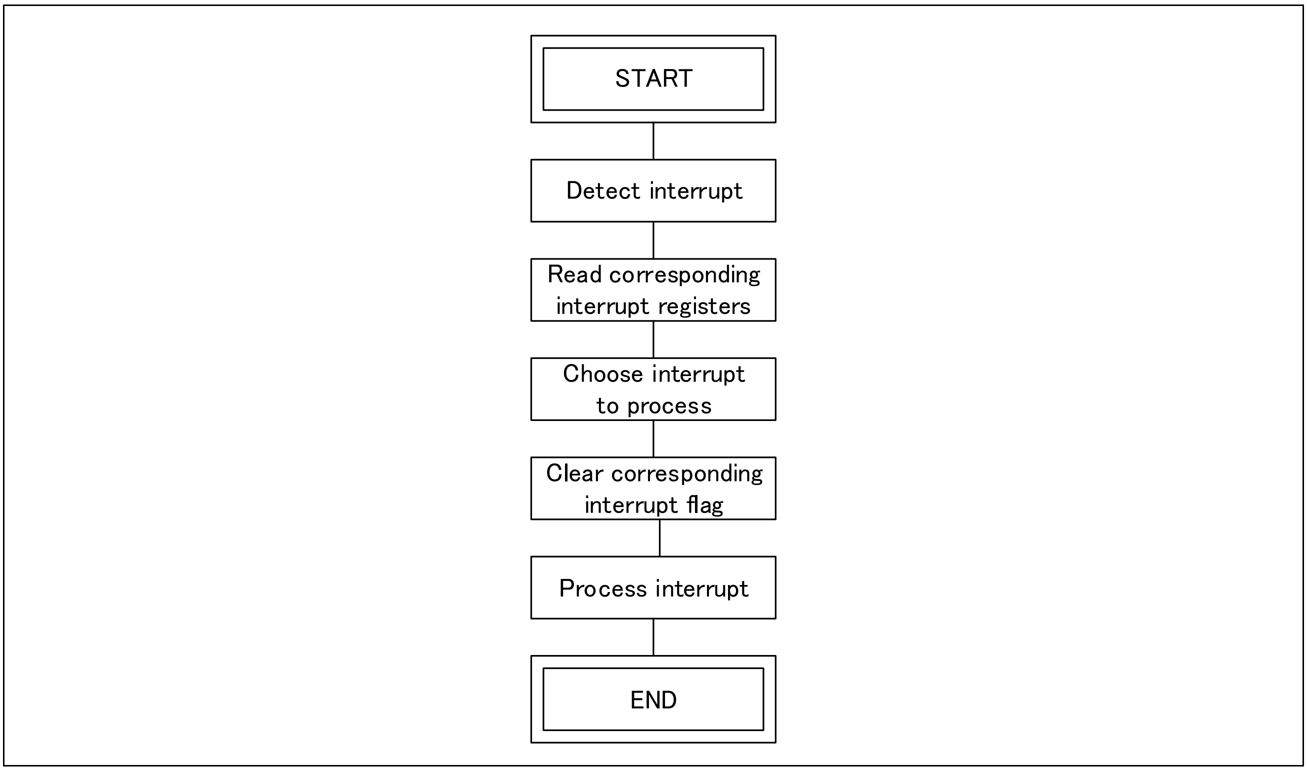

START
Detect interrupt
Read corresponding interrupt registers
Choose interrupt to process
Clear corresponding interrupt flag
Process interrupt
END

Fig 4.15: Interrupt handling flow

4.2.15 Called software flows

The flows described in this section can only be called from other flows thanks to a “flow link” box (Fig 4.2) and cannot be used alone.

4.2.15.1 Full setting flow

The full setting flow is described in Fig 4.16.

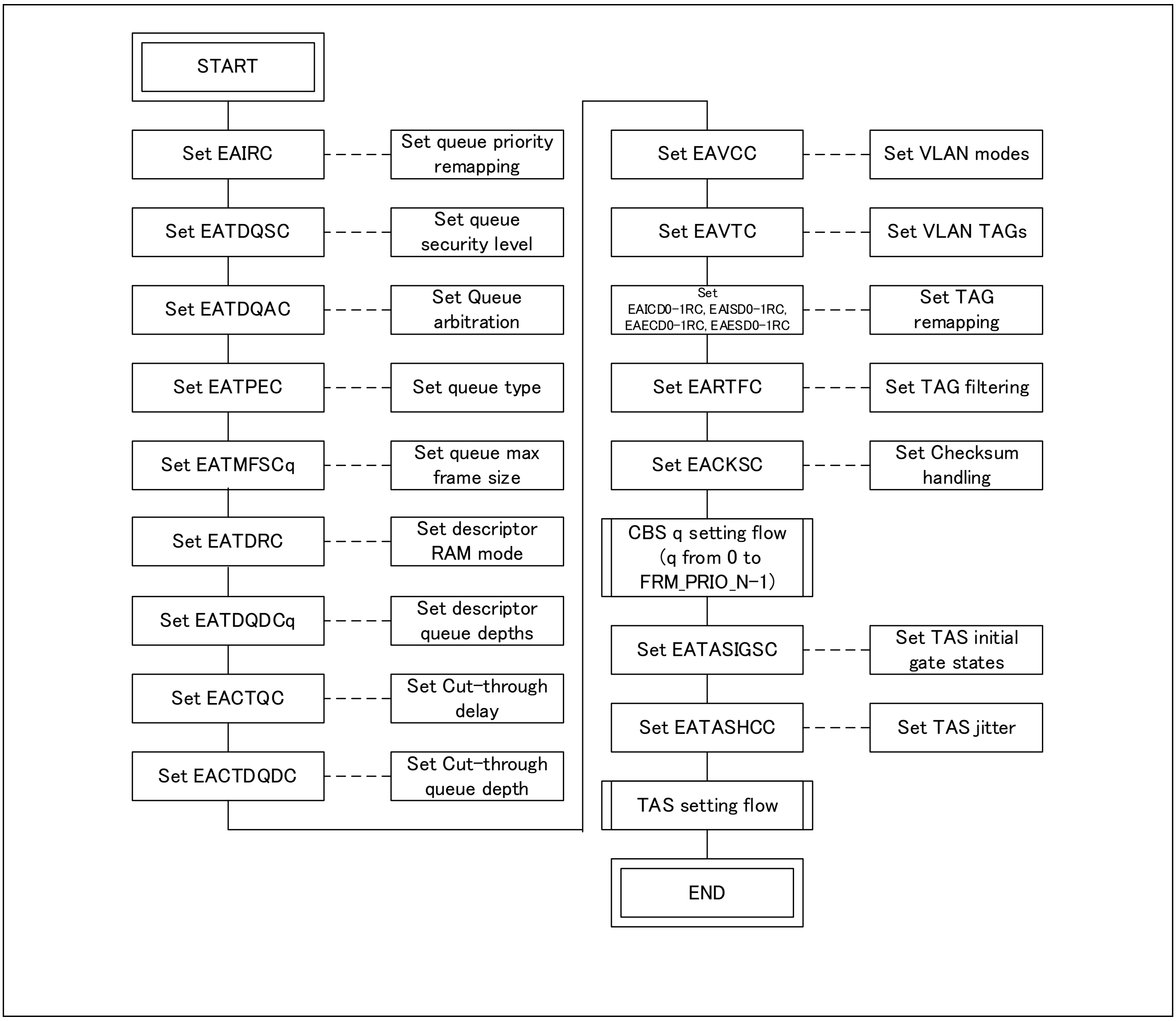


Fig 4.16: Full setting flow

4.2.16 Register writable without software flow

This section describes registers that have not been described so far. These registers can be changed dynamically. (However, it is necessary that the initial settings such as the clock enabling have been completed.)

5. Functional details

5.1 Data Transmission

Ethernet Agent allows data transmission through the TSNA TX data path, The TX data path is described in Fig 5.1.

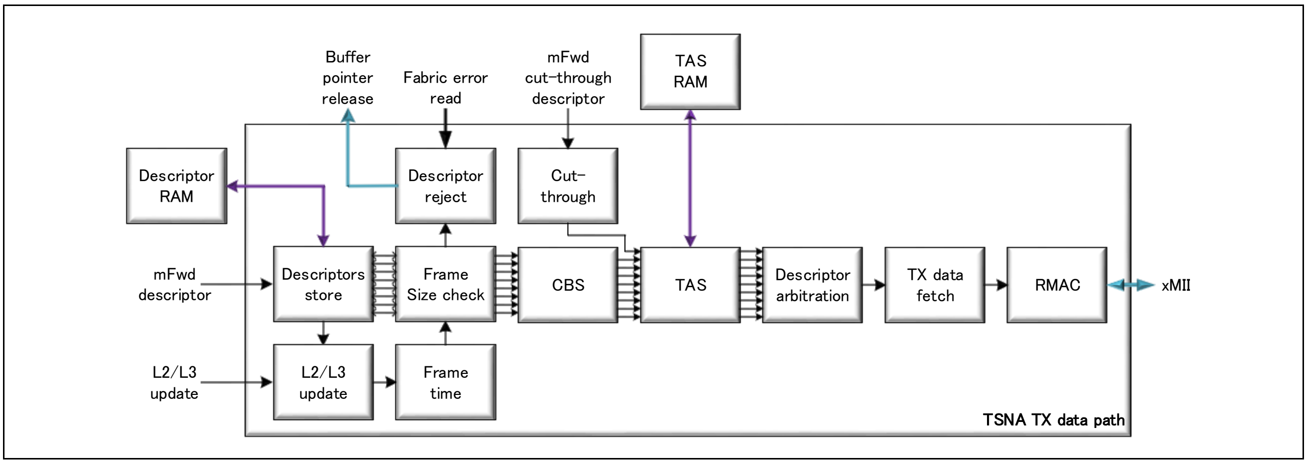


Fig 5.1: TSNA TX data path block diagram

The RX data path is separated in eleven blocks:

- Descriptor store: This block aims are storing the descriptor received from the Forwarding Engine [FWD] in the Descriptor RAM and read it to send it to the L2/L3 update.
- L2/L3 update: This block aims at fetching the L2/L3 update rules from the Forwarding Engine to update the frame while sending it.
- Frame time: This block aims at calculating how many ns a frame would take to be sent on the ethernet PHY for TAS arbitration. This block is here for TAS hardware purpose and its description is not required for switch utilization so it will not de described.
- Frame size check: This block checks the frame size and decide to forward or discard frames.
- Descriptor reject: This block aims at releasing the pointer of the frames rejected by Frame size control block. This block is here for switch hardware purpose and its description is not required for switch utilization so it will not de described.
- Cut-though: This block handles cut-through descriptor coming from forwarding engine [FWD].
- CBS: This block aims at shaping the data traffic per descriptor queue by controlling the data throughput per queue [802.1Qav].
- TAS: This block aims at shaping the data traffic per descriptor queue by controlling the data transmission with a schedule [802.1Qbv].
- Descriptor arbitration: Arbitrate between descriptors based on their descriptor queue using strict priority and/or WRR arbitration.
- TX data fetch: This block fetches the frame data from the local RAM, update it depending on the information obtained by L2/L3 update module and on VLAN control information and release the frame pointers. The pointer release is here for switch hardware purpose and its description is not required for switch utilization so it will not be described.
- RMAC: This block handles the data exchange with the Ethernet PHY. This block functionalities are described in the RMAC specification document [RMAC] so it will not de described.

5.1.1 Descriptor store

Descriptor store aims at storing the descriptors coming from the descriptor bus [FWD] in the descriptor RAM. This function is the same as the “Descriptor storage” function in the “Descriptor store” block in GWCA. Refer to GWCA specification for more details [GWCA] (The correspondence between TSNA and GWCA registers is described in Table 5-1).

Table 5-1: Descriptor store TSNA/GWCA register correspondence

Register/signal name in GWCA [GWCA]	Register name in TSNA
GWRDRC.RDRM	**EATDRC.TDRM**
GWIRC.IPVRi (i=0..7)	**EAIRC.IPVRi** (i=0..7)
GWRDQSC.RDQSL	**EATDQSC.TDQSL**
GWRDQC.RDQD	**EATDQC.TDQD**
GWRDQC.RDQP	**EATDQC.TDQP**
GWRDQDCq.DQDq (q=0.. FRM_PRIO_N-1)	**EATDQDCq.DQDq** (q=0.. FRM_PRIO_N-1)
GWRDQMq.DNQq (q=0.. FRM_PRIO_N-1)	**EATDQMq.DNQq** (q=0.. FRM_PRIO_N-1)
GWRDQMLMq.DMLQq (q=0.. FRM_PRIO_N-1)	**EATDQMLMq.DMLQq** (q=0.. FRM_PRIO_N-1)
GWEIS0.DSECCES	**EAEIS0.DSECCES**
GWEIS1.DQOES	**EAEIS2.DQOES**
GWEIS1.DQSES	**EAEIS2.DQSES**
GWEIS1.DIECCS	**EAEIS2.DIECCS**
GWLDCN.LDN	**EALDCN.LDN**

5.1.2 L2/L3 update

This Block is the same as the “L2/L3 update” block in GWCA. Refer to GWCA specification for more details [GWCA] (The correspondence between TSNA and GWCA registers is described in Table 5-2).

Table 5-2: L2/L3 update TSNA/GWCA register/signal correspondence

Register/signal name in GWCA [GWCA]	Register name in TSNA
GWEIS0.L23UECCES	EAEIS0.L23UECCES

5.1.3 Frame size check

Frame size check aims at checking if a frame has the right properties to be received by CPU using **EATMFSCq** (q=0..FRM_PRIO_N-1) registers.

Functions:

- **EATMFSCq** (q=0.. FRM_PRIO_N-1) registers are used to set the maximum frame size accepted for each descriptor queue. Any frame received for queue q with a size bigger than **EATMFSCq.MFSq** will be discarded and **EAEIS0.FSES[q]** flag will be set. The frame size considered is the size of frame on the PHY.

5.1.4 Cut-through

Cut-through controls the cut-through descriptors queue to store cut-through descriptors received from Forwarding Engine [FWD] using **EATDQC.TCTDQD**, **EACTQC.CTQD** and **EACTDQDC.CTDQD** registers and can be monitored using **EACTDQM.CTQDN** and **EACTDQMLM.CTDMLQ** registers.

Functions:

- **EATDQC.TCTDQD** register is used to disable the cut-through queue. If the cut-through queue is disabled, eha_ct_ready bit will be de-asserted, and no descriptor will be stored in it.
- **EACTQC.CTQD** is used to control the number of clocks that should be waited after receiving a cut-through descriptor to be able to send it for transmission. As a result, the Cut-through send queue that is delayed may be established later than the Mirroring send queue for the same frame.
- **EACTDQDC.CTDQD** register is used to set the maximum number of descriptors that can be stored in the cut-through descriptor queue. If the descriptor queue becomes full (**EACTDQDC.CTDQD** == **EACTDQM.CTQDN**) eha_ct_ready signal will be de-asserted. If a descriptor is received for cut-through descriptor queue while it is full, the descriptor will not be stored and **EAEIS2.CTDQOES** flag will be set.
- **EACTDQM.CTQDN** register is used to monitor the current number of descriptors in the cut-through descriptor queue.
- **EACTDQMLM.CTDMLQ** register is used to monitor the maximum number of descriptors that has been held in cut-through descriptor queue since the previous reset or previous time corresponding this has been read.

Restrictions:

- SW: All the cut-through registers which are not captured under the security level in IP spec will be only accessible by the APB secure interface.
- HW: Cut-through forwarding should not be set for ingress ports in Port-based VLAN mode because cut-through can only forward frame as they were received. (EAVCC.VIM set to 1’b0)

5.1.5 CBS (Credit-based shaper) [802.1Qav]

CBS aims at shaping the data traffic per descriptor queue by controlling the data throughput per queue using **EACAEC.CE**, **EACC.CC**, **EACAIVCq.CIVq** (q = 0..FRM_PRIO_N-1) and **EACAULCq.CULq** (q=0..FRM_PRIO_N-1) registers and can be monitored using **EACOEM.CE**, **EACOIVMq.CIVq** (q=0.. FRM_PRIO_N-1), **EACOULMq.CULq** (q = 0.. FRM_PRIO_N-1) and **EACGSM.CGS** registers and, **EAEIS1.CULES** interrupt register.

Functions:

- **EACAEC.CE** register is used to enable/disable CBS modules.
- **EACAIVCq.CIVq** register is used to configure descriptor queue q CBS throughput.
- **EACAULCq.CULq** register is used to configure the maximum credit number that can be stored in descriptor queue q CBS module. It is used to monitor CBS module corresponding descriptor queue has been applied a too long back pressure due to high priority non-CBS-Shaped queues interference traffic or due to queue open time partial overlapping in TAS. If too much back pressure has been applied to a CBS related descriptor queue, corresponding **EAEIS1.CULES** flag will be set.
- **EACC.CC** register is used to apply configurations set in **EACAEC.CE**, **EACC.CC**, **EACAIVCq.CIVq** and **EACAULCq.CULq** register by coping them to **EACOEM.CE**, **EACOIVMq.CIVq** and **EACOULMq.CULq** registers.
- **EACOEM.CE** register is used to monitor which descriptor queue as its CBS module enabled.
- **EACOIVMq.CIVq** register is used to monitor the current throughput set for descriptor queue q.
- **EACOULMq.CULq** register is used to monitor the current maximum credit number for descriptor queue q.
- **EACGSM.CGS** register is used to monitor which queue is allowed by CBS to transmit.
- **EAEIS1.CULES** register is used to flag CBS Upper Limit Errors.
- When using **MACsec**, the frame output by PHY-IF is expected to increase by 32 bytes or more compared to the frame output by MAC(ETHA)-IF. Please set **EACOIVMq.CIVq** taking this increase in frame amount. Specifically, "Credit of actual frame size = Subtracted credit + Credit of MACsec dedicated", so the actual transfer rate will be higher than the setting.

5.1.5.1 CBS setting

CBS setting should follow equation (1) and equation (2). While setting descriptor queue q CBS, CIV should be set to **EACAIVCq.CIVq** register and CUL should be set to **EACAULCq.CULq** register.

*(1) CIV[byte/cycle] = ((portTransmitRate[bps] / 8) * (bandwidthFraction[%] / 100)) / clk_f[Hz] * (CycleTime / GateOpenTime)*

For example, when *portTransmitRate* is 1Gbps, *bandwidthFraction* is 25% and clk_f is 200MHz, CIV is 0.15625. CIV[19:16] = 0 CIV[15:0] = 0.15625 * 2^16 = 10,240 = 2800H

*(2) CUL[byte] = (maxInterferenceSize[bit] * clk_f[Hz] / portTransmitRate[bps] + requestDelay[cyc]) * CIV[byte/cycle]*

Where:

- *portTransmitRate* is the TSNA link throughput [RMAC].
- *bandwidthFraction* is the percentage of bandwidth that the CBS corresponding should use.
- *clk_f* is the clock clk frequency.
- *(CycleTime / GateOpenTime)* is the ratio between the current TAS cycle time (**EATASCTM.TASOCT**) and the time the CBS corresponding descriptor queue TAS gate is opened during

EATASCTM.TASOCT time. When TAS is disabled, *(CycleTime / GateOpenTime)* is equal to 1.

- *maxInterferenceSize* is the maximum size of any burst of traffic that can delay the transmission of a frame that is available for transmission for this traffic class (refer to section 5.1.5.2).
- *requestDelay* is the time required from descriptor reception to frame transmission ready and is fixed to 50 cycles.

Restrictions:

- SW: All CBS *bandwidthFraction* sum should be smaller than 70% for queues with simultaneously opened TAS gates.

5.1.5.2 Maximum interference size calculation

The maximum interface *maxInterferenceSize* size can be calculated for a queue q using per equation (3). Each interference size includes preamble and inter frame gap (IFG).

(3) *maxInterferenceSize* = *queueInterferenceSizeLow* + *queueInterfereSize* + Σ_i (*queueInterferenceSizeHigh*[i])

Where:

- *queueInterferenceSizeLow* is the interference created by all the queues which have a priority smaller than q. This interface size is equal to (queueLowMaxSize+20)*8 where *queueLowMaxSize* is **EATMFSCi.MFSi** maximum values for all i smaller than q.
- *queueInterfereSize* is the interfere created by queue q and is equal to (**EATMFSCq.MFSq**+20)*8
- *queueInterferenceSizeHigh*[i] is the interference created by queue i with a higher priority than q. If CBS is enabled for queue i, *queueInterferenceSizeHigh*[i] is equal to (**EATMFSCi.MFSi** + *queueLowMaxSize* + 40)*8 where *queueLowMaxSize* is **EATMFSCj.MFSj** maximum values for all j smaller than i. If CBS is disabled for queue i, *queueInterferenceSizeHigh*[i] is equal to (*MaxFrameBurst****EATMFSCi.MFSi** + *queueLowMaxSize* + 40)*8 where *queueLowMaxSize* is **EATMFSCj.MFSj** maximum values for all j smaller than i and where *MaxFrameBurst* is the maximum number of frames that can be expected in a burst. *MaxFrameBurst* is a variable which is system dependent and cannot de defined here.

5.1.6 TAS (Time Aware Shaper) [802.1Qbv]

TAS aims at shaping the data traffic per descriptor queue by controlling a AVTP based schedule using **EATASC**, **EATASIGSC**, **EATASENCi** (i=0..FRM_PRIO_N-1), **EATASCTENC**, **EATASCSTC0/1**, **EATASCTC**, **EATASGL0/1**, **EATASGLR**, **EATASHCC** registers and can be monitored using **ENTASENMi** (q=0..FRM_PRIO_N-1) , **EATASCTENM**, **EATASCSTM0/1**, **EATASCTM**, **EATASGR**, **EATASGRR**, **EATASRIRM**, **EATASSM** registers and, **EAIEIS0.TASGEES**, **EAIEIS0.TASCTGEES**, **EAIEIS1.TASGES** and **EAIEIS1.TASCTGES** interrupt registers.

Functions:

- **EATASC** register is used for TAS configuration and enabling.
- **EATASIGSC** register is used to set the initial gate states.
- **EATASENCi** register is used to set the number of entries used in TAS RAM by gate i.
- **EATASCTENC** register is used to set the number of entries used in TAS RAM by cut-through gate.
- **EATASCSTC0/1** registers are used to set the cycle start time.
- **EATASCTC** register is used to set the cycle time.
- **EATASGL0/1** and **EATASGLR** registers are used to write entries in the TAS RAM.
- **EATASHCC** is used to set the TAS jitter caused by to the switch architecture.

- **ENTASENMi** register is used to monitor the number of entries used in the TAS RAM by gate i for the ongoing schedule.
- **EATASCTENM** register is used to monitor the number of entries used in the TAS RAM by cut-through gate for the ongoing schedule.
- **EATASCSTM0/1** registers are used to monitor the next cycle start time for the ongoing schedule.
- **EATASCTM** register is used to monitor the cycle time for the ongoing schedule.
- **EATASGR** and **EATASGRR** registers are used to read entries in the TAS RAM.
- **EATASRIRM** register is used to initialize TAS RAM entries to All0.
- **EATASSM** register is used to monitor TAS operation.
- **EAIEIS0.TASGEES** and **EAIEIS0.TASCTGEES** registers are used to monitor gate which returned in initial gate state because of an ECC error.
- **EAIEIS1.TASGES** and **EAIEIS1.TASCTGES** registers are used to monitor gates errors due to a lack of time to read the TAS RAM.

5.1.6.1 TAS general behavior

TAS module is a scheduler which allows frames to be transmitted depending on their descriptor queue and time by allowing frame transmission when its corresponding gate is opened. TAS scheduler uses as time reference a 64-bit AVTP timer (derived from gPTP timer [gPTP]). TAS general behavior can be summed up in four categories, TAS enabling, TAS configuration change, TAS disabling, and gPTP offset correction recovery [gPTP] explained in this section and TAS schedule itself is explained in section 5.1.6.2.

Restrictions:

- HW: TAS module does not support 64-bit AVTP timer overflows.

(1) TAS enabling

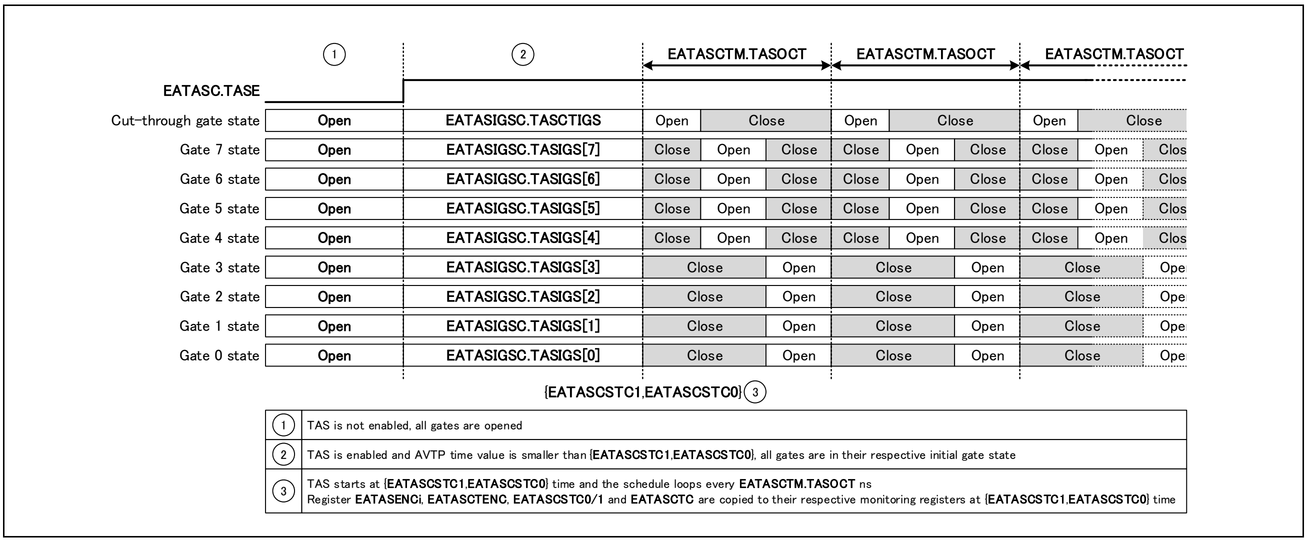
TAS enabling happens when TAS is set while being disabled (**EATASC.TASE** not set).

(a) TAS enabling in the future

TAS module can be such as its schedule starts in the future. TAS behavior when its schedule starts in the future is described in Fig 5.2. The schedule used in this figure is just an example and not correspond to anything relevant for TAS general behavior explanation.

Restrictions:

- SW: To enable TAS in the future, it should be insured that **EATASC.TASE** gets set before the AVTP timer reaches the following time: {**EATASCSTC1**, **EATASCSTC0**} – 2***EATASCTC.TASACT**.


① ② EATASCTM.TASOCT EATASCTM.TASOCT EATASCTM.TASOCT
EATASC.TASE
Cut-through gate state Open EATASIGSC.TASCTIGS Open Close Open Close Open Close
Gate 7 state Open EATASIGSC.TASIGS[7] Close Open Close Close Open Close Close Open Clos
Gate 6 state Open EATASIGSC.TASIGS[6] Close Open Close Close Open Close Close Open Clos
Gate 5 state Open EATASIGSC.TASIGS[5] Close Open Close Close Open Close Close Open Clos
Gate 4 state Open EATASIGSC.TASIGS[4] Close Open Close Close Open Close Close Open Clos
Gate 3 state Open EATASIGSC.TASIGS[3] Close Open Close Open Close Ope
Gate 2 state Open EATASIGSC.TASIGS[2] Close Open Close Open Close Ope
Gate 1 state Open EATASIGSC.TASIGS[1] Close Open Close Open Close Ope
Gate 0 state Open EATASIGSC.TASIGS[0] Close Open Close Open Close Ope
{EATASCSTC1,EATASCSTC0} ③

①	TAS is not enabled, all gates are opened
②	TAS is enabled and AVTP time value is smaller than {**EATASCSTC1**,**EATASCSTC0**}, all gates are in their respective initial gate state
③	TAS starts at {**EATASCSTC1**,**EATASCSTC0**} time and the schedule loops every **EATASCTM.TASOCT** ns Register **EATASENCi**, **EATASCTENC**, **EATASCSTC0/1** and **EATASCTC** are copied to their respective monitoring registers at {**EATASCSTC1**,**EATASCSTC0**} time

Fig 5.2: TAS behavior when its schedule starts in the future

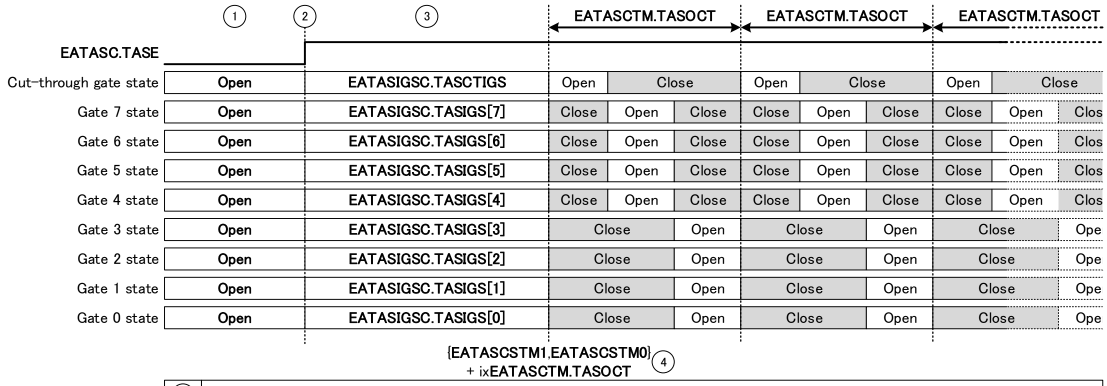
Restrictions:

- HW: When **EATASC.TASE** gets set an ongoing frame would cross a gate close state if its initial gate state is closed.
- HW: If **EATASC.TASE** gets set such as an ongoing frame does not have the time to finish transmitting before {**EATASCSTC1**, **EATASCSTC0**} is reached, the ongoing frame could cross a gate close state in the schedule first cycle.
- HW: If **EATASC.TASE** gets set between {**EATASCSTC1**, **EATASCSTC0**} – 2***EATASCTC.TASACT** and {**EATASCSTC1**, **EATASCSTC0**} there is a possibly that enabling could not happen properly and

TAS module will recover during 2*cycle_time[ns] from the start of the new cycle where cycle_time is the schedule cycle time set in **EATASCTC**.

(b) TAS enabling in the past

TAS module can be such as its schedule should have started in the past. TAS behavior when its schedule should have started in the past is described in Fig 5.3. The schedule used in this figure is just an example and not correspond to anything relevant for TAS general behavior explanation.



1	TAS is not enabled, all gates are opened
2	TAS gets enabled in the past Register **EATASENCi**, **EATASCTENC**, **EATASCSTC0/1** and **EATASCTC** are copied to their respective monitoring registers at **EATASC.TASE** set time
3	TAS recovery time. TAS needs time to calculate where it can restart safely.
4	TAS starts at {**EATASCSTM1**,**EATASCSTM0**} + ix**EATASCTM.TASOCT** time where i is an integer and the schedule loops every **EATASCTM.TASOCT** ns

Fig 5.3: TAS behavior when its schedule should have started in the past

TAS recovery time (*tas_recovery_time*) maximum value can be calculated by equation (4).

(4) *tas_recovery_time[ns] = clk_period[ns]*(tase_set_time[ns] – start_time[ns])/cycle_time[ns]* + 2**cycle_time[ns]*

Where:

- *clk_period* is the clock clk period.
- *tase_set_time* is the AVTP timer time at which the **EATASC.TASE** has been set.
- *start_time* is the AVTP timer time at which the schedule has been set to start ({**EATASCSTC1**, **EATASCSTC0**})
- *cycle_time* is the schedule cycle time set in **EATASCTC**.

Functions:

- Equation (4) comes from the fact that TAS will switch its start time by *cycle_time* every clock during recovery. A problem resulting from that is that the recovery time increase proportionality with *tase_set_time[ns] – start_time[ns].* To avoid this issue, TAS is able to switch its start time by 256^{i}**cycle_time* (i=0..6) every clock when start time is too far in the past.

Restrictions:

- HW: Because of the start setting in the past, the hardware guardband cannot be ensured from **EATASC.TASE** setting time to the end of the first cycle (frames could cross a gate close state).
- HW: Because of the start setting in the past, **EAEIS1.TASGES** and/or **EAEIS1.TASCTGES** gate errors

could be set.

(2) TAS configuration change

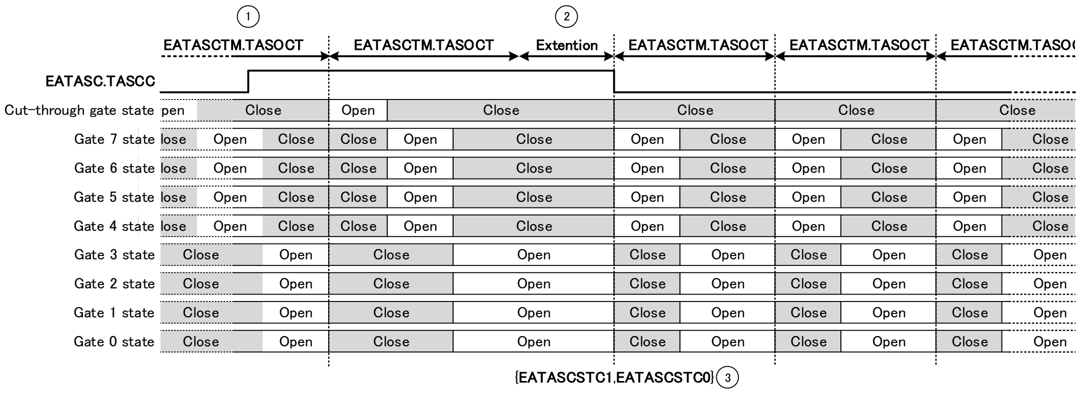
TAS configuration change happens when TAS is set while being enabled (**EATASC.TASE** already set).

(a) TAS configuration change in the future

TAS module can be such as its schedule is re-configured in the future. TAS behavior when its schedule is re-configured in the future is described in Fig 5.4. The schedule used in this figure is just an example and not correspond to anything relevant for TAS general behavior explanation.

Restrictions:

- SW: To enable TAS in the future, it should be insured that **EATASC.TASCC** gets set before the AVTP timer reaches the following time: {**EATASCSTC1**, **EATASCSTC0**} – 2***EATASCTC.TASACT**.



①
②
EATASCTM.TASOCT
EATASCTM.TASOCT
Extention
EATASCTM.TASOCT
EATASCTM.TASOCT
EATASCTM.TASO(
EATASC.TASCC
Cut-through gate state: pen, Close, Open, Close, Close, Close, Close
Gate 7 state: lose, Open, Close, Close, Open, Close, Open, Close, Open, Close, Open, Close
Gate 6 state: lose, Open, Close, Close, Open, Close, Open, Close, Open, Close, Open, Close
Gate 5 state: lose, Open, Close, Close, Open, Close, Open, Close, Open, Close, Open, Close
Gate 4 state: lose, Open, Close, Close, Open, Close, Open, Close, Open, Close, Open, Close
Gate 3 state: Close, Open, Close, Open, Close, Open, Close, Open, Close, Open
Gate 2 state: Close, Open, Close, Open, Close, Open, Close, Open, Close, Open
Gate 1 state: Close, Open, Close, Open, Close, Open, Close, Open, Close, Open
Gate 0 state: Close, Open, Close, Open, Close, Open, Close, Open, Close, Open
{EATASCSTC1,EATASCSTC0}③

①	TAS configuration change gets set during the ongoing schedule
②	If there is not enough time to complete a cycle for the ongoing schedule and the current cycle is finished the last gate state is extended until {**EATASCSTC1**,**EATASCSTC0**}
③	TAS starts at {**EATASCSTC1**,**EATASCSTC0**} time and the new schedule loops every **EATASCTM.TASOCT** ns Register **EATASENCi**, **EATASCTENC**, **EATASCSTC0/1** and **EATASCTC** are copied to their respective monitoring registers at {**EATASCSTC1**,**EATASCSTC0**} time

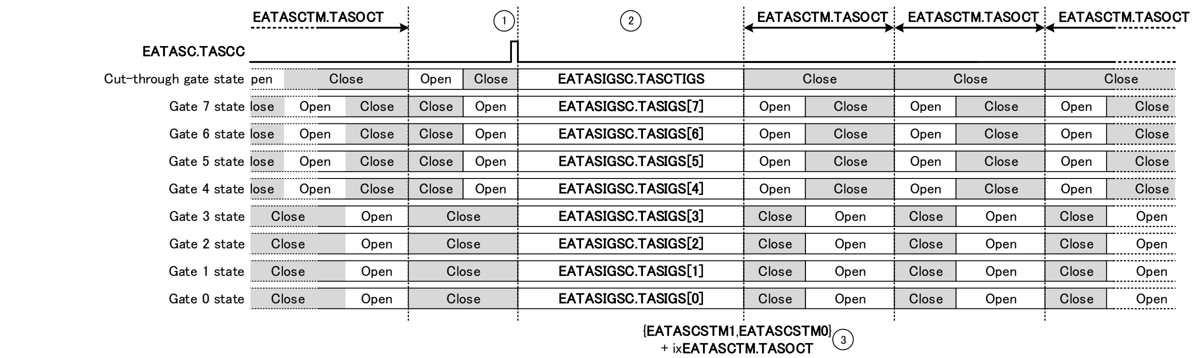
Fig 5.4: TAS behavior when its schedule is re-configured in the future

Restrictions:

- HW: If **EATASC.TASCC** gets set between {**EATASCSTC1**, **EATASCSTC0**} – 2***EATASCTC.TASACT** and {**EATASCSTC1**, **EATASCSTC0**} there is a possibly that configuration change could not happen properly and TAS module will recover during 2*cycle_time[ns] from the start of the new cycle where cycle_time is the schedule cycle time set in **EATASCTC**.

(b) TAS configuration change in the past

TAS module can be such as its schedule re-configuration should have happened in the past. TAS behavior when its schedule re-configuration should have happened in the past is described in Fig 5.5. The schedule used in this figure is just an example and not correspond to anything relevant for TAS general behavior explanation.


EATASCTM.TASOCT
①
②
EATASCTM.TASOCT
EATASCTM.TASOCT
EATASCTM.TASOCT
EATASC.TASCC
Cut-through gate state pen Close Open Close EATASIGSC.TASCTIGS Close Close Close
Gate 7 state lose Open Close Close Open EATASIGSC.TASIGS[7] Open Close Open Close Open Close
Gate 6 state lose Open Close Close Open EATASIGSC.TASIGS[6] Open Close Open Close Open Close
Gate 5 state lose Open Close Close Open EATASIGSC.TASIGS[5] Open Close Open Close Open Close
Gate 4 state lose Open Close Close Open EATASIGSC.TASIGS[4] Open Close Open Close Open Close
Gate 3 state Close Open Close EATASIGSC.TASIGS[3] Close Open Close Open Close Open
Gate 2 state Close Open Close EATASIGSC.TASIGS[2] Close Open Close Open Close Open
Gate 1 state Close Open Close EATASIGSC.TASIGS[1] Close Open Close Open Close Open
Gate 0 state Close Open Close EATASIGSC.TASIGS[0] Close Open Close Open Close Open
{EATASCSTM1,EATASCSTM0}
+ ixEATASCTM.TASOCT
③

①	TAS configuration change gets set during the ongoing schedule and start time is in the past Configuration change gets cleared right away Register **EATASENCi**, **EATASCTENC**, **EATASCSTC0/1** and **EATASCTC** are copied to their respective monitoring registers
②	TAS recovery time. TAS needs time to calculate where it can restart safely.
③	TAS starts at {**EATASCSTM1**,**EATASCSMC0**} + ix**EATASCTM.TASOCT** time where i is an integer and the schedule loops every **EATASCTM.TASOCT** ns

Fig 5.5: TAS behavior when its schedule re-configuration should have happened in the past

TAS recovery time (*tas_recovery_time*) maximum value can be calculated by equation (5).

(5) *tas_recovery_time[ns] = clk_period[ns]*(tascc_set_time[ns] – restart_time[ns])/cycle_time[ns]* + 2**cycle_time[ns]*

Where:

- *clk_period* is the clock clk period.
- *tascc_set_time* is the AVTP timer time at which the **EATASC.TASCC** has been set.
- *restart_time* is the AVTP timer time at which the schedule has been set to restart ({**EATASCSTC1**, **EATASCSTC0**})
- *cycle_time* is the schedule cycle time set in **EATASCTC**.

Functions:

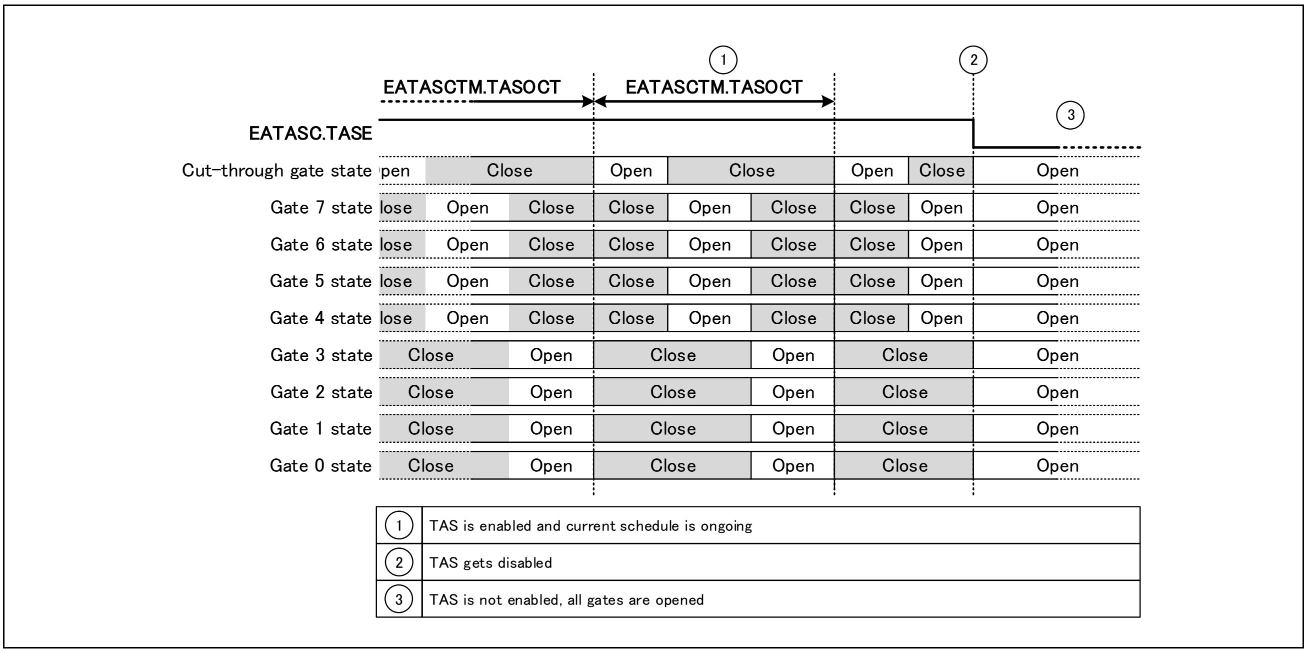
- Equation (5) comes from the fact that TAS will switch its start time by *cycle_time* every clock during recovery. A problem resulting from that is that the recovery time increase proportionality with *tase_set_time[ns] – start_time[ns].* To avoid this issue, TAS is able to switch its start time by 256^{i}**cycle_time* (i=0..6) every clock when start time is too far in the past.

Restrictions:

- HW: Because of the start setting in the past, the hardware guardband cannot be ensured from **EATASC.TASCC** setting time to the end of the first new cycle (frames could cross a gate close state).
- HW: Because of the start setting in the past, **EAEIS1.TASGES** gate errors could be set.

(3) TAS disabling

TAS module can be disabled during run-time. TAS behavior when its schedule is disabled is described in Fig 5.6. The schedule used in this figure is just an example and not correspond to anything relevant for TAS general behavior explanation.

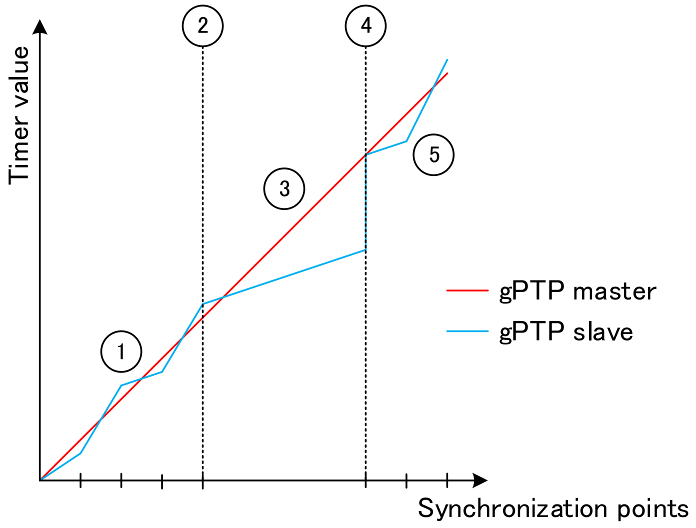


1	TAS is enabled and current schedule is ongoing
2	TAS gets disabled
3	TAS is not enabled, all gates are opened

Fig 5.6: TAS behavior when its schedule is disabled

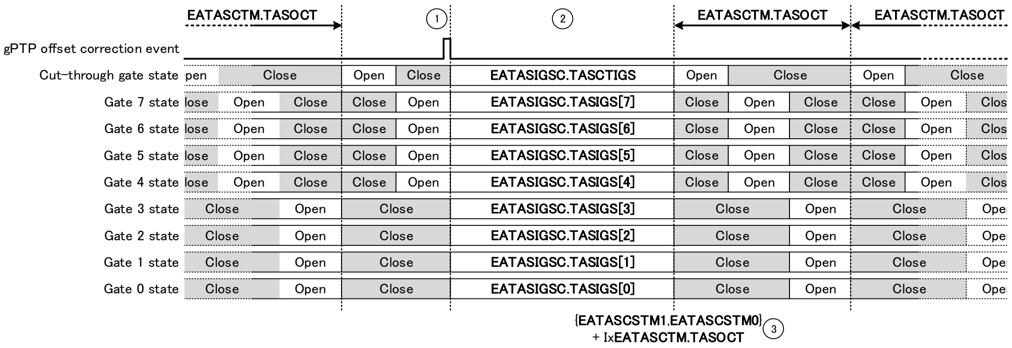
(4) gPTP offset correction recovery [gPTP]

gPTP timer control loop in normal condition correct the gPTP timer using the gPTP incremental value. In this case the gPTP timer should be incremented every clock by around *clk_period* ns (clock clk period) and TAS will operate normally. However, when the gPTP timer is deviating from its corresponding master timer value, it can happen that software corrects the gPTP offset value instead of the incremental value, so, the gPTP timer (so also the AVTP timer) will jump in time like described in Fig 5.7. In this case TAS won't be able to operate normally and has to recover as described in Fig 5.8.



①	The slave is synchronized to the master
②	Last successful synchronization point
③	Synchronization is lost. The gPTP slave is deriving
④	Synchronization is done successfully, the gPTP slave offset gets corrected.
⑤	The slave is synchronized to the master

Fig 5.7: gPTP offset correction


EATASCTM.TASOCT
1
2
EATASCTM.TASOCT
EATASCTM.TASOCT
gPTP offset correction event
Cut-through gate state pen Close Open Close EATASIGSC.TASCTIGS Open Close Open Close
Gate 7 state lose Open Close Close Open EATASIGSC.TASIGS[7] Close Open Close Close Open Clos
Gate 6 state lose Open Close Close Open EATASIGSC.TASIGS[6] Close Open Close Close Open Clos
Gate 5 state lose Open Close Close Open EATASIGSC.TASIGS[5] Close Open Close Close Open Clos
Gate 4 state lose Open Close Close Open EATASIGSC.TASIGS[4] Close Open Close Close Open Clos
Gate 3 state Close Open Close EATASIGSC.TASIGS[3] Close Open Close Ope
Gate 2 state Close Open Close EATASIGSC.TASIGS[2] Close Open Close Ope
Gate 1 state Close Open Close EATASIGSC.TASIGS[1] Close Open Close Ope
Gate 0 state Close Open Close EATASIGSC.TASIGS[0] Close Open Close Ope
{EATASCSTM1,EATASCSTM0}
+ IxEATASCTM.TASOCT
3

1	gPTP offset gets corrected by the software so gPTP timer jumps in time
2	TAS recovery time. TAS needs time to calculate where it can restart safely.
3	TAS resynchronize at {**EATASCSTM1**,**EATASCSMC0**} + Ix**EATASCTM.TASOCT** time where I is an integer and the schedule loops every **EATASCTM.TASOCT** ns

Fig 5.8: TAS recovery during gPTP offset correction

TAS recovery time (*tas_recovery_time*) maximum value can be calculated by equation (6).

(6) *tas_recovery_time[ns] = clk_period[ns]*gptp_jump_time[ns]/cycle_time[ns]* + 2**cycle_time[ns]*

Where:

- *clk_period* is the clock clk period.
- *gptp_jump_time* is the time that has been jumped by the gPTP time.
- *cycle_time* is the schedule cycle time contained in **EATASCTM** register.

Functions:

- For gPTP offset correction in the future, equation (6) comes from the fact that TAS will switch its start time by *cycle_time* every clock during recovery. A problem resulting from that is that the recovery time increase proportionality with *tase_set_time[ns] – start_time[ns].* To avoid this issue, TAS is able to switch its start time by 256^i**cycle_time* (i=0..6) every clock when AVTP timer is too far in the future.

Restrictions:

- HW: Because of the gPTP offset correction, the hardware guardband cannot be ensured from the gPTP offset correction event time to the end of the first new cycle (frames could cross a gate close state).
- HW: Because of the gPTP offset correction, **EAEIS1.TASGES** and/or **EAEIS1.TASCTGES** gate errors could be set.
- HW: If *gptp_jump_time* is too small (less than 2**cycle_time*), it can happen that the TAS cannot go back to initial gate states. The recovery will still happen.
- HW: For gPTP offset correction in the past, TAS is not able to switch its start time by 256^i**cycle_time* (i=0..6) every clock so a gPTP offset correction too far in the past could result in a too long recovery time.

5.1.6.2 TAS schedule

TAS schedule defines at which moment of a cycle a queue is allowed to transmit frames. A TAS schedule should be learnt in the TAS RAM using TAS RAM learning functionality. TAS entry format, TAS entry learning function, TAS entry reading function and conversion between TAS schedule to a set of TAS entries are described in this section.

(1) TAS entry format

The TAS RAM is used to store TAS entries which compose the TAS schedule. A TAS entry contains the information of a gate state. All the fields in this table, if quoted, will be written **TAS.{Field name}**. Table 5-3 describes fields contained in a TAS entry.

Table 5-3: TAS entry format

Field name	Field size (bit)	Field Explanation
GS	1	Gate state Values: - 1'b0: Gate state is closed - 1'b1: Gate state is opened
GT	28	Gate time Functions: - Values of time in nanoseconds associated to the entry gate state **TAS.GS**

(2) TAS entry learning

Learning is used to overwrite entries in the TAS RAM.Table 5-4 describes register used to learn an entry in the TAS RAM. There is no learning result because learning never fails.

Table 5-4: TAS entry learn registers

Register name	Field name/corresponding field in TAS RAM	Field explanation
EATASGL0.TASGAL	Entry address	Not present in TAS RAM, new entry will be written at this address
EATASGL1.TASGSL	**TAS.GS**	Refer to section 5.1.6.2(1)
EATASGL1.TASGTL	**TAS.GT**	Refer to section 5.1.6.2(1)

(3) TAS entry reading

Reading is used to read entries in the TAS RAM. Table 5-5 describes register used to read an entry in the TAS RAM. Table 5-6 describes the read results.

Table 5-5: TAS entry read registers

Register name	Field name/corresponding field in TAS RAM	Field explanation
EATASGR.TASGAR	Entry address	Not present in TAS RAM, entry will be read from this address

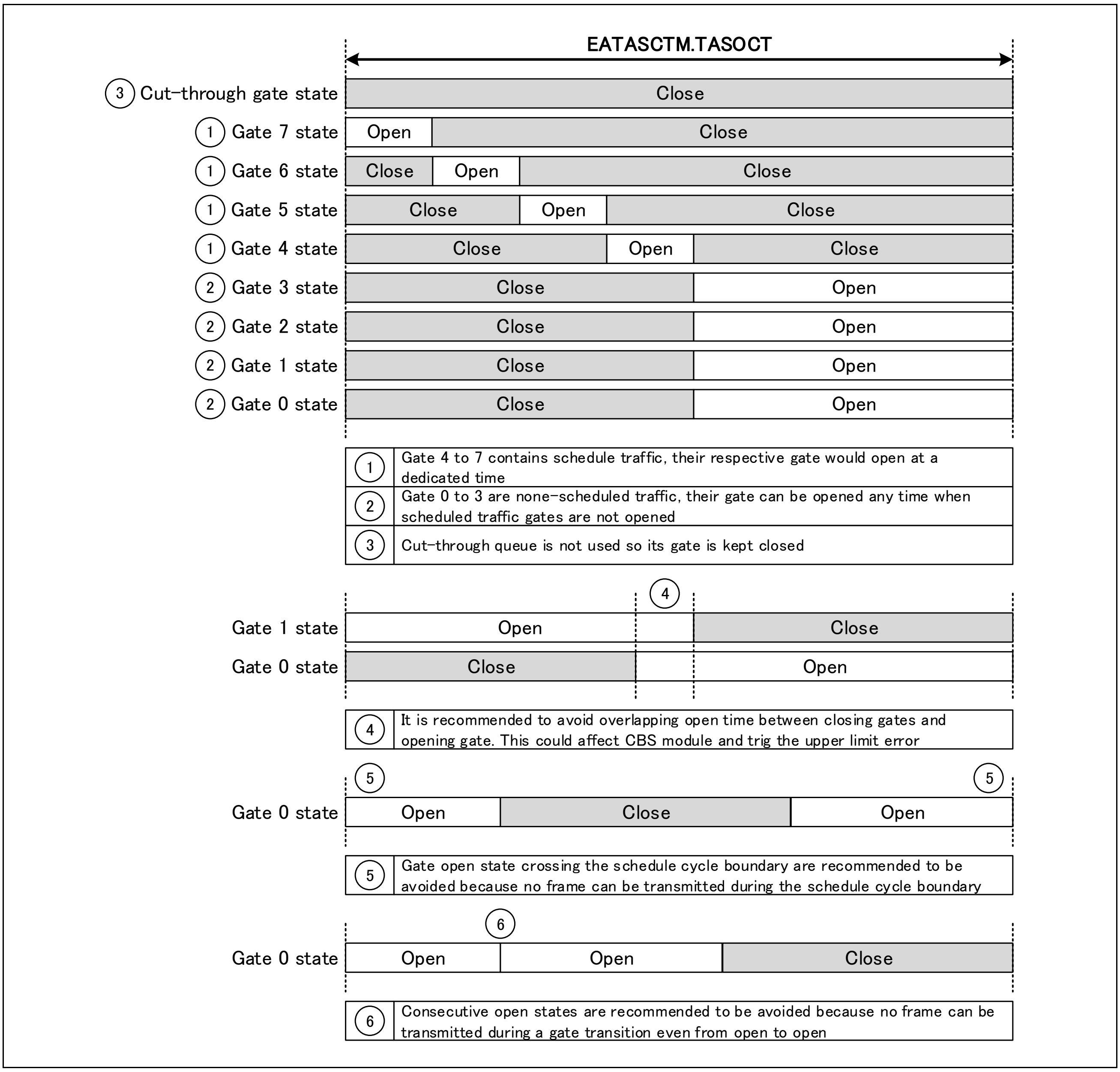
Table 5-6: TAS entry read result

Register name	Field name/corresponding field in TAS RAM	Field explanation
EATASGRR.TASGSR	**TAS.GS**	Refer to section 5.1.6.2(1)
EATASGRR.TASGTR	**TAS.GT**	Refer to section 5.1.6.2(1)
EATASGRR.TASREF	Reading ECC Fail	Reading failed because of an ECC error.

(4) Conversion between TAS schedule to a set of TAS entries

(a) Schedule example

Fig 5.9 describes a basic schedule example with basic gate behavior and recommendations about what not to do while designing a schedule.



1 Gate 4 to 7 contains schedule traffic, their respective gate would open at a dedicated time

2 Gate 0 to 3 are none-scheduled traffic, their gate can be opened any time when scheduled traffic gates are not opened

3 Cut-through queue is not used so its gate is kept closed

4 It is recommended to avoid overlapping open time between closing gates and opening gate. This could affect CBS module and trig the upper limit error

5 Gate open state crossing the schedule cycle boundary are recommended to be avoided because no frame can be transmitted during the schedule cycle boundary

6 Consecutive open states are recommended to be avoided because no frame can be transmitted during a gate transition even from open to open

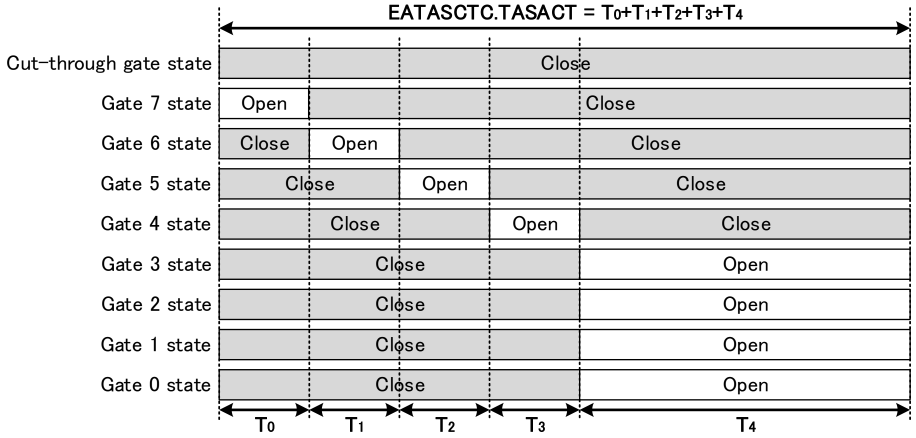
Fig 5.9: Schedule example and recommendations

Restrictions:

- SW: All p-Frames queues (**EATPEC.TTQ** corresponding bit set to 1’b1) should be opened simultaneously. For preemption, refer to section 5.1.7.2

(b) Schedule setting

Fig 5.10 describes how to convert a schedule into a set of TAS entries and how to set the schedule corresponding register through an example.



Corresponding gate	Entry number	TAS.GS	TAS.GT	Entry number
① Gate 0 states	② 0	0	$T_0+T_1+T_2+T_3$	EATASENC0.TASAEN0 = 1
	1	1	③ **0**	
Gate 1 states	2	0	$T_0+T_1+T_2+T_3$	EATASENC1.TASAEN1 = 1
	3	1	**0**	
Gate 2 states	4	0	$T_0+T_1+T_2+T_3$	EATASENC2.TASAEN2 = 1
	5	1	**0**	
Gate 3 states	6	0	$T_0+T_1+T_2+T_3$	EATASENC3.TASAEN3 = 1
	7	1	**0**	
Gate 4 states	8	0	$T_0+T_1+T_2$	④ EATASENC4.TASAEN4 = 2
	9	1	T_3	
	10	0	**0**	
Gate 5 states	11	0	T_0+T_1	EATASENC5.TASAEN5 = 2
	12	1	T_2	
	13	0	**0**	
Gate 6 states	14	0	T_0	EATASENC6.TASAEN6 = 2
	15	1	T_1	
	16	0	**0**	
Gate 7 states	17	1	T_0	EATASENC7.TASAEN7 = 1
	18	0	**0**	
Cut-through gate states	19	0	⑤ **0**	EATASCTENC.TASCTAEN = 0

①	Entries should be written in the TAS RAM from gate 0 to cut-through gate
②	For each gate, entries should be written written along with their time in order of appearance
③	A gate corresponding last entry time should always be set to 0, the last gate will be extended to the cycle end
④	In a gate corresponding entry number register should be set the number of entries − 1 used by the gate
⑤	If a gate has a constant state, the corresponding entry number should be set to 0 and the corresponding constant state should be set in TAS RAM with a gate time of 0

Fig 5.10: Schedule example

Restrictions:

- SW: Each gate time (T_0, T_1, T_2, T_3, T_4 in Fig 5.10) should be greater or equal to 50ns + EATASHCC.TASJ.

(5) Gate open time calculation

Restrictions:

- SW: In order to avoid TAS module from clogging, gate open time should respect a minimum time value *minimumOpenTime* per queue. This time can be calculated by equation (7) for e-queues and equation (8) for p-queues (for pre-emption explanation, refer to section 5.1.7.2).

(7) *minimumOpenTime[ns]* = (*maximumFrameSize[byte]**8 + 160) / *portTransmitRate[Gbps]* + *jitter[ns]* + *clk_period[ns]*

(8) *minimumOpenTime[ns]* = (*MinimumFragmentSize[byte]*8* + 704) / *portTransmitRate[Gbps]* + *jitter[ns]* + *clk_period[ns]*

Where:

- *minimumOpenTime* is the smallest open time for a gate in the schedule
- *maximumFrameSize* is the maximum frame size a queue can receive. Its value for descriptor queue q is equal to **EATMFSCq.MFSq**.
- *portTransmitRate* is the TSNA link throughput [RMAC].
- *Jitter* is the internal jitter set in **EATASHCC.TASJ** register.
- *clk_period* is the clock clk period.
- *MinimumFragmentSize* is the minimum fragment size set in **EATPEC.AFS** register.

5.1.6.3 TAS hardware calibration

TAS transmission authorization is given to a frame before the frame read by TX data fetch block appends and before frame is sent to RMAC for transmission. Because of that, there is an internal minimum latency and an internal maximum jitter (Fig 5.11) that should be taken in account while setting TAS module. Latency calibration should be handled by SW and jitter calibration should be set by SW and will be handled by HW. In this section will be described how to calibrate the TAS module in latency and jitter until the PHY interface [RMAC], but, by adding, for example, the minimum latency and the maximum jitter until the next IP PSFP gate fileting module, it can be possible to calibrate the TAS to it.

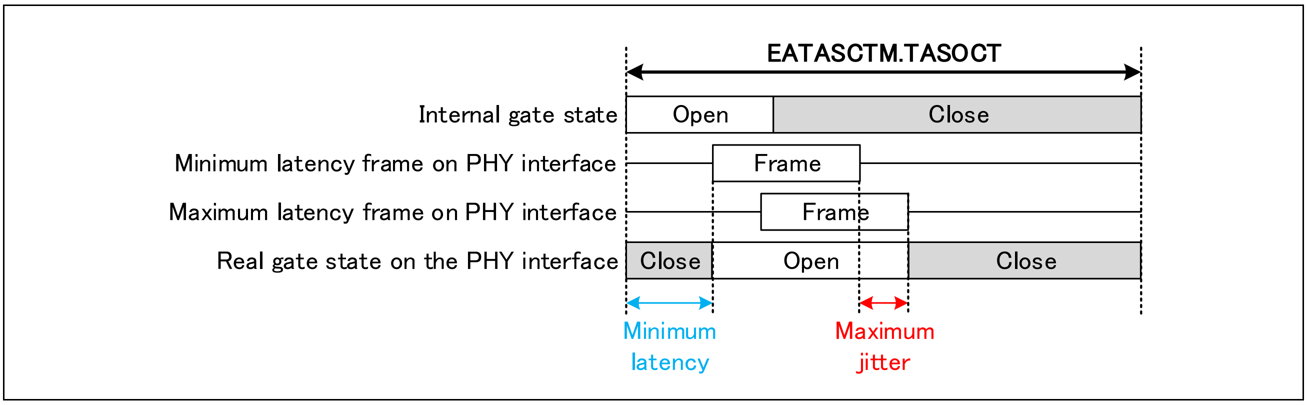

EATASCTM.TASOCT
Internal gate state
Open
Close
Minimum latency frame on PHY interface
Frame
Maximum latency frame on PHY interface
Frame
Real gate state on the PHY interface
Close
Open
Close
Minimum latency
Maximum jitter

Fig 5.11: Minimum latency and maximum jitter

(1) Latency calibration

Because of the internal minimum latency, with a non-calibrated schedule a frame would always start transmitting late on the PHY interface [RMAC]. To correct this phenomenon, the TAS schedule should always be in advance compared to the expected schedule of the minimum latency between the TAS transmission decision and the actual transmission on the PHY interface (Fig 5.12). This minimum latency is described in Table 5-7. To calibrate the schedule, the software should always subtract the minimum latency from the cycle start time before setting it in **EATASCSTC0/1** registers.

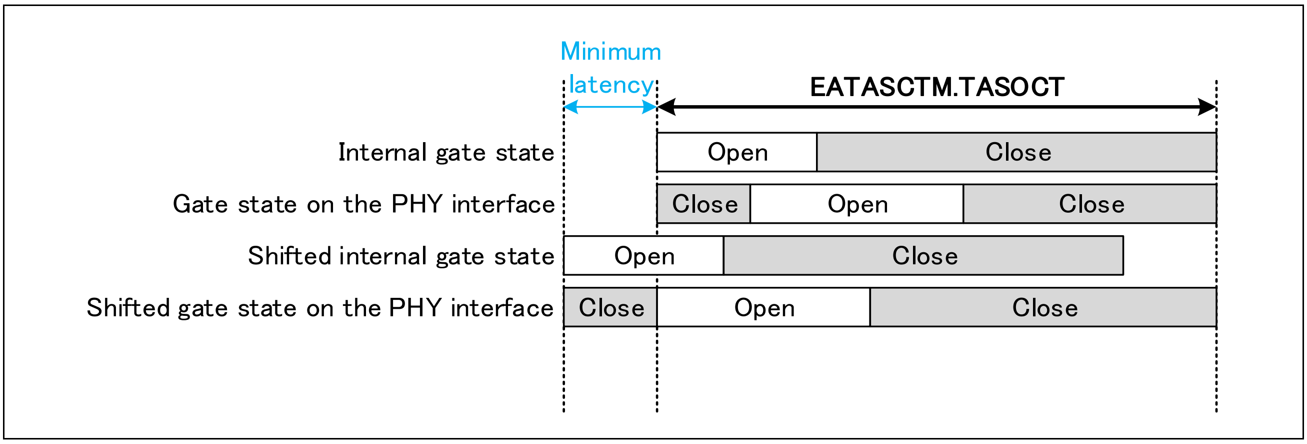

Minimum
latency
EATASCTM.TASOCT
Internal gate state
Open
Close
Gate state on the PHY interface
Close
Open
Close
Shifted internal gate state
Open
Close
Shifted gate state on the PHY interface
Close
Open
Close

Fig 5.12: Minimum latency calibration

Table 5-7: Minimum latency values

PHY speed	Minimum latency when pre-emption disabled (EATPEC.TTQ set to All0)
10Mbps	10**clk_period[ns]* + 11**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
100Mbps	10**clk_period[ns]* + 12**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
1Gbps	10**clk_period[ns]* + 13**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
2.5Gbps	10**clk_period[ns]* + 13**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
5Gbps	10**clk_period[ns]* + 12**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
10Gbs	10**clk_period[ns]* + 11**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*

PHY speed	Minimum latency when pre-emption enabled (at least one bit of EATPEC.TTQ set to 1'b1)
10Mbps	3**clk_period[ns]* + 11**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
100Mbps	3**clk_period[ns]* + 12**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
1Gbps	3**clk_period[ns]* + 13**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
2.5Gbps	3**clk_period[ns]* + 13**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
5Gbps	3**clk_period[ns]* + 12**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*
10Gbs	3**clk_period[ns]* + 11**clk_phy_tx_period[ns]* + *fabricLatency[ns]* + *MACsecLatency[ns]*

Where:

- *clk_period* is the clock clk period.
- *clk_phy_tx_period* is the clock clk_phy_tx period.
- *fabricLatency* is the latency induced by the fabric. Refer to fabric specification document for calculation [FAB].
- *MACsecLatency = Please refer to Latency_TX_MIN [MACsec] (If disabling MACsec, this is 0)*

(2) Jitter calibration

Because of the internal maximum jitter, with a non-calibrated schedule a frame would always finish transmitting late on the PHY interface [RMAC]. To correct this phenomenon, a TAS schedule opened gate should always be closed in advance compared to the expected close time of the maximum jitter (Fig 5.13). This maximum jitter should be calculated using equation (9). To calibrate the schedule, software should set the maximum jitter in **EATASHCC.TASJ** register.

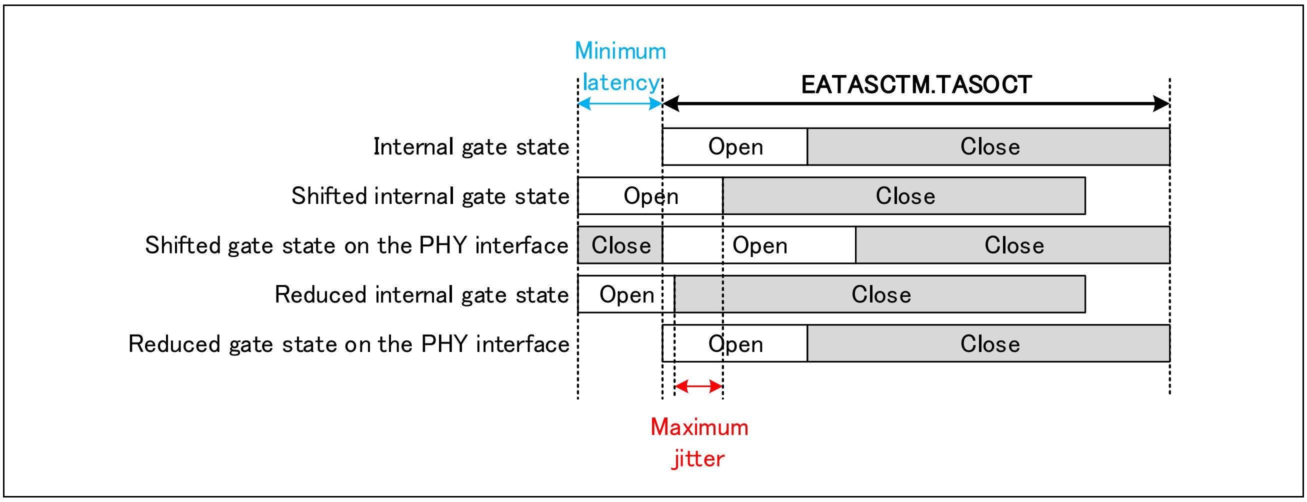

Minimum latency
EATASCTM.TASOCT
Internal gate state
Open
Close
Shifted internal gate state
Open
Close
Shifted gate state on the PHY interface
Close
Open
Close
Reduced internal gate state
Open
Close
Reduced gate state on the PHY interface
Open
Close
Maximum jitter

Fig 5.13: Maximum jitter calibration

(9) *jitter[ns] = tasInternalJitter[ns] + fabricJitter[ns] + preemptionJitter[ns] + MACsecAdditional[ns] + MACsecJitter[ns] + asynchronousJitter[ns] + gptpSyncJitter[ns] + clkDerivationJitter[ns]*

Where:

- *tas_internal_jitter* is the jitter induced by TAS module and is equal to 2**clk_period[ns]* where *clk_period* is the clock clk period.
- *fabricJitter* is the jitter induced by the fabric. Refer to fabric specification document for calculation [FAB].
- *MACsecAdditional[ns] = Actual transmitting time of "TAG and ICV adding by MACsec".*
  MACsecJitter[ns] = Please refer to Latency_TX_MAX – Latency_TX_MIN [MACsec] (If disabling MACsec, they are 0)
- *asynchronousJitter* is the jitter induced by the asynchronous conversion in RMAC [RMAC] and is equal to *clk_period[ns]* + *clk_phy_tx_period[ns]* where *clk_period* is the clock clk period and *clk_phy_tx_period*s the clock clk_phy_tx period.
- *gptpSyncJitter* is the jitter induced by gPTP synchronization between the gPTP master clock and the gPTP slave clock.
- *clkDerivationJitter* is the jitter induced by clock derivation between the gPTP clock clk and the PHY TX clock clk_phy_tx and is equal to *clkRelativeDerivation*maxGateTime[ns]* where *clkRelativeDerivation* is the clock derivation and *maxGateTime* is the longest opened gate in the TAS schedule. *clkRelativeDerivation* is equal to 0 when TAS corresponding gPTP timer is the master and the gPTP clock is phase synchronous to the TX PHY clock.
- *preemptionJitter* is the jitter induced by preemption and is equal to 5**clk_period[ns]* where *clk_period* is the clock clk period when preemption is enabled (at least one bit of **EATPEC.TTQ** register set to 1'b1) and 0ns otherwise.

5.1.6.4 TAS debug interface

This interface aims at debugging the TAS gate states by giving it out of the IP.

Functions:

- eha_race_tas_gate_state[FRM_TPRIO_N-1] reflects the cut-through queue current TAS gate state.
- eha_race_tas_gate_state[FRM_PRIO_N-1:0] **THE PRODUCT I/O PIN NAME SHOULD BE WRITTEN HERE WHILE CREATING USER MANUAL** reflects the current TAS gate states.

5.1.7 TX data fetch

TX data fetch is the same as “RX data fetch” in the GWCA except that it can handle pre-emption and that it also handles the data fetching for cut-through queues. Refer to GWCA specification for more details [GWCA] about data update in case of none-cut-through frames (The correspondence between TSNA and GWCA registers/signals is described in Table 5-1).

Table 5-8: TX data fetch TSNA/GWCA register correspondence

Register/signal name in GWCA [GWCA]	Register name in TSNA
GWVCC.VEM	**EAVCC.VEM**
{GWMAC0.MAUP, GWMAC1.MADP}	{**MRMAC0.MAUP**, **MRMAC1.MADP**} [RMAC]
GWRGC.RCPT	Fixed to 1 (RMAC will add an FCS to all frames sent without it)
GWVTC	**EAVTC**
GWECD0RC, GWECD1RC, GWESD0RC, GWESD1RC	**EAECD0RC, EAECD1RC, EAESD0RC, EAESD1RC**

5.1.7.1 Cut-through frames

Cut-through frames handling by data fetched is transparent for TX data fetch. All cut-through frames are handled has E-Frames for transmission.

5.1.7.2 Pre-emption

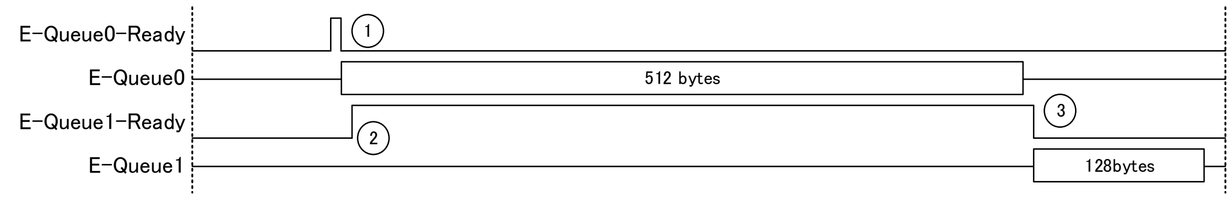
Pre-emption functions allows express frames (e-Frames) to interrupt a preemptable frame (p-Frame) to start transiting faster (a preemptable frame can also be interrupt by TAS gate close event) using **EATPEC** register. Fig 5.14 describes a transmission between two e-queues, Fig 5.15 describes the same transmission but with the highest priority queue being a p-queue and Fig 5.16 describes how preemption happens for a p-Frame.

Functions:

- **EATPEC.AFS** is used to set the minimum fragment size.
- **EATPEC.TTQ[q]** is used to set descriptor queue q traffic to p-traffic or e-traffic.

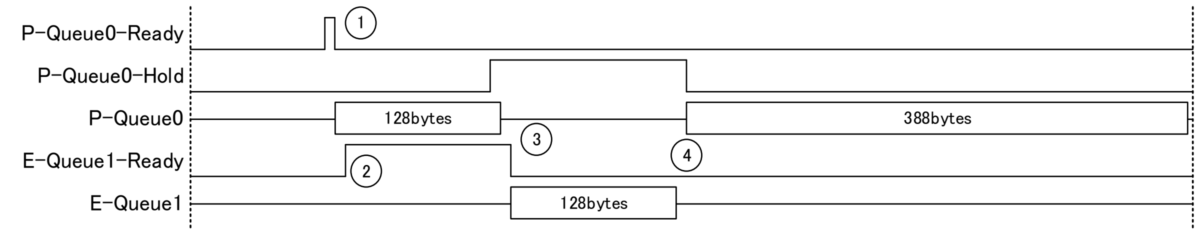
Restrictions:

- HW: In TSN agent, preemption happens on a 64-bit bus. As a results preemption can only be performed every 16 bytes in a frame.
- HW: P-Frames can only be pre-empted by e-Frames with a higher priority. As a result, if the user wants p-Frames to always be pre-empted by e-Frames, low priority queues should be set to p-Frames and high priority queue to e-Frames. Register **EAIRC** can be used to remap IPVs coming fro forwarding engine [FWD] to p-queues or e-queues.



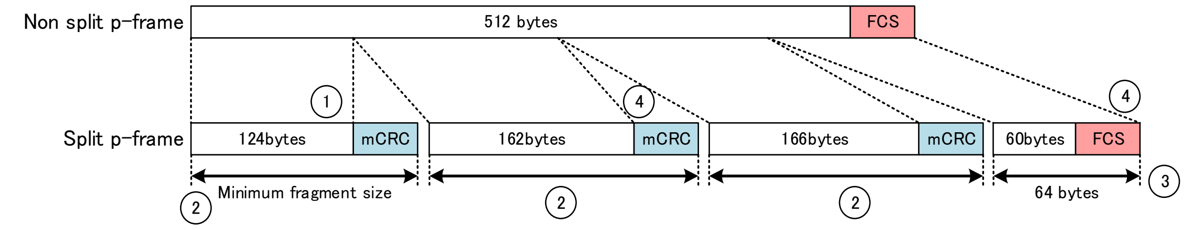
1	A 512 byte frame for queue 0 is ready and starts
2	A 128 byte frame for queue 1 is ready but cannot start because queue 0 frame is ongoing
3	Queue 0 finished transmitting so queue 1 frame can start

Fig 5.14: E-Frame transmission



1	A 512 byte frame for queue 0 is ready and starts
2	A 128 byte frame for queue 1 is ready but cannot start because queue 0 frame hasn't transmit the minimum fragment size
3	Queue 0 frame minimum fragment size is reached so queue 0 frame stops and queue 1 frame starts
4	Queue 1 finished transmitting so queue 0 frame can restart

Fig 5.15: E-Frame and P-Frame transmission (**EATPEC.AFS** == 2'b01)



1	When a frame is preempted, an mCRC a attached to the frame (mCRC is an FCS but with 2 bytes inverted)
2	Any fragment should be bigger or equal to the minimum fragment size (including mCRC) except for the last fragment containing the FCS.
3	The last fragment should be bigger or equal to 64 bytes (including FCS)
4	Detecting an mCRC at the end of a fragment tells that the frame is not finished. Detecting an FCS tells that the frame is finished. mCRC and FCS can be detected thanks to their different magic numbers.

Fig 5.16: P-Frame preemption (**EATPEC.AFS** == 2'b01)

5.1.8 RMAC TX interface control

After TX data fetch, data is sent to the RMAC through MHD TX interface [RMAC]. Table 5-9 shows how MHD TX interface status information are provided to the RMAC.

Table 5-9: MHD TX interface status provided by TSNA to RMAC

Signal name in RMAC [RMAC]	Bus width	Value provided by TSNA
FCS in	1	Set to 1'b1 if A & ~(B & C) & ~D. A: Corresponding transmit frames as an FCS (descriptor is received from forwarding engine with **FDESCR.FI** equal to 1) and if the frame hasn't been modified by the switch (No update due to VLAN tagging/un-tagging and no update requested by L23 update [FWD]). B: **MTTFC.DPAD** == 1'b0 [RMAC]. C: "Payload length of frame (including FCS)" less than 64 bytes. D: Changing PCP/DEI by **EAICD0RC/EAICD1RC/EAISD0RC/EAISD1RC/EAECD0RC/EAECD1RC/EAESD0RC/EAESD1RC** on Ingress/Egress ports.
Timestamp capture	1	Set to **LDESCR.TXC**
Insert egress timestamp	1	Set to **LDESCR.IET**
Calculate residence time	1	Set to **LDESCR.CRT**
Timer number	PTP_TN_W	Set to **LDESCR.TN**
Timestamp unique number	8+PTP_TUNES = 8+PORT_W	Set to {SPN,**LDESCR.TSUN**} where SPN is the source port number of the corresponding frame.

5.2 Data reception

Ethernet Agent allows data reception through the TSNA RX data path, The RX data path is described in Fig 5.17.

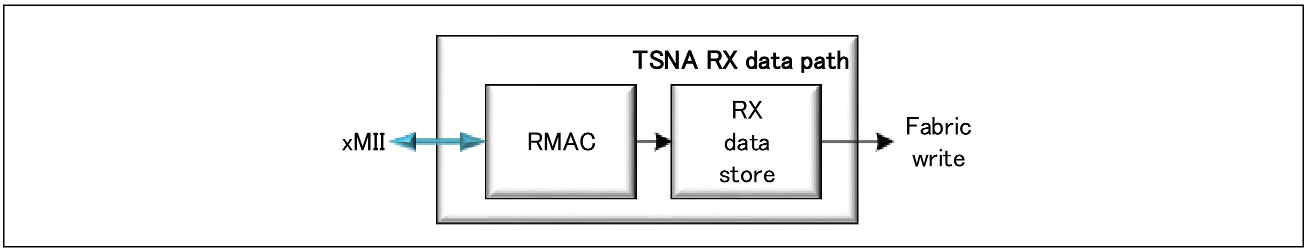


Fig 5.17: TSNA RX data path block diagram

The RX data path is separated in two blocks:

- RMAC: This block handles the data exchange with the Ethernet PHY. This block functionalities are described in the RMAC specification document [RMAC] so it will not de described.
- RX data store: This block extracts TAG information for frames, applies VLAN tagging and save the frames in the local RAM. This block can also release pointers while going out of OPERATION mode. The pointer release is here for switch hardware purpose and its description is not required for switch utilization so it will not be described.

Restrictions:

- SW: The switch maximum input frame size is 60Kbytes. RMAC should be set so it doesn’t send frames bigger than 60Kbytes (**MRFSCE.EMXS** and **MRFSCP.PMXS** should be set to a value smaller or equal to 61440 [RMAC]).

5.2.1 RX data store

RX data store has follow functions, it extracts TAG information for frames, applies VLAN tagging, check checksums (In TSNA, direct descriptor is not available, so checksum calculation function is not available) in frames and save the frames in the local RAM. This function is the same as the “TX data store” function in GWCA except for the local descriptor format. Refer to GWCA specification for more details [GWCA] (The correspondence between TSNA and GWCA registers is described in Table 5-10).

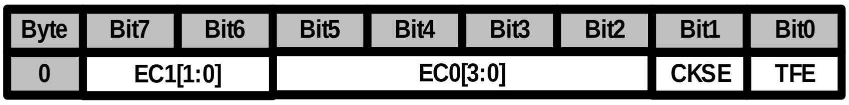
Table 5-10: Data store TSNA/GWCA register correspondence

Register/signal name in GWCA [GWCA]	Register name in TSNA
GWTTFC	EARTFC
GWVCC.VIM	EAVCC.VIM
GWVTC	EAVTC
GWEIS0.USMFSES	EAEIS0.USMFSES
GWICD0RC, GWICD1RC, GWISD0RC, GWISD1RC	EAICD0RC, EAICD1RC, EAISD0RC, EAISD1RC

As local descriptor format, TSNA can only use ethernet local descriptors. Ethernet local descriptor is explained in GWCA specification and the field corresponding values for ethernet agent is described in Table 5-11.

Table 5-11: Ethernet local descriptor field description

Field name	Bit width	Value for TSNA
FI	1	Set to RMAC MHD Rx Interface “FCS included” field [RMAC]
FMT	1	Set to 1’b0 for ethernet descriptors
TXC	1	Set to RMAC MHD Rx Interface “Timestamp capture at TX side” field [RMAC]
IET	1	Set to 1’b0
CRT	1	Set to 1’b0
TN	PTP_TN_W	Set to RMAC MHD Rx Interface “Time domain number” field [RMAC]
TSUN	8	Set to RMAC MHD Rx Interface “Timestamp unique number” field [RMAC]
SAEF	8	Refer to Fig 5.18 and Table 5-12 If multiple errors happen for the same frame (for example : TFE (TAG filtering error) , EC0 (RMAC Frame filtering error) , EC1 (Oversize error)), SAEF will be updated for all these errors as each signal is independent. And if exceptional path is set for any of these errors, then Frame will be forwarded via exceptional path.
VCTRL	3	VCTRL[2] values: - Set to **EAVCC.VIM.** VCTRL[1:0] values - 2’b00: For No TAG, R-TAG and Unknow TAG frames. - 2’b01: For C-TAG and CR-TAG frames. - 2’b10: For SC-TAG and SCR-TAG frames. - 2’b11: For CoS-TAG and CoSR-TAG frames.
RTGI	1	Values: - 1’b0: For No TAG, CoS-TAG, C-TAG, SC-TAG and Unknow TAG frames. - 1’b1: For R-TAG, CoSR-TAG, CR-TAG and SCR-TAG frames.
TPL	FRM_TPL_W	Payload length of frame. It only includes the data saved in the data RAM [FAB].
TSV	1	Values: - 1’b0: Timestamp has not been received along with the frame from RMAC [RMAC] - 1’b1: Timestamp has been received along with the frame from RMAC [RMAC]
TSD	1	Set to RMAC MHD Rx Interface “Timestamp is a default timestamp” field [RMAC]
TSNS	30	Set to 30’b0 if timestamp has not been received. Set to timestamps nanosecond part if timestamp has not been received
TSS	32	Set to 32’b0 if timestamp has not been received. Set to timestamps second part if timestamp has not been received
RSV	- -	Set to 0



Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	EC1[1:0]		EC0[3:0]				CKSE	TFE

Fig 5.18: Source agent error flag format

Table 5-12: Source agent error flag field description

Field name	Bit width	Description
TFE	1	TAG Filtering Error Set conditions: - HW: When a frame does not fit the required TAG ingress format. Refer to **EARTFC** register explanation.
CKSE	1	ChecKSum Error Set conditions: - HW: When a frame has been filtered because of a checksum error, refer to GWCA specification [GWCA].
EC0	4	Error Code 0 [RMAC] Values: - 4’d0: No error - 4’d1: Frame received from RMAC with “PHY error” bit set - 4’d2: Frame received from RMAC with “PCH CRC error” bit set - 4’d3: Frame received from RMAC with “Nibble error” bit set - 4’d4: Frame received from RMAC with “FCS (mCRC) error” bit set - 4’d5: Frame received from RMAC with “Final fragment missing error” bit set - 4’d6: Frame received from RMAC with “C Fragment SMD Error” bit set - 4’d7: Frame received from RMAC with “C Fragment FRAG_COUNT Error” bit set - 4’d8: Frame received from RMAC with “RMAC Frame filtered” bit set - 4’d9: Frame received from RMAC with “Reception partially out of operation” bit set - 4’d10: Frame received from MACsec with “MACsec any error” bit set - Others: Reserved Restrictions: - HW: Because of the coding mechanism only one error can be flagged in this field. - HW: If several errors happen for the same frame, the error with the smallest EC0 value will be flagged.
EC1	2	Error Code 1 [RMAC] Values: - 2’d0: No error - 2’d1: Frame received from RMAC with “Buffer overflow error” bit set - 2’d2: Frame received from RMAC with “Undersize error” bit set - 2’d3: Frame received from RMAC with “Oversize error” bit set Restrictions: - HW: Because of the coding mechanism only one error can be flagged in this field. - HW: If several errors happen for the same frame, the error with the smallest EC0 value will be flagged.
RSV	- -	Reserved Set to 0

6. Precautions

6.1 Precautions

NA.

6.2 Restrictions (Including known problems)

NA.

Published by Renesas Electronics Corporation