User's Manual

GWCA
(R-Switch-3 GateWay CPU 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

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

GWCA (GateWay CPU Agent) is a CPU interface.

The GWCA consists of an agent interface module allowing communication within the Rswitch [TOP]. It handles the data exchange between the Rswitch and the GWCPU subsystem. SW involvement is required for configuration of the GateWay AXI bus interface.

1.1 Features

GWCA Features are described Table **1-1**.

Table 1-1 GWCA Feature List

Function		Details
Interfaces	Clock/Reset interface	Clock/Reset interface [TOP]
	Switch mode	Interface to receive the switch mode [FWD]
	Pause interface	Interface to pause TX queues [COMA]
	Inter IP interface	Interface to process communication between GWCA and any other IP in HW.
	AXI Master interface	Master interface conforming to AXI 3 AMBA protocol. It is used for data transfer between GWCPU and GWCA. ・Data bit width: 128 bits ・Number of outstanding read: AXI_RD_OUT_N ・Number of outstanding write: AXI_WR_OUT_N ・Number of ID read: AXI_RD_OUT_N (from 0 to AXI_RD_OUT_N-1) ・Number of ID write: AXI_WR_OUT_N (from 0 to AXI_WR_OUT_N-1) ・Incremental Access only ・Supports unaligned transactions ・Supports out of order ・Supports interleave on read bus
	SFR interface	Interface to access GWCA SFRs [COMA]
	Interrupts	GWCA 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 receive TX timestamps and send it 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	AXI TX Descriptor (CPU to GWCA) AXI RX Descriptor (GWCA to CPU) Local Ethernet Descriptor, Local Direct Descriptor (GWCA to Forwarding engine) Forwarding descriptor (Forwarding engine to GWCA)
	Timestamp	TX timestamp for transmission to CPU RAM [RMAC]

1.2 GWCA block diagram

Fig 1.1 shows GWCA block diagram.

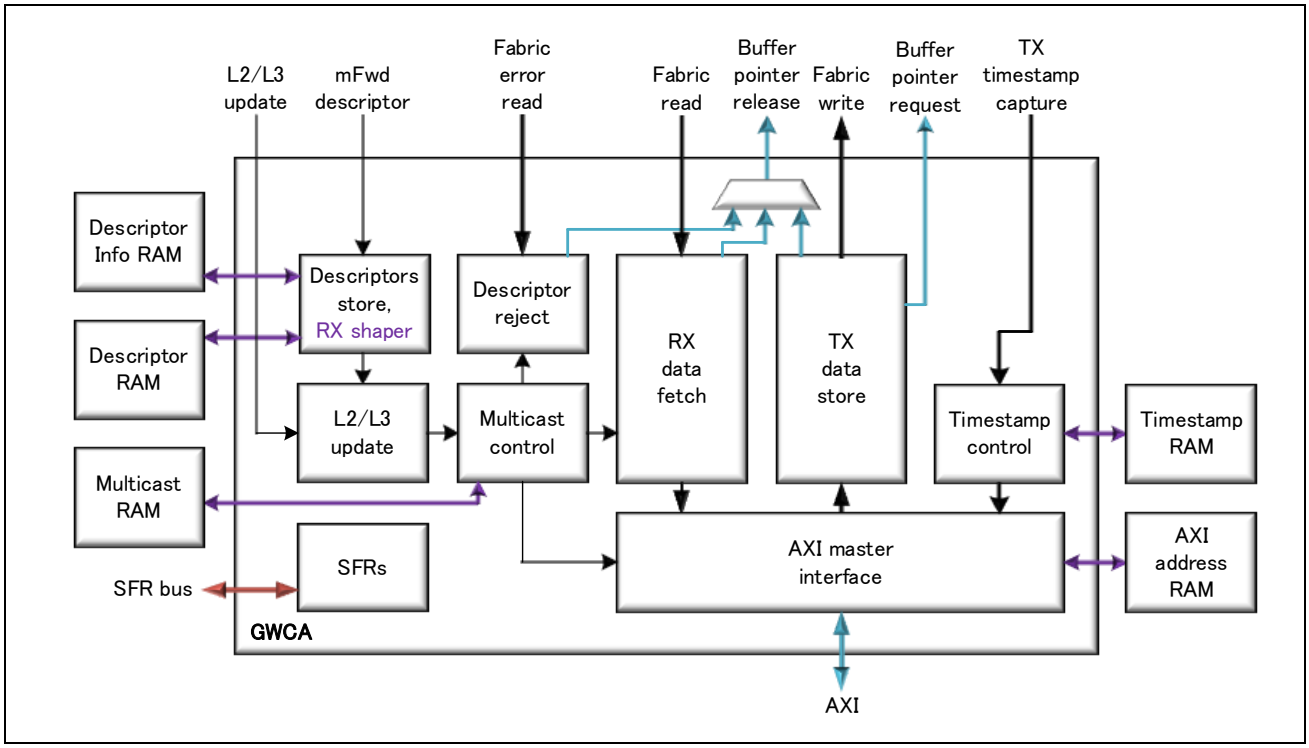


Fig 1.1: GWCA Block diagram

Table 1-2: GWCA Functional Blocks

Block name	Function
TX data store	Transmits frames from AXI master to the local RAM
RX data fetch	Transmits frames from local RAM to AXI master
Descriptor store, RX shaper	Receives the descriptors from forwarding engine and controls the priority between descriptors and shapes the traffic on RX side
L2/L3 update	Fetches L2/L3 update information
Descriptor reject	Reject descriptors
Multicast control	Controls multicast transactions
Timestamp control	Receives timestamps for TSN agents, saves them and transmits them to the AXI master.
AXI master interface	AXI master for data exchange with the CPU
SFRs	GWCA SFRs

2. Parameter list

GWCA global parameter list is shown in Table **2-2**.

GWCA local parameter list is shown in Table **2-3**.

Table 2-1: Define list

Parameter Name	Define state	Explanation
none		

Table 2-2: Global parameter list

Parameter Name	RSW3 Values	Explanation
SFRs		
PADDR_NBR_GWCA	8192/4	Number of addresses used by GWCA SFRs
My port number		
MY_PORT_N	13-14 for GWCA0-1	My port number Refer to Fabric specification to know an agent port number [FAB].
AXI master		
AXI_CHAIN_N	128	AXI descriptor chain number
AXI_RINC_N	8	RX incremental descriptor chain number
AXI_TLIM_N	32	TX rate limiter number
AXI_DW	128	AXI bus data width
AXI_AW	40	AXI bus address width
AXI_BST_SZ	256	AXI burst size Restrictions: - AXI_BST_SZ maximum value is AXI_DW*2.
AXI_RD_OUT_N	16	AXI read outstanding number Also equal to the number of read IDs.
AXI_WR_OUT_N	16	AXI write outstanding number Also equal to the number of write IDs.
AXI_RD_BUF_DP	128	Read data bus buffer depth
AXI_WR_BUF_DP	128	Write data bus buffer depth
AXI_RX_DESCR_PRE_N	4	RX descriptor prefetch number
AXI_TX_DESCR_PRE_N	4	TX descriptor prefetch number
Port Number [TOP]		
PORT_TSNA_N	**13**	TSN Agent Number [TSNA]
PORT_GWCA_N	**2**	CPU Agent Number [GWCA]
Local RAM [FAB]		
LCL_RAM_SZ	1024	Local RAM size in Kbytes

Parameter Name	RSW3 Values	Explanation
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)
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 routing 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
Pause frame		
PAS_LVL_N	2	Pause level number
Descriptor RAM		
DES_RAM_DP	2048	Descriptor RAM depth
Timestamp RAM		
TS_RAM_DP	256	Timestamp RAM depth
Meta-information		
GWC_META_INFO_W	24	GWCA meta-information bus width (information to be added to the RX AXI descriptors)

Table 2-3: Local parameter list

Parameter Name	RSW3.0	Explanation
SFRs		
PADDR_NBR_GWCA_W	11	Number of addresses used by GWCA SFRs bus width
AXI master		
AXI_CHAIN_W	7	AXI descriptor chain width
AXI_RINC_W	2	AXI RX incremental chain width
AXI_STRB_W	16	AXI strobe width
AXI_STRB_WW	4	AXI strobe width width
AXI_RD_OUT_W	3	AXI read outstanding number width
AXI_RD_OUT_W1	4	AXI read outstanding number +1 width
AXI_WR_OUT_W	3	AXI write outstanding number width
AXI_WR_OUT_W1	4	AXI write outstanding number +1 width
AXI_BST_SZ_W	8	AXI burst size width
Port Number		
PORT_N	15	Port number on the switch
PORT_W	4	Port number on the switch bus width
PORT_TSNA_W	4	TSN Agent Number bus width [TSNA]
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
Frame		
LCL_RAM_DW_FAB	512	Local RAM data width
LCL_RAM_AW_FAB	14	Local RAM addr width
FRM_PRIO_W	3	Priority width
FRM_MTN_W	5	Frame multicast number width
Forwarding		
LTH_RRULE_W	11	Layer 3 routing number width
gPTP timer		
PTP_TN_W	1	gPTP Timer number width
Descriptor RAM		
DES_RAM_AW	11	Descriptor RAM address width
DES_RAM_AW1	12	Descriptor RAM address width + 1 width
DES_RAM_DW	104	Descriptor RAM data width
Multicast RAM		
RMS_RAM_DW	22	RX multicast RAM data width
Timestamp RAM		
TS_RAM_DW	77	Timestamp RAM data width
TS_RAM_AW	8	Timestamp RAM address width
TS_RAM_AW1	9	Timestamp RAM address + 1 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 GWCA register list is described in Table 3-3. GWRO (GWCA Register Offset) indicates base address of address space allocated to GWCA by the system. All registers representations are done with the default values of the section 2. If the GWCA is not use with default parameters, it should be taken in account by the user while reading the SFR representation.

Write Modes:

- Any: Register can be written in any mode
- R: Register can be written in RESET mode
- D: Register can be written in DISABLE mode
- C: Register can be written in CONFIG mode
- O: Register can be written 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 GWCA registers

Address	Register name	Abbreviation	Write Access Mode
GWRO + 0000H	GWCA Mode Configuration	GWMC	Any
GWRO + 0004H	GWCA Mode Status	GWMS	- -
GWRO + 0008H	GWCA RX Descriptor RAM Configuration	GWRDRC	C
GWRO + 0010H	GWCA IPV Remapping Configuration	GWIRC	C
GWRO + 0014H	GWCA RX Descriptor Queue Security Configuration	GWRDQSC	C, O
GWRO + 0018H	GWCA RX Descriptor Queue Control	GWRDQC	O
GWRO + 001CH	GWCA RX Descriptor Queue Arbitration Control	GWRDQAC	C
GWRO + 0020H	GWCA RX General Configuration	GWRGC	C
GWRO + 0024H	GWCA CSD Remapping Configuration	GWCSDRC	C
GWRO + 0040H + 4H*q	GWCA Reception Maximum Frame Size Configuration q (q=0.. FRM_PRIO_N-1)	GWRMFSCq	C, O
GWRO + 0060H + 4H*q	GWCA Reception Descriptor Queue Depth Configuration q (q=0.. FRM_PRIO_N-1)	GWRDQDCq	C
GWRO + 0080H + 4H*q	GWCA RX Descriptor Queue q Monitoring (q=0.. FRM_PRIO_N-1)	GWRDQMq	- -
GWRO + 00A0H + 4H*q E: GWRO + 00C0H + 4H*q	GWCA RX Descriptor Queue q Max Level Monitoring (q=0.. FRM_PRIO_N-1)	GWRDQMLMq	- -
GWRO + 0100H	GWCA Multicast and DCNC Table Initialization Register Monitoring	GWMTIRM	C, O
GWRO + 0104H	GWCA Multicast and DCNC Table Learning Setting	GWMSTLS	C, O
GWRO + 0108H	GWCA Multicast and DCNC Table Learning Result	GWMSTLR	- -
GWRO + 010CH	GWCA Multicast and DCNC Table Searching Setting	GWMSTSS	C, O
GWRO + 0110H	GWCA Multicast and DCNC Table Searching Result	GWMSTSR	- -
GWRO + 0120H	GWCA MAC Address Configuration 0	GWMAC0	C, O
GWRO + 0124H	GWCA MAC Address Configuration 1	GWMAC1	C, O
GWRO + 0130H	GWCA VLAN Control Configuration	GWVCC	C
GWRO + 0134H	GWCA VLAN TAG Configuration	GWVTC	C
GWRO + 0138H	GWCA Transmission TAG Filtering Configuration	GWTTFC	C
GWRO + 013CH	GWCA CheckSum Configuration	GWCKSC	C
GWRO + 0140H + 8H*s	GWCA TimeStamp Descriptor Chain Address Configuration 0 s (s=0 to PTP_TN-1)	GWTDCAC0s	C, O

Address	Register name	Abbreviation	Write Access Mode
GWRO + 0144H + 8H*s	GWCA TimeStamp Descriptor Chain Address Configuration 1 s (s=0 to PTP_TN-1)	GWTDCAC1s	C, O
GWRO + 0160H + 4H*s	GWCA TimeStamp Descriptor Chain Configuration s (s=0 to PTP_TN-1)	GWTSDCCs	C
GWRO + 0180H	GWCA Agent TimeStamp Number Monitoring	GWTSNM	C, O
GWRO + 0184H E: GWRO + 0188H	GWCA Agent TimeStamp Maximum Number Monitoring	GWTSMNM	C, O
GWRO + 01A0H + 20H*t	GWCA AVTP Timer Monitoring 0 t (t=0..PTP_TN-1)	GWAVTPTM0t	- -
GWRO + 01A4H + 20H*t	GWCA AVTP Timer Monitoring 1 t (t=0..PTP_TN-1)	GWAVTPTM1t	- -
GWRO + 01A8H + 20H*t	GWCA gPTP Timer Monitoring 0 t (t=0..PTP_TN-1)	GWGPTPTM0t	- -
GWRO + 01ACH + 20H*t	GWCA gPTP Timer Monitoring 1 t (t=0..PTP_TN-1)	GWGPTPTM1t	- -
GWRO + 01B0H + 20H*t	GWCA gPTP Timer Monitoring 2 t (t=0..PTP_TN-1)	GWGPTPTM2t	- -
GWRO + 01E0H	GWCA AXI Control	GWAC	O
GWRO + 01E4H	GWCA Descriptor Chain Base Address Configuration 0	GWDCBAC0	C
GWRO + 01E8H	GWCA Descriptor Chain Base Address Configuration 1	GWDCBAC1	C
GWRO + 01F0H	GWCA Maximum Descriptor Number Configuration	GWMDNC	C
GWRO + 0200H + 4H*i	GWCA AXI Transmission Request Configuration i (i = 0 to AXI_CHAIN_N/32-1)	GWTRCi	O
GWRO + 0300H + 4H*p	GWCA Transmission Pause Configuration p (p = 0 to PAS_LVL_N-1)	GWTPCp	C
GWRO + 0380H	GWCA AXI RAM Initialization Register Monitoring	GWARIRM	C, O
GWRO + 0400H + 4H*i	GWCA AXI Descriptor chain Configuration i (i = 0 to AXI_CHAIN_N-1)	GWDCCi	C, O
GWRO + 0800H	GWCA AXI Address RAM Searching Setting	GWAARSS	C, O
GWRO + 0804H	GWCA AXI Address RAM Searching Result 0	GWAARSR0	- -
GWRO + 0808H	GWCA AXI Address RAM Searching Result 1	GWAARSR1	- -
GWRO + 0840H + 4H*i	GWCA Incremental Data Area Used Area Size i (i=0 to AXI_RINC_N -1)	GWIDAUASi	C, O, D
GWRO + 0880H + 4H*i	GWCA Incremental Data Area Size Monitoring i (i=0 to AXI_RINC_N -1)	GWIDASMi	- -
GWRO + 0900H + 8H*i	GWCA Incremental Data Area Start Address Monitoring 0 i (i=0 to AXI_RINC_N -1)	GWIDASAM0i	- -
GWRO + 0904H + 8H*i	GWCA Incremental Data Area Start Address Monitoring 1 i (i=0 to AXI_RINC_N -1)	GWIDASAM1i	- -
GWRO + 0980H + 8H*i	GWCA Incremental Data Area Current Address Monitoring 0 i (i=0 to AXI_RINC_N -1)	GWIDACAM0i	- -
GWRO + 0984H + 8H*i	GWCA Incremental Data Area Current Address Monitoring 1 i (i=0 to AXI_RINC_N -1)	GWIDACAM1i	- -
GWRO + 0A00H	GWCA Global Rate Limiter Configuration	GWGRLC	C, O
GWRO + 0A04H	GWCA Global Rate Limiter Upper Limit Configuration	GWGRLULC	C, O
GWRO + 0A80H + 8H*i	GWCA Rate Limiter incremental value Configuration i (i = 0 to AXI_TLIM_N-1)	GWRLCi	C, O
GWRO + 0A84H + 8H*i	GWCA Rate Limiter Upper Limit Configuration i (i = 0 to AXI_TLIM_N-1)	GWRLULCi	C, O
GWRO + 0B80H	GWCA Interrupt Delay Prescaler Configuration	GWIDPC	C
GWRO + 0C00H + 4H*i	GWCA Interrupt Delay Configuration i (i = 0 to AXI_CHAIN_N-1)	GWIDCi	C, O
GWRO + 1000H E: GWRO + 1080H	GWCA Received Data CouNter	GWRDCN	- -
GWRO + 1004H E: GWRO + 1084H	GWCA Transmitted Data CouNter	GWTDCN	- -
GWRO + 1008H E: GWRO + 1088H	GWCA TimeStamp CouNter	GWTSCN	- -
GWRO + 100CH E: GWRO + 108CH	GWCA TimeStamp OVerFlow Error CouNter	GWTSOVFECN	- -
GWRO + 1010H E: GWRO + 1090H	GWCA Under Switch Minimum Frame Size Error CouNter	GWUSMFSECN	- -
GWRO + 1014H E: GWRO + 1094H	GWCA TAG Filtering Error CouNter	GWTFECN	- -

Address	Register name	Abbreviation	Write Access Mode
GWRO + 1018H E: GWRO + 1098H	GWCA SEQuence Error CouNter	GWSEQECN	- -
GWRO + 1020H E: GWRO + 10A0H	GWCA TX Descriptor Number Error CouNter	GWTXDNECN	- -
GWRO + 1024H E: GWRO + 10A4H	GWCA Frame Size Error CouNter	GWFSECN	- -
GWRO + 1028H E: GWRO + 10A8H	GWCA Timestamp Descriptor Full Error CouNter	GWTDFECN	- -
GWRO + 102CH E: GWRO + 10ACH	GWCA Timestamp Descriptor Number Error CouNter	GWTSDNECN	- -
GWRO + 1030H E: GWRO + 10B0H	GWCA Descriptor Queue Overflow Error CouNter	GWDQOECN	- -
GWRO + 1034H E: GWRO + 10B4H	GWCA Descriptor Queue Security Error CouNter	GWDQSECN	- -
GWRO + 1038H E: GWRO + 10B8H	GWCA Descriptor Full Error CouNter	GWDFECN	- -
GWRO + 103CH E: GWRO + 10BCH	GWCA Descriptor Security Error CouNter	GWDSECN	- -
GWRO + 1040H E: GWRO + 10C0H	GWCA Data SiZe Error CouNter	GWDSZECN	- -
GWRO + 1044H E: GWRO + 10C4H	GWCA Descriptor Chain Type Error CouNter	GWDCTECN	- -
GWRO + 1048H E: GWRO + 10C8H	GWCA RX Descriptor Number Error CouNter	GWRXDNECN	- -
GWRO + 104CH E: GWRO + 10CCH	GWCA CheckSum Error CouNter	GWCKSECN	- -
GWRO + 107CH E: GWRO + 10FCH	GWCA Lost Descriptor CouNter	GWLDCN	- -
GWRO + 1100H + 10H*i	GWCA Data Interrupt Status i (i = 0 to AXI_CHAIN_N/32-1)	GWDISi	C, O, D
GWRO + 1104H + 10H*i	GWCA Data Interrupt Enable i (i = 0 to AXI_CHAIN_N/32-1)	GWDIEi	C, O, D
GWRO + 1108H + 10H*i	GWCA Data Interrupt Disable i (i = 0 to AXI_CHAIN_N/32-1)	GWDIDi	C, O, D
GWRO + 110CH + 10H*i	GWCA Data Interrupt Delayed Status i (i = 0 to AXI_CHAIN_N/32-1)	GWDIDSi	- -
GWRO + 1180H	GWCA TimeStamp Data Interrupt Status	GWTSDIS	C, O, D
GWRO + 1184H	GWCA TimeStamp Data Interrupt Enable	GWTSDIE	C, O, D
GWRO + 1188H	GWCA TimeStamp Data Interrupt Disable	GWTSDID	C, O, D
GWRO + 1190H	GWCA Error Interrupt Status 0	GWEIS0	C, O, D
GWRO + 1194H	GWCA Error Interrupt Enable 0	GWEIE0	C, O, D
GWRO + 1198H	GWCA Error Interrupt Disable 0	GWEID0	C, O, D
GWRO + 11A0H	GWCA Error Interrupt Status 1	GWEIS1	C, O, D
GWRO + 11A4H	GWCA Error Interrupt Enable 1	GWEIE1	C, O, D
GWRO + 11A8H	GWCA Error Interrupt Disable 1	GWEID1	C, O, D
GWRO + 1200H + 10H*i	GWCA Error Interrupt Status 2 i (i = 0 to AXI_CHAIN_N/32-1)	GWEIS2i	C, O, D
GWRO + 1204H + 10H*i	GWCA Error Interrupt Enable 2 i (i = 0 to AXI_CHAIN_N/32-1)	GWEIE2i	C, O, D
GWRO + 1208H + 10H*i	GWCA Error Interrupt Disable 2 i (i = 0 to AXI_CHAIN_N/32-1)	GWEID2i	C, O, D
GWRO + 1280H	GWCA Error Interrupt Status 3	GWEIS3	C, O, D
GWRO + 1284H	GWCA Error Interrupt Enable 3	GWEIE3	C, O, D
GWRO + 1288H	GWCA Error Interrupt Disable 3	GWEID3	C, O, D

Address	Register name	Abbreviation	Write Access Mode
GWRO + 1290H	GWCA Error Interrupt Status 4	GWEIS4	C, O, D
GWRO + 1294H	GWCA Error Interrupt Enable 4	GWEIE4	C, O, D
GWRO + 1298H	GWCA Error Interrupt Disable 4	GWEID4	C, O, D
GWRO + 12A0H	GWCA Error Interrupt Status 5	GWEIS5	C, O, D
GWRO + 12A4H	GWCA Error Interrupt Enable 5	GWEIE5	C, O, D
GWRO + 12A8H	GWCA Error Interrupt Disable 5	GWEID5	C, O, D
GWRO + 1800H	GWCA Security Configuration Register 0	GWSCR0	C, O
GWRO + 1804H	GWCA Security Configuration Register 1	GWSCR1	C, O
GWRO + 1900H + 4H*i	GWCA Security Configuration Register 2 i (i = 0 to AXI_CHAIN_N/32-1)	GWSCR2i	C, O
GWRO + 1A00H	GWCA Ingress C-TAG DEI 0 Remapping Configuration	GWICD0RC	C, O, D
GWRO + 1A04H	GWCA Ingress C-TAG DEI 1 Remapping Configuration	GWICD1RC	C, O, D
GWRO + 1A08H	GWCA Ingress S-TAG DEI 0 Remapping Configuration	GWISD0RC	C, O, D
GWRO + 1A0CH	GWCA Ingress S-TAG DEI 1 Remapping Configuration	GWISD1RC	C, O, D
GWRO + 1A10H	GWCA Egress C-TAG DEI 0 Remapping Configuration	GWECD0RC	C, O, D
GWRO + 1A14H	GWCA Egress C-TAG DEI 1 Remapping Configuration	GWECD1RC	C, O, D
GWRO + 1A18H	GWCA Egress S-TAG DEI 0 Remapping Configuration	GWESD0RC	C, O, D
GWRO + 1A1CH	GWCA Egress S-TAG DEI 1 Remapping Configuration	GWESD1RC	C, O, D
GWRO + 1A20H + 4H*q E: GWRO + 1A40H + 4H*q	GWCA Descriptor Queue Overflow Error CouNter Priority q (q=0..FRM_PRIO_N-1)	GWDQOECNPq	- -

3.3 Register detailed explanation

This section describes SFR details.

3.3.1 GWCA Function registers

3.3.1.1 GWCA Mode function registers

(1) GWMC

GWCA 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 GWCA mode. For more details, refer to section 4.1. 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 **GWMS.OPS.**

(2) GWMS

GWCA 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 GWCA mode. For more details, refer to section 4.1. Values: - 2'b00: RESET mode - 2'b01: DISABLE mode - 2'b10: CONFIG mode - 2'b11: OPERATION mode Update conditions: - HW: Mode change completed. - This register is change at "Maximum frame communication time (exp: 64Kbyte / 100Mbps = 5.12ms)" from OPERATION to DISABLE.

3.3.1.2 Reception function registers

(1) GWRDRC

GWCA Rx 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															RDRM

Bits	Bit name	RW-P	Initial value	Function description
31: 1	RSV	R0-P	0H	Reserved area. On read, 0 will be returned.
0	RDRM	RW-P	0B	Rx Descriptor RAM Mode Values : - 1'b0: Rx Descriptor RAM Separation Mode - 1'b1: Rx Descriptor RAM Share Mode Functions: - "Separation Mode": Rx Descriptor RAM is sepalated for each queues by values of **GWRDQDCq.DQDq**. "Share Mode": All area of Descriptor RAM is shared by all queues.

(2) GWIRC

GWCA 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) GWRDQSC

GWCA RX Descriptor Queue Security 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								RDQSL[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: FRM_PRIO	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
i i=0..FRM_PRIO_N-1	RDQSL	RW-RP	0H	RX 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) GWRDQC

GWCA RX Descriptor Queue Control.

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

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.
i+16 i=0..FRM_PRIO_N-1	RDQP	R!=W-P	0H	RX Descriptor Queue Pause i Values: - 1'b0: Queue i active - 1'b1: Queue i paused Functions: - Avoid frames to be sent to the CPU 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 (**GWMC.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.
15: FRM_PRIO_N	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

i i=0.. FRM_PRIO_N-1	RDQD	R!=W-P	0H	RX 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 **GWRMFSCq.MFSq** register can be set to 0 in order to avoid more frames to go to the CPU 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 (**GWMC.OPC** != 2'b11).

(5) GWRDQAC

GWCA RX Descriptor Queue Arbitration Control.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDQA7[3:0]				RDQA6[3:0]				RDQA5[3:0]				RDQA4[3:0]			
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDQA3[3:0]				RDQA2[3:0]				RDQA1[3:0]				RDQA0[3:0]			

Bits	Bit name	RW-P	Initial value	Function description
[4*(i+1)-1:4*i] i=0..FRM_PRIO_N-1	RDQAi	RW-P	0H	RX Descriptor Queue Arbitration i Values: - 4'b0: Queue i strict arbitration - Others: Queue i WRR arbitration Functions: - For more details, refer to 5.3.1.2. Restrictions: - SW: This function is for Ethernet agent (ETHA/TSNA), and recommend invalid it (setting All 0 in GWCA). - SW: In hybrid arbitration mode, all the queues with a **GWRDQAC.RDQAi** value different than 4'd0 should have a consecutive priority (priority refers to the queue number i). - SW: In all arbitration mode except strict priority arbitration mode, at least two queues should have a **GWRDQAC.RDQAi** different than 4'd0.

(6) GWRGC

GWCA RX General 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															RCPT

Bits	Bit name	RW-P	Initial value	Function description
31:1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
0	RCPT	RW-P	0H	Receive CRC Pass Through [802.3] This bit selects to pass FCS field on the reception frame. Values: - 1'b0: FCS is not passed to GWCPU. If a descriptor is received from forwarding engine with **FDESCR.FI** equal to 1, GWCA remove the last 4 bytes of the Frame Data (FCS). - 1'b1: FCS is passed to GWCPU. GWCA don't remove FCS from the Frame which has one (a descriptor is received from forwarding engine with **FDESCR.FI** equal to 1) if the frame hasn't been modified by the switch (No update due to VLAN tagging/un-tagging and no update requested by L3 routing [FWD]). Restrictions: - HW: GWCA does not check FCS integrity.

(7) GWCSDRC

GWCA CSD Remapping Configuration.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV	CSDR7 [FRM_PRIO_W-1:0]			RSV	CSDR6 [FRM_PRIO_W-1:0]			RSV	CSDR5 [FRM_PRIO_W-1:0]			RSV	CSDR4 [FRM_PRIO_W-1:0]		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV	CSDR3 [FRM_PRIO_W-1:0]			RSV	CSDR2 [FRM_PRIO_W-1:0]			RSV	CSDR1 [FRM_PRIO_W-1:0]			RSV	CSDR0 [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	CSDRi	RW-P	0H	CSD Remapping i Values: - This is CSD value. The descriptor of (IPV == i and CSD == 0) will be changed CSD value to **CSDRE.CSDRi** value. (when a descriptor is received) from forwarding engine with **FDESCR.IPV** [FWD] == i and **FDESCR.CSD** [FWD] == 0, this descriptor will be changed **FDESCR.CSD** to **CSDRE.CSDRi**).

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

GWCA Reception 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 maximum frame size comparison takes in account the frame size which will actually be written by the AXI master (after VLAN un-tagging, R-TAG insertion and L2L3 update) but always counts in this this size FCS even if not received by CPU.

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

GWCA Reception 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 : - Number of descriptors that can contain descriptor queue q. - For details refer to section 5.3.1.1. Restrictions: - SW: When **GWRDRC.RDRM** is 0, the sum of DQD fields should always be smaller or equal to DES_RAM_DP. When **GWRDRC.RDRM** is 1, the maximum value for this field is DES_RAM_DP.

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

GWCA RX 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: - Indicate the current number of descriptors stored in RX Descriptor Queue q. 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, there is no descriptor security error and the queue is not disabled. Decrement conditions: - HW: Decremented by 1 when a descriptor is read by GWCA to send data to AXI Master Interface or to reject data.

(11) GWRDQMLMq (q=0.. FRM_PRIO_N-1)

GWCA RX 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	R-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: - Indicates the maximum number of descriptors that has been stored in RX Descriptor Queue q. Clear conditions: - HW: Being in RESET mode will clear this register (**GWMS.OPS** == 2'b00). - SW: By reading this register, it is cleared to **GWRDQMq.DNQq.** Increment conditions: - HW: Increment to **GWRDQMq.DNQq** value when smaller than **GWRDQMq.DNQq.**

3.3.1.3 Reception multicast function registers

(1) GWMTIRM

GWCA Multicast and Descriptor Chain Number Change (DCNC) Table 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														MTR	MTIOG

Bits	Bit name	RW-P	Initial value	Function description
31:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	MTR	R-RP	0H	Multicast and DCNC Table Ready Set conditions: - When **GWMTIRM.MTIOG** is getting cleared. Clear conditions: - By writing 1 to **GWMTIRM.MTIOG**. - Being in RESET mode.
0	MTIOG	R!=W-RP	0B	Multicast and DCNC Table Initialization OnGoing Set conditions: - SW: By writing 1 to this register. It starts Multicast and DCNC Table initialization. Clear conditions: - HW: This bit is cleared when Multicast and DCNC Table initialization is finished. Min clk_period * AXI_CHAIN_N = [exp] (1000/320) * 128 = 400ns

(2) GWMSTLS

GWCA Multicast and Descriptor Chain Number Change (DCNC) Table Learning Setting.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV	DCNCENL[AXI_CHAIN_W-1:0]							RSV	MSENL[AXI_CHAIN_W-1:0]						
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV / DCNCSPNL[PORT_W-1:0]				DCNCEL	MNL[2:0]			RSV	MNRCNL[AXI_CHAIN_W-1:0]						

Bits	Bit name	RW-P	Initial value	Function description
31:AXI_CHAIN_W+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W+23:24	DCNCENL	RW-D	0H	DCNC to Entry Number Learn Functions: - This register is used to set the descriptor chain number change to entry number that should be learnt in **GWMATLS.MSENL** multicast and DCNC table entry.
23:AXI_CHAIN_W+16	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W+15:16	MSENL	RW-D	0H	Multicast and DCNC Setting Entry Number Learn Functions: - This register is used to set in which address the current learning should happen.
15:PORT_W+12	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PORT_W+11:12	DCNCSPNL	RW-D	0H	DCNC target Source Port Number Learn Functions: - This register is used to set the descriptor chain number change target source port number that should be learnt in **GWMATLS.MSENL** multicast and DCNC table entry.
11	DCNCEL	RW-D	0H	DCNC Enable Learn Functions: - This register is used to set the descriptor chain number change enable that should be learnt in **GWMATLS.MSENL** multicast and DCNC table entry.
10:8	MNL	RW-D	0H	Multicast Number Learn Values: - 3'd0: multicast 0 (1 target, no multicast) - 3'd1: multicast 1 (2 targets) - … - 3'd7: multicast 7 (8 targets) Functions: - This register is used to set the multicast number that should be learnt in **GWMATLS.MSENL** multicast and DCNC table entry.
7:AXI_CHAIN_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W-1:0	MNRCNL	RW-D	0H	Multicast Next Descriptor Chain Number Learn Functions: - This register is used to set the next address in the multicast and DCNC table to be learnt in **GWMATLS.MSENL** multicast and DCNC table entry, refer to section 5.3.3.4.

(3) GWMSTLR

GWCA Multicast and Descriptor Chain Number Change (DCNC) Table Learning Result

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

Bits	Bit name	RW-P	Initial value	Function description
31	MTL	R-D	0H	Multicast and DCNC Table Learning Set conditions: - SW: Writing **GWMSTLS** register will set this bit. Clear conditions: - HW: This bit will be de-asserted when the Multicast RAM write is completed.
30:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	MTLSF	R-D	0H	Multicast and DCNC Table Learning Security Fail Values: - 1'b0: Entry learning didn't fail because of a security error. - 1'b1: Entry learning failed because of a security error. Update Conditions: - HW: **GWMSTLR.MTL** clear event. Restrictions: - HW: This bit never sets for the secure APB.
0	MTLF	R-D	0H	Multicast and DCNC Table Learning Fail Values: - 1'b0: Entry learning didn't fail because the Multicast and DCNC table is not ready. - 1'b1: Entry learning failed the Multicast and DCNC table is not ready. Update Conditions: - HW: **GWMSTLR.MTL** clear event.

(4) GWMSTSS

GWCA Multicast and Descriptor Chain Number Change (DCNC) Table Searching Setting.

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									MSENS[AXI_CHAIN_W-1:0]						

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_CHAIN_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W-1:0	MSENS	RW-D	0H	Multicast and DCNC Setting Entry Number Search Functions: - This register is used to read multicast and DCNC table.

(5) GWMSTSR

GWCA Multicast and Descriptor Chain Number Change (DCNC) Table Searching Result.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MTS	DCNCENR[AXI_CHAIN_W-1:0]							RSV							MTSEF
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV / DCNCSPNR[PORT_W-1:0]				DCNCER	MNR[2:0]			RSV	MNRCNR[AXI_CHAIN_W-1:0]						

Bits	Bit name	RW-P	Initial value	Function description
31	MTS	R-D	0H	Multicast and DCNC Table Searching Set conditions: - SW: Writing **GWMSTSS** register will set this bit. Clear conditions: - HW: This bit will be de-asserted when the Multicast RAM read is completed. Functions: - This register is used to read multicast and DCNC table.
30: AXI_CHAIN_W+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W+23:24	DCNCENR	R-D	0H	DCNC to Entry Number Result Update conditions: - HW: **GWMSTSR.MTS** clear event. Functions: - This register is used to read multicast and DCNC table
23:17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
16	MTSEF	R-D	0H	Multicast and DCNC Table Searching ECC Fail Values: - 1'b0: Entry searching didn't fail because of an ECC error. - 1'b1: Entry searching failed because of an ECC error. Update Conditions: - HW: **GWMSTSR.MTS** clear event.
15:PORT_W+12	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PORT_W+11:12	DCNCSPNR	R-D	0H	DCNC target Source Port Number Result Update conditions: - HW: **GWMSTSR.MTS** clear event. Functions: - This register is used to read multicast and DCNC table.
11	DCNCER	R-D	0H	DCNC Enable Result Update conditions: - HW: **GWMSTSR.MTS** clear event. Functions: - This register is used to read multicast and DCNC table.
10:8	MNR	R-D	0H	Multicast Number Result Update conditions: - HW: **GWMSTSR.MTS** clear event. Functions: - This register is used to read multicast and DCNC table.

7: AXI_CHAIN_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W-1:0	MNRCNR	R-D	0H	Multicast Next RX Descriptor Chain Number Result Update conditions: - HW: **GWMSTSR.MTS** clear event. Functions: - This register is used to read multicast and DCNC table.

3.3.1.4 MAC function registers

(1) GWMAC0

GWCA MAC Address Configuration 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
MAUP[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	MAUP	RW-P	0H	MAC Address Upper Part Functions: - These bits are used to set the 16 upper-order bits of the MAC address. For example, if the MAC address is 01-23-45-67-89-AB (hexadecimal), set H'0123 in this register.

(2) GWMAC1

GWCA MAC Address Configuration 1.

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

Bits	Bit name	RW-P	Initial value	Function description
31:0	MADP	RW-P	0H	MAC Address Downer Part Functions: - These bits are used to set the 32 lower-order bits of the MAC address. For example, if the MAC address is 01-23-45-67-89-AB (hexadecimal), set H'456789AB in this register.

3.3.1.5 TAG function registers

(1) GWVCC

GWCA 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, frames are transmitted with no VLAN. - 3'b001: C-TAG VLAN mode, frames are transmitted with ingress C-TAG if there is one stored in the Local RAM. - 3'b010: HW C-TAG VLAN mode, frames are transmitted with the C-TAG stored in local RAM. - 3'b011: SC-TAG VLAN mode, frames are transmitted with ingress C-TAG and S-TAG if they were stored in the Local RAM. - 3'b100: HW SC-TAG VLAN mode, frames are transmitted with the C-TAG and S-TAG stored in local RAM.
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.
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 Values : - 1'b0: Incoming VLAN mode. - 1'b1: Port based VLAN mode. Restrictions : - SW: This register shouldn't be set to 1'b1 is the switch when in no VLAN mode (In forwarding engine, **FWGC.SVM** is set to 2'b00 [FWD]).

(2) GWVTC

GWCA 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
30:28	STP	RW-P	0H	S-TAG PCP
27:16	STV	RW-P	1H	S-TAG VLAN
15	CTD	RW-P	0H	C-TAG DEI
14:12	CTP	RW-P	0H	C-TAG PCP
11:0	CTV	RW-P	1H	C-TAG VLAN

(3) GWTTFC

GWCA Transmission 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						RSV	UT	SCRT	SCT	CRT	CT	CSRT	CST	RT	NT

Bits	Bit name	RW-P	Initial value	Function description
31:10	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
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) GWICD0RC

GWCA 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) GWICD1RC

GWCA 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) GWISD0RC

GWCA 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) GWISD1RC

GWCA 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) GWECD0RC

GWCA 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) GWECD1RC

GWCA 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) GWESD0RC

GWCA 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) GWESD1RC

GWCA 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.

3.3.1.6 Checksum function registers

(1) GWCKSC

GWCA 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. **Example of "Switch Error Frame":** **- DMAC(6) + SMAC(6) + EtherType(2) + Payload(34)** **- DMAC(6) + SMAC(6) + EtherType(2) + Payload(30) + FCS(4)** **- DMAC(6) + SMAC(6) + TAGs(any) + EtherType(2) + Payload(34)** **- DMAC(6) + SMAC(6) + TAGs(any) + EtherType(2) + Payload(30) + FCS(4)** - 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) + C-TAG(4) + EtherType(2) + Payload(13)** Restrictions: - SW: If enabling this function, don't add "FCS" on the frames. Because padding bits are added at the end of the frame. If there is an FCS, padding bits will be given after FCS.
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.

3.3.1.7 Timestamp function registers

(1) GWTDCAC0s (s=0 to PTP_TN-1)

GWCA Timestamp Descriptor Chain Address Configuration 0 s.

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								TSCCAUPs[AXI_AW-33:0]							

Bits	Bit name	RW-P	Initial value	Function description
31:AXI_AW-32	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_AW -33:0	TSCCAUPs	R!=W-P	0H	TS Descriptor Chain s Current Address Upper Part Update conditions: - SW: Writing to this field with update it to the write value. - HW: When an FEMPTY_ND descriptor is read for descriptor queue s, this field value is automatically updated to the next descriptor address upper part. - HW: When a LINK or LINKFIX descriptor is read, this register value is updated to **DESCR.PTR**[AXI_AW-1:32]. Restrictions: - SW: To overwrite the current address, **GWTDCAC0s** and **GWTDCAC1s** should always both be written stating by **GWTDCAC1s** and finishing by **GWTDCAC0s.**

(2) GWTDCAC1s (s=0 to PTP_TN-1)

GWCA Timestamp Descriptor Chain Address Configuration 1 s.

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

Bits	Bit name	RW-P	Initial value	Function description
31:0	TSCCADPs	R!=W-P	0H	TS Descriptor Chain s Current Address Downer Part Update conditions: - SW: Writing **GWTDCAC0s** register with update this field to the previous value written on it by the same APB. - HW: When an FEMPTY_ND descriptor is read for descriptor queue s, this field value is automatically updated to the next descriptor address downer part. - HW: When a LINK or LINKFIX descriptor is read, this register value is updated to **DESCR.PTR**[31:0]. Restrictions: - SW: To overwrite the current address, **GWTDCAC0s** and **GWTDCAC1s** should always both be written stating by **GWTDCAC1s** and finishing by **GWTDCAC0s.**

(3) GWTSDCCs (s=0 to PTP_TN-1)

GWCA TimeStamp Descriptor Chain Configuration s.

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					OSIDs[2:0]			RSV						DCSs	TEs

Bits	Bit name	RW-P	Initial value	Function description
31:11	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
10:8	OSIDs	RW-P	0B	OS ID s Functions: - When a memory access is done for timer number s, the OSID set in this register will be used.
7: PTP_TN_W+2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN_W+1:1	DCSs	RW-P	0B	Descriptor Chain Select Functions: - Select to which chain timestamps taken with timer s will be received.
0	TEs	RW-P	0H	Timer Enable s Functions: - Enable timestamp reception for timestamps taken with timer s. Restrictions: - HW: If a timestamp is received from RMAC for a timer which is disabled for timestamp reception, the timestamp will be ignored and lost. This is a valid setting (Another GWCA could have taken the timestamp) so there is no flag to monitor this event.

(4) GWTSNM

GWCA TimeStamp 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							TNTR[TS_RAM_AW1-1:0]								

Bits	Bit name	RW-P	Initial value	Function description
31: TS_RAM_AW1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
TS_RAM_AW1-1:0	TNTR	R-P	0B	Timestamp Number in Timestamp RAM Increment conditions: - HW: Incremented by 1 when a timestamp is received from a TSN Agent. Decrement conditions: - HW: Decremented by 1 when a timestamp is read from timestamp RAM.

(5) GWTSMNM

GWCA TimeStamp Maximum 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							TMNTR[TS_RAM_AW1-1:0]								

Bits	**Bit name**	**RW-P**	**Initial value**	**Function description**
31: TS_RAM_AW+1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
TS_RAM_AW:0	TMNTR	RC-P	0B	Timestamp Maximum Number in Timestamp RAM Update conditions: - HW: Set to **GWTSNM.TNTR** when **GWTSNM.TNTR> GWTSNM.TMNTR**. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Reading this register clears it to **GWTSNM.TNTR** value**.**

(6) GWAVTPTM0t (t=0..PTP_TN-1)

GWCA AVTP Timer Monitoring 0 t.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AVTPP0t[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AVTPP0t[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	AVTPP0t	R-P	0H	AVTP timer value Part 0 t Functions: - AVTP time is in nanoseconds and derive from gPTP timer t. **GWAVTPTM0t.AVTPP0t** represents either timer t 32-bit AVTP timer or timer t 64-bit AVTP timer lower bits. - This register is always equal to timer t 32-bit AVTP timer or timer t 64-bit AVTP timer lower bits.

(7) GWAVTPTM1t (t=0..PTP_TN-1)

GWCA AVTP Timer Monitoring 1 t.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AVTPP1t[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AVTPP1t[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	AVTPP1t	R-P	0H	AVTP timer value Part 0 t Functions: - AVTP time is in nanoseconds and derive from gPTP timer t. **GWAVTPTM1t.AVTPP1t** reprensents timer t 64-bit AVTP timer upper bits. Update conditions: - HW: Reading**GWAVTPTM0t** register update this register to timer t 64-bit AVTP timer upper bits.

(8) GWGPTPTM0t (t=0..PTP_TN-1)

GWCA GPTP Timer Monitoring 0 t.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV		GPTPP0t[29:16]													
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPTPP0t[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:30	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
29:0	GPTPP0t	R-P	0H	GPTP timer value Part 0 t Functions: - **GWGPTPTM0t.GPTPP0t** reprensents timer t gPTP timer nanosecond part. - This register is always equal to timer t gPTP timer nanosecond part.

(9) GWGPTPTM1t (t=0..PTP_TN-1)

GWCA GPTP Timer Monitoring 1 t

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPTPP1t[31:16]															
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPTPP1t[15:0]															

Bits	Bit name	RW-P	Initial value	Function description
31:0	GPTPP1t	R-P	0H	GPTP timer value Part 1 t Functions: - **GWGPTPTM1t.GPTPP1t** reprensents timer t gPTP timer second lower 32 bit part. Update conditions: - HW: Reading **GWGPTPTM0t** register update this register to timer t gPTP timer second lower 32-bit part.

(10) GWGPTPTM2t (t=0..PTP_TN-1)

GWCA gPTP Timer Monitoring 2 t.

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
GPTPP2t[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	GPTPP2t	R-P	0H	GPTP timer value Part 2 t Functions: - **GWGPTPTM2t.GPTPP2t** reprensents timer t gPTP timer second upper 16 bit part. Update conditions: - HW: Reading **GWGPTPTM0t** register update this register to timer t gPTP timer second upper 16-bit part.

3.3.1.8 AXI master registers

(1) GWAC

GWCA AXI Control.

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														AMP	AMPR

Bits	Bit name	RW-P	Initial value	Function description
31: 2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	AMP	R-F	0H	AXI Master Paused Values: - 1'b0: AXI master not paused (transaction ongoing). - 1'b1: AXI master paused (no transaction ongoing). Set conditions: - HW: When GWAC.AMPR is set to 1'b1 and all ongoing AXI transactions are completed, this bit will be set. Clear conditions: - HW: When GWAC.AMPR is set to 1'b0, this bit will be set to 1'b0 (it can take time because of asynchronous modules). Functions: "AXI Master Pause" stop only the final processing of the AXI interface. (Internal processing cannot be stopped.) For example, the AXI address for chain [i] is loaded immediately before "AXI Master Paused". This address will used after "AXI Master not paused" regardless of AXI address re-configuration (by **GWDCCi**) during "AXI Master Pause".
0	AMPR	RW-F	0H	AXI Master Pause Request Values: - 1'b0: AXI master pause not requested. - 1'b1: AXI master pause requested. Functions: - This register blocks the AXI master from making new accesses (Because the AXI master has register slices for timing purpose few AXI accesses can be emitted before stopping).

(2) GWDCBAC0

GWCA Descriptor Chain Base Address Configuration 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								DCBAUP[AXI_AW-33:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_AW-32	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_AW-33:0	DCBAUP	RW-P	0H	Descriptor Chain Base Address Upper Part Functions: - For further setting information, refer to section 5.1.3.1.

(3) GWDCBAC1

GWCA Descriptor Chain Base Address Configuration 1.

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

Bits	Bit name	RW-P	Initial value	Function description
31:0	DCBADP	RW-P	0H	Descriptor Chain Base Address Downer Part. Functions: - For further setting information, refer to section 5.1.3.1.

(4) GWMDNC

GWCA Maximum Descriptor Number Configuration.

!!Caution!! Be sure to change values in the initial settings. Especially when using LINK or LINK_FIX.

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

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	TSDMN	RW-RP	1H	TimeStamp Descriptor Maximum Number Functions: - Configures the maximum number of descriptors that can be processed at once for a TS queue (It limits to **GWMDNC.TSDMN**+1 the number of descriptors read that can be done for a timestamp). Refer to **GWEIS0.TSDNES**.
15:13	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
12:8	TXDMN	RW-RP	1H	TX Descriptor Maximum Number Functions: - Configures the maximum number of descriptors that can be processed at once for a TX queue (It limits to **GWMDNC.TXDMN**+1 the number of descriptors read that can be done for a TX frame). Refer to **GWEIS0.TXDNES**. Restrictions: - SW: The switch maximum input frame size is 63488 bytes so this register should be set to a value smaller or equal to 30. To avoid any unexpected frame loss due to this register, any software sending frames bigger than 58Kbytes should avoid having more than one consecutive LINK/LINKFIX descriptor in a descriptor chain. Cautions: - SW: This maximum descriptor number takes in account all descriptors including LINK, LINKFIX descriptors and descriptors remaining from a previous error frame like FMID and FEND descriptors. It should be taken in account while setting this register.
7:5	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

4:0	RXDMN	RW-RP	1H	RX Descriptor Maximum Number Functions: - Configures the maximum number of descriptors that can be processed at once for an RX queue (It limits to **GWMDNC.RXDMN**+1 the number of descriptors read that can be done for an RX frame). Refer to **GWEIS0.RXDNES** register explanation. Cautions: - SW: This maximum descriptor number takes in account all descriptors including LINK, LINKFIX descriptors and descriptors remaining from a previous error frame like FMID_EMPTY and FEND_EMPTY descriptors. It should be taken in account in account while setting this register.

(5) GWTRCi (i = 0 to AXI_CHAIN_N/32-1)

GWCA Transmission Request Configuration i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSR {i*32+31}	TSR {i*32+30}	TSR {i*32+29}	TSR {i*32+28}	TSR {i*32+27}	TSR {i*32+26}	TSR {i*32+25}	TSR {i*32+24}	TSR {i*32+23}	TSR {i*32+22}	TSR {i*32+21}	TSR {i*32+20}	TSR {i*32+19}	TSR {i*32+18}	TSR {i*32+17}	TSR {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSR {i*32+15}	TSR {i*32+14}	TSR {i*32+13}	TSR {i*32+12}	TSR {i*32+11}	TSR {i*32+10}	TSR {i*32+9}	TSR {i*32+8}	TSR {i*32+7}	TSR {i*32+6}	TSR {i*32+5}	TSR {i*32+4}	TSR {i*32+3}	TSR {i*32+2}	TSR {i*32+1}	TSR {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b=0..31	TSRt t=32*i+b	R!=W-P	0H	Transmission Start Request t Functions: - This register is used to start transfer from CPU to switch for its corresponding TX queue. Values: - 1'b0: TX Descriptor Chain is empty. - 1'b1: Transmission request is ongoing. Set conditions (only for TX queues): - SW: Writing 1 to this bit will this bit set if queue t is a transmission queue (**GWDCCt.DQTt** set). - HW: A transmission request received through **INTER_TX**[t] interface will set this bit if queue t is a transmission queue (**GWDCCt.DQTt** is set). Clear conditions: - HW: When HW read a descriptor other than FSINGLE, FSTART, FMID, FEND, LINK and LINKFIX. * - HW: When an AXI error occurs while reading a descriptor in the corresponding AXI descriptor queue. * - HW: When an ECC error occurs while reading the corresponding entry in the AXI Address RAM. * - HW: When a descriptor number error occurs while reading the corresponding AXI descriptor queue. * - HW: Going out of OPERATION mode will clear this register. Restrictions: - HW: For RX queues, this bit cannot be set.

(6) GWTPCp (p = 0 to PAS_LVL_N-1)

GWCA Transmission Pause Configuration p.

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								PPPLp[FRM_PRIO_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: FRM_PRIO_N	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
i i=0.. FRM_PRIO_N	PPPLp	RW-P	0H	Per Priority Pause Level p i Values: - 1'b0: Transmission queues with their priority **GWDCCi.DCPi** set to i are not paused when pause level p is set (**PAUSE[MY_PORT_N][p]** is set [COMA]). - 1'b1: Transmission queues with the priority **GWDCCi.DCPi** set to i are paused when pause level p is set (**PAUSE[MY_PORT_N][p]** is set [COMA]).

(7) GWARIRM

GWCA AXI 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														ARR	ARIOG

Bits	Bit name	RW-P	Initial value	Function description
31:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	ARR	R-RP	0H	AXI RAM Ready Set conditions: - When **GWARIRM.ARIOG** is getting cleared. Clear conditions: - By writing 1 to **GWARIRM.ARIOG**. - Being in RESET mode.
0	ARIOG	R!=W-RP	0B	AXI RAM Initialization Ongoing Set conditions: - SW: By writing 1 to this register. It starts AXI RAM initialization. Clear conditions: - HW: This bit is cleared when AXI RAM initialization is finished. Min clk_period * AXI_CHAIN_N = [exp] (1000/320) * 128 = 400ns

(8) GWDCCi (i = 0 to AXI_CHAIN_N-1)

GWCA Descriptor chain Configuration i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSV	OSIDi[2:0]			RSV			BALRi	RSV					DCPi [FRM_PRIO_W-1:0]		
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV				DQTi	SLi	ETSi	EDEi	RSV						SMi[1:0]	

Bits	Bit name	RW-P	Initial value	Function description
31	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
30:28	OSIDi	RW-P	0H	OS ID i Functions: - When a memory access is done for descriptor queue number i the OSID set in this register will be used.
27:25	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
24	BALRi	R!=W-P	0H	Base Address Load Request i Functions: - This bit is used to reset current_address field in the AXI address RAM to {{**GWDCBAC.DCBAUP**, **GWDCBAC.DCBADP**} + i*8}. Set conditions: - SW: Writing 1 to this bit will set it. (Low priority than Clear conditions) Clear conditions: - HW: This bit will be cleared when the current_address field in the AXI address RAM is reset.
23: FRM_PRIO_W+16	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_W+15:16	DCPi	R!=W-P	0H	Descriptor Chain Priority i Functions: - This register will be set to 3'd0 if the unsecure APB interface try to write a secure priority in this register (refer to **GWSCR0.APRSL**). Values: - 3'd7: highest priority - 3'd0: lowest priority Restrictions: - HW: This register is only valid for TX queues. - SW: This register value shouldn't be changed for a TX queue when **GWTRCj.TSRi** is set. (When transmission is ongoing for the corresponding queue)
15:12	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
11	DQTi	RW-P	0H	Descriptor Queue Type i Values: - 1'b0: Descriptor queue i is a reception queue. - 1'b1: Descriptor queue i is a transmission queue.

10	SLi	RW-RP	0H	Security Level i Values: - 1'b0: Chain i unsecure - 1'b1: Chain i secure Functions: - When a chain is secured, an unsecure descriptor cannot enter it (when a descriptor is received from forwarding engine [FWD] with **FDESCR.SEC** equal to 0). Restrictions: - HW: This register is only valid for RX queues.
9	ETSi	RW-P	0H	Enable Timestamp Storage i Values: - 1'b0: Timestamp is not added in the Descriptor. - 1'b1: Timestamp is added in the Descriptor. Restrictions: - HW: This register is only valid for RX queues.
8	EDEi	RW-P	0H	Extended Descriptor Enable i Values: - 1'b0: Basic Descriptor - 1'b1: Extended Descriptor
7:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1:0	SMi	RW-P	0H	Synchronization Mode i Values: - 2'b00: Normal mode (full descriptor writeback) - 2'b01: No-write-back mode (no descriptor writeback) - 2'b10: Keep-DT mode (no update of DT field at descriptor write back) - 2'b11 : Reserved

(9) GWAARSS

GWCA AXI Address RAM Searching Setting.

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									AARA[AXI_CHAIN_W-1:0]						

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_CHAIN_W	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W-1:0	AARA	RW-D	0H	AXI Address RAM Address Functions: - This register is used to read the current address used in AXI master for **GWAARSS.AARA** descriptor queue. Restrictions: - SW: Don't write this register from the unsecure APB. Because this is for debugging. - HW: Due to the pipeline architecture of the AXI master, the AXI read can just give an idea of which address is processing the AXI master and is not precise enough to be used for HW/SW synchronization. This register is only here for debug purpose.

(10) GWAARSR0

GWCA AXI Address RAM Searching Result 0.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AARS	RSV													AARSSF	AARSEF
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSV								ACARUP[AXI_AW-33:0]							

Bits	Bit name	RW-P	Initial value	Function description
31	AARS	R-D	0H	AXI Address RAM Searching Functions: - This register is used to read an address in AXI address RAM. Set conditions: - SW: Writing **GWAARSS** will set this bit. Clear conditions: - HW: This bit will be de-asserted when the AXI address RAM search is completed.
30:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	AARSSF	R-D	0H	AXI Address RAM Search Security fail Functions: - Shows when a security error happened while reading an entry. Update conditions: - HW: **GWAARSR0.AARS** clear event. [Notes] This bit has no purpose because unsecure APB was prohibited.
16	AARSEF	R-D	0H	AXI Address RAM Search ECC fail Functions: - Shows when an ECC error happened while reading an entry. Update conditions: - HW: **GWAARSR0.AARS** clear event.
15: AXI_AW-32	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_AW-33:0	ACARUP	R-D	0H	AXI Current Address Result Upper Part Functions: - Displays AXI address read value. Update conditions: - HW: **GWAARSR0.AARS** clear event. [Notes] AXI address read will **not** restart from the **base address** of chain after RESET mode (Because being in RESET mode will **not** clear this register).

(11) GWAARSR1

GWCA AXI Address RAM Searching Result 1.

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

Bits	Bit name	RW-P	Initial value	Function description
31:0	ACARDP	R-D	0H	AXI Current Address Result Downer Part Functions: - Displays AXI address read value. Update conditions: - HW: **GWAARSR0.AARS** clear event. [Notes] AXI address read will **not** restart from the **base address** of chain after RESET mode (Because being in RESET mode will **not** clear this register).

(12) GWIDAUASi (i=0 to AXI_RINC_N -1)

GWCA Incremental Data Area Used Area Size i.

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

Bits	Bit name	RW-P	Initial value	Function description
31:24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
23:0	IDAUASi	R!=W-P	0H	Incremental Data Area Used Size of RX Descriptor Chain i Functions: - Represents the byte numbers that have been used in the corresponding incremental area. Clear conditions: - HW: Being in RESET mode will clear this register. - HW: When a FEMPTY_IS descriptor is read in the corresponding descriptor queue, this register gets cleared. Increment conditions: - HW: When a frame has been stored in the corresponding incremental area, this register is incremented by the frame size. Decrement conditions: - SW: By writing a value, SW will decrement this register by this value. Restrictions: - SW: should always write a value smaller than the current register value.

(13) GWIDASMi (i=0 to AXI_RINC_N-1)

GWCA Incremental Data Area Size Monitoring i.

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

Bits	Bit name	RW-P	Initial value	Function description
31:24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
23:0	IDASi	R-P	0H	Incremental Data Area Size of RX descriptor chain i Functions: - This register indicates the size of RX descriptor chain i incremental area in bytes. Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - HW: When a FEMPTY_IS descriptor is read in the corresponding descriptor queue, this register is update to **DESCR.DS***4096.

(14) GWIDASAM0i (i=0 to AXI_RINC_N-1)

GWCA Incremental Data Area Start Address Monitoring 0 i.

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								IDASAUPi[AXI_AW-33:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_AW-32	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_AW-33:0	IDASAUPi	R-P	0H	Incremental Data Area Start Address Upper Part of RX descriptor chain i Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - HW: When a FEMPTY_IS descriptor is read in the corresponding descriptor queue, this register is update to **DESCR.PTR**[AXI_AW-1:32].

(15) GWIDASAM1i (i=0 to AXI_RINC_N-1)

GWCA Incremental Data Area Start Address Monitoring 1 i.

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

Bits	Bit name	RW-P	Initial value	Function description
31:0	IDASADPi	R-P	0H	Incremental Data Area Start Address Downer Part of RX descriptor chain i Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - HW: When a FEMPTY_IS descriptor is read in the corresponding descriptor queue, this register is update to **DESCR.PTR**[31:0].

(16) GWIDACAM0i (i=0 to AXI_RINC_N-1)

GWCA Incremental Data Area Current Address Monitoring 0 i.

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								IDACAUPi[AXI_AW-33:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_AW-32	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_AW-33:0	IDACAUPi	R-P	0H	Incremental Data Area Current Address Upper Part of RX descriptor chain i Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - HW: When a FEMPTY_IS descriptor is read in the corresponding descriptor queue, this register is update to **DESCR.PTR**[AXI_AW-1:32]. - HW: When a burst is decided to be sent in the incremental area, {**GWIDACAM0i.IDACAUPi**, **GWIDACAM1i.IDACADPi** } is incremented by the burst size. - HW: {**GWIDACAM0i.IDACAUPi**, **GWIDACAM1i.IDACADPi** } overflows at { **GWIDASAM0i.IDASAUPi** , **GWIDASAM1i.IDASADPi}** + **GWIDASMi.IDASi**-1**.**

(17) GWIDACAM1i (i=0 to AXI_RINC_N-1)

GWCA Incremental Data Area Current Address Monitoring 1 i.

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

Bits	Bit name	RW-P	Initial value	Function description
31:0	IDACADPi	R-P	0H	Incremental Data Area Current Address Downer Part of RX descriptor chain i Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - HW: When a FEMPTY_IS descriptor is read in the corresponding descriptor queue, this register is update to **DESCR.PTR**[31:0]. - HW: When a burst is decided to be sent in the incremental area, {**GWIDACAM0i.IDACAUPi**, **GWIDACAM1i.IDACADPi** } is incremented by the burst size. - HW: {**GWIDACAM0i.IDACAUPi**, **GWIDACAM1i.IDACADPi** } overflows at { **GWIDASAM0i.IDASAUPi , GWIDASAM1i.IDASADPi}** + **GWIDASMi.IDASi**-1**.**

3.3.1.9 Rate limiter function registers

(1) GWGRLC

GWCA Global Rate Limiter Configuration.

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

Bits	Bit name	RW-P	Initial value	Function description
31:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	GRLULRS	R!=W-P	0H	Global Rate Limiter Upper Limit Reached Status Set conditions: - HW: When the global rate limit reaches its upper limit **GWGRLULC.GRLUL**, this bit gets set. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing one to this bit will clear it. Functions: - If this bit gets set when only the global rate limiter is enabled (per port rate limiters disabled), it means that the AXI master is too slow to provide the throughput set in the global rate limiter or that the AXI latency has been underestimated. - This bit can get set while using per port rate limiters. In this case it should be ignored.
16	GRLE	RW-P	0H	Global Rate Limiter Enable Values: - 1'b0: Global Rate limiter disabled - 1'b1: Global Rate limiter enabled Functions: - The global rate limiter limits the throughput through all TX queues to avoid the switch to take in too much data at once.
15:0	GRLIV	RW-P	0H	Global Rate Limiter Incremental Value Functions: - For further setting information refer to section 5.2.1.2. Cautions: - SW: Setting this register to 0 or a value close to 0 would set GWCA to a very low throughput or could stuck GWCA in transmission.

(2) GWGRLULC

GWCA Global Rate Limiter Upper Limit Configuration.

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

Bits	Bit name	RW-P	Initial value	Function description
31:24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
23:0	GRLUL	RW-P	0H	Global Rate Limiter Upper Limit Functions: - Set the maximum number of credits in bit rate limiter that can accumulate before stopping. - For further setting information refer to section 5.2.1.2. Restrictions: - SW: This register shouldn't be set to a value bigger than 24'h800000.

(3) GWRLCi (i = 0 to AXI_TLIM_N-1)

GWCA Rate Limiter Configuration i.

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

Bits	Bit name	RW-P	Initial value	Function description
31:17	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
16	RLEi	RW-P	0H	Rate Limiter Enable i Values: - 0b Rate limiter i disabled - 1b Rate limiter i enabled Functions: - Rate limiter i limit the through put for AXI descriptor queue number AXI_CHAIN_N-1-I. For details, refer to section 5.1.1.
15:13	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
12:0	RLIVi	RW-P	0H	Rate Limiter Incremental Value for rate limiter i Functions: - For further setting information refer to section 5.2.1.2. Cautions: - SW: Setting this register to 0 or a value close to 0 would set corresponding queue to a very low throughput or could stuck the corresponding queue.

(4) GWRLULCi (i = 0 to AXI_TLIM_N-1)

GWCA Rate limiter upper limit configuration i.

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

Bits	Bit name	RW-P	Initial value	Function description
31:24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
23:0	RLULi	RW-P	0H	Rate Limiter Upper Limit for rate limiter i Functions: - Set the maximum number of credits in bit rate limiter i can accumulate before stopping. - This register can be used if a lot of traffic happens on the AXI bus so burst on queues i make the switch overflow. If not required, set this register to 24'h80_0000. - For further setting information refer to section 5.2.1.2. Restrictions: - SW: This register shouldn't be set to a value bigger than 24'h80_0000.

3.3.1.10 Interrupt function registers

(1) GWIDPC

GWCA Interrupt Delay Prescaler 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						IDPV[9:0]									

Bits	Bit name	RW-P	Initial value	Function description
31:10	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
9:0	IDPV	RW-RP	0H	Interrupt Delay Prescaler Value Functions: - This register is used to create an internal clock. This internal clock period will be *clk_delay_period* = *clk_period**(**GWIDPC.IDPV**+1) where *clk_period* is the period of clk. - For further information, refer to section 5.6.

(2) GWIDCi (i = 0 to AXI_CHAIN_N)

GWCA Interrupt Delay 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				IDVi[11:0]											

Bits	Bit name	RW-P	Initial value	Function description
31:12	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
11:0	IDVi	RW-P	0H	Interrupt Delay Value i Functions: - Configures the delay between the interrupt status flag **GWDISi.DISt** setting and the actual interrupt event in us. - If this register is set to 0, data interrupt will occur without delay. - If value X is set to this register, the interrupt will be delayed by a time between X-1**clk_delay_period* us and X**clk_delay_period* us where *clk_delay_period* is the internal clock created using **GWIDPC** register period. - For further information, refer to section 5.6.

3.3.2 GWCA Counter registers

(1) GWRDCN

GWCA Received Data 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
RDN[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
COUNT_MED_W-1:0	RDN	RC-P	0H	Received Data Number Functions: - This register counts data that has been received by GWCA (data aborted by AXI error are accounted here). 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 is passed from multicast control to AXI master interface and if this register has a value different than {{COUNT_MED_W}{1'b1}}.

(2) GWTDCN

GWCA Transmitted Data 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
TDN[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
COUNT_MED_W-1:0	TDN	RC-P	0H	Transmitted Data Number Functions: - This register counts data that has been transmit by GWCA (this includes data transmitted with an error and frames smaller than 32 bytes). 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 is passed from AXI master interface to TX data store and if this register has a value different than {{COUNT_MED_W}{1'b1}}.

(3) GWTSCN

GWCA TimeStamp 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
TN[COUNT_MED_W-1:0]															

Bits	Bit name	RW-P	Initial value	Function description
COUNT_MED_W-1:0	TN	RC-P	0H	Timestamp Number Functions: - This register counts timestamp processed by GWCA (TS lost because of AXI errors are accounted here). 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 Timestamp is saved in timestamp RAM and if this register has a value different than {{COUNT_MED_W}{1'b1}}.

(4) GWTSOVFECN

GWCA TimeStamp 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
TSOVFEN[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	TSOVFEN	RC-P	0H	TimeStamp OVerFlow Error Number Functions: - This register counts the number of timestamps lost because of timestamp RAM overflow. 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 timestamp overflow error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(5) GWUSMFSECN

GWCA Under 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 transmit 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}}.

(6) GWTFECN

GWCA 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 transmit 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}}.

(7) GWSEQECN

GWCA SEQuence 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
SEQEN[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	SEQEN	RC-P	0H	SEQuence Error Number Functions: - This register counts the number of transmit data lost because of a sequence 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 sequence error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(8) GWTXDNECN

GWCA TX Descriptor Number 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
TXDNEN[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	TXDNEN	RC-P	0H	TX Descriptor Number Error Number Functions: - This register counts the number of transmit data lost because of a TX Descriptor Number 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 TX Descriptor Number Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(9) GWFSECN

GWCA 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 receive 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}}.

(10) GWTDFECN

GWCA Timestamp Descriptor Full 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
TDFEN[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	TDFEN	RC-P	0H	Timestamp Descriptor Full Error Number Functions: - This register counts the number of timestamps lost because of a Timestamp Descriptor Full 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 Timestamp Descriptor Full Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(11) GWTSDNECN

GWCA Timestamp Descriptor Number 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
TSDNEN[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	TSDNEN	RC-P	0H	Timestamp Descriptor Number Error Number Functions: - This register counts the number of timestamps lost because of a Timestamp Descriptor Number 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 Timestamp Descriptor Number Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(12) GWDQOECN

GWCA 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 receive 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}}.

(13) GWDQSECN

GWCA 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 receive 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}}.

(14) GWDFECN

GWCA Descriptor Full 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
DFEN[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	DFEN	RC-P	0H	Descriptor Full Error Number Functions: - This register counts the number of receive data lost because of a Descriptor Full 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 Full Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(15) GWDSECN

GWCA Descriptor 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
DSEN[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	DSEN	RC-P	0H	Descriptor Security Error Number Functions: - This register counts the number of receive data lost because of a Descriptor 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 Security Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(16) GWDSZECN

GWCA Data 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
DSZEN[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	DSZEN	RC-P	0H	Data SiZe Error Number Functions: - This register counts the number of receive data lost because of a Data 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 Data Size Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(17) GWDCTECN

GWCA Descriptor Chain Type 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
DCTEN[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	DCTEN	RC-P	0H	Descriptor Chain Type Error Number Functions: - This register counts the number of receive data lost because of a Descriptor Chain Type 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 Chain Type Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(18) GWRXDNECN

GWCA RX Descriptor Number 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
RXDNEN[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	RXDNEN	RC-P	0H	RX Descriptor Number Error Number Functions: - This register counts the number of receive data lost because of a RX Descriptor Number 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 RX Descriptor Number Error happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

(19) GWCKSECN

GWCA 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 transmit 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}}.

(20) GWLDCN

GWCA 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 **GWRDRC.RDRM**==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}}.

(21) GWDQOECNPq (q=0..FRM_PRIO_N-1)

GWCA 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 receive 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 priority q happens and if this register has a value different than {{COUNT_LOW_W}{1'b1}}.

3.3.3 GWCA Interrupt registers

This section describes, SFR for GWCPU interrupt and the set / clear condition of each status.

3.3.3.1 Data interrupt registers

(1) GWDISi (i = 0 to AXI_CHAIN_N/32-1)

GWCA Data Interrupt Status i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIS {i*32+31}	DIS {i*32+30}	DIS {i*32+29}	DIS {i*32+28}	DIS {i*32+27}	DIS {i*32+26}	DIS {i*32+25}	DIS {i*32+24}	DIS {i*32+23}	DIS {i*32+22}	DIS {i*32+21}	DIS {i*32+20}	DIS {i*32+19}	DIS {i*32+18}	DIS {i*32+17}	DIS {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIS {i*32+15}	DIS {i*32+14}	DIS {i*32+13}	DIS {i*32+12}	DIS {i*32+11}	DIS {i*32+10}	DIS {i*32+9}	DIS {i*32+8}	DIS {i*32+7}	DIS {i*32+6}	DIS {i*32+5}	DIS {i*32+4}	DIS {i*32+3}	DIS {i*32+2}	DIS {i*32+1}	DIS {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	DISt t=32*i+b	R!=W-P	0H	Data Interrupt Status t Set conditions TX: - HW: A descriptor has been processed in the corresponding queue and **DESCR.DIE** was set in the corresponding read descriptor. [About set timing information for normal TX] A descriptor is considered as processed when written back. All descriptor which are not written back are considered as processed at the timing it should have been written back so that SW can know that all the previous at the actual descriptor has been fully processed. Set conditions RX: - HW: A frame has been processed in the corresponding queue and **DESCR.DIE** was set in at least one of the corresponding read descriptors. [About set timing information for normal RX] A frame is considered as processed when frame FSTART or FSINGLE descriptor is processed. A descriptor is considered as processed when written back. All descriptor which are not written back are considered as processed at the timing it should have been written back so that SW can know that all the previous at the actual descriptor has been fully processed. All unknown descriptors are never considered as processed (Refer to **GWEIS2i.DFESt**). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it.

(2) GWDIEi (i = 0 to AXI_CHAIN_N/32-1)

GWCA Data Interrupt Enable i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIE {i*32+31}	DIE {i*32+30}	DIE {i*32+29}	DIE {i*32+28}	DIE {i*32+27}	DIE {i*32+26}	DIE {i*32+25}	DIE {i*32+24}	DIE {i*32+23}	DIE {i*32+22}	DIE {i*32+21}	DIE {i*32+20}	DIE {i*32+19}	DIE {i*32+18}	DIE {i*32+17}	DIE {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIE {i*32+15}	DIE {i*32+14}	DIE {i*32+13}	DIE {i*32+12}	DIE {i*32+11}	DIE {i*32+10}	DIE {i*32+9}	DIE {i*32+8}	DIE {i*32+7}	DIE {i*32+6}	DIE {i*32+5}	DIE {i*32+4}	DIE {i*32+3}	DIE {i*32+2}	DIE {i*32+1}	DIE {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	DIEt t=32*i+b	R!=W-P	0H	Data Interrupt Enable t 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 **GWDIDi.DIDt** register will clear this bit.

(3) GWDIDi (i = 0 to AXI_CHAIN_N/32-1)

GWCA Data Interrupt Disable i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DID {i*32+31}	DID {i*32+30}	DID {i*32+29}	DID {i*32+28}	DID {i*32+27}	DID {i*32+26}	DID {i*32+25}	DID {i*32+24}	DID {i*32+23}	DID {i*32+22}	DID {i*32+21}	DID {i*32+20}	DID {i*32+19}	DID {i*32+18}	DID {i*32+17}	DID {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DID {i*32+15}	DID {i*32+14}	DID {i*32+13}	DID {i*32+12}	DID {i*32+11}	DID {i*32+10}	DID {i*32+9}	DID {i*32+8}	DID {i*32+7}	DID {i*32+6}	DID {i*32+5}	DID {i*32+4}	DID {i*32+3}	DID {i*32+2}	DID {i*32+1}	DID {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	DIDt t=32*i+b	R0W-P	0H	Data Interrupt Disable t Functions: - Writing 1 to this bit will clear **GWDIEi.DIEt** register.

(4) GWDIDSi (i = 0 to AXI_CHAIN_N/32-1)

GWCA Data Interrupt Delayed Status i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIDS {i*32+31}	DIDS {i*32+30}	DIDS {i*32+29}	DIDS {i*32+28}	DIDS {i*32+27}	DIDS {i*32+26}	DIDS {i*32+25}	DIDS {i*32+24}	DIDS {i*32+23}	DIDS {i*32+22}	DIDS {i*32+21}	DIDS {i*32+20}	DIDS {i*32+19}	DIDS {i*32+18}	DIDS {i*32+17}	DIDS {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIDS {i*32+15}	DIDS {i*32+14}	DIDS {i*32+13}	DIDS {i*32+12}	DIDS {i*32+11}	DIDS {i*32+10}	DIDS {i*32+9}	DIDS {i*32+8}	DIDS {i*32+7}	DIDS {i*32+6}	DIDS {i*32+5}	DIDS {i*32+4}	DIDS {i*32+3}	DIDS {i*32+2}	DIDS {i*32+1}	DIDS {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	DIDSt t=32*i+b	R-P	0H	Data Interrupt Delayed status t Functions: - Displays the interrupt **GWDISi.DISt** delayed by interrupt delay function.

(5) GWTSDIS

GWCA TimeStamp Data Interrupt 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														TSDIS [PTP_TN-1:0]	

Bits	Bit name	RW-P	Initial value	Function description
31:PTP_TN	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
s s=0..PTP_TN-1	TSDISs	R!=W-P	0H	Time Stamp Data Interrupt Status s Set conditions: - HW: A descriptor is processed in the corresponding queue and **DESCR.DIE** was set in the corresponding read descriptor. A descriptor is considered as processed when written back. All descriptor which are not written back are considered as processed at the timing it should have been written back so that SW can know that all the previous at the actual descriptor has been fully processed. All unknown descriptors are never considered as processed (Refer to **GWEIS0.TDFES**). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it.

(6) GWTSDIE

GWCA TimeStamp Data Interrupt Enable.

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														TSDIE [PTP_TN-1:0]	

Bits	Bit name	RW-P	Initial value	Function description
31:PTP_TN	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
s s=0..PTP_TN-1	TSDIEs	R!=W-P	0H	TimeStamp Data Interrupt Enable s 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 **GWTSDID.TSDIDs** register will clear this bit.

(7) GWTSDID

GWCA TimeStamp Data Interrupt Disable.

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														TSDID [PTP_TN-1:0]	

Bits	Bit name	RW-P	Initial value	Function description
31:PTP_TN	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
s s=0..PTP_TN-1	TSDIDs	R0W-P	0H	TimeStamp Data Interrupt Disable s Functions: - Writing 1 to this bit will clear **GWTSDIE.TSDIEs** register.

3.3.3.2 Error interrupt registers

This subsubsection describes SFR for GWCA error interrupt.

(1) GWEIS0

GWCA Error Interrupt Status 0.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CKSES	RSV	TSDNES [PTP_TN-1:0]		RSV		TDFES [PTP_TN-1:0]		FSES[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSHES	TXDNES	RSV	SEQES	TFES	USMFSES	TSOVFES	L23UECCES	TSECCES	AECCES	MECCES	DSECCES	PECCES	TECCES	DECCES	AES

Bits	Bit name	RW-P	Initial value	Function description
31	CKSES	R!=W-P	0H	ChecKSum Error Status Set conditions: - HW: A checksum error has been detected. (Refer to section 5.2.3.4). 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 5.2.3.5). - SW: System dependent, cannot de defined here.
30	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN+27:28	TSDNES	R!=W-F	0H	TimeStamp Descriptor Number Error Status Set conditions: - HW: Bit i sets when **GWMDNC.TSDMN**+1 descriptor has already been used to process one timestamp for descriptor chain i and the timestamp processing is not finished. 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: Timestamp lost - SW: Software error, software should be reviewed.
27: PTP_TN+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

PTP_TN+23:24	TDFES	R!=W-P	0H	Timestamp Descriptor Full Error Status Set conditions: - HW: Bit i sets when a timestamp has been received for descriptor chain i which is full (read any other descriptor than a FEMPTY_ND, LINKFIX or LINK descriptor). 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: Timestamp lost. - SW: Nothing if it is acceptable to lose a timestamp else switch reset. - SW: AXI is too slow to save all timestamps in the CPU RAM. Less timestamps should be captured or more bandwidth on AXI should be reserved for R-switch.
FRM_PRIO_N+15:16	FSES	R!=W-P	0H	Frame Size Error Status Set conditions: - HW: Bit q of this register is set when a frame bigger than **GWRMFSC.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.
15	TSHES	R!=W-P	0H	TimeStamp Hardware Error Status Set conditions: - HW: This error happens when timestamps are coming too fast to GWCA to be stored in time, so timestamp are lost even if the timestamp RAM is not full. This error should never happen and tells that the number of captured timestamps is too high or the bus clock clk has a too low frequency. 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: timestamp lost. - SW: No recovery, this error shouldn’t happen.

14	TXDNES	R!=W-F	0H	TX Descriptor Number Error Status Set conditions: - HW: **GWMDNC.TXDMN**+1 descriptor has already been used to process one frame and the frame processing is not finished. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Restrictions: - HW: This flag is only available for TX descriptor queues. Error recovery: - HW: If a TX Descriptor Number Error happens before a frame transfer started, **GWTRCi.TSRt** bit is cleared. - HW: If an TX Descriptor Number Error happens after a frame transfer started, the ongoing frame is padded with 8'h0, sent to Forwarding Engine [FWD] with DNE error set (section 5.2.3.5(3)) and corresponding **GWTRCi.TSRt** bit is cleared. - SW: Software error, software should be reviewed.
13	RSV	R0-U	0H	- Reserved area. On read, 0 will be returned.
12	SEQES	R!=W-P	0H	SEQuence Error Status Set conditions: - HW: A FMID, FEND has been received before a frame transfer started. - HW: A descriptor different than FMID, FEND, LINK and LINKFIX has been received after a frame transfer started. - HW: An FSTART, FMID, FEND or FSINGLE descriptor has been received with **DESCR.DS** set to 0. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to one of these bits will clear it. Restrictions: - HW: This flag is only available for TX descriptor queues. Error recovery: - HW: If a SEQuence Error happens before a frame transfer started, nothing happens. - HW: If a SEQuence Error happens after a frame transfer started, The ongoing frame is padded with 8'h0 and sent to Forwarding Engine [FWD] with SEQE error set (section 5.2.3.5(3)). - SW: Software error, software should be reviewed.
11	TFES	R!=W-P	0H	TAG Filtering Error Status Set conditions: - HW: An unauthorized TAG format has been detected. (Refer to section 5.2.3.2). 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 5.2.3.5). - SW: Software error, software should be reviewed.

10	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 CPU. 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: Software error, software should be reviewed.
9	TSOVFES	R!=W-P	0H	TimeStamp OVerFlow Error Status Set conditions: - HW: When a timestamp is received when the timestamp RAM is full. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: A timestamp is lost. - SW: Nothing if it is acceptable to lose a timestamp else switch reset.
8	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 GWCA using L2/L3 update bus. The error is flag to GWCA 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]
7	TSECCES	R!=W-P	0H	TimeStamp ECC Error Status Set conditions: - HW: When an ECC error has been detected while reading a timestamp from timestamp RAM. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: A timestamp is lost. - SW: Nothing if it is acceptable to lose a timestamp else switch reset.

6	AECCES	R!=W-P	0H	AXI ECC Error Status Set conditions: - HW: When an ECC error has been detected while reading an AXI address from the AXI Address Table. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Restrictions: - SW: The entry number is not given by GWCA for error recovery, this value should be read from the AXI address RAM ECC module. Error recovery: - HW: For reception, if an AXI ECC error happens before a frame transfer started, the frame/timestamp is rejected. - HW: For reception, if an AXI error happens during descriptor read after a frame/timestamp transfer started, the remaining part of the frame/timestamp is rejected and **DESCR.ERR** is set in the frame/timestamp corresponding first written back data descriptor, which is FSTART in normal synchronization mode (for **GWDCCi.SMi** == 2'b00). - HW: For transmission, if an AXI ECC error happens before a frame transfer started, corresponding **GWTRCi.TSRt** bit is cleared and no frame is transmitted. - HW: For transmission, if an AXI ECC error happens after a frame transfer started, the ongoing frame is padded with 8'h0, sent to Forwarding Engine [FWD] with AREE error set (section 5.2.3.5(3)) and corresponding **GWTRCi.TSRt** bit is cleared. - SW: Clear descriptor queue in URAM and reset entry by setting corresponding **GWDCCi.BALRi** register.
5	MECCES	R!=W-P	0H	Multicast ECC Error Status Set conditions: - HW: When an ECC error has been detected while reading multicast entry from multicast and DCNC table (This flag is not set when SW reads the multicast RAM and an ECC error happens. In this case, ECC error detection should happen using **GWMSTSR.MTSEF** register). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Restrictions: - SW: The entry number is not given by GWCA for error recovery, this value should be read from the Multicast and DCNC Table RAM ECC module. - HW: This bit is not set during SW read through **GWMSTSS/GWMSTSR** registers. Error recovery: - HW: The current frame multicast operation will stop. Next frames will be processed normally. The current frame will be disabled DCNC and will be selected incoming CSD. - SW: Reset entry by **GWMSTLS** and **GWMSTLR** registers.

4	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.
3	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: Set **DESCR.ERR** bit in RX descriptor. - SW: Discard data which have **DESCR.ERR** bit set. - SW: Nothing if it is acceptable to operate with a reduced Local RAM else Switch reset.
2	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 and one or more TAGs are forwarded to the CPU (forwarded and not overwritten by L2/L3 update function). 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: Set **DESCR.ERR** bit in RX descriptor. - SW: Discard data which have **DESCR.ERR** bit set.
1	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: Set **DESCR.ERR** bit in RX descriptor. - SW: Discard data which have **DESCR.ERR** bit set.

0	AES	R!=W-P	0H	AXI Error Status Set conditions: - HW: When an error is detected on an AXI access (xRESP != 2'b00). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Error recovery: - HW: For reception, if an AXI error happens during descriptor read before a frame/timestamp transfer started, the frame/timestamp is rejected. - HW: For reception, if an AXI error happens during descriptor read after a frame transfer started, the remaining part of the frame is rejected and **DESCR.AXIE** is set in the frame corresponding first written back data descriptor, which is FSTART in normal synchronization mode (for **GWDCCi.SMi** == 2'b00). - HW: For reception, if an AXI error happens during data write, **DESCR.AXIE** is set in the frame corresponding first written back data descriptor, which a FSTART or a FSINGLE descriptor in normal synchronization mode (for **GWDCCi.SMi** == 2'b00). - HW: For reception, if an AXI error happens during RX/TS descriptor write, nothing happens. - HW: For transmission, if an AXI error happens during descriptor read and no frame is ongoing, corresponding **GWTRCi.TSRt** bit is cleared and no frame is transmitted. - HW: For transmission, if an AXI error happens during descriptor read and a frame is ongoing, the ongoing frame is padded with 8'h0, sent to Forwarding Engine [FWD] with AXE error set (section 5.2.3.5(3)) and corresponding **GWTRCi.TSRt** bit is cleared. - HW: For transmission, if an AXI error happens during data read, the ongoing frame is sent to Forwarding Engine [FWD] with AXE error set (section 5.2.3.5(3)). - HW: For transmission, if an AXI error happens during descriptor write, nothing happens. - SW: System reset.

(2) GWEIE0

GWCA Error Interrupt Enable 0.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CKSEE	RSV	TSDNEE [PTP_TN-1:0]		RSV		TDFEE [PTP_TN-1:0]		FSEE[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSHEE	TXDNEE	RSV	SEQEE	TFEE	USMFSEE	TSOVFEE	L23UECCEE	TSECCEE	AECCEE	MECCEE	DSECCEE	PECCEE	TECCEE	DECCEE	AEE

Bits	Bit name	RW-P	Initial value	Function description
31	CKSEE	R!=W-P	0H	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 **GWEID0.CKSED** register will clear this bit.
30	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN+27:28	TSDNEE	R!=W-F	0H	TimeStamp Descriptor Number 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 one of **GWEID0.TSDNED** bits will clear the corresponding bit in this register.
27: PTP_TN+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN+23:24	TDFEE	R!=W-P	0H	Timestamp Descriptor Full 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 **GWEID0.TDFEID** bits will clear the corresponding bit in this register.

FRM_PRIO_N+15: 16	FSEE	R!=W-P	0H	Frame Size 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 one of **GWEID0.FSED** bits will clear the corresponding bit in this register.
15	TSHEE	R!=W-P	0H	TimeStamp Hardware 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 **GWEID0.TSHED** register will clear this bit.
14	TXDNEE	R!=W-F	0H	TX Descriptor Number 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 **GWEID0.TXDNED** register will clear this bit.
13	RSV	R0-U	0H	- Reserved area. On read, 0 will be returned.
12	SEQEE	R!=W-P	0H	SEQuence 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 **GWEID0.SEQED** register will clear this bit.
11	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 **GWEID0.TFED** register will clear this bit.

10	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 **GWEID0.USMFSED** register will clear this bit.
9	TSOVFEE	R!=W-P	0H	TimeStamp 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 **GWEID0.TSOVFED** register will clear this bit.
8	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 **GWEID0.L23UECCED** register will clear this bit.
7	TSECCEE	R!=W-P	0H	TimeStamp 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 **GWEID0.AECCED** register will clear this bit.
6	AECCEE	R!=W-P	0H	AXI 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 **GWEID0.AECCED** register will clear this bit.

5	MECCEE	R!=W-P	0H	Multicast 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 **GWEID0.MECCED** register will clear this bit.
4	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 **GWEID0.DSECCEED** register will clear this bit.
3	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 **GWEID0.PECCED** register will clear this bit.
2	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 **GWEID0.TECCED** register will clear this bit.
1	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 **GWEID0.DECCED** register will clear this bit.

0	AEE	R!=W-P	0H	AXI 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 **GWEID0.AED** register will clear this bit.

(3) GWEID0

GWCA Error Interrupt Disable 0.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CKSED	RSV	TSDNED [PTP_TN-1:0]		RSV		TDFED [PTP_TN-1:0]		FSED[FRM_PRIO_N-1:0]							
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSHED	TXDNED	RSV	SEQED	TFED	USMFSED	TSOVFED	L23UECCED	TSECCES	AECCED	MECCED	DSECCED	PECCED	TECCED	DECCED	AED

Bits	Bit name	RW-P	Initial value	Function description
31	CKSED	R0W-P	0H	ChecKSum Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.CHSEE** register.
30	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN+27:28	TSDNED	R0W-F	0H	TimeStamp Descriptor Number Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **GWEIE0.TSDNED** register.
27: PTP_TN+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN+23:24	TDFED	R0W-P	0H	Timestamp Descriptor Full Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **GWEIE0.TDFEIE** register.
FRM_PRIO_N+15:16	FSED	R0W-P	0H	Frame Size Error Disable Functions: - Writing 1 to one of these bits will clear the corresponding bit in **GWEIE0.FSEE** register.
15	TSHED	R0W-P	0H	TimeStamp Hardware Full Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.TSHEE** register.
14	TXDNED	R0W-F	0H	TX Descriptor Number Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.TXDNEE** register.
13	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
12	SEQED	R0W-P	0H	SEQuence Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.SEQEE** register.
11	TFED	R0W-P	0H	TAG Filtering Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.TFEE** register.
10	USMFSED	R0W-P	0H	Under Switch Minimum Frame Size Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.USMFSEE** register.
9	TSOVFED	R0W-P	0H	TimeStamp OVerFlow Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.TSOVFEE** register.

8	L23UECCED	R0W-P	0H	Layer 2/3 Update ECC Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.L23UECCEE** register.
7	TSECCED	R0W-P	0H	TimeStamp ECC Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.TSECCEE** register.
6	AECCED	R0W-P	0H	AXI ECC Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.AECCEE** register.
5	MECCED	R0W-P	0H	Multicast ECC error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.MECCEE** register.
4	DSECCED	R0W-P	0H	Descriptor ECC Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.DSECCEE** register.
3	PECCED	R0W-P	0H	Pointer ECC error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.PECCEE** register.
2	DECCED	R0W-P	0H	Data ECC error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.DECCEE** register.
1	TECCED	R0W-P	0H	TAG ECC error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.TECCEE** register.
0	AED	R0W-P	0H	AXI Error Disable Functions: - Writing 1 to this bit will clear **GWEIE0.AEE** register.

(4) GWEIS1

GWCA Error Interrupt Status 1.

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								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 **GWRDQMq.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 (**GWRDQMq.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 (**GWRDQSC.RDQSL[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 GWCA and will not be forwarded to CPU. - SW: System dependent, cannot de defined here.
15:8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

FRM_PRIO_N-1:0	DQOES	R!=W-P	0H	Descriptor Queue Overflow Error Status [Cond1] [GWCA is ongoing] (**GWMC.OPC** == 2'b11 and **GWMS.OPS** == 2'b11) and a descriptor is received for queue q. [Cond2] Descriptor queue is "full (**GWRDQDCq.DQDq** == **GWRDQMq.DNQq**)" and "not disabled (**GWRDQC.TDQD[q]** is not set). [Cond3] "**GWRDRC.RDRM** is 1", (Descriptor queue is "full (**GWRDQDCq.DQDq** == **GWRDQMq.DNQq**)" or "descriptor RAM is full (Sum of **GWRDQMq.DNQq [q=0..FRM_PRIO_N-1]** == (**DES_RAM_DP - FRM_PRIO_N *2**))") and "not disabled (**GWRDQC.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 GWCA and will not be forwarded to CPU. - SW: Too many descriptors are received for the corresponding queue. **GWRDQDCq.DQDq** setting should be reviewed.

(5) GWEIE1

GWCA Error Interrupt Enable 1.

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								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 **GWEID1.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 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 **GWEID1.DQSED** register will clear the bit q in this register.
15:8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N-1:0	DQOEE	R!=W-P	0H	Descriptor Queue Overflow 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 **GWEID1.DQOED** register will clear the bit q in this register.

(6) GWEID1

GWCA Error Interrupt Disable 1.

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								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 **GWEIE1.DIECCEE** register.
FRM_PRIO_N+15:16	DQSED	R0W-F	0H	Descriptor Queue Security Error Disable Functions: - Writing 1 to bit q in this register will clear bit q in **GWEIE1.DQSEE** register.
15:8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
FRM_PRIO_N-1:0	DQOED	R0W-P	0H	Descriptor Queue Overflow Error Disable Functions: - Writing 1 to bit q in this register will clear bit q in **GWEIE1.DQOEE** register.

(7) GWEIS2i (i = 0 to AXI_CHAIN_N/32-1)

GWCA Error Interrupt Status 2 i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DFES {i*32+31}	DFES {i*32+30}	DFES {i*32+29}	DFES {i*32+28}	DFES {i*32+27}	DFES {i*32+26}	DFES {i*32+25}	DFES {i*32+24}	DFES {i*32+23}	DFES {i*32+22}	DFES {i*32+21}	DFES {i*32+20}	DFES {i*32+19}	DFES {i*32+18}	DFES {i*32+17}	DFES {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DFES {i*32+15}	DFES {i*32+14}	DFES {i*32+13}	DFES {i*32+12}	DFES {i*32+11}	DFES {i*32+10}	DFES {i*32+9}	DFES {i*32+8}	DFES {i*32+7}	DFES {i*32+6}	DFES {i*32+5}	DFES {i*32+4}	DFES {i*32+3}	DFES {i*32+2}	DFES {i*32+1}	DFES {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	DFESt t=32*i+b	R!=W-P	0H	Descriptor Full Error Status t. Set conditions: - HW: When a frame has been received for descriptor queue t which is full (read a descriptor different than FEMPTY, FEMPTY_IS, FEMPTY_IC, FEMPTY_ND, FEMPTY_START, FEMPTY_MID, FEMPTY_END, LINKFIX and LINK). This flag is set at the timing the frame last descriptor write-back should have been completed. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Restrictions: - HW: This register is only valid for RX queues. Error recovery: - HW: set of **DESCR.ERR** bit in RX descriptor if a part of the frame has been already sent. - HW: Current frame or remaining part of the frame will be discarded, and next frames will be processed normally (as long as software doesn't insert new descriptors for reception the queue will continue overflowing). - SW: Discard data which have **DESCR.ERR** bit set.

(8) GWEIE2i (i = 0 to AXI_CHAIN_N/32-1)

GWCA Error Interrupt Enable 2 i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DFEE {i*32+31}	DFEE {i*32+30}	DFEE {i*32+29}	DFEE {i*32+28}	DFEE {i*32+27}	DFEE {i*32+26}	DFEE {i*32+25}	DFEE {i*32+24}	DFEE {i*32+23}	DFEE {i*32+22}	DFEE {i*32+21}	DFEE {i*32+20}	DFEE {i*32+19}	DFEE {i*32+18}	DFEE {i*32+17}	DFEE {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DFEE {i*32+15}	DFEE {i*32+14}	DFEE {i*32+13}	DFEE {i*32+12}	DFEE {i*32+11}	DFEE {i*32+10}	DFEE {i*32+9}	DFEE {i*32+8}	DFEE {i*32+7}	DFEE {i*32+6}	DFEE {i*32+5}	DFEE {i*32+4}	DFEE {i*32+3}	DFEE {i*32+2}	DFEE {i*32+1}	DFEE {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	DFEEt t=32*i+b	R!=W-P	0H	Descriptor Full Error Enable t 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 **GWEID2i.DFEDt** register will clear this bit.

(9) GWEID2i (i = 0 to AXI_CHAIN_N/32-1)

GWCA Error Interrupt Disable 2 i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DFED{i*32+31}	DFED{i*32+30}	DFED{i*32+29}	DFED{i*32+28}	DFED{i*32+27}	DFED{i*32+26}	DFED{i*32+25}	DFED{i*32+24}	DFED{i*32+23}	DFED{i*32+22}	DFED{i*32+21}	DFED{i*32+20}	DFED{i*32+19}	DFED{i*32+18}	DFED{i*32+17}	DFED{i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DFED{i*32+15}	DFED{i*32+14}	DFED{i*32+13}	DFED{i*32+12}	DFED{i*32+11}	DFED{i*32+10}	DFED{i*32+9}	DFED{i*32+8}	DFED{i*32+7}	DFED{i*32+6}	DFED{i*32+5}	DFED{i*32+4}	DFED{i*32+3}	DFED{i*32+2}	DFED{i*32+1}	DFED{i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	DFEDt t=32*i+b	R0W-P	0H	Descriptor Full Error Enable t Functions: - Writing 1 to this bit will clear **GWEIE2i.DFEEt** register.

(10) GWEIS3

GWCA Error Interrupt Status 3.

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								IAOES[AXI_RINC_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_RINC_N	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
i i=0..AXI_RINC_N	IAOES	R!=W-P	0H	Incremental Area Overflow Error Status i Set conditions: - HW: When incremental area i overflows, for further information refer to 5.3.5.3. This flag is only set when **GWIDAUASt.IDAUASi** crosses **GWIDASMi.IDASi** value (if incremental area is already overflowing this flag won't be set again exept if **GWIDAUASt.IDAUASi** overflows). Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Restrictions: - HW: This register is only valid for RX incremental queues. Error recovery: - HW: The data coming for this incremental area will still be written. - SW: Set a new incremental area.

(11) GWEIE3

GWCA Error Interrupt Enable 3.

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								IAOEE[AXI_RINC_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_RINC_N	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
i i=0..AXI_RINC_N	IAOEE	R!=W-P	0H	Incremental Area Overflow Error Enable i 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 **GWEID3.IAOEDi** register will clear this bit.

(12) GWEID3

GWCA Error Interrupt Disable 3.

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								IAOED[AXI_RINC_N-1:0]							

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_RINC_N	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
i i=0..AXI_RINC_N	IAOED	R0W-P	0H	Incremental Area Overflow Error Disable i Functions: - Writing 1 to this bit will clear **GWEIE3.IAOEDi** register.

(13) GWEIS4

GWCA Error Interrupt Status 4.

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

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_CHAIN_W+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W+23 :24	DSECN	R-P	0H	Data Size Error Chain Number Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - This field is update to the descriptor chain number in which a Data Size Error occurred when **GWEIS4.DSEIS** is being set. Restrictions: - HW: This register is only valid for RX queues.
23:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	DSEIOS	R!=W-P	0H	Data Size Error Interrupt Overflow Status Set conditions: - When a Data Size Error occurs and **GWEIS4.DSES** is already set to one. In this case, the descriptor chain number in which the new error occurred is lost. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Restrictions: - HW: This register is only valid for RX queues. Error recovery: - SW: Switch reset.

16	DSES	R!=W-P	0H	Data Size Error Status Set conditions: - HW: When a frame has been received with an unexpected size, for more information refer to 5.3.5.2. This flag is set after the frame last descriptor write back is completed. Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Restrictions: - HW: This register is only valid for RX queues. Error recovery: - HW: set of **DESCR.DSE** bit in FSTART RX descriptor. - HW: If the frame was an undersized frame the frame is written with the normal descriptor sequence (FSINGLE or FSTART/ FEND or FSTART/n*FMID/ FEND). - HW: If the frame was an undersized frame, when a new frame comes for the corresponding queue, the AXI master will skip all following descriptors until a empty descriptor which is not FEMPTY_MID or FEMPTY_END (at start of a frame, FEMPTY_MID and FEMPTY_END descriptors are always ignored). - If the frame is an oversized frame, the frame will be truncated. - SW: Discard data which have **DESCR.DSE** bit set. - SW: Because **DESCR.DSE** is set in RX descriptors with a size error, it is not mandatory to read this flag for error recovery.
15: AXI_CHAIN_W+8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W+7: 8	DSSECN	R-F	0H	Descriptor Security Error Chain Number Restrictions: - HW: This register is only valid for RX queues. Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - This field is update to the descriptor chain number in which a Descriptor Security Error occurred when **GWEIS4.DSSES** is being set.
7:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	DSSEIOS	R!=W-F	0H	DeScriptor Security Error Interrupt Overflow Status Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Set conditions: - When a Descriptor Security Error occurs and **GWEIS4.DSSES** is already set to one. In this case, the descriptor chain number in which the new error occurred is lost. Restrictions: - HW: This register is only valid for RX queues. Security restrictions: - HW: This register cannot be accessed by the unsecure APB interface. Error recovery: - SW: Switch reset.

0	DSSES	R!=W-F	0H	DeScriptor Security Error Status Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Set conditions: - HW: During reception, when an unsecure descriptor has been received for a secure queue (**GWDCCi.SLi** set). Security restrictions: - HW: This register cannot be accessed by the unsecure APB interface. Error recovery: - HW: During reception, the current frame will be lost. Following data will be processed normally. - SW: Secure CPU should read the unsecure forwarding engine registers to find which entry is trying to use a secure queue for unsecure traffic and suppress it. - SW: Secure CPU could block access to switch from unsecure CPU.

(14) GWEIE4

GWCA Error Interrupt Enable 4.

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

Bits	Bit name	RW-P	Initial value	Function description
31:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	DSEIOE	R!=W-F	0H	Data Size Error Interrupt Overflow Interrupt 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 **GWEID4.DSEIOD** register will clear this bit.
16	DSEE	R!=W-F	0H	Data 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 **GWEID4.DSED** register will clear this bit.
15:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	DSSEIOE	R!=W-P	0H	DeScriptor Security Error Interrupt Overflow 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 **GWEID4.DSSEIOD** register will clear this bit. Security restrictions: - HW: This register cannot be accessed by the unsecure APB interface.

0	DSSEE	R!=W-P	0H	DeScriptor Security 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 **GWEID4.DSSED** register will clear this bit. Security restrictions: - HW: This register cannot be accessed by the unsecure APB interface.

(15) GWEID4

GWCA Error Interrupt Disable 4.

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

Bits	Bit name	RW-P	Initial value	Function description
31:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	DSEIOD	R0W-P	0H	Data Size Error Interrupt Overflow Disable Functions: - Writing 1 to this bit will clear **GWEIE4.DSEIOE** register.
16	DSED	R0W-P	0H	Data Size Error Disable Functions: - Writing 1 to this bit will clear **GWEIE4.DSEE** register.
15:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	DSSEIOD	R0W-F	0H	Descriptor Security Error Interrupt Overflow Disable Functions: - Writing 1 to this bit will clear **GWEIE4.DSSEIOE** register. Security restrictions: - HW: This register cannot be accessed by the unsecure APB interface.
0	DSSED	R0W-F	0H	Descriptor Security Error Disable Functions: - Writing 1 to this bit will clear **GWEIE4.DSSEE** register. Security restrictions: - HW: This register cannot be accessed by the unsecure APB interface.

(16) GWEIS5

GWCA Error Interrupt Status 5.

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

Bits	Bit name	RW-P	Initial value	Function description
31: AXI_CHAIN_W+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
AXI_CHAIN_W+23 :24	RXDNECN	R-F	0H	RX Descriptor Number Error Chain Number Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - This field is update to the descriptor chain number in which a RX Descriptor Number Error occurred when **GWEIS5.RXDNES** is being set.
23:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	RXDNEIOS	R!=W-F	0H	RX Descriptor Number Error Interrupt Overflow Status Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Set conditions: - RX Descriptor Number Error occurs and **GWEIS5.RXDNES** is already set to one. In this case, the descriptor chain number in which the new error occurred is lost. Error recovery: - SW: Switch reset.
16	RXDNES	R!=W-F	0H	RX Descriptor Number Error Status Set conditions: - HW: **GWMDNC.RXDMN**+1 descriptor has already been used to process one frame and the frame processing is not finished. 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: set of **DESCR.ERR** bit in RX descriptor if a part of the frame has been already sent. - HW: Current frame or remaining part of the frame will be discarded, and next frames will be processed normally. - SW: Discard data which have **DESCR.ERR** bit set. - SW: Software error, software should be reviewed.
15: AXI_CHAIN_W+8	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.

AXI_CHAIN_W+7:8	DCTECN	R-P	0H	Descriptor Chain Type Error Chain Number Clear conditions: - HW: Being in RESET mode will clear this register. Update conditions: - This field is update to the descriptor chain number in which a Descriptor Chain Type Error occurred when **GWEIS5.DCTES** is being set.
7:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	DCTEIOS	R!=W-P	0H	Descriptor Chain Type Error Interrupt Overflow Status Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Set conditions: - Descriptor Chain Type Error occurs and **GWEIS5.DCTES** is already set to one. In this case, the descriptor chain number in which the new error occurred is lost. Error recovery: - SW: Switch reset.
0	DCTES	R!=W-P	0H	Descriptor Chain Type Error Status Clear conditions: - HW: Being in RESET mode will clear this register. - SW: Writing 1 to this bit will clear it. Set conditions: - HW: When a descriptor is received for a transmission descriptor queue (**GWDCCt.DQTt** set). Error recovery: - HW: When a descriptor is received for a transmission descriptor queue, the corresponding frame is discarded. - SW: It is a software error. Software should be reviewed.

(17) GWEIE5

GWCA Error Interrupt Enable 5.

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

Bits	Bit name	RW-P	Initial value	Function description
31:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	RXDNEIOE	R!=W-F	0H	RX Descriptor Number Error Interrupt Overflow 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 **GWEID5.RXDNEIOD** register will clear this bit.
16	RXDNEE	R!=W-F	0H	RX Descriptor Number 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 **GWEID5.RXDNED** register will clear this bit.
15:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	DCTEIOE	R!=W-P	0H	Descriptor Chain Type Error Interrupt Overflow 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 **GWEID5.DCTEIOD** register will clear this bit.

Bits	Bit name	RW-P	Initial value	Function description
0	DCTEE	R!=W-P	0H	Descriptor Chain Type 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 **GWEID5.DCTED** register will clear this bit.

(18) GWEID5

GWCA Error Interrupt Disable 5.

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

Bits	Bit name	RW-P	Initial value	Function description
31:18	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
17	RXDNEIOD	R0W-F	0H	RX Descriptor Number Error Interrupt Overflow Disable Functions: - Writing 1 to this bit will clear **GWEIE5.RXDNEIOE** register.
16	RXDNED	R0W-F	0H	RX Descriptor Number Error Disable Functions: - Writing 1 to this bit will clear **GWEIE5.RXDNEE** register.
15:2	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
1	DCTEIOD	R0W-P	0H	Descriptor Chain Type Error Interrupt Overflow Disable Functions: - Writing 1 to this bit will clear **GWEIE5.DCTEIOE** register.
0	DCTED	R0W-P	0H	Descriptor Chain Type Error Disable Functions: - Writing 1 to this bit will clear **GWEIE5.DCTEE** register.

3.3.4 GWCA Security registers

(1) GWSCR0

GWCA Security Configuration Register 0.

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

Bits	Bit name	RW-P	Initial value	Function description
31: PTP_TN+28	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN+27:28	TRSL	RW-F	0H	Timer Register Security Level Values: - 1'b0 for bit q: Timer q registers can only be accessed by the APB secure interface. - 1'b1 for bit q: Timer q registers can be accessed by both APBs. Timer q registers include the following registers: - **GWTSDCCq** - **GWAVTPTM0/1q** - **GWGPTPTM0/1/2q**
27: PTP_TN+24	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
PTP_TN+23:24	TSQRSL	RW-F	0H	TimeStamp Queue Register Security Level Values: - 1'b0 for bit q: Timestamp queue q registers can only be accessed by the APB secure interface. - 1'b1 for bit q: Timestamp queue q registers can be accessed by both APBs. Timestamp queue q registers include the following registers: - **GWTDCAC0q** - **GWTDCAC1q** - **GWTSDIS.TSDIS[q]** - **GWTSDIE.TSDIE[q]** - **GWTSDID.TSDID[q]** - **GWEIS0.TDFESq** - **GWEIE0.TDFEEq** - **GWEID0.TDFEDq**

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: - **GWRDQC.RDQD[q]** - **GWRDQC.RDQP[q]** - **GWRMFSCq** - **GWRDQDCq** - **GWRDQMq** - **GWRDQMLq** - **GWEIS0.FSES[q]** - **GWEIE0.FSEE[q]** - **GWEID0.FSED[q]** - **GWEIS1.DQOES[q]** - **GWEIE1.DQOEE[q]** - **GWEID1.DQOED[q]** - **GWEIS1.DIECCES[q]** - **GWEIE1.DIECCEE[q]** - **GWEID1.DIECCED[q]**
FRM_PRIO_N+7:8	APRSL	RW-F	0H	AXI Priority Register Security Level Values: - 1'b0 for bit q: AXI Priority q registers can only be accessed by the APB secure interface. - 1'b1 for bit q: AXI Priority q registers can be accessed by both APBs. AXI registers include the following registers: - **GWTPCp.PPPLp[q]** - **GWDCCi.DCPi (Writable value q)** [Example] if **APRSL[7]** == 1'b1, unsecure APB can write **7** to GWDCCi.DCPi.

7	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: - **GWEIS0.AES** - **GWEIS0.DECCES** - **GWEIS0.TECCES** - **GWEIS0.PECCES** - **GWEIS0.DSECCES** - **GWEIS0.MECCES** - **GWEIS0.AECCES** - **GWEIS0.TSECCES** - **GWEIS0.L23UECCES** - **GWEIS0.TSOVFES** - **GWEIS0.USMFSES** - **GWEIS0.TFES** - **GWEIS0.CKSES** - **GWEIS0.SEQES** - **GWEIS0.IIPES** - **GWEIS0.TSHES** - **GWEIE0.AEE** - **GWEIE0.DECCEE** - **GWEIE0.TECCEE** - **GWEIE0.PECCEE** - **GWEIE0.DSECCEE** - **GWEIE0.MECCEE** - **GWEIE0.AECCEE** - **GWEIE0.TSECCEE** - **GWEIE0.L23UECCEE** - **GWEIE0.TSOVFEE** - **GWEIE0.USMFSEE** - **GWEIE0.TFEE** - **GWEIE0.CKSEE** - **GWEIE0.SEQEE** - **GWEIE0.IIPEE** - **GWEIE0.TSHEE** - **GWEID0.AED** - **GWEID0.DECCED** - **GWEID0.TECCED** - **GWEID0.PECCED** - **GWEID0.DSECCED** - **GWEID0.MECCED** - **GWEID0.AECCED** - **GWEID0.TSECCED** - **GWEID0.L23UECCED** - **GWEID0.TSOVFED** - **GWEID0.USMFSED** - **GWEID0.TFED** - **GWEID0.CKSED** - **GWEID0.SEQED**

				- **GWEID0.IIPED** - **GWEID0.TSHED** - **GWEIS4.DSES** - **GWEIS4.DSEIOS** - **GWEIS4.DSECN** - **GWEIE4.DSEE** - **GWEIE4.DSEIOE** - **GWEID4.DSED** - **GWEID4.DSEIOD** - **GWEIS5.DCTES** - **GWEIS5.DCTEIOS** - **GWEIS5.DCTECN** - **GWEIE5.DCTEE** - **GWEIE5.DCTEIOE** - **GWEID5.DCTED** - **GWEID5.DCTEIOD**
6	AXRSL	RW-F	0H	AXI Register Security Level Values: - 1’b0: AXI registers can only be accessed by the APB secure interface. - 1’b1: AXI registers can be accessed by both APBs. AXI registers include the following registers: - **GWDCBAC0** - **GWDCBAC1** - **GWMDNC** - **GWARIRM** - **GWGRLC** - **GWGRLULC** - **GWIDPC**
5	TSRSL	RW-F	0H	TimeStamp Register Security Level Values: - 1’b0: Timestamp registers can only be accessed by the APB secure interface. - 1’b1: Timestamp registers can be accessed by both APBs. Timestamp registers include the following registers: - **GWTSNM** - **GWTSMNM**

4	TGRSL	RW-F	0H	TAG Register Security Level Values: - 1'b0: TAG (also checksum) registers can only be accessed by the APB secure interface. - 1'b1: TAG (also checksum) registers can be accessed by both APBs. TAG registers include the following registers: - **GWVCC** - **GWVTC** - **GWTTFC** - **GWCKSC** - **GWICD0RC** - **GWICD1RC** - **GWISD0RC** - **GWISD1RC** - **GWECD0RC** - **GWECD1RC** - **GWESD0RC** - **GWESD1RC**
3	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: - **GWMAC0** - **GWMAC1**
2	MTRSL	RW-F	0H	Multicast and DCNC Table Register Security Level Values: - 1'b0: Multicast and DCNC Table registers can only be accessed by the APB secure interface. - 1'b1: Multicast and DCNC Table registers can be accessed by both APBs. Multicast and DCNC Table registers include the following registers: - **GWMTIRM**
1	RRSL	RW-F	0H	Reception Register Security Level Values: - 1'b0: Reception registers can only be accessed by the APB secure interface. - 1'b1: Reception registers can be accessed by both APBs. Reception registers include the following registers: - **GWRDRC** - **GWIRC** - **GWRDQSC** - **GWRDQAC** - **GWRGC** - **GWCSDRC**
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: - **GWMC** - **GWMS**

(2) GWSCR1

GWCA Security Configuration Register 1.

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															CRSL

Bits	Bit name	RW-P	Initial value	Function description
31:1	RSV	R0-U	0H	Reserved area. On read, 0 will be returned.
0	CRSL	RW-F	0H	Counter Register Security Level Values: - 1'b0 for bit q: Timer q registers can only be accessed by the APB secure interface. - 1'b1 for bit q: Timer q registers can be accessed by both APBs. Timer q registers include the following registers: - **GWRDCN** - **GWTDCN** - **GWTSCN** - **GWTSOVFECN** - **GWUSMFSECN** - **GWTFECN** - **GWSEQECN** - **GWIIPECN** - **GWTXDNECN** - **GWFSECN** - **GWTDFECN** - **GWTSDNECN** - **GWDQOECN** - **GWDQSECN** - **GWDFECN** - **GWDSECN** - **GWDSZECN** - **GWDCTECN** - **GWRXDNECN** - **GWCKSECN** - **GWLDCN**

(3) GWSCR2i (i = 0 to AXI_CHAIN_N/32-1)

GWCA Security Configuration Register 2 i.

B31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACRSL {i*32+31}	ACRSL {i*32+30}	ACRSL {i*32+29}	ACRSL {i*32+28}	ACRSL {i*32+27}	ACRSL {i*32+26}	ACRSL {i*32+25}	ACRSL {i*32+24}	ACRSL {i*32+23}	ACRSL {i*32+22}	ACRSL {i*32+21}	ACRSL {i*32+20}	ACRSL {i*32+19}	ACRSL {i*32+18}	ACRSL {i*32+17}	ACRSL {i*32+16}
B15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACRSL {i*32+15}	ACRSL {i*32+14}	ACRSL {i*32+13}	ACRSL {i*32+12}	ACRSL {i*32+11}	ACRSL {i*32+10}	ACRSL {i*32+9}	ACRSL {i*32+8}	ACRSL {i*32+7}	ACRSL {i*32+6}	ACRSL {i*32+5}	ACRSL {i*32+4}	ACRSL {i*32+3}	ACRSL {i*32+2}	ACRSL {i*32+1}	ACRSL {i*32}

Bits	Bit name	RW-P	Initial value	Function description
b	ACRSLt t=32*i+b	RW-F	0H	AXI Chain Register Security Level t Values: - 1’b0 for bit t: AXI chain t registers can only be accessed by the APB secure interface. - 1’b1 for bit t: AXI chain t registers can be accessed by both APBs. Descriptor queue q registers include the following registers: - **GWTRCi.TSRt** - **GWDCCt** - **GWIDAUASt** - **GWIDASMt** - **GWIDASAM0t** - **GWIDASAM1t** - **GWIDACAM0t** - **GWIDACAM1t** - **GWRLCt** - **GWRLULCt** - **GWIDCt** - **GWDISi.DISt** - **GWDIEi.DIEt** - **GWDIDi.DIDt** - **GWDIDSi.DIDSt** - **GWEIS2i.DFESt** - **GWEIE2i.DFEEt** - **GWEID2i.DFEDt** - **GWEIS3.IAOES[t]** (Only for t = 0..AXI_RINC_N-1) - **GWEIE3.IAOEE[t]** (Only for t = 0..AXI_RINC_N-1) - **GWEID3.IAOED[t]** (Only for t = 0..AXI_RINC_N-1)

4. Register utilization

4.1 Operation Modes

Table 4-1 describes GWCA operation modes.

Table 4-1: GWCA Operation Modes

Operation mode	GWMS.OPS value	Description
DISABLE	2'd1	- No transaction is ongoing. But there can be tolerance (left behind) of transactions* from GWCA when the agent has moved from OPERATION to DISABLE. (* : max AXI_RD_OUT_N read transactions or max AXI_WR_OUT_N write transactions) - 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 GWCA 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 **GWMC.OPC**. In that case, mode transition will be confirmed by **GWMS.OPS**.

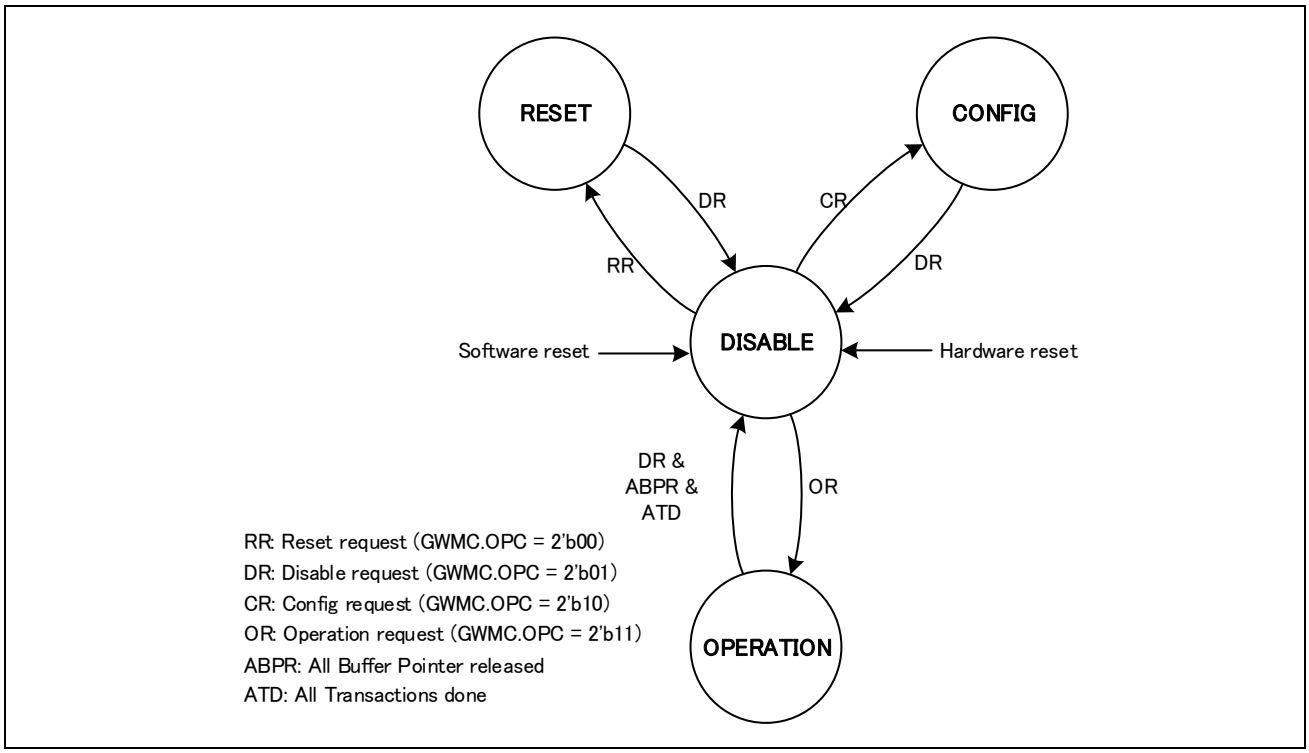


Fig 4.1: Operating mode transitions

Complementary information on transition conditions

ABPR:

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

ATD:

- All the descriptors already received are processed and corresponding frames are sent to CPU and AXI descriptors are written back.

All ongoing transmit frames are fully sent to the forwarding engine [FWD] and AXI descriptors are written back.

[Restrictions]

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

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.

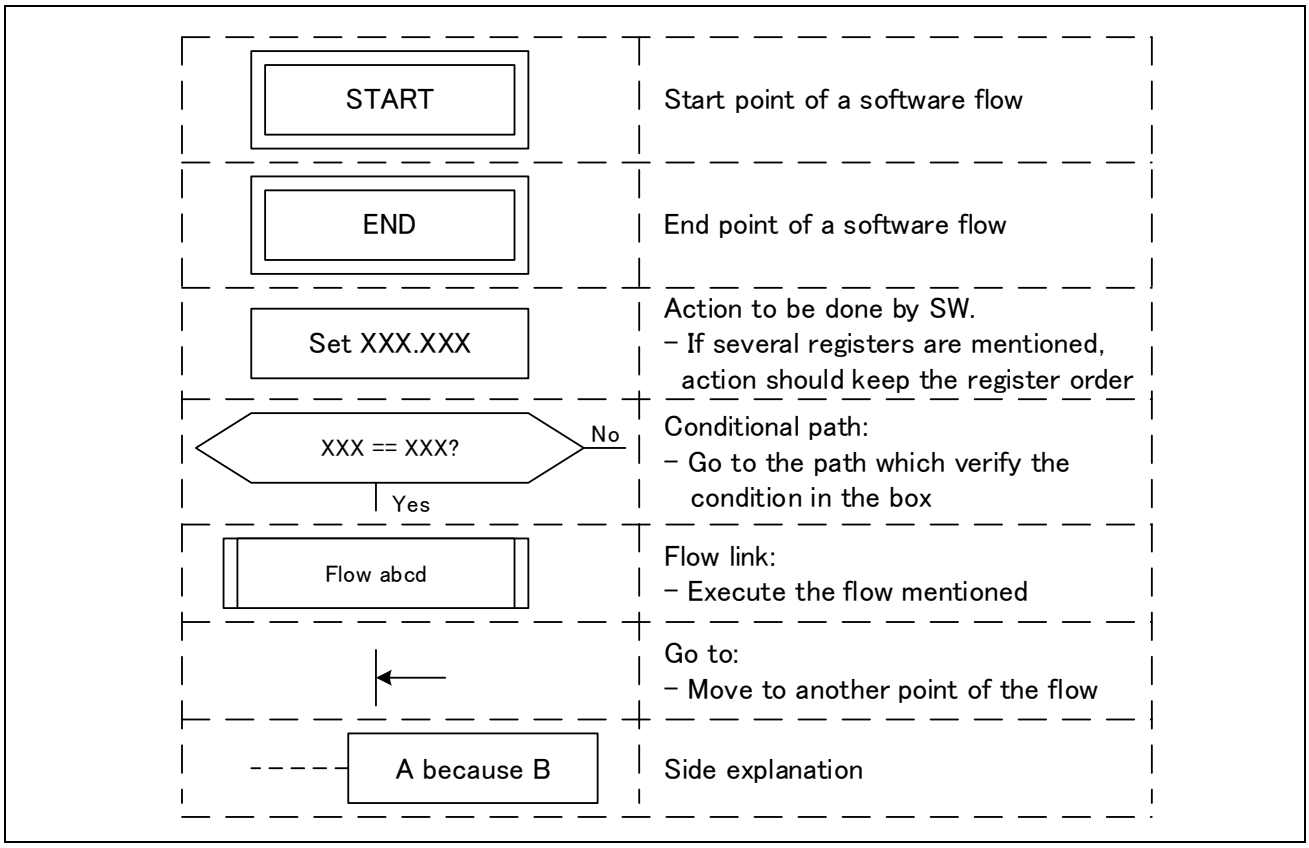


Fig 4.2: Software flow legend

4.2.2 Mode transition flow

The mode transition flow is described in Fig 4.3.

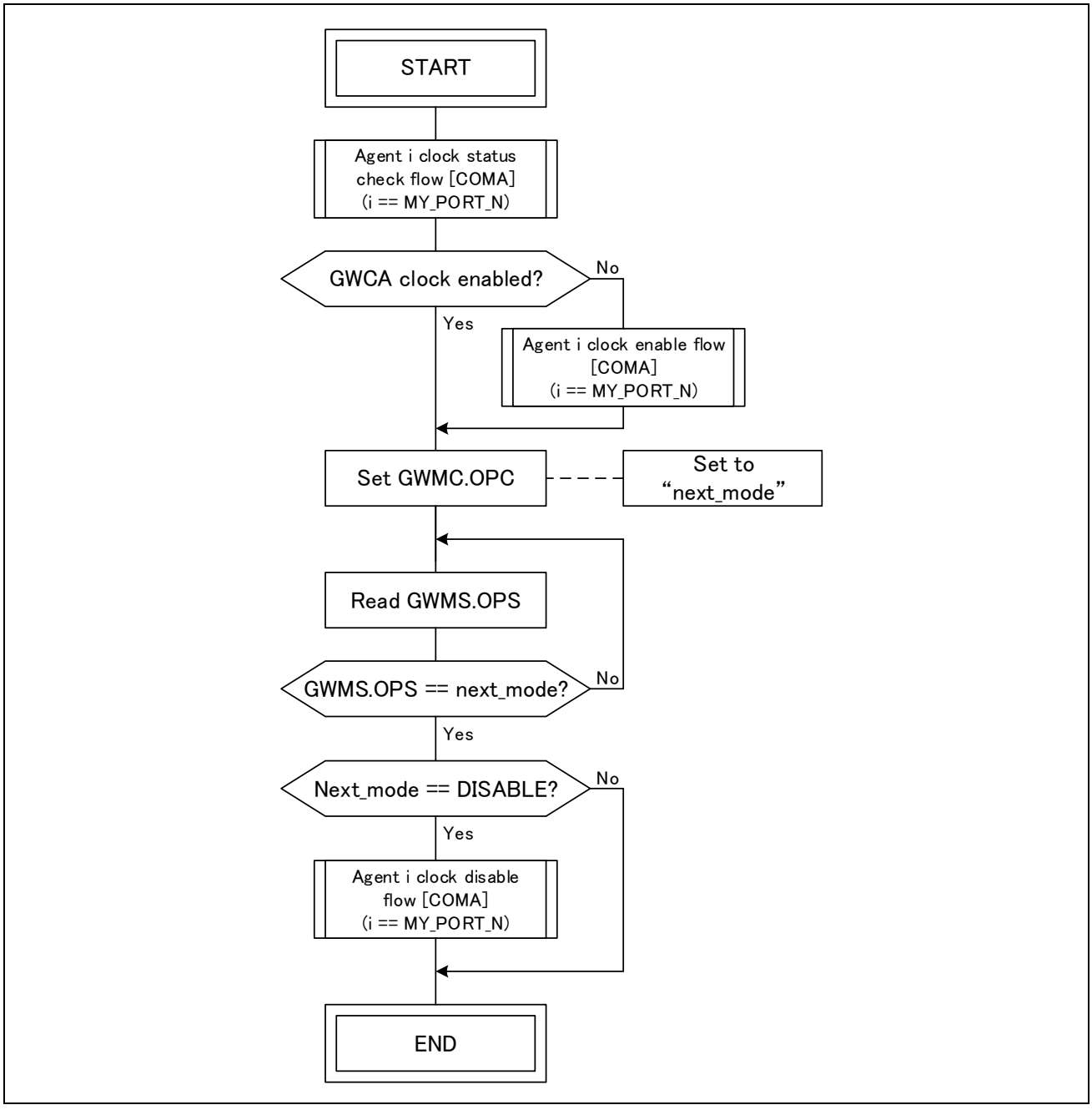


Fig 4.3: Mode transition flow

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.

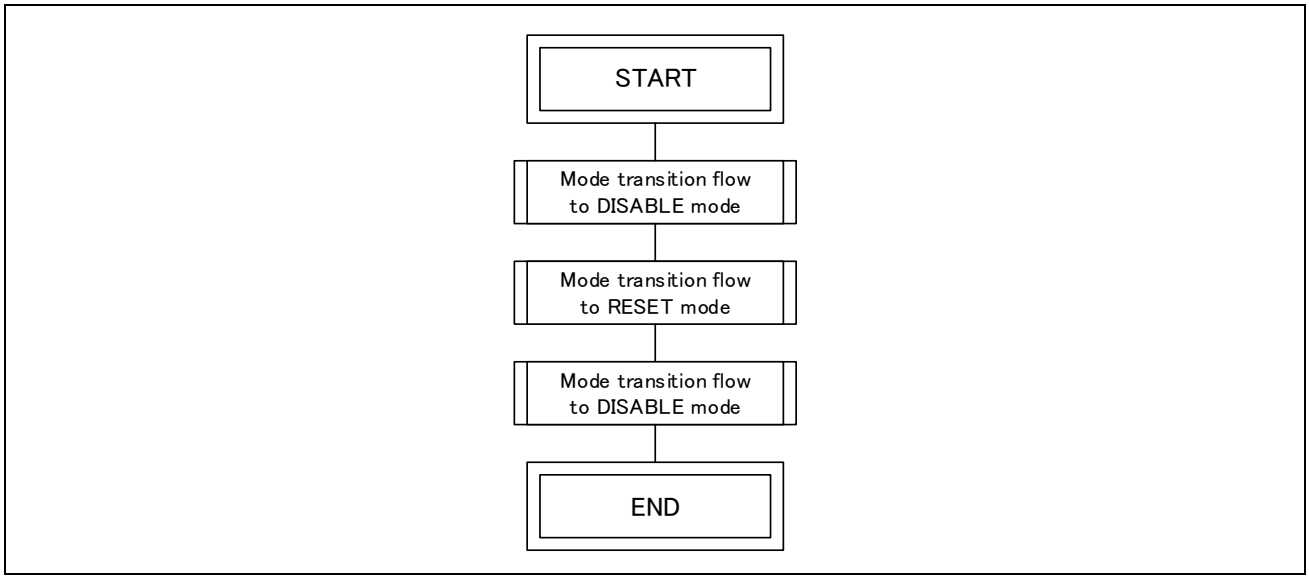


Fig 4.4: Reset flow

4.2.4 Initialization flow

The initialization flow is described in Fig 4.5.

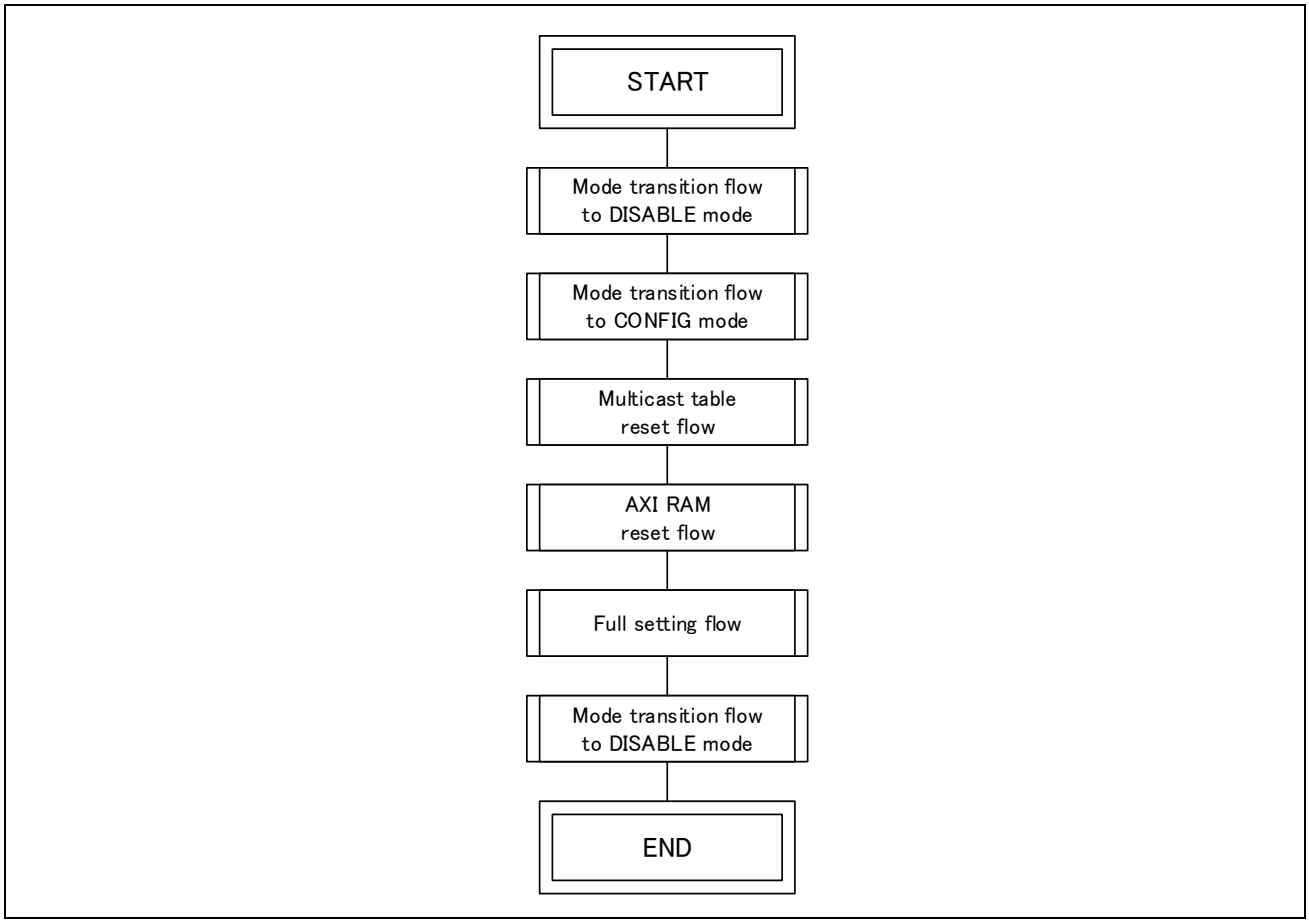


Fig 4.5: Initialization flow

4.2.5 Reinitialization flow

The reinitialization flow is described in Fig 4.6.

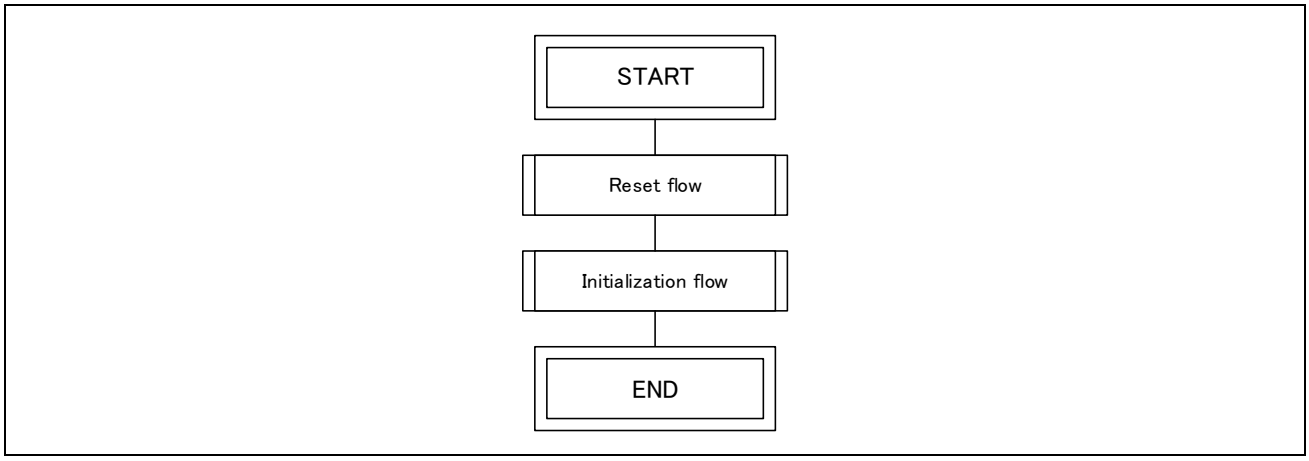


Fig 4.6: Reinitialization flow

4.2.6 Multicast and DCNC table reset flow

The multicast and DCNC table reset flow is described in Fig 4.7.

Restrictions:

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

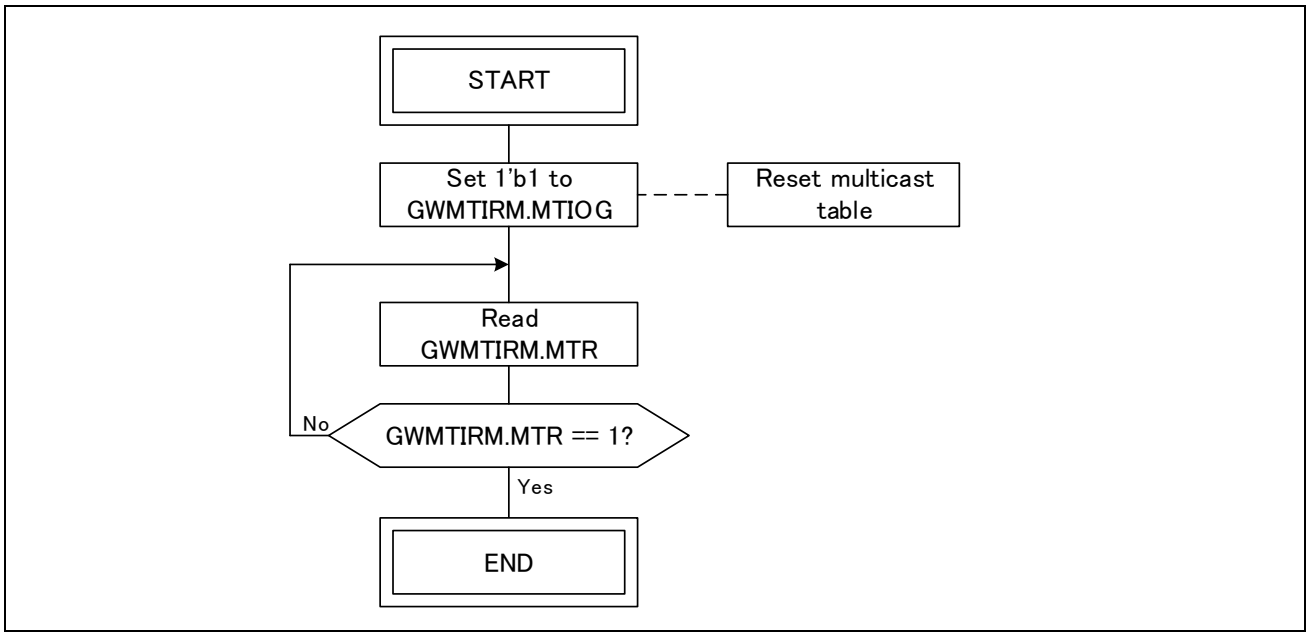


Fig 4.7: Multicast and DCNC table reset flow

4.2.7 Multicast and DCNC table setting flow

The multicast and DCNC table setting flow is described in Fig 4.8.

Restrictions:

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

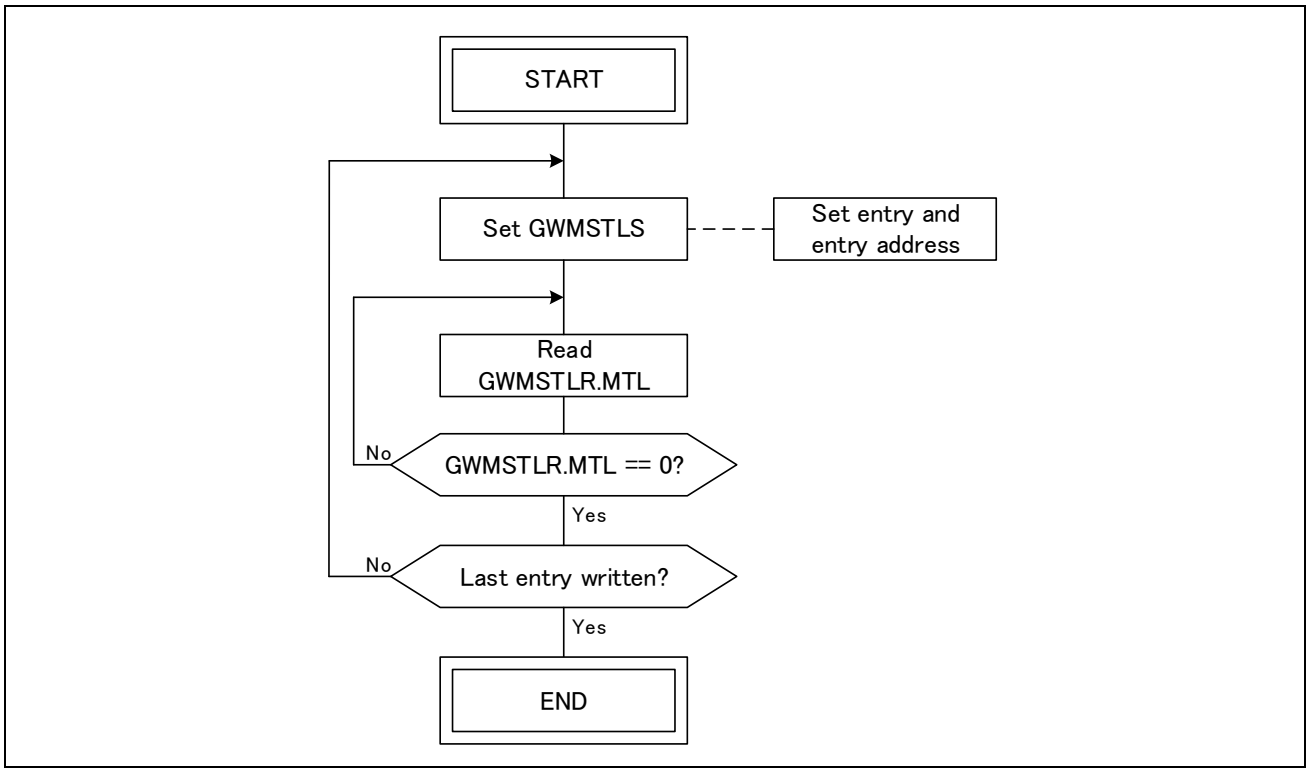


Fig 4.8: Multicast and DCNC table setting flow

4.2.8 Multicast and DCNC table read flow

The multicast and DCNC table read flow is described in Fig 4.8.

Restrictions:

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

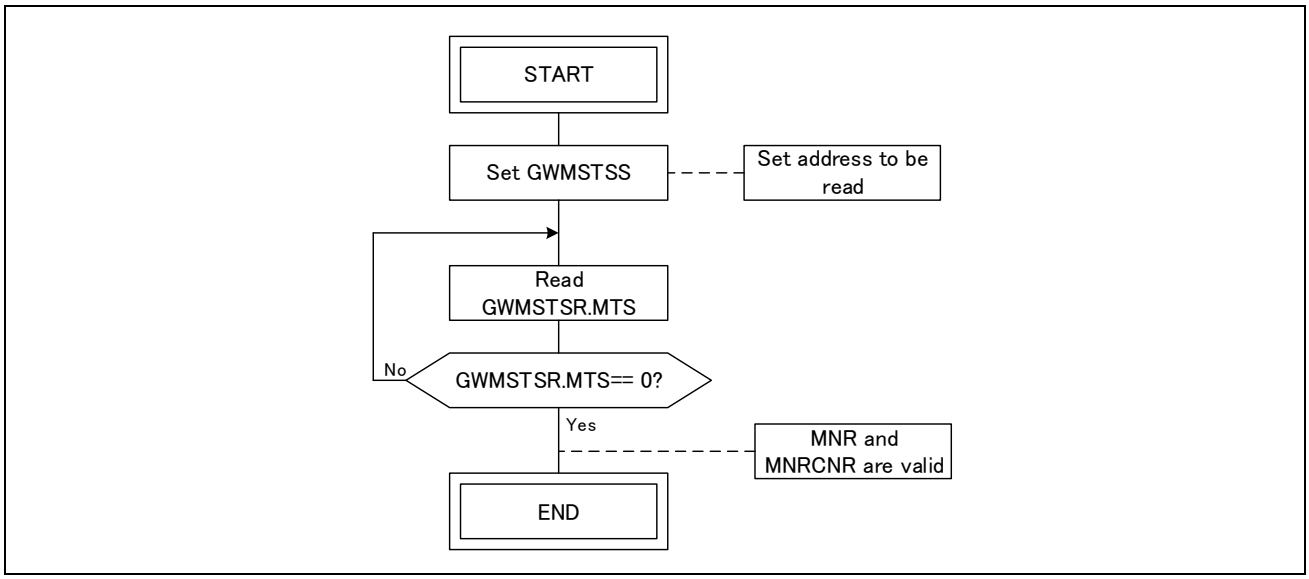


Fig 4.9: Multicast and DCNC table read flow

4.2.9 AXI RAM reset flow

The AXI RAM reset flow is described in Fig 4.10.

Restrictions:

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

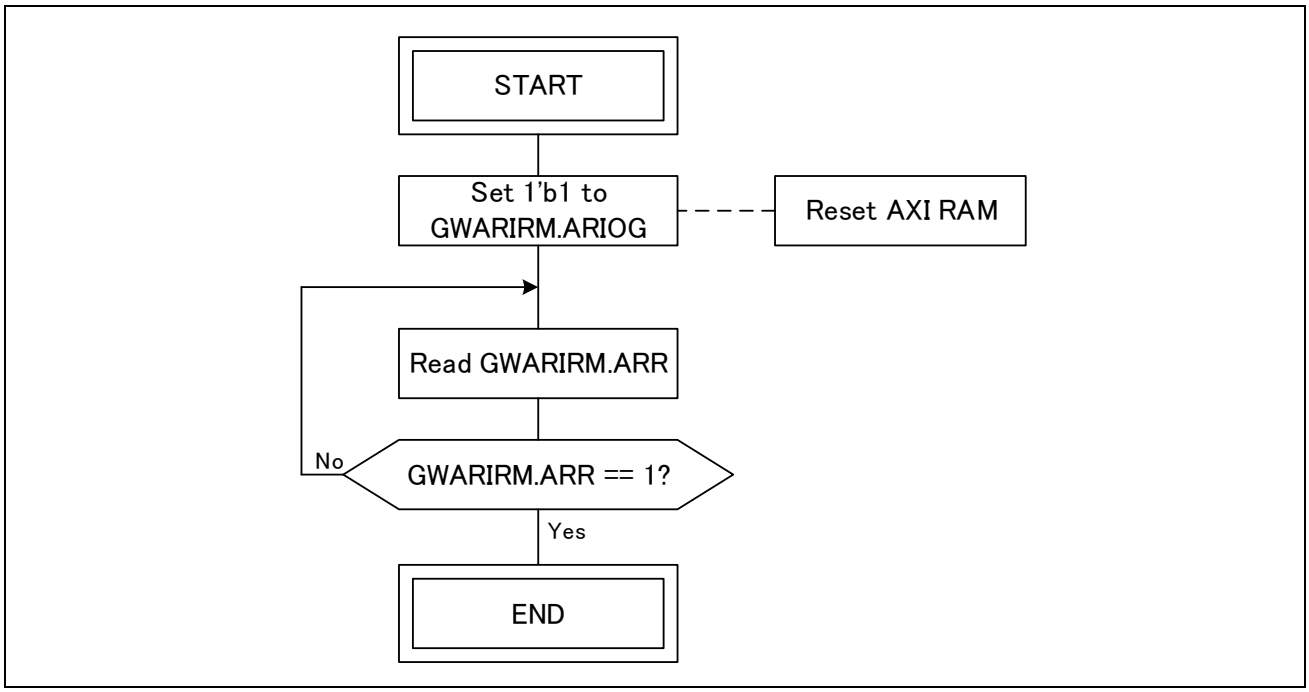


Fig 4.10: AXI RAM reset flow

4.2.10 AXI RAM read flow

The AXI RAM read flow is described in Fig 4.11.

Restrictions:

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

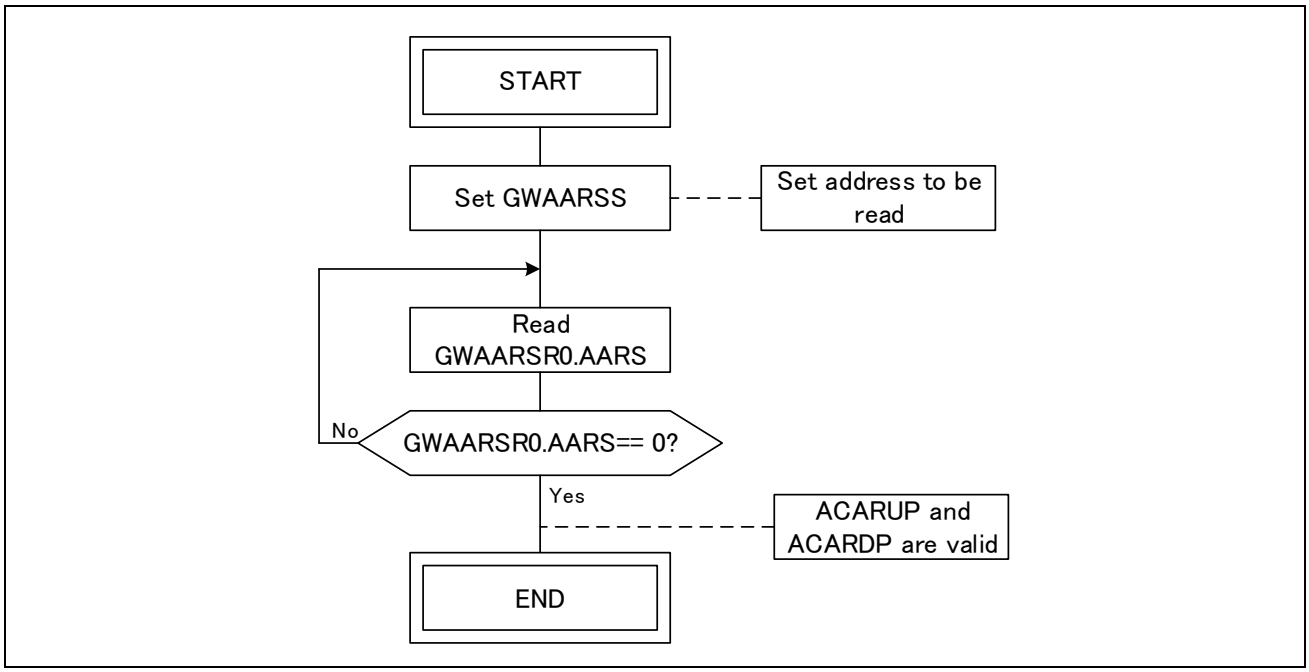


Fig 4.11: AXI RAM read flow

4.2.11 Global rate limiter setting flow

The global rate limiter setting flow is described in Fig 4.12.

Restrictions:

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

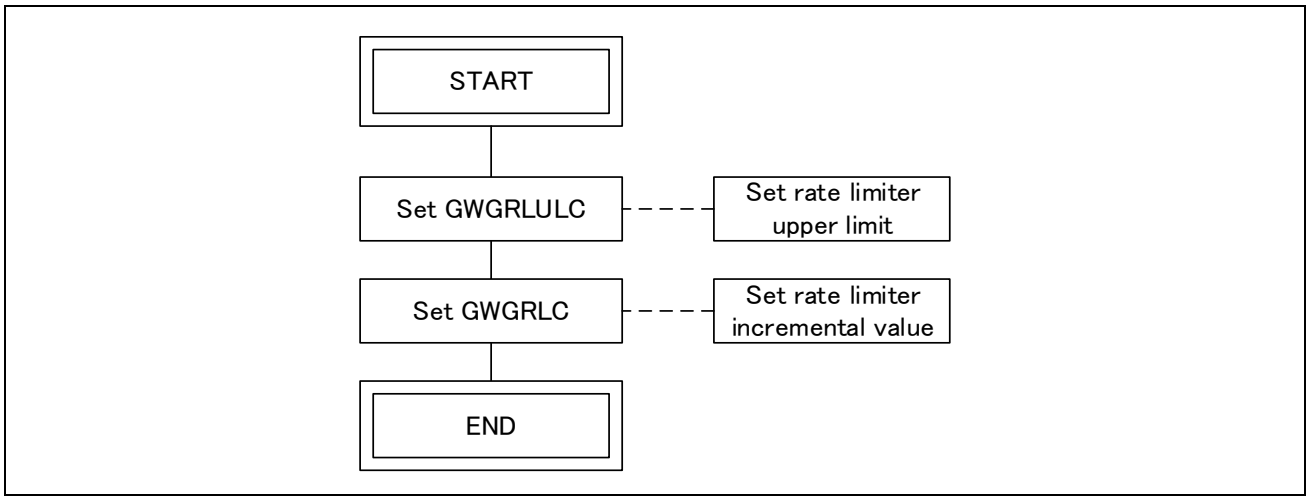


Fig 4.12: Global Rate limiter setting flow

4.2.12 Global rate limiter disabling flow

The global rate limiter disabling flow is described in Fig 4.13.

Restrictions:

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

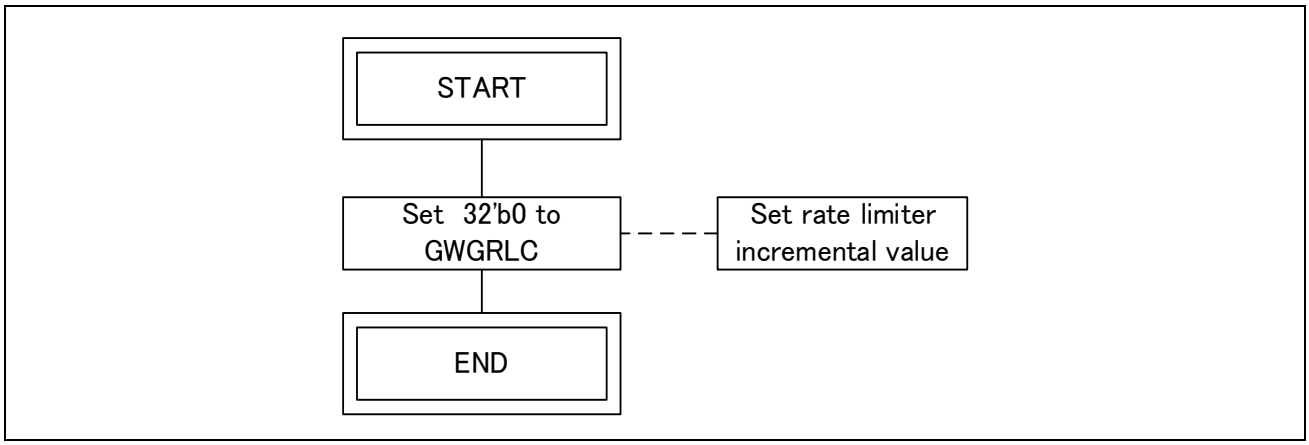


Fig 4.13: Global Rate limiter disabling flow

4.2.13 Rate limiter i setting flow

The rate limiter i setting flow is described in Fig 4.14.

Restrictions:

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

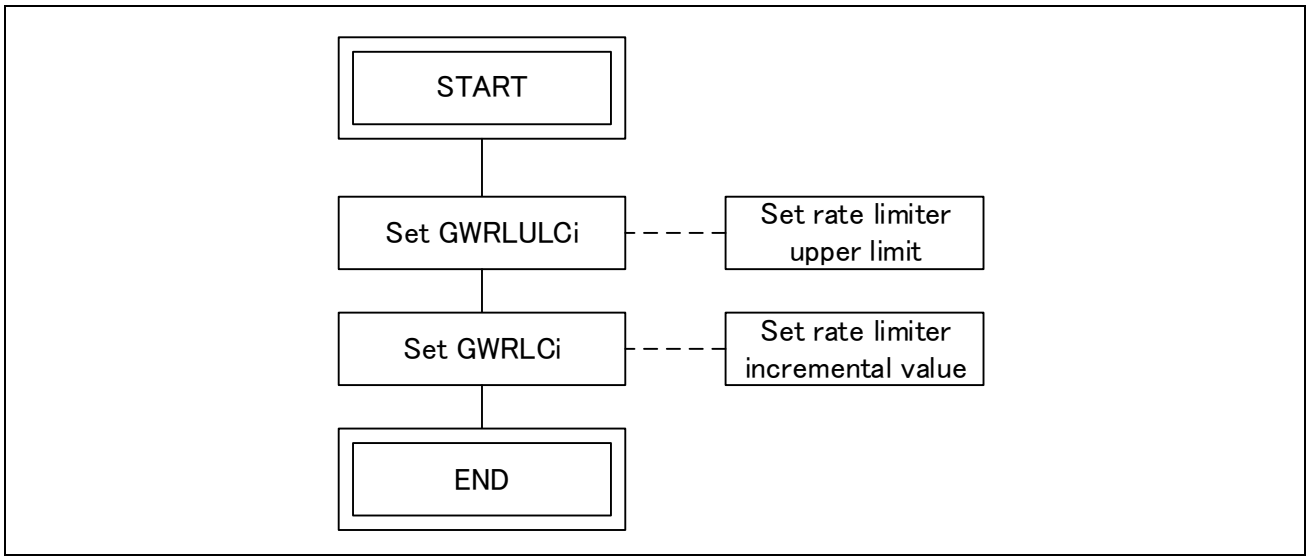


Fig 4.14: Rate limiter i setting flow

4.2.14 Rate limiter i disabling flow

The rate limiter i disabling flow is described in Fig 4.15.

Restrictions:

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

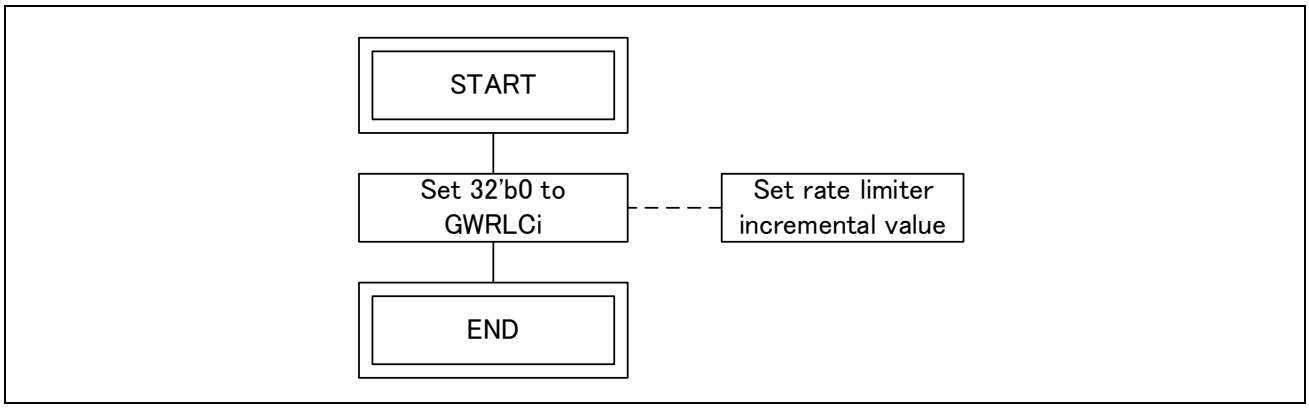


Fig 4.15: Rate limiter i disabling flow

4.2.15 AXI table read flow

The AXI table read flow is described in Fig 4.16.

Restrictions:

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

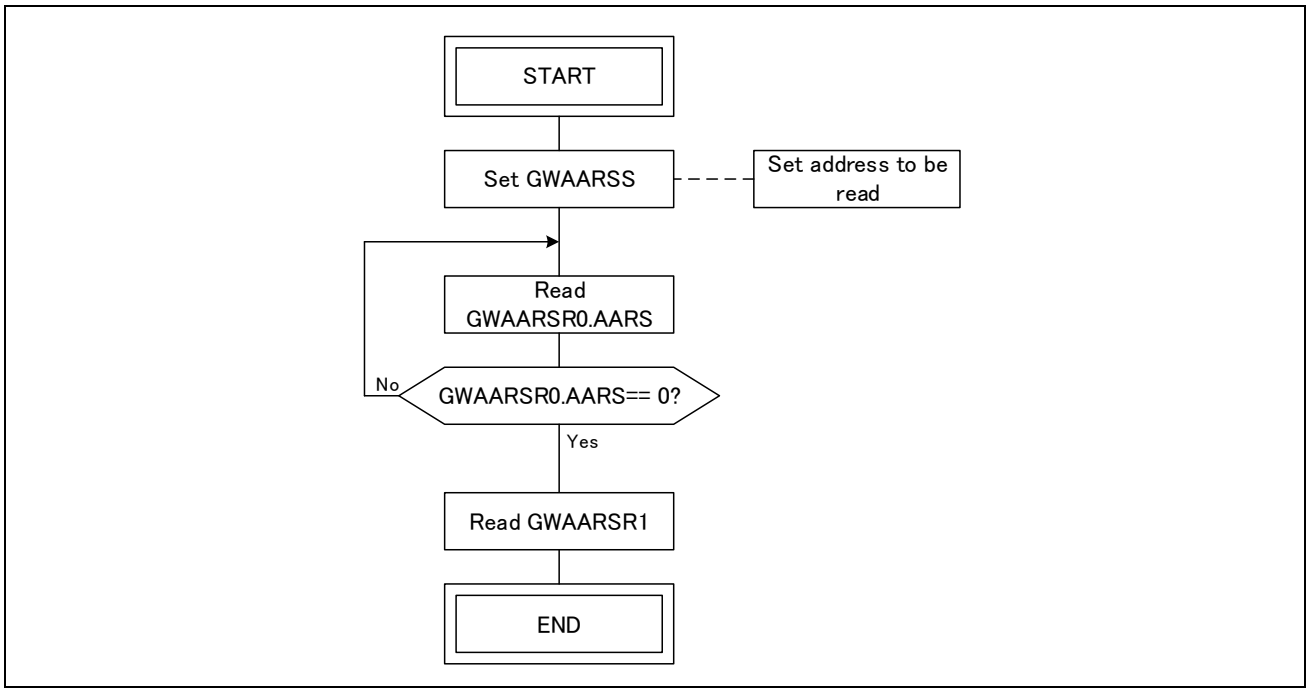


Fig 4.16: AXI table read flow

4.2.16 Interrupt handling flow

The interrupt handling flow is described in Fig 4.17.

Restrictions:

- This flow is not usable in RESET mode.

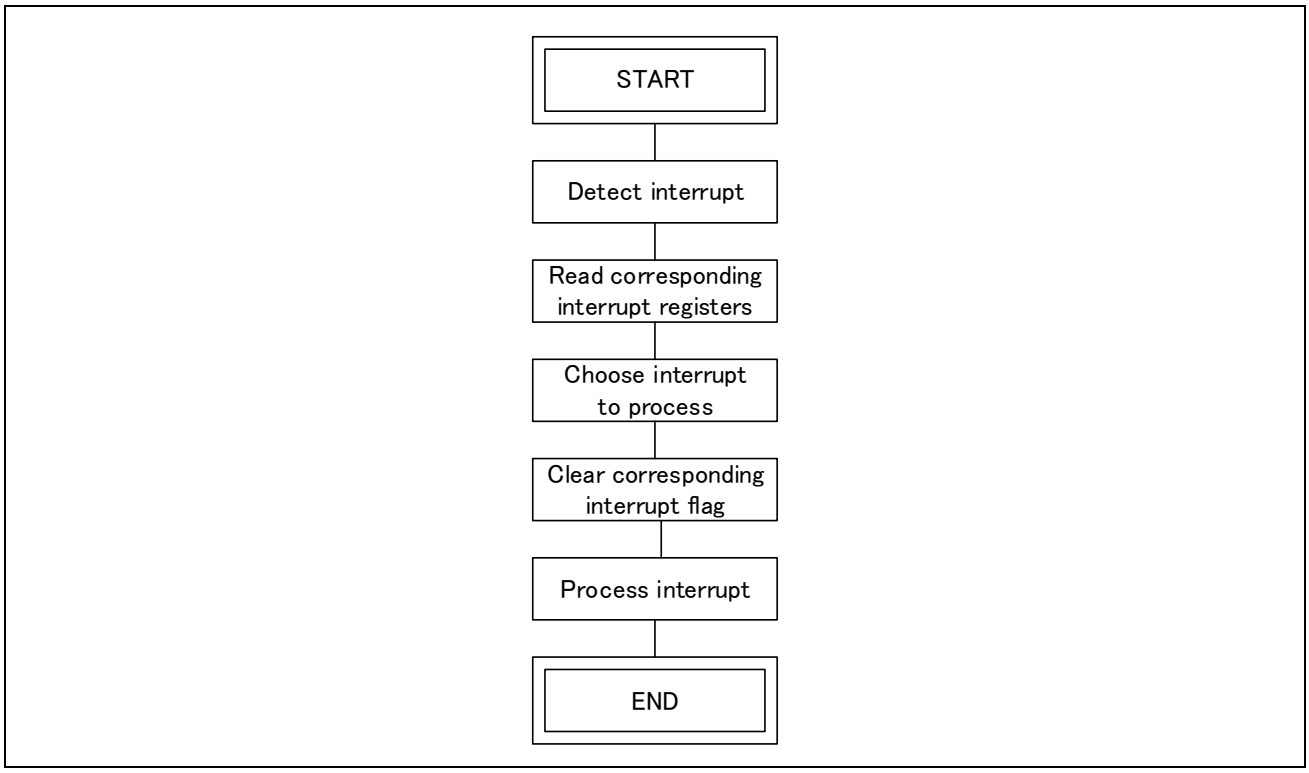


Fig 4.17: Interrupt handling flow

4.2.17 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.

4.2.17.1 Full setting flow

The full setting flow is described in Fig 4.18.
For LINKFIX table setting refer to section 5.1.3.1.

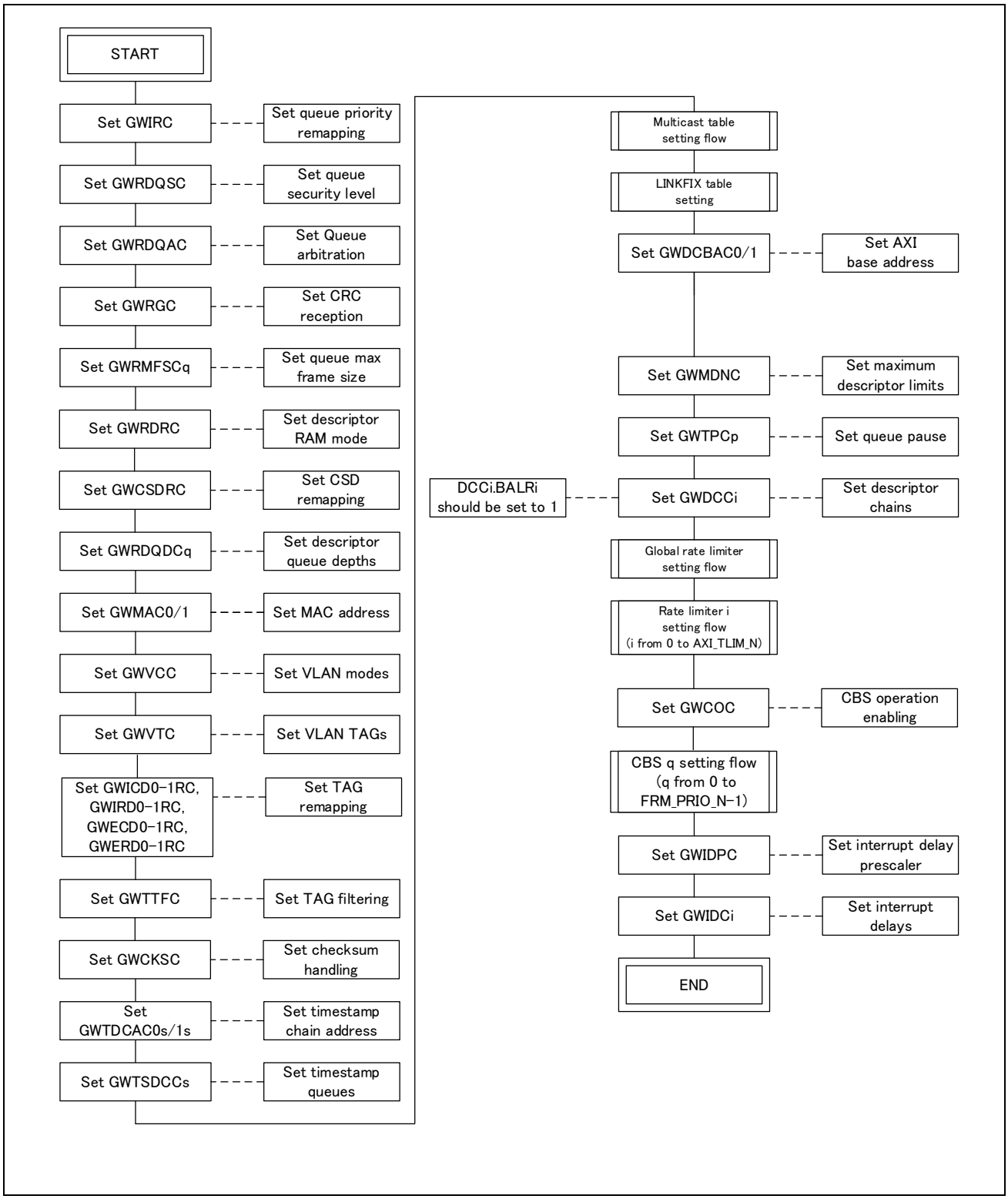


Fig 4.18: Full setting flow

4.2.17.2 AXI emergency stop flow

The AXI emergency stop flow is described in Fig 4.19.

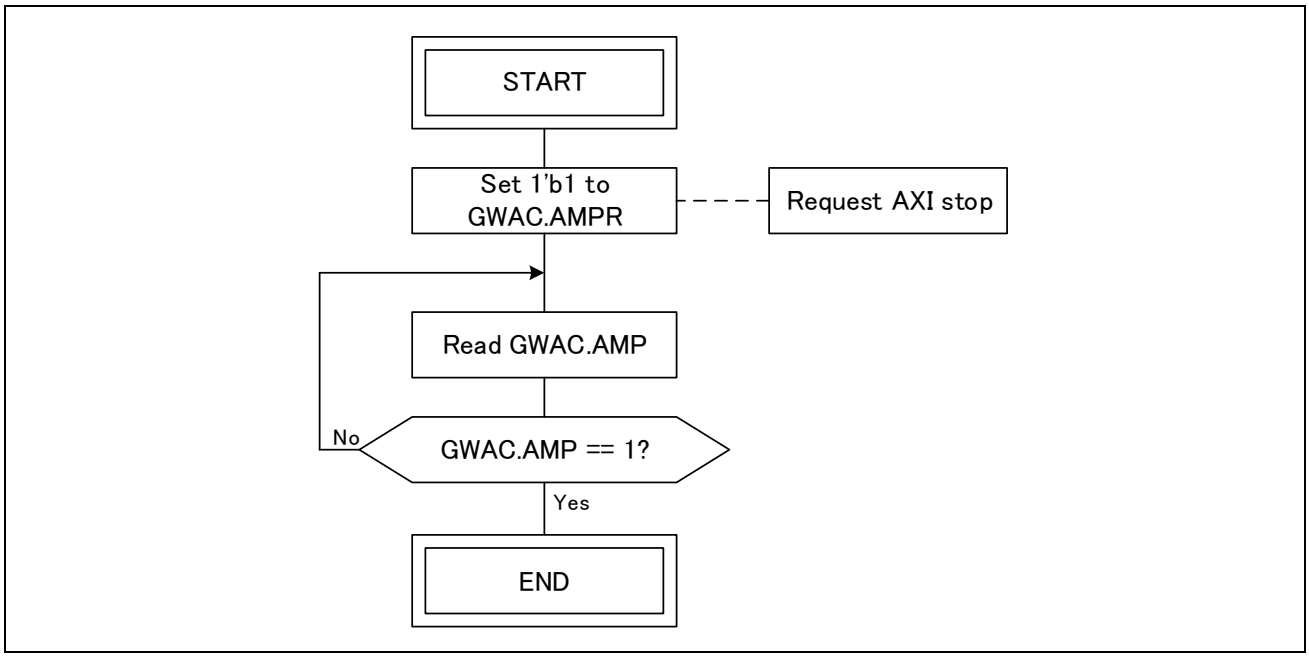


Fig 4.19: AXI emergency stop flow

4.2.18 Register writable without software flow

These registers can be changed dynamically. (However, it is necessary that the initial settings such as the clock enabling have been completed.)

- GWMAC0 and GWMAC1 (if in CONFIG)
- Other (not been described so far)

[Notes]

When a frame has reached GWCA and before the frame has been written into URAM by AXI BMI if GWDCC configurations of target chain is reconfigured by the user then the frame will be handled using the GWDCC register settings before reconfiguration and not using the reconfigured GWDCC settings.

5. Functional details

5.1 AXI master general information

In this section is described the knowledge required to utilize the GWCA AXI master. It contains the AXI address table handling, the AXI descriptor general description, and the AXI IDs utilization description.

5.1.1 AXI descriptor queue mapping

The AXI descriptor queue mapping is described along with its mapping to the **INTER_TX** interface, the incremental queues and the rate limiters in Fig 5.1. The global rate limiter is not represented because it limits throughput for all the TX queues.

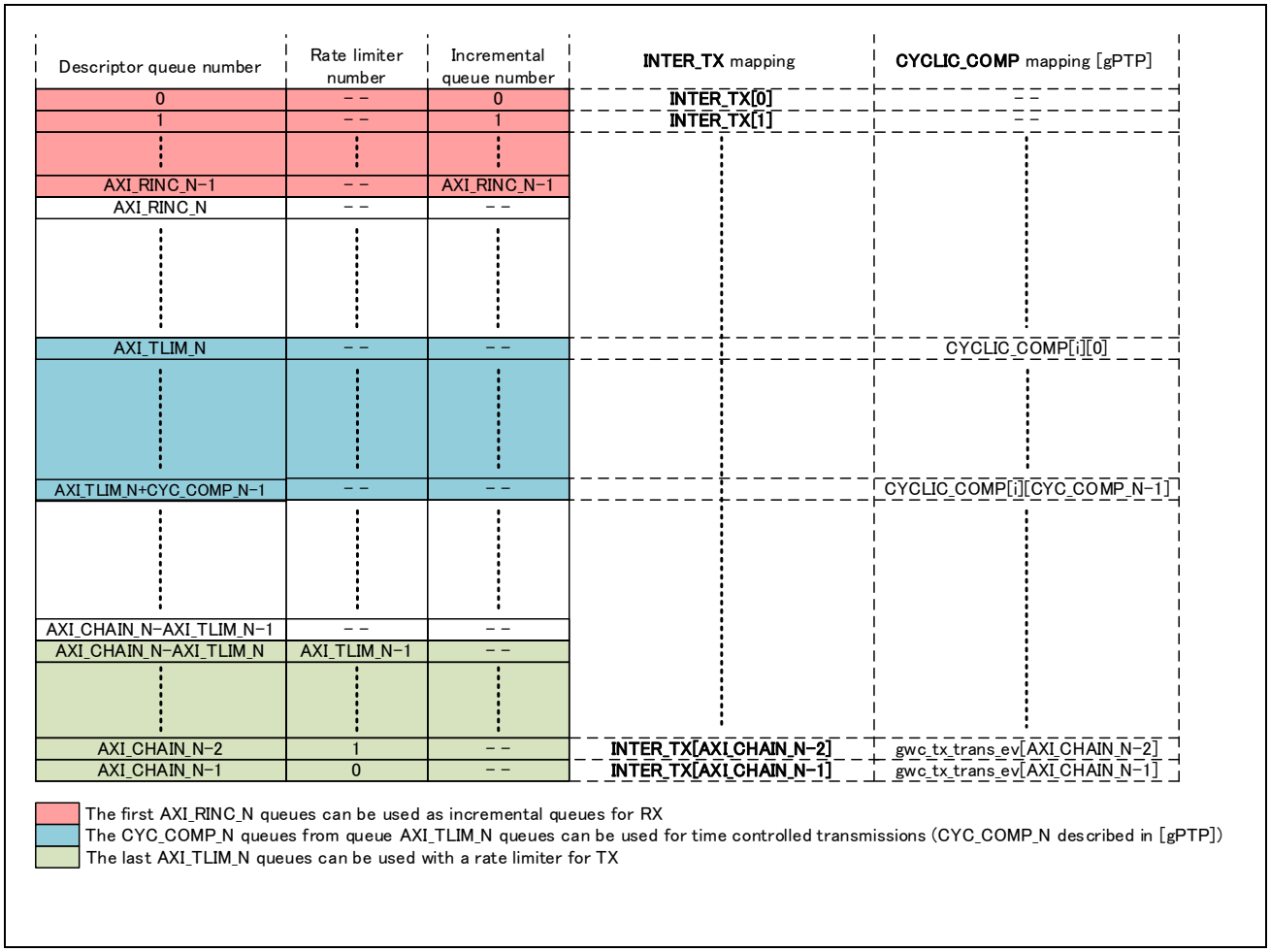


Fig 5.1: AXI descriptor queue mapping

5.1.2 AXI descriptor general description

5.1.2.1 General formats

AXI descriptor general formats are described in Fig 5.2, Fig 5.3, Fig 5.4 and Fig 5.5. The fields are explained in Table 5-1. The fields utilization will be explained per transaction type (Data reception, Data transmission and Timestamp reception).

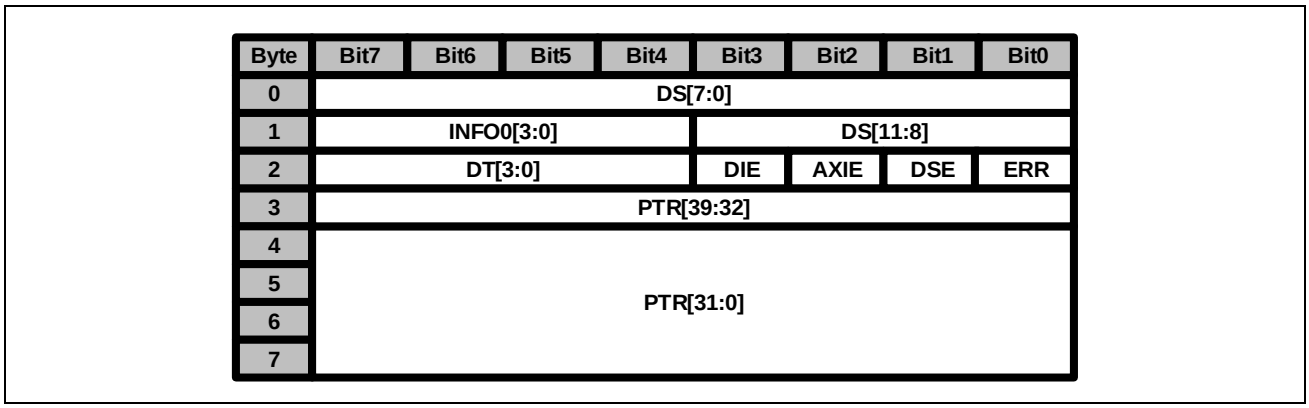


Fig 5.2: Basic descriptor (**GWDCCi.ETSi** == 0 and **GWDCCi.EDEi** == 0)

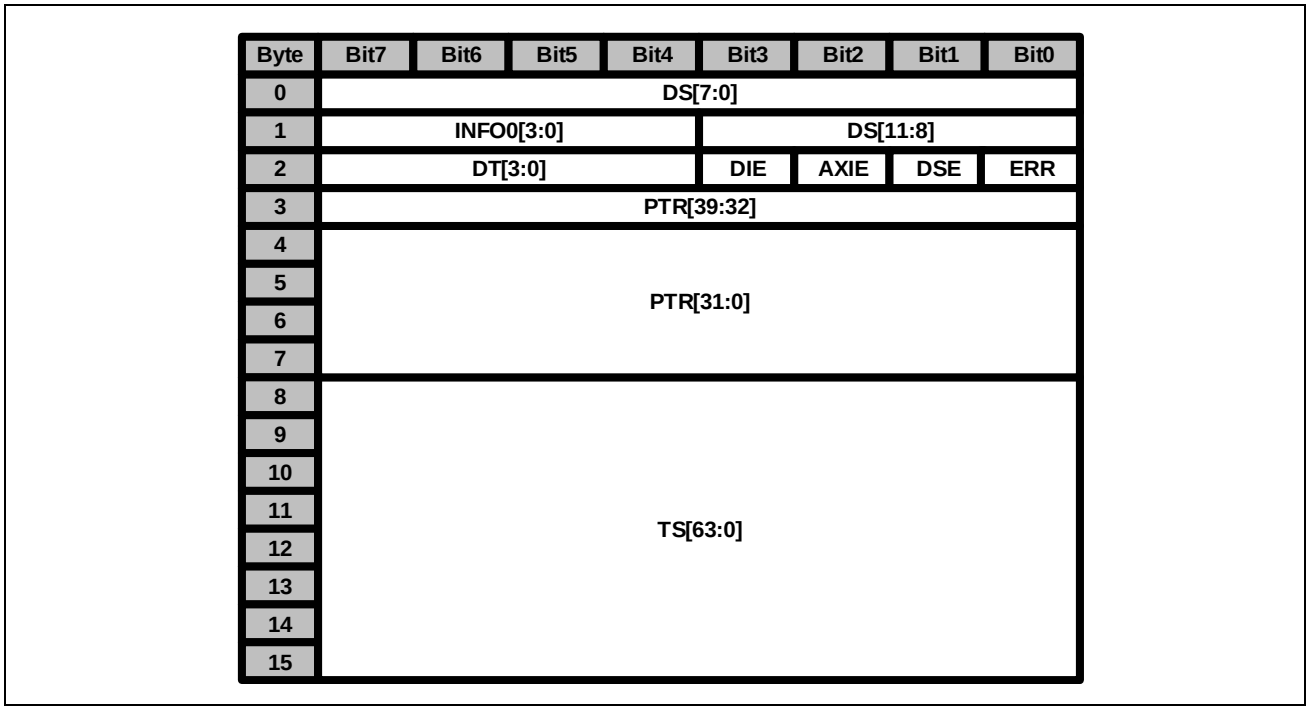


Fig 5.3: Timestamp descriptor (**GWDCCi.ETSi** == 1 and **GWDCCi.EDEi** == 0)

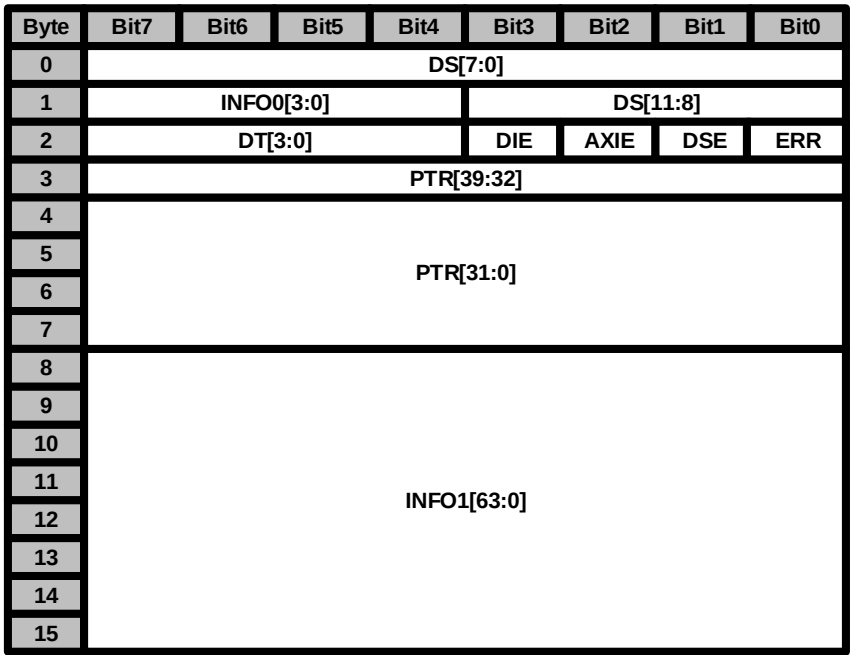


Fig 5.4: Extended descriptor (**GWDCCi.ETSi** == 0 and **GWDCCi.EDEi** == 1)

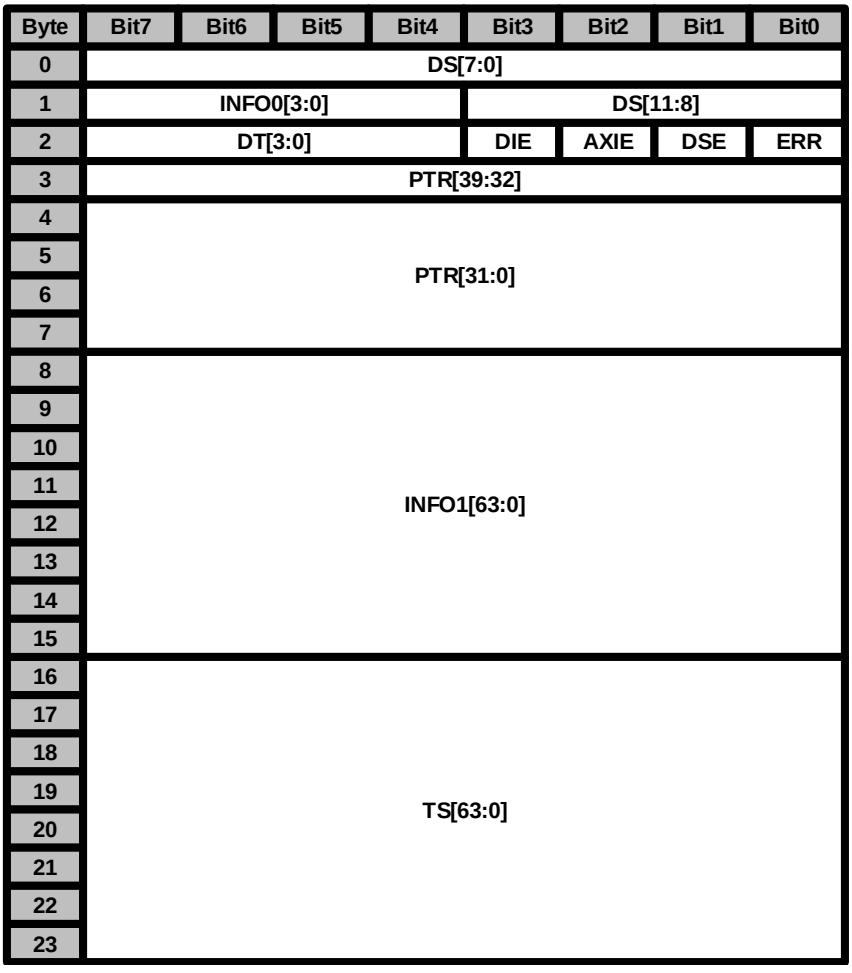


Fig 5.5: Extended timestamp descriptor (**GWDCCi.ETSi** == 1 and **GWDCCi.EDEi** == 1)

Table 5-1: General format descriptor field explanation

Field name	Bit width	Description
DS	12	Descriptor Size Restrictions: - SW: DS field maximum value is 2048. - SW: For other descriptors that LINKFIX, LEMPTY, EEMPTY, LINK and EOS, DS should not be set to 0.
INFO0	4	INFOrmation 0
ERR	1	ERRor
DSE	1	Data Size Error
AXIE	1	AXI bus Error
DIE	1	Descriptor Interrupt Enable This (DESCR.DIE) field should not be set for LINK/LINKFIX descriptors in Rx chain. - 1'b0: Interrupt disable - 1'b1: Interrupt enable
DT	4	Descriptor Type, refer to section 5.1.2.2
PTR	40	PoinTeR
INFO1	64	INFOrmation 1
TS	64	TimeStamp

Notes:

- General format descriptor field utilization is described per use-case later in the following sections.

5.1.2.2 Descriptor types

The table below gives an overview of all descriptor types used by GWCA. The first column shows the descriptor type name and the second column the 4-bit number to be written to **DESCR.DT**.

Table 5-2: Descriptor Type

Name	DT	Description
Empty data descriptors		
FEMPTY_IS	1	**Frame EMPTY Incremental Start** Frame data related descriptor without valid frame data. Used to set an incremental area.
FEMPTY_IC	2	**Frame EMPTY Incremental Continue** Frame data related descriptor without valid frame data. Used to continue using an incremental area.
FEMPTY_ND	3	**Frame EMPTY No Data Storage** Frame data related descriptor without valid frame data. Used to reject data.
FEMPTY	4	**Frame EMPTY** Frame complete data related descriptor without valid frame data. Used to save frame data.
FEMPTY_START	5	**Frame EMPTY START** Frame start data related descriptor without valid frame data. Used to save first part of a frame.
FEMPTY_MID	6	**Frame EMPTY MIDdle** Frame continues data related descriptor without valid frame data. Used to save middle part of a frame.
FEMPTY_END	7	**Frame EMPTY END** Frame end data related descriptor without valid frame data. Used to save end part of a frame.
Data descriptors		
FSINGLE	8	**Frame SINGLE** Storage Element has valid frame data of a complete frame.
FSTART	9	**Frame START** Storage Element has valid frame data. Frame starts with this descriptor and continues in next descriptor.
FMID	10	**Frame MIDdle** Storage Element has valid frame data. Frame has started with a previous descriptor and continues in next descriptor.
FEND	11	**Frame END** Storage Element has valid frame data. Frame has started with a previous descriptor and ends with this descriptor.
Chain control / HW/SW arbitration		
LINKFIX	0	**LINKFIX** Control element pointing to next descriptor in chain.
LEMPTY	12	**Link EMPTY** Used control element pointing to next descriptor in chain.
EEMPTY	13	**EOS EMPTY** Used control element pausing a transmit chain.
LINK	14	**LINK** Control element pointing to next descriptor in chain.
EOS	15	**End Of Set** Control element pausing a transmit chain.

5.1.3 LINKFIX table

5.1.3.1 LINKFIX table initialization

The usage of AXI address table requires the setting of a LINKFIX table in the CPU USER RAM at {**GWDCBAC.DCBAUP**, **GWDCBAC.DCBADP**} location. This table will be used to set the first address of descriptor chains. It can be used any time to change a chain base address or to return to the previous base address.

The LINKXFIX table setting is described in Fig 5.6 through an example. Note than SW and HW initialization are happening simultaneously.

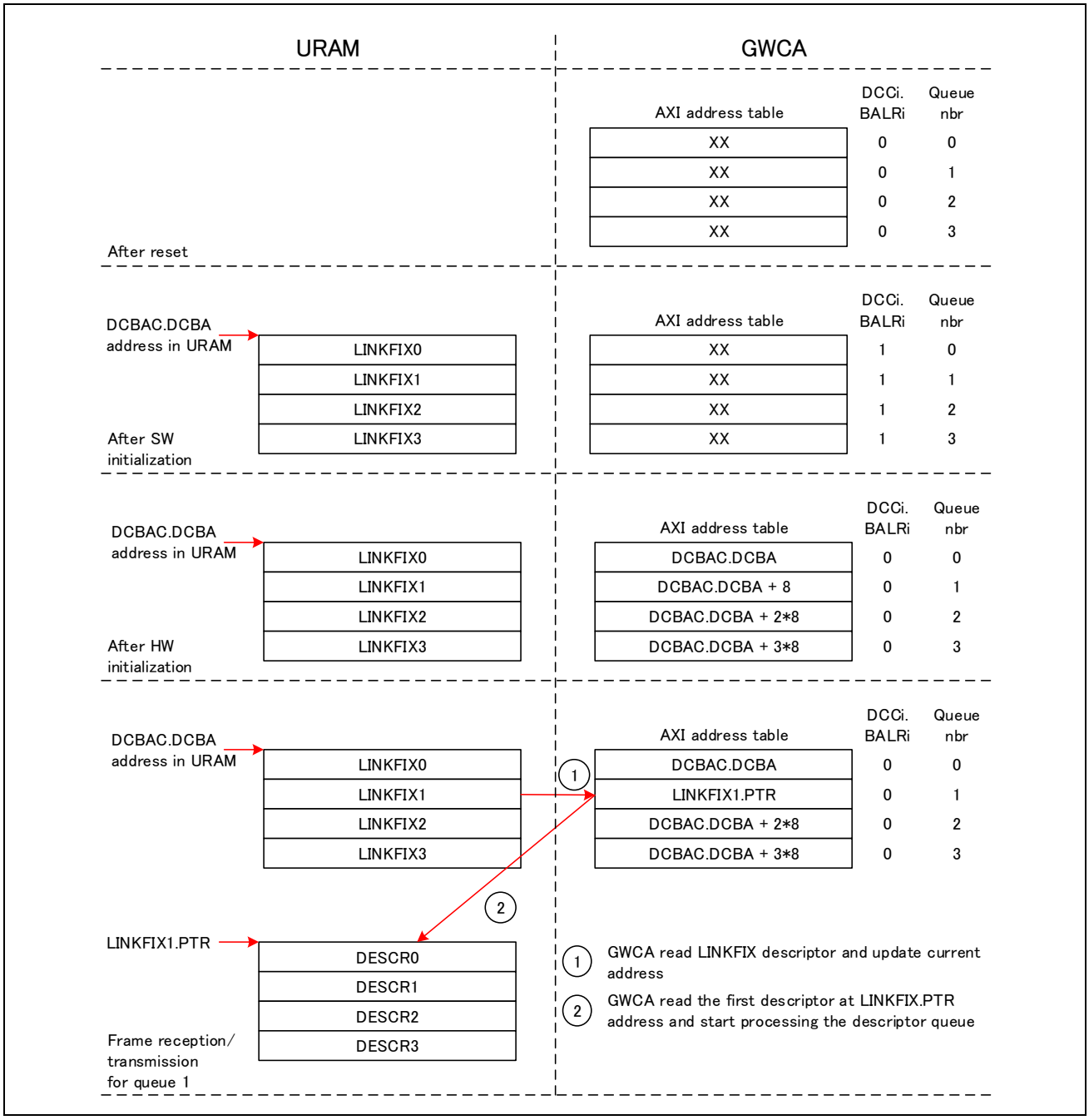


Fig 5.6: LINKXFIX table setting example

Restrictions:

- SW: A LEMPTY descriptor queues must be written for preventing them from starting.

- HW: Add descriptors in LINKFIX table are 8 byte-descriptors independently from **GWDCCi.EDEi** and **GWDCCi.ETSi** register settings.

Note:

- LINKFIX and LINK descriptors are interchangeable. The only difference is that the LINK descriptor can be written back by HW.

5.1.3.2 LINKFIX table descriptors

For the LINKFIX table, a LINKFIX descriptor is used to start a descriptor queue, a LEMPTY descriptor is used to disable a TX queues (**GWDCCi.DQTi** == 1'b1) and a LEMPTY descriptor is used to disable a RX queues (**GWDCCi.DQTi** == 1'b0). In the LINKFIX table, only basic descriptors can be used (section 5.1.2.1). The descriptor formats for LINKFIX table are describes in Fig 5.7 and Table 5-3. If a reception disabled queue is started despite being disabled in the LINKFIX table, the descriptor queue be considered as full (refer to register **GWEIS2i** explanation).

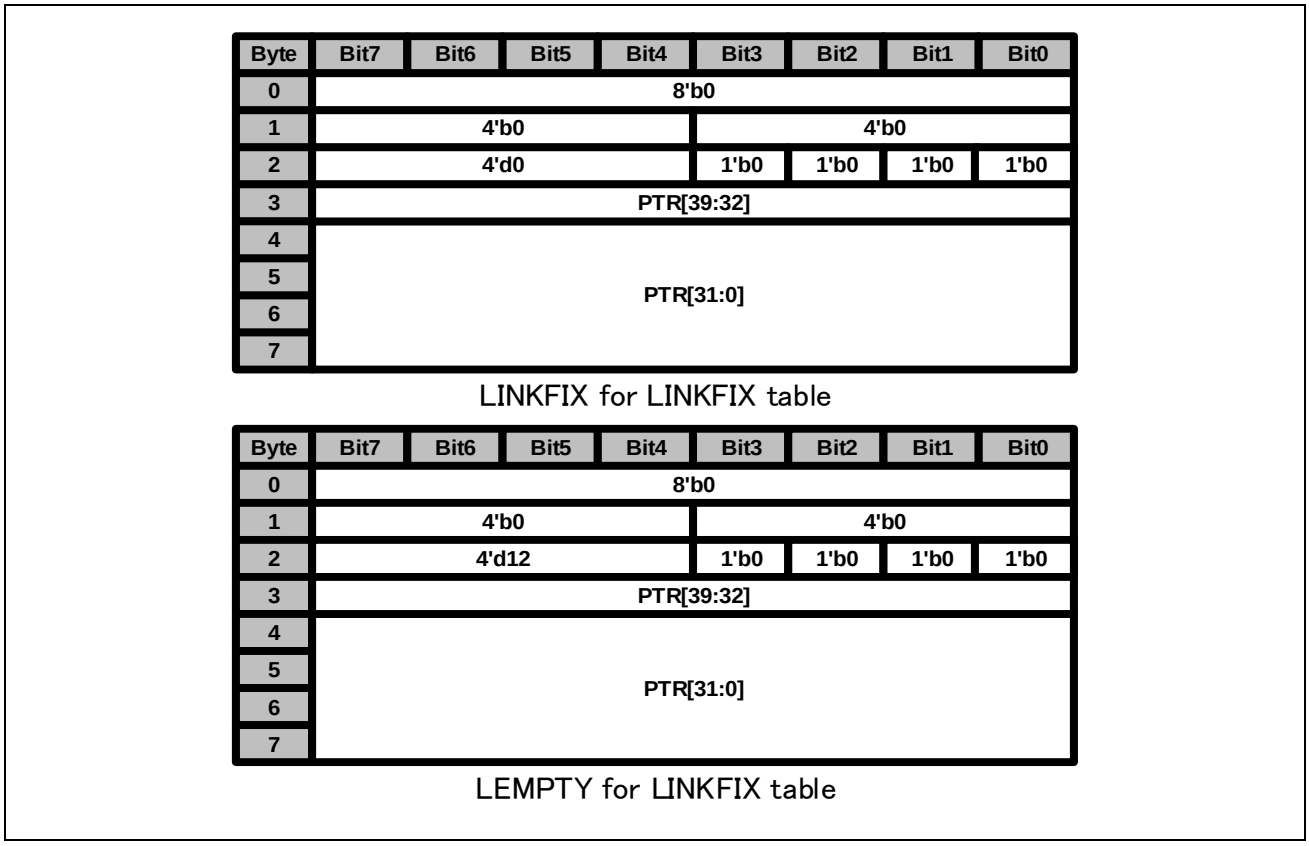


Fig 5.7: LINKFIX table descriptor formats

Table 5-3: LINKFIX table descriptor field description

Field name	LINKFIX	LEMPTY
DS	12'b0	12'b0
INFO0	4'b0	4'b0
ERR	1'b0	1'b0
DSE	1'b0	1'b0
AXIE	1'b0	1'b0
DIE	1'b0	1'b0
DT	4'd0	4'd12
PTR	PTR Pointer pointing to the first descriptor of the descriptor chain	40'b0
INFO1 (GWDCCi.EDEi == 1'b1)	NA for LINKFIX table	NA for LINKFIX table
TS (GWDCCi.ETSi == 1'b1)	NA for LINKFIX table	NA for LINKFIX table

5.1.4 AXI common descriptor utilization

AXI common descriptors are the descriptors which are not reserved to a special use-case and can be inserted in any descriptor chain at any time. It includes LINKFIX and LINK descriptors.

5.1.4.1 LINKFIX descriptor utilization

A LINKFIX descriptor is used to change a descriptor queue pointer in the GWCA to move it to another place and is described in Fig 5.8. The LINKFIX descriptor is never written back to decrease the AXI load.

The LINKFIX descriptor format is described in Fig 5.9 and Table 5-4.

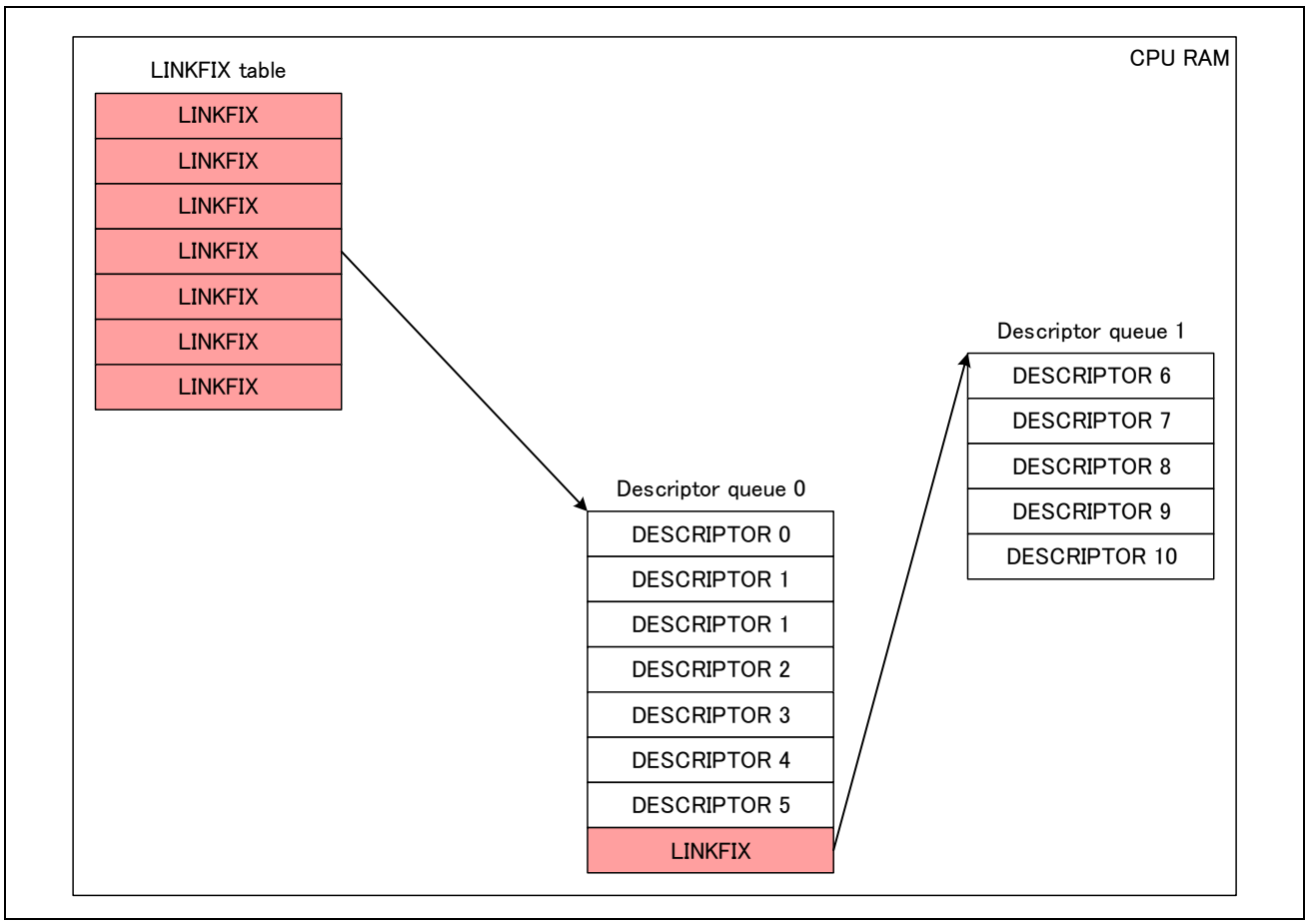


Fig 5.8: LINKFIX descriptor utilization

Notes:

- The LINKFIX table is described in section 5.1.3.
- A LINKFIX descriptor can be placed anywhere in a descriptor queue.
- LINKFIX and LINK descriptors are interchangeable. The only difference is that the LINK descriptor can be written back by HW.

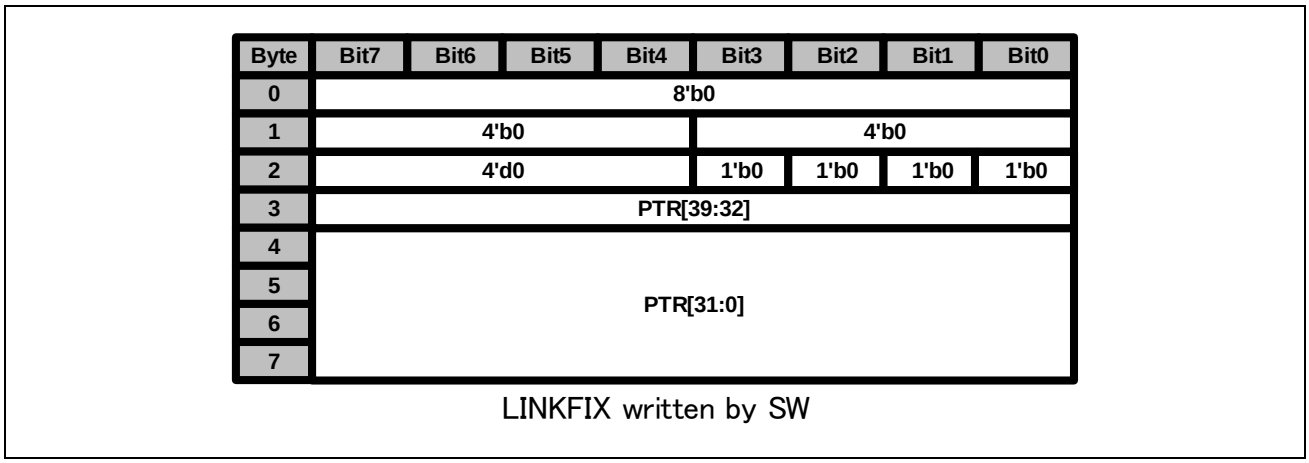


Fig 5.9: LINKFIX utilization format (for basic descriptor, section 5.1.2.1)

Table 5-4: LINKFIX utilization field description

Field name	Field description written by SW	Field description after HW write-back
DS	12’b0	NA: LINKFIX descriptors are never written back
INFO0	4’b0	NA: LINKFIX descriptors are never written back
ERR	1’b0	NA: LINKFIX descriptors are never written back
DSE	1’b0	NA: LINKFIX descriptors are never written back
AXIE	1’b0	NA: LINKFIX descriptors are never written back
DIE	1’b0	NA: LINKFIX descriptors are never written back
DT	4’d0	NA: LINKFIX descriptors are never written back
PTR	PTR Pointer pointing to the new descriptor queue	NA: LINKFIX descriptors are never written back
INFO1 (GWDCCi.EDEi == 1’b1)	64’b0	NA: LINKFIX descriptors are never written back
TS (GWDCCi.ETSi == 1’b1)	64’b0	NA: LINKFIX descriptors are never written back

5.1.4.2 LINK descriptor utilization

A LINK descriptor has the same usage as a LINKFIX descriptor (refer to section 5.1.4.1). Contrary to the LINKFIX descriptor the LINK descriptor can be written back by HW. The LINK descriptor format is described in Fig 5.10 and Table 5-5.

Notes:

- LINKFIX and LINK descriptors are interchangeable. The only difference is that the LINK descriptor can be written back by HW.

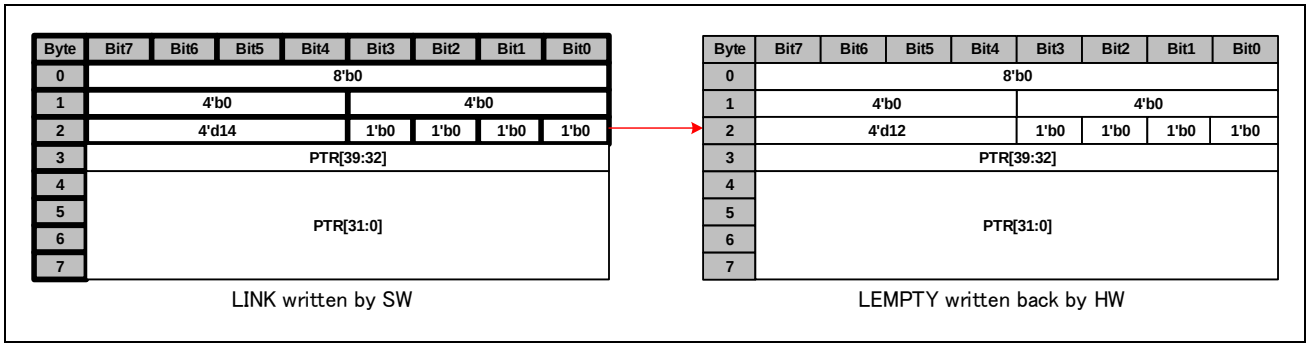


Fig 5.10: LINK utilization format (for basic descriptor, section 5.1.2.1)

Table 5-5: LINK utilization field description

Field name	Field description written by SW	Field description after HW write-back
DS	12'b0	12'b0 (Value is copied from read descriptor)
INFO0	4'b0	4'b0 (Value is copied from read descriptor)
ERR	1'b0	1'b0 (Value is copied from read descriptor)
DSE	1'b0	1'b0 (Value is copied from read descriptor)
AXIE	1'b0	1'b0 (Value is copied from read descriptor)
DIE	1'b0	1'b0 (Value is copied from read descriptor)
DT	4'd14	4'd12 (Value is copied from read descriptor)
PTR	PTR (user setting) Pointer pointing to the new descriptor queue	PTR (Value is copied from read descriptor)
INFO1 (GWDCCi.EDEi == 1'b1)	64'b0	NA: INFO1 not written back for LINK descriptors
TS (GWDCCi.ETSi == 1'b1)	64'b0	NA: TS not written back for LINK descriptors

5.1.5 AXI descriptor queue format

The AXI master accepts two kinds of descriptor queues: linear descriptor queues and cyclic descriptor queues. A linear descriptor queue is a queue which is ended by a terminate descriptor. A linear descriptor queue example is described in figure Fig 5.11. A cyclic descriptor queue is a queue which is looping on itself. A cyclic descriptor queue example is described in figure Fig 5.12.

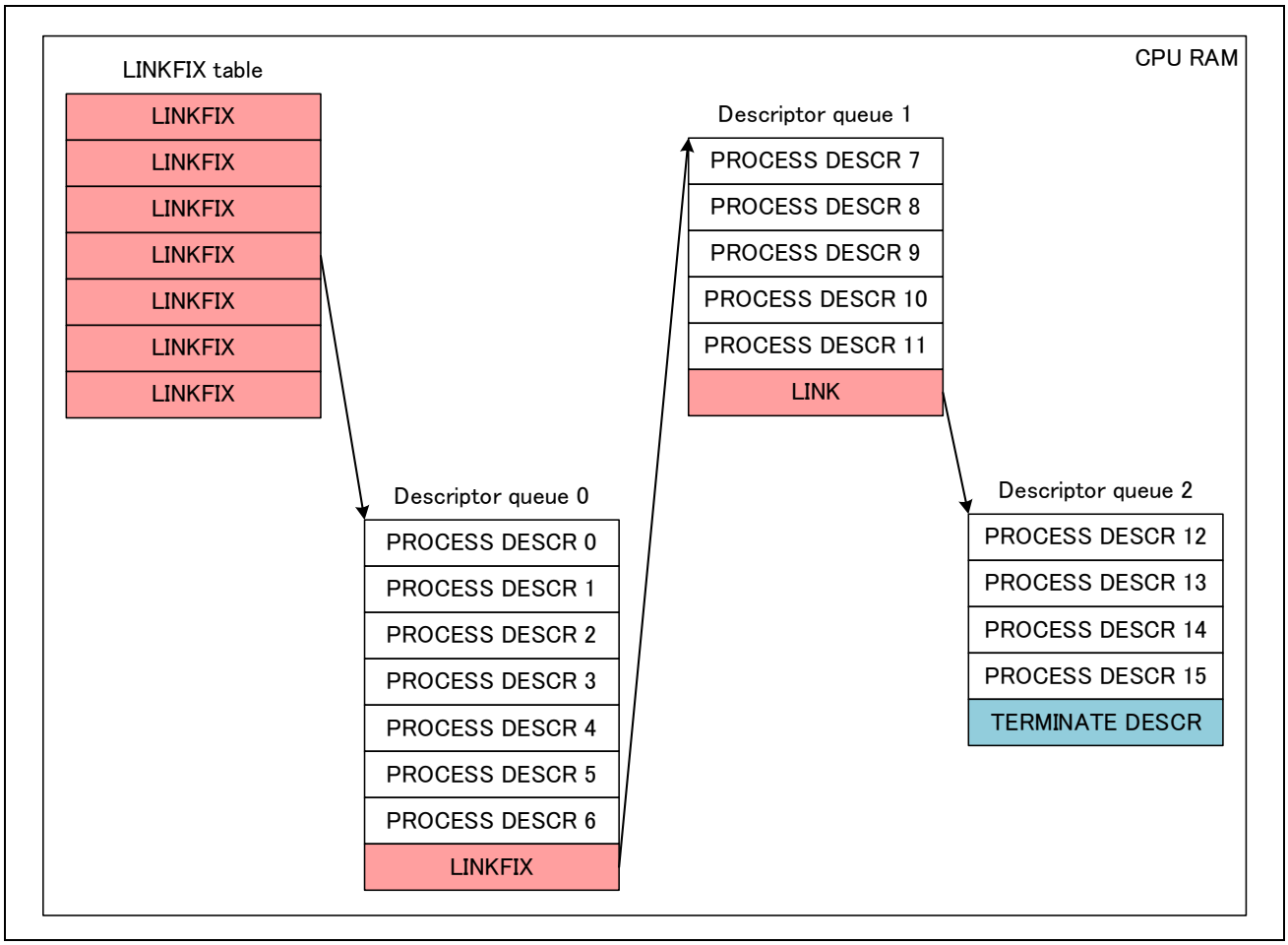


Fig 5.11: Linear descriptor queue example

Restrictions:

- SW: To add data to a linear queue, the SW should write the data and process/terminate descriptor and overwrite the previous terminate descriptor with a LINK/LINKFIX descriptor pointing to the new descriptor chain.

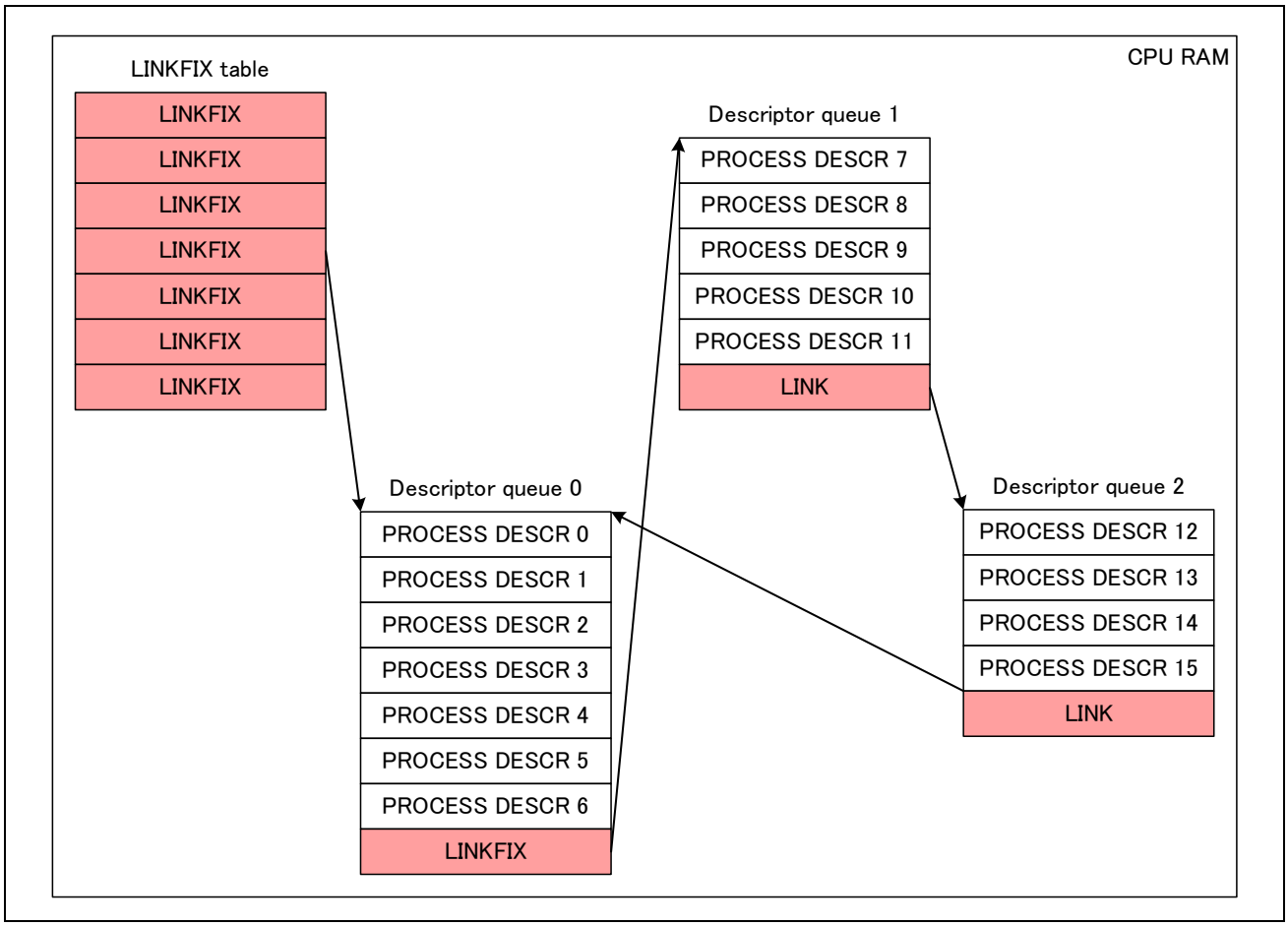


Fig 5.12: Cyclic descriptor queue example

Notes :

- The LINKFIX table is described in section 5.1.3.
- The LINK, LINKFIX utilization is described in section 5.1.4.
- A terminate descriptor (TERMINATE DESCR in Fig 5.11) correspond to a FSINGLE or a LEMPTY descriptor for an RX queue, a FEMPTY or a LEMPTY descriptor for a TX queue and a FSINGLE or a LEMPTY descriptor for a TS queue.
- A process descriptor (PROCESS DESCR in Fig 5.11 and Fig 5.12) correspond to a FEMPTY_IS, FEMPTY_IC, FEMPTY_ND, FEMPTY, FEMPTY_START, FEMPTY_MID or FEMPTY_END descriptor for an RX queue (refer to section 5.3.5), to a FSINGLE, FSTART, FMID, FEND or EOS descriptor for a TX queue (refer to section 5.2.2) and to a FEMPTY_ND for a TS queue (refer to section 5.4.2).

5.1.6 AXI address RAM searching

Searching functions is used to read entries in the AXI address RAM. Table 5-6 describes register used to read an entry in the AXI RAM. Table 5-7 describes the read results.

Table 5-6: AXI Address read registers

Register name	Field name/corresponding field in MAC table	Field explanation
GWAARSS.AARA	Entry address Set the descriptor queue number from which AXU address should be read	Not present in AXI address RAM, entry will be read from this address

Table 5-7: AXI Address read result

Register name	Field name/corresponding field in MAC table	Field explanation
{GWAARSR0.ACARUP, GWAARSR1.ACARDP}	AXI address value	Address of the next descriptor to be processed for AXI descriptor queue number **GWAARSS.AARA**
GWAARSR0.AARSEF	Reading ECC Fail	Reading failed because of an ECC error. During an ECC fail, **GWAARSR0.ACAUP** and **GWAARSR1.ACARDP** are not valid.
GWAARSR0.AARSSF	Reading Security Fail	Reading failed because of a security error. During a security error, **GWAARSR0.ACAUP** and **GWAARSR1.ACARDP** are not valid.

Cautions:

- {**GWAARSR0.ACAUP**, **GWAARSR1.ACARDP**} indicates the address of the next descriptor to be processed for AXI descriptor queue number but it does not mean that the previous descriptor process is completed. That's why this register only has debugging purpose and not HW/SW synchronization purposes.

5.2 Data transmission

GWCA allows data transmission through the GWCA TX data path, The TX data path is described in Fig 5.13.

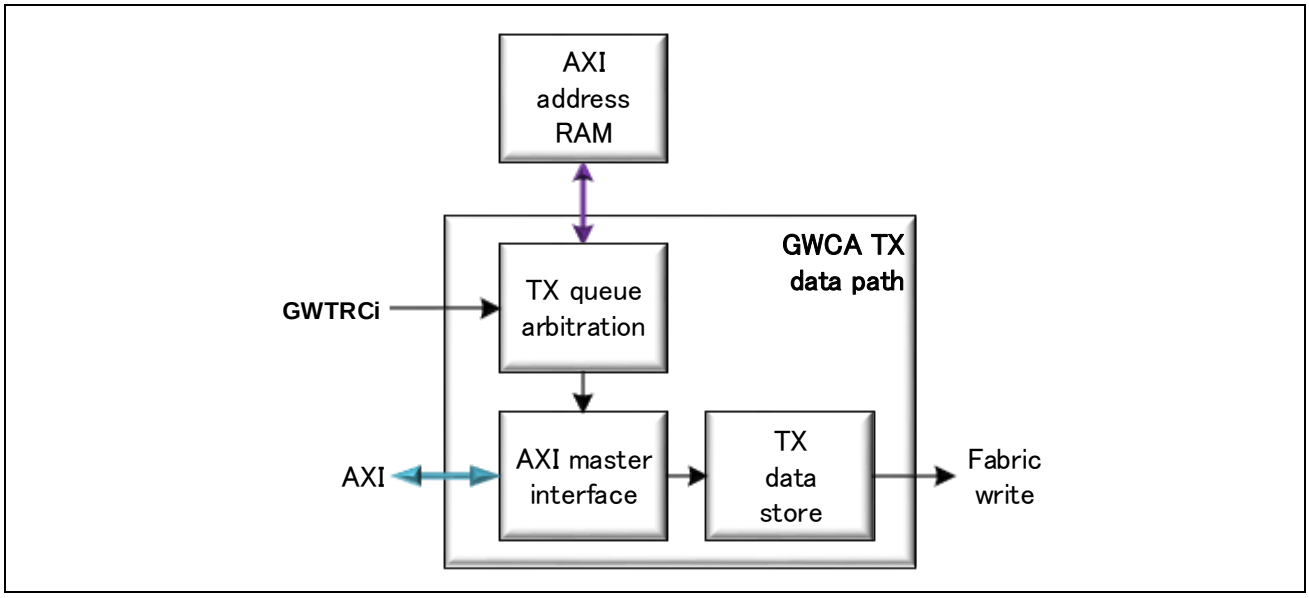


Fig 5.13: GWCA TX data path block diagram

The TX data path is separated in Thee blocks:

- TX queue arbitration: This block arbitrates between TX queues when several queues are requesting for transmission.
- AXI master interface: This block handles the data/AXI descriptor exchange with the CPU.
- TX 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.

5.2.1 TX queue arbitration

TX queue arbitration has four functions. Pausing certain queues when switch is about to overflow, limiting TX queue data rates, arbitrating between TX descriptor queue with a strict priority algorithm and arbitrating between same priority queues with a round robin algorithm.

5.2.1.1 Per priority pause

Per priority pause aims at pausing descriptor queue transmission depending on their priority using “hardware pause” (**PAUSE[MY_PORT_N]** values) [COMA] and **GWTPCp** register. It avoids the AXI master to inject uncritical data in the switch when it is about to overflow.

Functions:

- **GWTPCp** sets the priorities that should be paused when **PAUSE[MY_PORT_N][p]** is set.

5.2.1.2 Rate limiters

Rate limiters are controlling the input flow of the switch. Using these limiters could prevent the switch from overflowing. GWCA has two types of rate limiters, a global rate limiter and AXI_TLIM_N per-queue rate limiters. By setting the rate limiter configurations correctly, expected bandwidth with a tolerance value of less than ± 2% can be achieved.

(1) Per-queue rate-limiter

A per-queue rate-limiter aims at limiter the data throughput per one descriptor queue using **GWRLCi.RLEi**、**GWRLCi.RLIVi** and **GWRLULCi.RLULi** registers. For per-port rate limiter/descriptor queue mapping information refer to section 5.1.1.

Functions:

- **GWRLCi.RLEi** register enables rate-limiter number i.
- **GWRLCi.RLIVi** register sets the maximum throughput accepted by rate limiter number i and should be set respecting equation (1).
- **GWRLULCi.RLULi** register sets the maximum collision time accepted by rate limiter number i and should be set respecting equation (2).

(1) **GWRLCi.RLIVi** = 256 x *ACLK_period[ns]* x *maxBandwidth[Gbps]*

(2) **GWRLULCi.RLULi** = *maximumBurst[bit]*

Where:

- *ACLK_period* is ACLK clock period.
- *maxBandwidth* is the expected maximum throughput accepted by rate limiter number i.
- *maximumBurst* is the maximum burst in bit accepted for corresponding queue.

(2) Global rate limiter

The global rate limiter aims at limiting all AXI data transmission throughput using **GWGRLC.GRLE**, **GWGRLC.GRLIV** and **GWGRLULC.GRLUL** registers and can be monitored using **GWGRLC.GRLULRS**.

Functions:

- **GWGRLC.GRLE** register enables the global rate limiter.
- **GWGRLC.GRLIV** register sets the maximum throughput accepted by AXI data transmission and should be set respecting equation (1).
- **GWGRLULC.GRLUL** register sets the maximum number of credits that the global rate limiter can stack and should be set respecting equation (2).
- **GWGRLC.GRLULRS** register shows if some credit has been lost because of a too high AXI latency or because of a too slow AXI.

(1) **GWGRLC.GRLIV** = 256 x *ACLK_period[ns]* x *max_bandwidth[Gbps]*

(2) **GWGRLULC.GRLUL** = *ACLK_period[ns]* x *max_bandwidth[Gbps]* x 2**maximum_axi_latency[cycle]*

Where:

- ACLK_period is ACLK clock period in ns.
- max_bandwidth is the expected maximum throughput accepted by GWCA.
- maximum_axi_latency is the AXI maximum latency in ACLK clock number.

Restrictions:

- SW: The sum of all **GWRLCi.RLIVi** registers should always be smaller or equal to **GWGRLC.GRLIV**.

5.2.1.3 Strict priority

Strict priority arbitrates between TX queue which are authorized by rate limiters to transmit using **GWDCCi.DCPi**. The strict priority algorithm operates as the one in the data reception descriptor store. Refer to section 5.3.1.2(a) for more information.

Functions:

- **GWDCCi.DCPi** register set descriptor queue i priority. The highest priority is the biggest value.

5.2.1.4 Round Robin

Round robin arbitrates between the highest priority queues selected by the strict arbiter. The round robin operates as the one in the data reception descriptor store. Refer to section 5.3.1.2(b) for more information.

5.2.2 AXI master interface

The AXI master interface handles the data/AXI descriptor exchange with the CPU. It receives the queue number of the descriptor queue number authorised to transmit by the TX queue arbitration, read the corresponding data from the CPU RAM and send the data to TS data store. To do so, it uses **GWTRCi.TSRt**, **GWDCCi.EDEi**, **GWDCCi.ETSi**, **GWDCCi.SMi** and can be monitored using **GWAARSS**, **GWAARSR0** and **GWAARSR1** registers and **GWEIS0.AECCES** interrupt register.

Functions:

- **GWTRCi.TSRt** register is used to start transmission when a descriptor queue is ready.
- **GWDCCi.EDEi** and **GWDCCi.ETSi** registers set the descriptor format that will be used for descriptor queue number i refer to section 5.1.2.1.
- **GWDCCi.SMi** register sets how GWCA will write back descriptors. In this section all explanation will use **GWDCCi.SMi** equal to 2'b00. For **GWDCCi.SMi** equal to 2'b01 or 2'b10 refer to the register explanation.
- For **GWAARSS**, **GWAARSR0** and **GWAARSR1** registers refer to section 5.1.6.
- For **GWEIS0.AECCES** interrupt register shows that some frames may have been lost because of and AXI address RAM ECC error.

Restrictions:

- SW: Only basic descriptors and extended descriptors are authorized for data transmission, refer to section 5.1.2.1.

Data transmission can be done using linear or cyclic descriptor queues (refer to section 5.1.5). Data transmission process descriptors can be set two different ways, through the basic data transmission descriptor chain (section 5.2.2.1) and the time-controlled data transmission descriptor chain (section 5.2.2.2).

5.2.2.1 Basic data transmission

Basic data transmission process descriptor utilization is described in Fig 5.14. Basic data transmission process descriptor format is described in Fig 5.15 and Table 5-8.

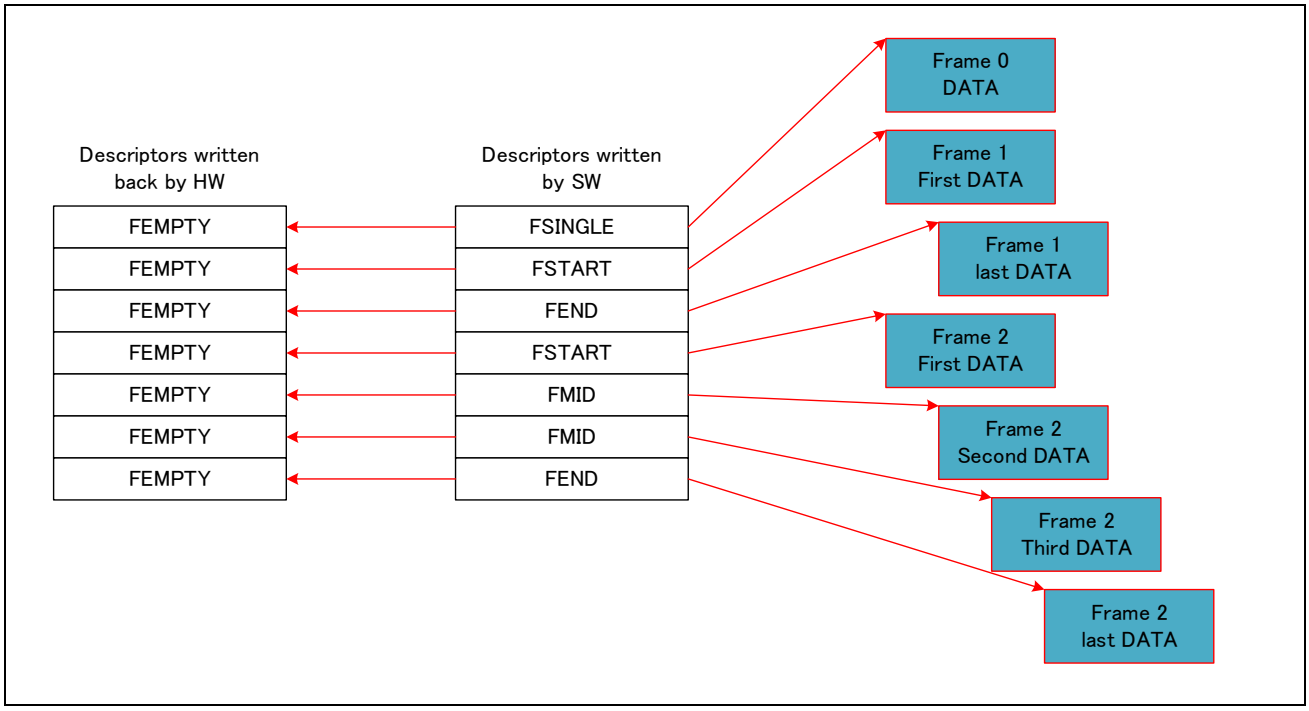


Fig 5.14: Data transmission process descriptor utilization

Notes:

- If needed, the SW can split a frame between several descriptors. The first descriptor should be a FSTART descriptor, the last descriptor should be a FEND descriptor and all other descriptors should be FMID descriptors.
- When data is ready, CPU should set the corresponding **GWTRCi.TSRt** bit to start transfer. Refer to **GWTRCi.TSRt** description.
- **GWTRCi.TSRt** can be set anytime even if data is not available. The GWCA will in this case read the terminate descriptor and clear **GWTRCi.TSRt bit**.

Restrictions:

- SW: When a frame is split, the FSTART descriptor should be written after FMIDs and FEND descriptor for a given frame.

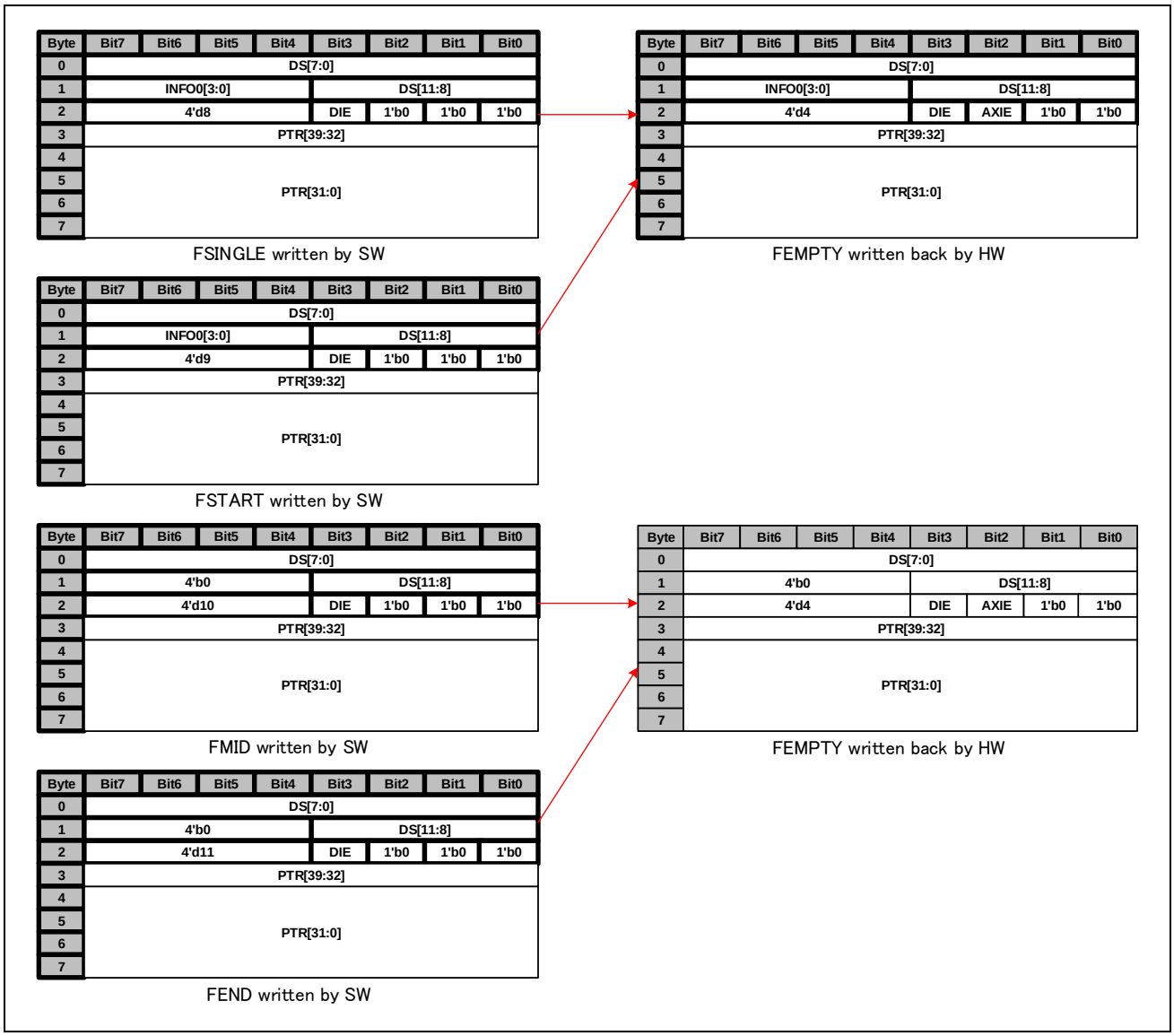


Fig 5.15: Data transmission process descriptor utilization format (for basic descriptor, section 5.1.2.1)

Table 5-8: Data transmission process descriptor utilization field description

Field name	Field description written by SW	Field description after HW write-back
DS	Data what has been written to the URAM by SW	DS (Value is copied from read descriptor)
INFO0	For FSINGLE and FSTART refer to section 5.2.2.3 4'b0 for FMID and FEND.	INFO0 (Value is copied from read descriptor)
ERR	1'b0	1'b0
DSE	1'b0	1'b0
AXIE	1'b0	Set conditions: - HW: An AXI error happened while reading a data (RRESP != 2'b00).
DIE	DIE	DIE (Value is copied from read descriptor)

Field name	Field description written by SW	Field description after HW write-back
DT	4'd8 for FSINGLE 4'd9 for FSTART 4'd10 for FMID 4'd11 for FEND	4'd4 for FEMPTY
PTR	Address where data is written in the URAM	PTR (Value is copied from read descriptor)
INFO1 (**GWDCCi.EDEi** == 1'b1)	For FSINGLE and FSTART refer to section 5.2.2.3 4'b0 for FMID and FEND.	NA: INFO1 not written back for transmission
TS (**GWDCCi.ETSi** == 1'b1)	NA: No timestamp descriptor for transmission	NA: No timestamp descriptor for transmission

5.2.2.2 Time-controlled data transmission

Time-controlled data transmission process descriptor utilization is described in Fig 5.16. Time-controlled data transmission process descriptor format is described in Fig 5.17 and Table 5-9. FSINGLE, FSTART, FMID, FEND and FEMPTY formats are not described in this section, for their description refer to Fig 5.15 and Table 5-8.

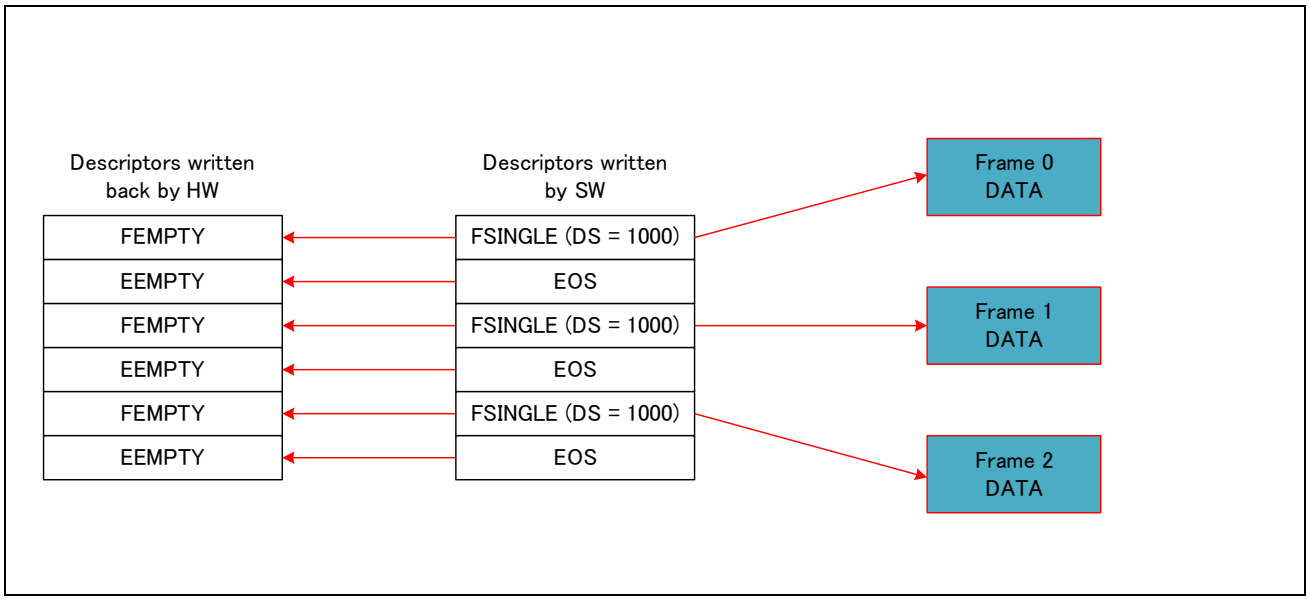


Fig 5.16: Data transmission process descriptor utilization

Notes:

- If needed, the SW can split a frame between several descriptors. The first descriptor should be a FSTART descriptor, the last descriptor should be a FEND descriptor and all other descriptors should be FMID descriptors.
- When data is ready, CPU should set the corresponding **GWTRCi.TSRt** bit to start transfer. Refer to **GWTRCi.TSRt** description.
- Reading an EOS descriptor will have per effect to clear **GWTRCi.TSRt** bit. So, if the CPU set **GWTRCi.TSRt** bit at a cyclic timing, CPU can control the data throughput (If TSR is already set when the CPU tries to set it the next time, only one frame will be sent so a decrease in throughput can happen).
- **GWTRCi.TSRt** bit can be set automatically using gPTP timer [gPTP]. Refer to section 5.5.
- **GWTRCi.TSRt** can be set anytime even if data is not available. The GWCA will in this case read the terminate descriptor and clear **GWTRCi.TSRt bit**.

Restrictions:

- SW: When a frame is split, the FSTART descriptor should be written after FMIDs and FEND descriptor for a given frame.

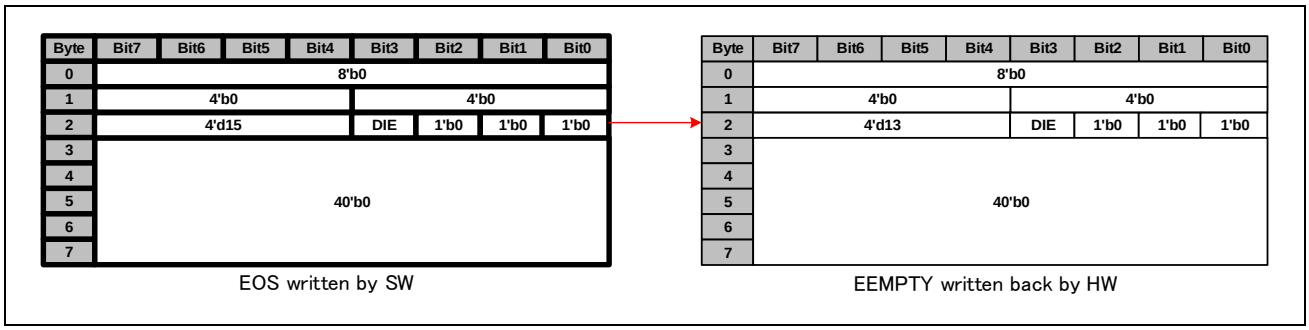


Fig 5.17: Time-controlled data transmission process descriptor utilization format

(for basic descriptor, section 5.1.2.1)

Table 5-9: Time-controlled data transmission process descriptor utilization field description

Field name	Field description written by SW	Field description after HW write-back
DS	12'b0	DS (Value is copied from read descriptor)
INFO0	4'b0	INFO0 (Value is copied from read descriptor)
ERR	1'b0	1'b0
DSE	1'b0	1'b0
AXIE	1'b0	1'b0
DIE	DIE	DIE (Value is copied from read descriptor)
DT	4'd15	4'd13
PTR	40'b0	PTR (Value is copied from read descriptor)
INFO1 (**GWDCCi.EDEi** == 1'b1)	64'b0	NA: INFO1 not written back for EOS descriptors
TS (**GWDCCi.ETSi** == 1'b1)	NA: No timestamp descriptor for transmission	NA: No timestamp descriptor for transmission

5.2.2.3 Data transmission common descriptor format

In this section are described the fields which are common for any data transmission (INFO0 and INFO1). There are two types of formats, the transmission direct descriptor and the transmission ethernet descriptor formats. the transmission direct descriptor could be used, for example, for non-routable protocol forwarding (network control) or for CPU-CPU communication and the transmission ethernet descriptor could be used for all normal ethernet transmissions. All the fields contained in AXI descriptors, if quoted, will be written **ADESCR.{Field name}**.

(1) Transmission direct descriptor

Transmission direct descriptor is used by a CPU to send frames by deciding their target directly (Forwarding engine processing will be by-passed [FWD]). Transmission direct descriptor INFO formats are described in follow.

Restrictions:

- SW: The Transmission direct descriptor can only be and extended descriptor, refer to section 5.1.2.1.

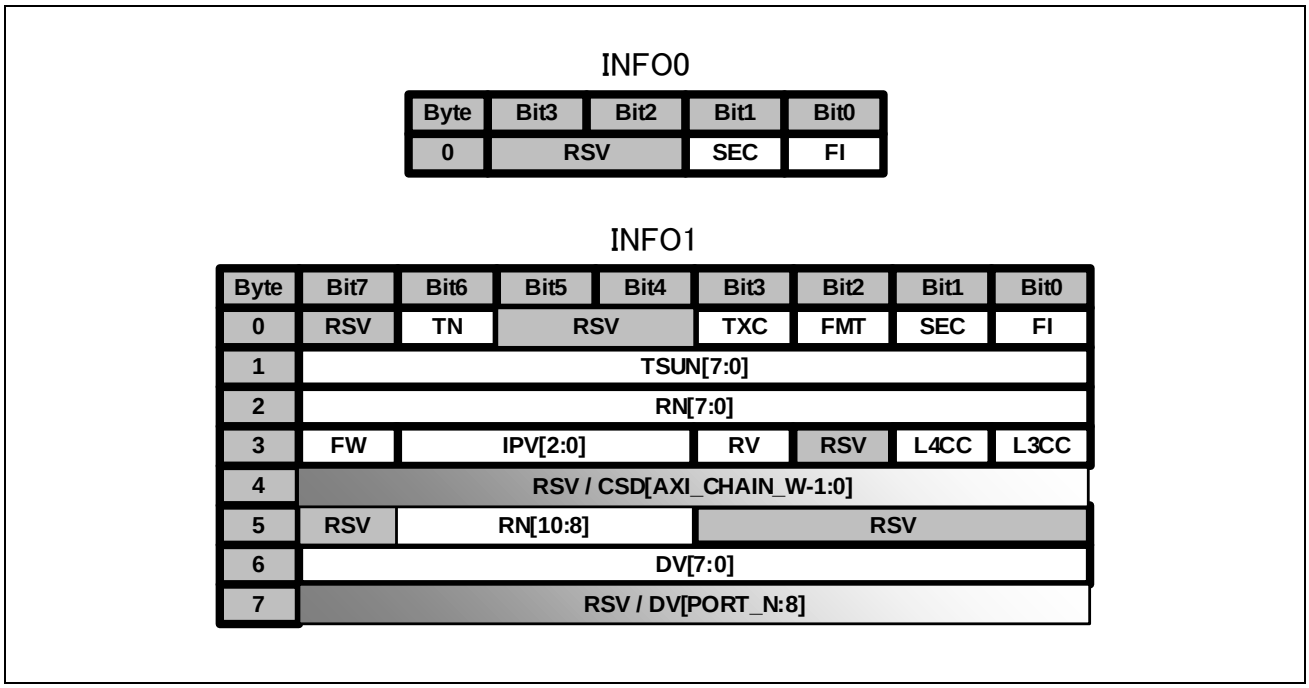


Fig 5.18: Transmission direct descriptor INFO formats

Table 5-10: Transmission direct descriptor INFO field descriptions

Field name	Bit width	Description
FI	1	FCS in.
SEC	1	Secure descriptor
FMT	1	Descriptor format Restrictions: - SW: Fixed to 1'b1 for Direct descriptor.
TXC	1	TX timestamp capture [RMAC]. Used to set corresponding bit in RMAC TX interface.

Field name	Bit width	Description
TN	PTP_TN_W	Timer utilized for capture/insertion [RMAC]. Used to set corresponding bit in RMAC TX interface.
TSUN	8	Timestamp unique number [RMAC]. Used to set corresponding bit in RMAC TX interface.
L3CC	1	Layer 3 checksum calculation Used to calculate and insert Layer 3 checksum in corresponding frame. Refer to section 5.2.3.4
L4CC	1	Layer 4 checksum calculation Used to calculate and insert Layer 4 checksum in corresponding frame. Refer to section 5.2.3.4
RV	1	Routing valid. Refer to section 5.3.2, to ethernet agent L2/3 update [TSN] and to forwarding engine L3 touting table [FWD]
RN	LTH_RRULE_W	Routing number. Refer to section 5.3.2, to ethernet agent L2/3 update [TSN] and to forwarding engine L3 touting table [FWD]
IPV	3	Internal priority value Target priority of the frame.
FW	1	The FCS contained in the frame is wrong (It will be corrected in the switch)
CSD	AXI_CHAIN_W	CPU sub destination for GWCA (It is common to for all GWCAs).
DV	PORT_N	Destination vector [FWD]
RSV	- -	Reserved area Restrictions: - SW: these fields should be set to 0.

(2) Transmission ethernet descriptor

Transmission ethernet descriptor is used by a CPU to send frames that will be forwarded by the Forwarding engine [FWD]. Transmission ethernet descriptor INFO formats are described in Fig 5.19 and Table 5-11.

Restrictions:

- SW: Because FW field (refer to section 5.2.2.3(1)) is not present in an Ethernet descriptor. Any transmit frame should have either a correct FCS or no FCS.

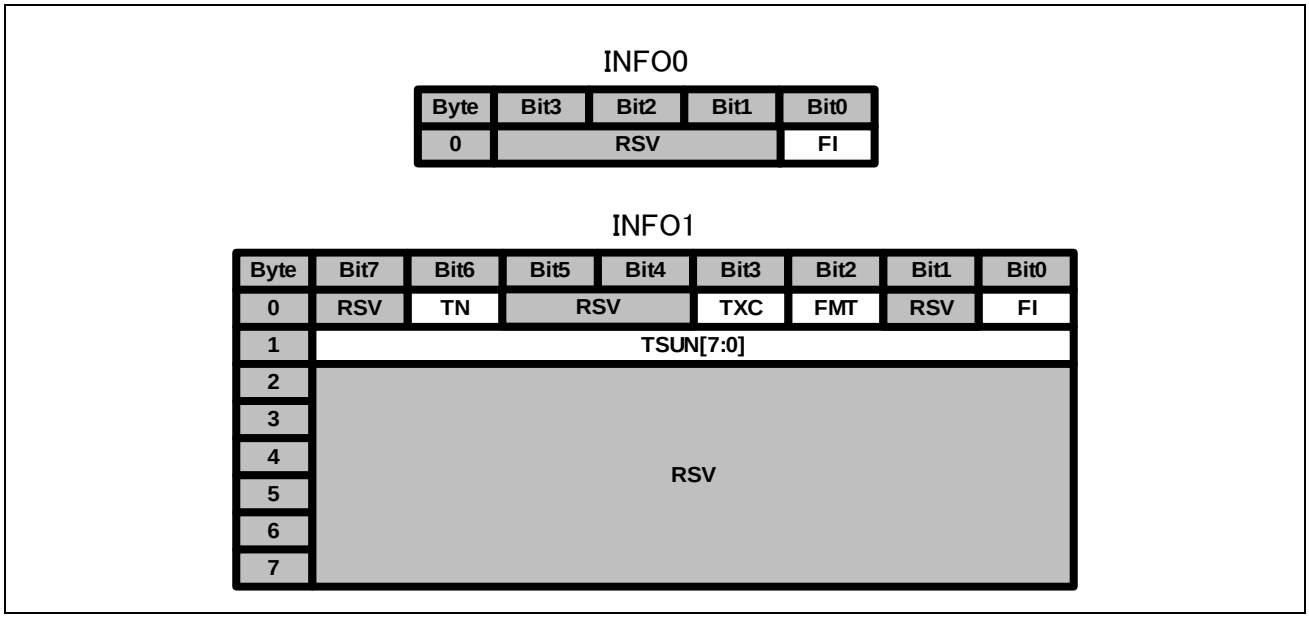


Fig 5.19: Transmission ethernet descriptor INFO formats

Table 5-11: Transmission ethernet descriptor INFO field descriptions

Field name	Bit width	Description
FI	1	FCS in.
FMT	1	Descriptor format Restrictions: - SW: Fixed to 1'b0 for Ethernet descriptors.
TXC	1	TX timestamp capture [RMAC]. Used to set corresponding bit in RMAC TX interface.
TN	PTP_TN_W	Timer utilized for capture/insertion [RMAC]. Used to set corresponding bit in RMAC TX interface.
TSUN	8	Timestamp unique number [RMAC]. Used to set corresponding bit in RMAC TX interface.
RSV	- -	Reserved area Restrictions: - SW: these fields should be set to 0.

5.2.3 TX data store

TX data store has follow functions, it extracts TAG information for frames, applies VLAN tagging, check/calculate checksums in frames and save the frames in the local RAM. These functions are applied to all frames independently from their descriptor format.

5.2.3.1 TAG information extraction

TAG information extraction is done in HW without setting intervention. The extracted format can be filtered using **GWTTFC** register. TAG information extraction is described in Fig 5.20 along with **GWTTFC** register bit explanation for TAG filtering. The extracted fields are SN, CoS-TCI, C-TCI and S-TCI.

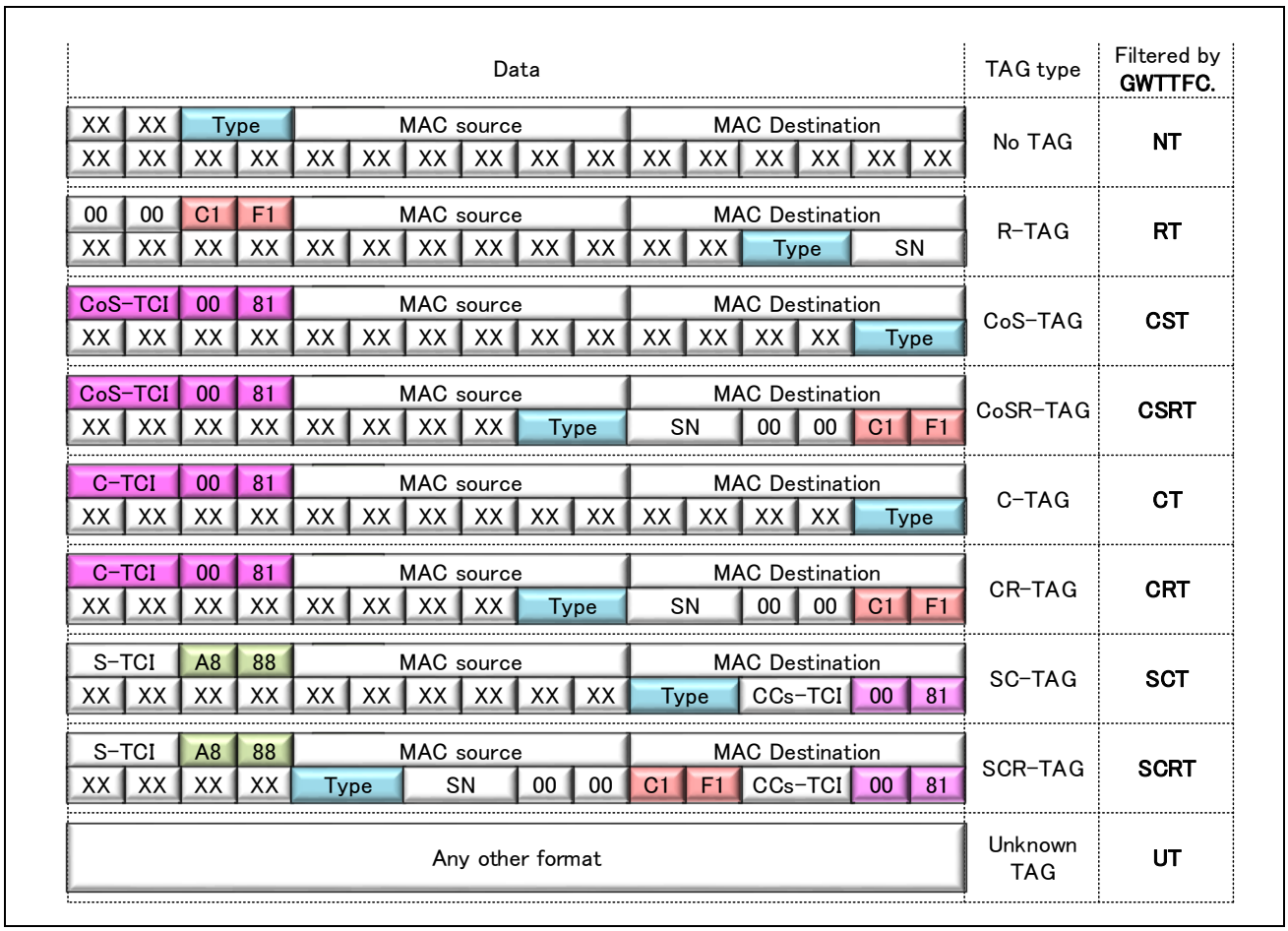


Fig 5.20: TAG information extraction

Notes:

- Only colored part in Fig 5.20 are considered for TAG information extraction.
- HF1C1 corresponds to R-TAG TPID and SN correspond to R-TAG sequence number [802.1CB].
- H8100 corresponds to C-TAG TPID, C-TCI corresponds to a C-TAG TCI with a non-null VID, CoS-TCI corresponds to a C-TAG TCI with a null VID and CCs-TCI corresponds to C-TCI or CoS-TCI [802.1Q].
- H88A8 corresponds to S-TAG TPID and S-TCI corresponds to a S-TAG TCI [802.1Q].
- Type corresponds to any Ethernet type value which is not HF1C1, H88A8 or H8100.
- TPID values can be configured in Forwarding Engine [FWD]. Values for TPIDs can be set in forwarding engine (in **FWTTC0** and **FWTTC1** registers).
- If a Double-R-TAG frame passes through Unknown-TAG filtering(GWTTFC.UT=1'b0), that frame will be treated as a Single-R-TAG frame (with 1st R-TAG) (2nd R-TAG will be treated as payload). However, if

Unknown-TAG filtering is Enabled(GWTTFC.UT=1'b1), a Double-R-TAG frame will be treated as a Double-R-TAG frame and will be filtered(rejected).

Restrictions:

- HW: Because 32 bytes are needed for TAG extraction, GWCA will reject transmit frames smaller than 32 bytes and set **GWEIS0.USMFSES** interrupt register.
- SW: In R-TAG, the reserved part should always be set to 0. If the reserved part is not set to 0 and corresponding frame contains an FCS, the frame could be forwarded with an error.

5.2.3.2 Priority Code Point Remapping

This function is recognized as disable by default values.

This function can replace PCP and DEI for TAGs int the ingress frames. This function is processed first before all subsequent frame conversions. It will affect for **DEI and PCP** in C-TAG, SC-TAG and CoS-TAG. It will not affect **VID** for S-TAG and C-TAG.

This feature is placed in front of the “VLAN tagging”. Therefore, HW-TAG is not remapped but locations not replaced by mask are affected.

Functions:

- The PCP and DEI in C-TAG is replaced by the value in the corresponding register **GWICD0/1RC**. When DEI = 1, **GWICD1RC** register replaces the PCP and DEI in C-TAG. When DEI = 0, **GWICD0RC** register replaces the PCP and DEI in C-TAG. For example, by setting {**GWICD0RC.ICD0PR6**, **GWICD0RC**.**ICD0DR6}** to {2, 1}, the C-TAG frame of {PCP,DEI}=={6, 0} will be remapped to {PCP, DEI}=={2, 1}.
- The PCP and DEI in S-TAG is replaced by the value in the corresponding register **GWISD0/1RC**. When DEI = 1, **GWISD1RC** register replaces the PCP and DEI in S-TAG. When DEI = 0, **GWISD0RC** register replaces the PCP and DEI in S-TAG. For example, by setting {**GWISD0RC.ISD0PR6**, **GWISD0RC**.**ISD0DR6}** to {2, 1}, the S-TAG frame of {PCP,DEI}=={6, 0} will be remapped to {PCP, DEI}=={2, 1}.

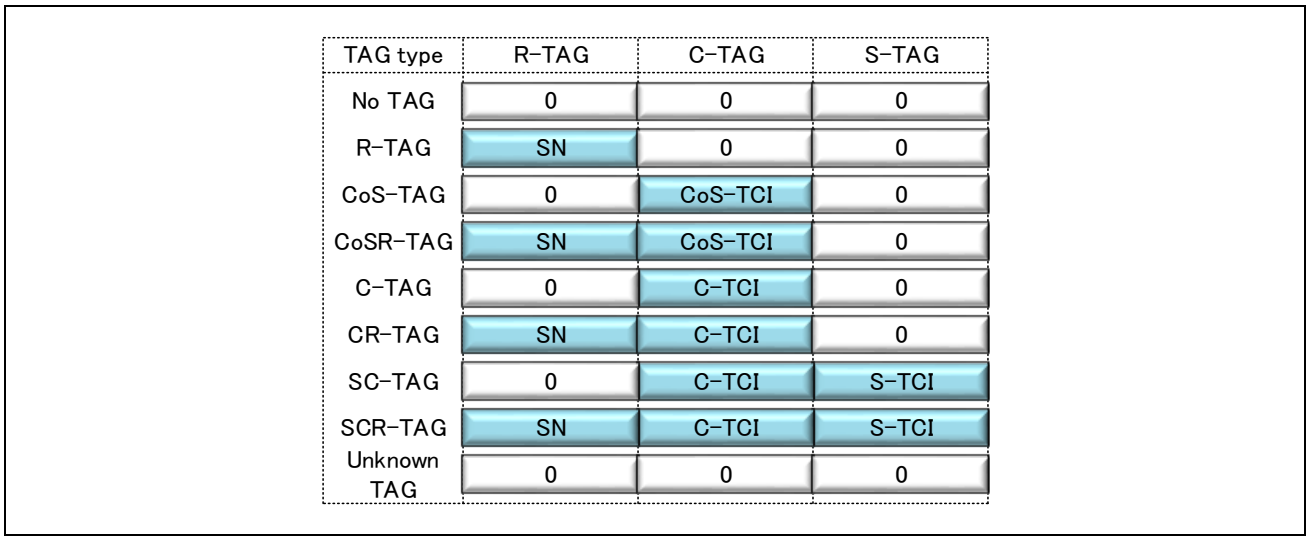
5.2.3.3 VLAN tagging

VLAN tagging replaces or overwrites present TAGs in a frame using the format given by TAG information extraction (section 5.2.3.1), switch VLAN mode (**FWGC.SVM** register in forwarding engine [FWD]) and, **GWVCC** and **GWVTC** registers. After VLAN tagging, any frame has three TAGs, one R-TAG, one C-TAG and one S-TAG which will be directly saved in the TAG RAM [FAB].

Functions:

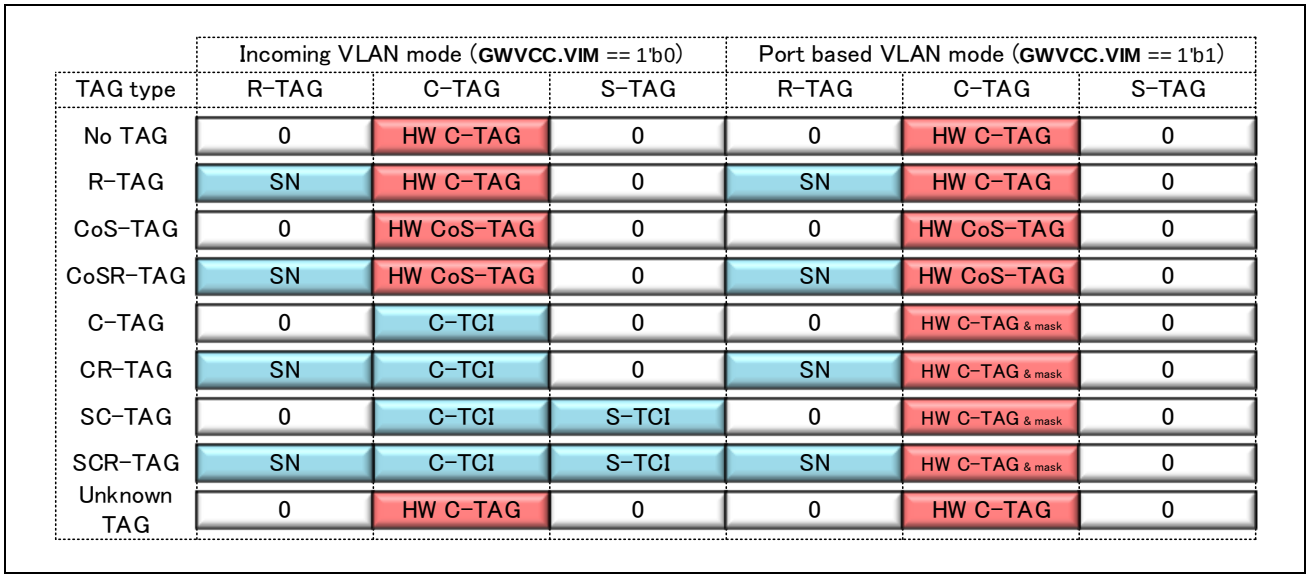
- **GWVCC.VIM** register sets the TX data path in Port based VLAN (all incoming frames are sent to the forwarding engine with the same VLAN ID). **GWVCC.VIM** setting is ignored when the switch in is No VLAN mode (**FWGC.SVM** register in forwarding engine set to 2’b00) [FWD].
- **GWVTC** register is used to set the HW TAGs. There are three possible HW TAGs. The HW C-TAG made from **GWVTC.CTV** as VID, **GWVTC.CTP** as PCP and **GWVTC.CTD** as DEI. The HW S-TAG made from **GWVTC.STV** as VID, **GWVTC.STP** as PCP and **GWVTC.STD** as DEI. The HW CoS-TAG made from **GWVTC.CTV** as VID, the incoming frame PCP as PCP and the incoming frame DEI as DEI.
- **GWVCC.STDUM/STPUM/STVUM/CTDUM/CTPUM/CTVUM** register fields are used to mask the re-tagging HW TAGs in ingress. If any of these fields of the register is set to 1, then re-tagging is disabled for that corresponding field only. “Re-tagging” means overwriting the TAGs, which is tagging for the ethernet incoming frame with the TAGs. This function does not affect “Tagging” for incoming frames without the TAGs. And this function does not affect “CoS-TCI” to “HW CoS-TAG” conversion.

TAGs after VLAN tagging are described in Fig 5.21, Fig 5.22 and Fig 5.23 depending on Switch VLAN mode (**FWGC.SVM** register in forwarding engine [FWD]) and the type of frame given by TAG information extraction (section 5.2.3.1). In the figures, the data extracted from the incoming frames are in blue and the added HW TAGs are in red.



TAG type	R-TAG	C-TAG	S-TAG
No TAG	0	0	0
R-TAG	SN	0	0
CoS-TAG	0	CoS-TCI	0
CoSR-TAG	SN	CoS-TCI	0
C-TAG	0	C-TCI	0
CR-TAG	SN	C-TCI	0
SC-TAG	0	C-TCI	S-TCI
SCR-TAG	SN	C-TCI	S-TCI
Unknown TAG	0	0	0

Fig 5.21: TAGs after VLAN tagging: Switch in No-VLAN mode [FWD]



	Incoming VLAN mode (**GWVCC.VIM** == 1'b0)			Port based VLAN mode (**GWVCC.VIM** == 1'b1)		
TAG type	R-TAG	C-TAG	S-TAG	R-TAG	C-TAG	S-TAG
No TAG	0	HW C-TAG	0	0	HW C-TAG	0
R-TAG	SN	HW C-TAG	0	SN	HW C-TAG	0
CoS-TAG	0	HW CoS-TAG	0	0	HW CoS-TAG	0
CoSR-TAG	SN	HW CoS-TAG	0	SN	HW CoS-TAG	0
C-TAG	0	C-TCI	0	0	HW C-TAG & mask	0
CR-TAG	SN	C-TCI	0	SN	HW C-TAG & mask	0
SC-TAG	0	C-TCI	S-TCI	0	HW C-TAG & mask	0
SCR-TAG	SN	C-TCI	S-TCI	SN	HW C-TAG & mask	0
Unknown TAG	0	HW C-TAG	0	0	HW C-TAG	0

Fig 5.22: TAGs after VLAN tagging: Switch in C-TAG mode [FWD]

	Incoming VLAN mode (**GWVCC.VIM** == 1'b0)			Port based VLAN mode (**GWVCC.VIM** == 1'b1)		
TAG type	R-TAG	C-TAG	S-TAG	R-TAG	C-TAG	S-TAG
No TAG	0	HW C-TAG	HW S-TAG	0	HW C-TAG	HW S-TAG
R-TAG	SN	HW C-TAG	HW S-TAG	SN	HW C-TAG	HW S-TAG
CoS-TAG	0	HW CoS-TAG	HW S-TAG	0	HW CoS-TAG	HW S-TAG
CoSR-TAG	SN	HW CoS-TAG	HW S-TAG	SN	HW CoS-TAG	HW S-TAG
C-TAG	0	C-TCI	HW S-TAG	0	C-TCI	HW S-TAG
CR-TAG	SN	C-TCI	HW S-TAG	SN	C-TCI	HW S-TAG
SC-TAG	0	C-TCI	S-TCI	0	C-TCI	HW S-TAG & mask
SCR-TAG	SN	C-TCI	S-TCI	SN	C-TCI	HW S-TAG & mask
Unknown TAG	0	HW C-TAG	HW S-TAG	0	HW C-TAG	HW S-TAG

Fig 5.23: TAGs after VLAN tagging: Switch in SC-TAG mode [FWD]

5.2.3.4 Checksum handling

Checksum handling can perform checksum check or checksum calculation for IPv4, IPv4/TCP, IPv4/UDP, IPv4/ICMP, IPv6/TCP, IPv6/UDP or IPv6/ICMP (The calculated checksum insertion in the frame takes place at the egress, refer to section 5.3.4.3). Checksum check can be controlled using **GWCKSC** register and checksum insertion is controlled by ADESCR AXI descriptor fields. Checksum handling is described in Fig 5.24, checksum errors are described in Table 5-12 and frame type attribution is described in Fig 5.25.

Restrictions:

- HW: IPv6 extension headers are not supported.

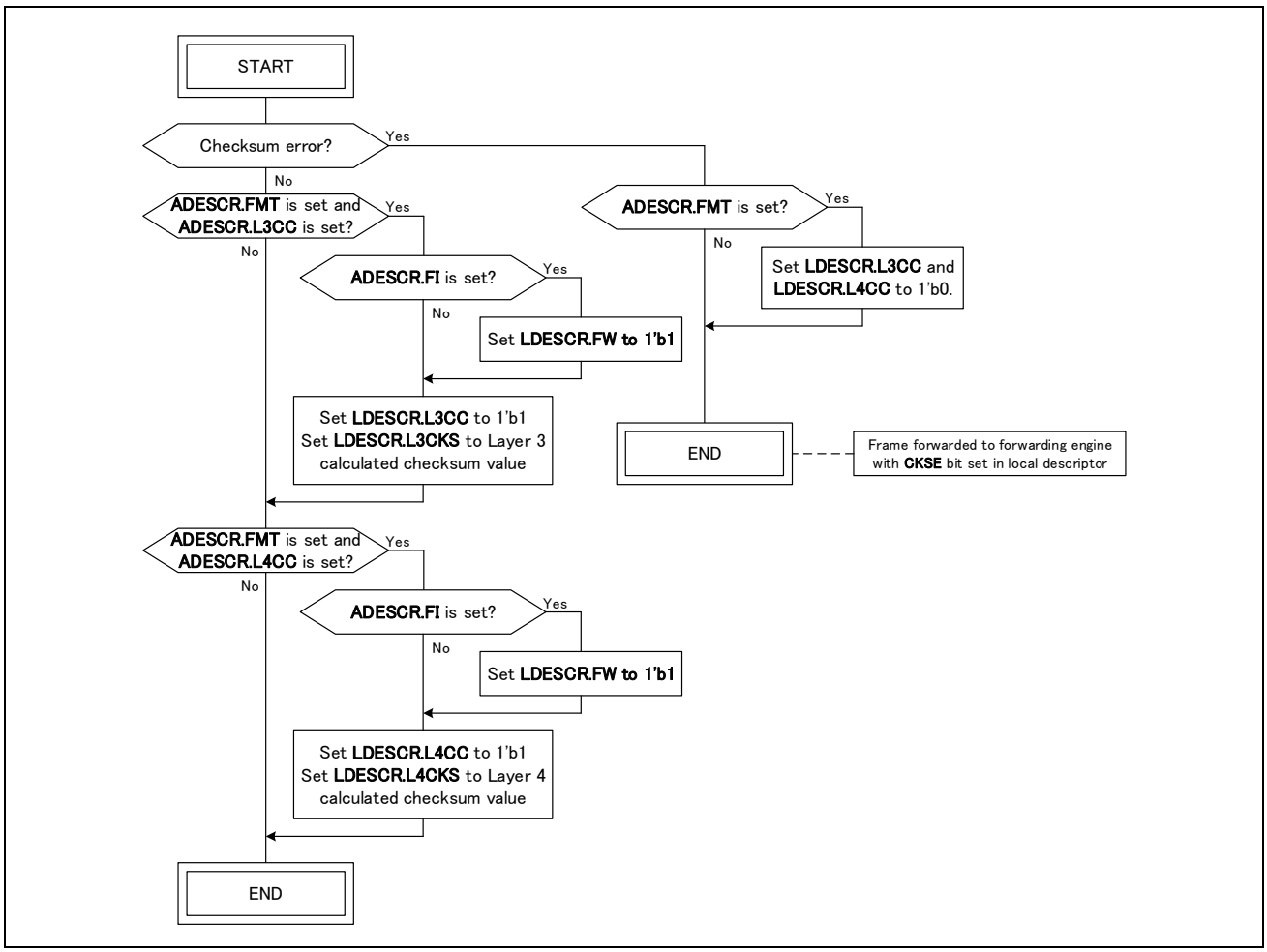


Fig 5.24: Checksum handling flow

Table 5-12: Checksum Errors

Error name	Error conditions	Error interrupt
Format error 0	Frame is not IPv4 **ADESCR.FMT** is set **ADESCR.L3CC** is set	**GWEIS0.CKSES**
Format error 1	Frame is not IPv4/TCP, IPv4/UDP, IPv4/ICMP, IPv6/TCP, IPv6/UDP nor IPv6/ICMP **ADESCR.FMT** is set **ADESCR.L4CC** is set	**GWEIS0.CKSES**
Checksum error IPv4	Frame is IPv4 **GWCKSC.IP4CKSE** is set **ADESCR.FMT** is not set or **ADESCR.L3CC** is not set Checksum included in IPv4 header is wrong (Calculated checksum does not correspond)	**GWEIS0.CKSES**
Checksum error UDP	Frame is IPv4/UDP or IPv6/UDP **GWCKSC.UDPCKSE** is set **ADESCR.FMT** is not set or **ADESCR.L4CC** is not set Checksum included in UDP header is wrong (For IPv4: Calculated checksum does not correspond and checksum present in the frame is not null. For IPv6: Calculated checksum does not correspond or checksum present in the frame is null)	**GWEIS0.CKSES**
Checksum error TCP	Frame is IPv4/TCP or IPv6/TCP **GWCKSC.TCPCKSE** is set **ADESCR.FMT** is not set or **ADESCR.L4CC** is not set Checksum included in TCP header is wrong (Calculated checksum does not correspond)	**GWEIS0.CKSES**
Checksum error ICMP	Frame is IPv4/ICMP or IPv6/ICMP **GWCKSC.ICMPCKSE** is set **ADESCR.FMT** is not set or **ADESCR.L4CC** is not set Checksum included in ICMP header is wrong (Calculated checksum does not correspond)	**GWEIS0.CKSES**

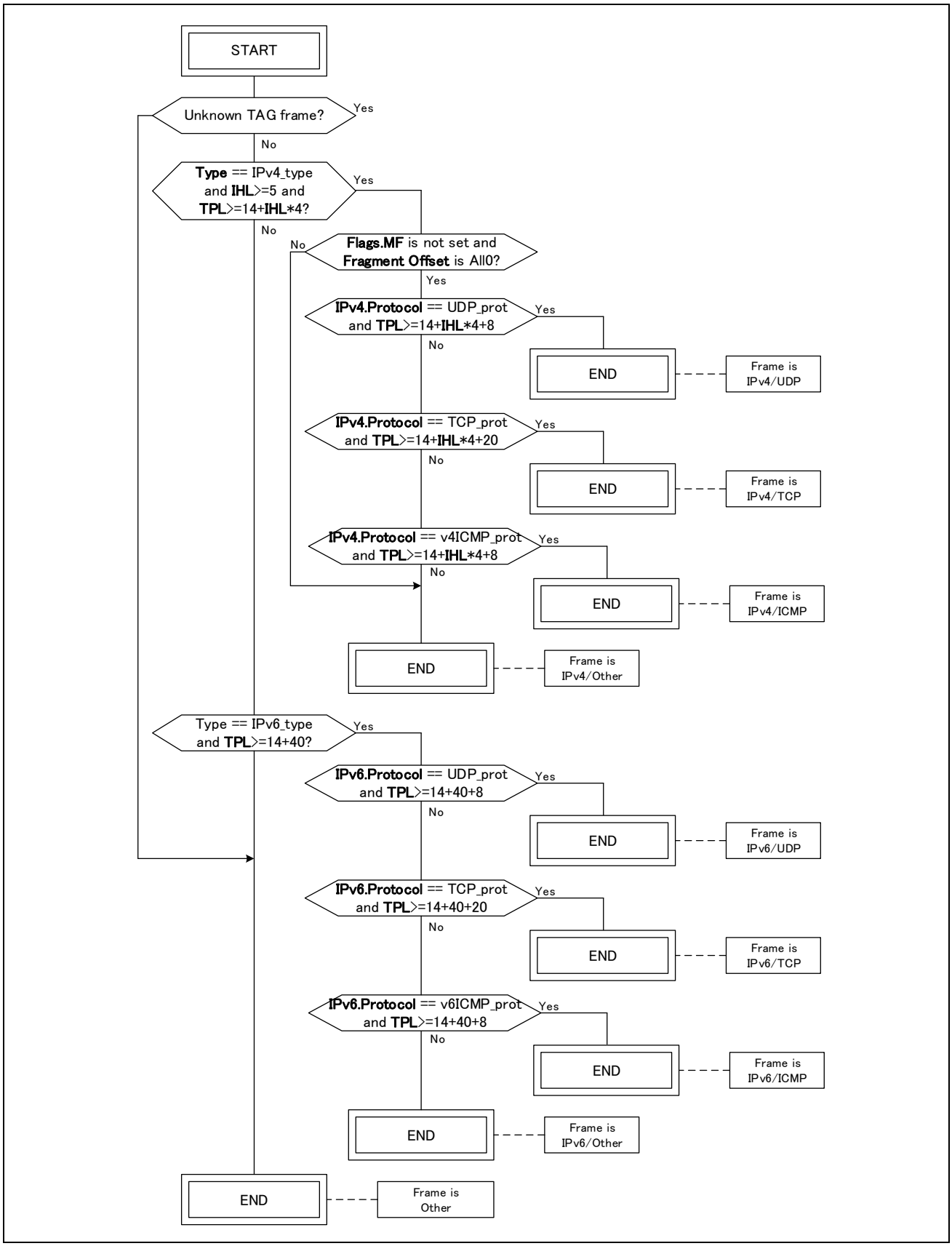


Fig 5.25: Frame type attribution flow

Where:

- “Unknown TAG frame” is described in section 5.2.3.1.
- **Type** refers to the type field described in section 5.2.3.1.

- **TPL** is the frame size without FCS.
- **IPv4_type** is equal to H0800 [RFC791]
- **IHL** refers to IPv4 header Internet Header Length field [RFC791]
- **Flags.MF** refers to to IPv4 header More Fragments Flag bit in Flags field [RFC791]
- **IPv4.Protocol** refers to IPv4 Protocol field [RFC791]
- **IPv6_type** is equal to H86DD [RFC8200]
- **IPv6.Protocol** refers to IPv6 Next Header field [RFC8200]
- **TCP_prot** is equal to H06
- **UDP_prot** is equal to H11
- **v4ICMP_prot** is equal to H01
- **v6ICMP_prot** is equal to H3A

5.2.3.5 Frame saving in Local RAM

For frame saving in the local RAM, refer to Fabric specification document [FAB], section “Data representation in RAMs”. Only the frame descriptor mentioned in Fabric specification document [FAB] will be described in this section.

There are two types of Local RAM descriptors, direct local descriptors and the ethernet local descriptors. They are respectively created from direct descriptors and ethernet descriptors, refer to section 5.2.2.3.

All the fields in these descriptors, if quoted, will be written **LDESCR.{Field name}**.

(1) Direct local descriptor

Direct local descriptor format is described in follow.

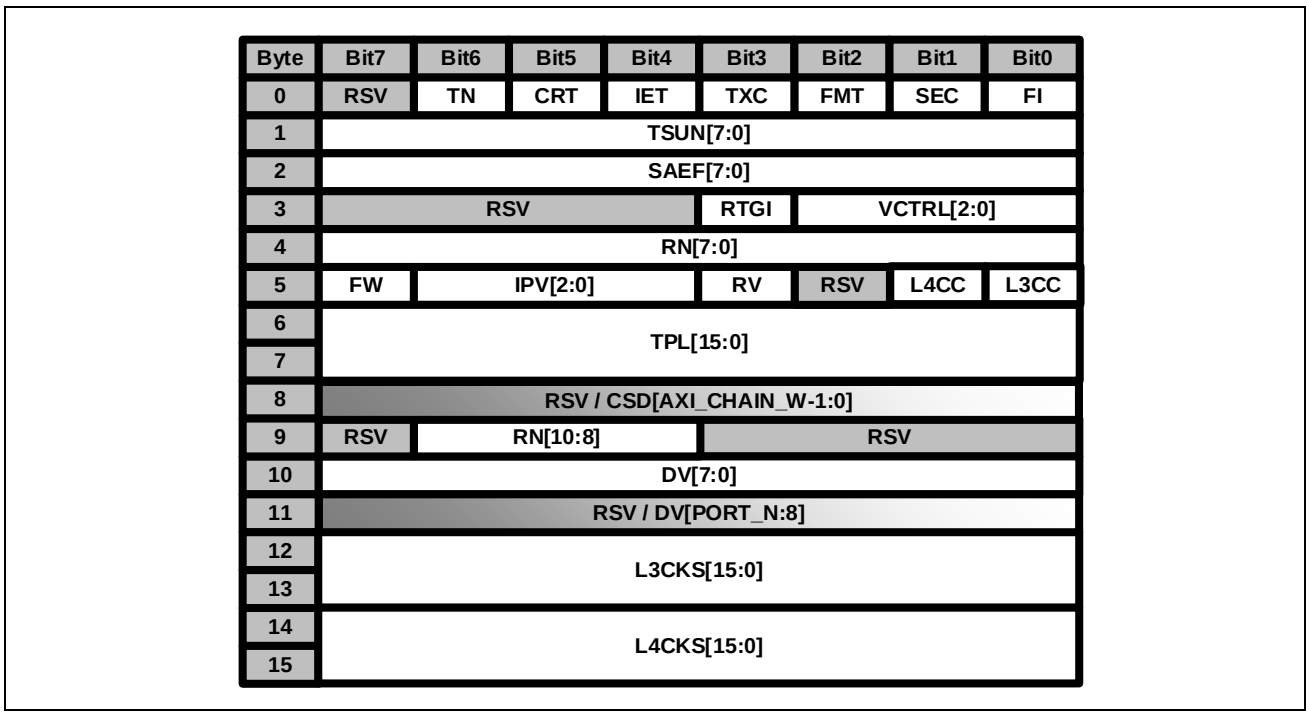


Fig 5.26: Direct local descriptor format

Table 5-13: Direct local descriptor field description

Field name	Bit width	Description	Value for GWCA
FI	1	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
SEC	1	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
FMT	1	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
TXC	1	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
IET	1	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
CRT	1	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
TN	PTP_TN_W	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
TSUN	8	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
SAEF	8	Source agent error flags	Refer to 5.2.3.5(3).

Field name	Bit width	Description	Value for GWCA
VCTRL	FRM_VCTRL_W	VLAN control	VCTRL[2] values: - Set to **GWVCC.VIM.** VCTRL[1:0] values - 2'b00: For No TAG, R-TAG and Unknow TAG frames, refer to section 5.2.3.1. - 2'b01: For C-TAG and CR-TAG frames, refer to section 5.2.3.1. - 2'b10: For SC-TAG and SCR-TAG frames, refer to section 5.2.3.1. - 2'b11: For CoS-TAG and CoSR-TAG frames, refer to section 5.2.3.1.
RTGI	1	R-TAG in	Values: - 1'b0: For No TAG, CoS-TAG, C-TAG, SC-TAG and Unknow TAG frames, refer to section 5.2.3.1. - 1'b1: For R-TAG, CoSR-TAG, CR-TAG and SCR-TAG frames, refer to section 5.2.3.1.
RV	1	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
RN	LTH_RRULE_W	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
L3CC	1	Refer to Table 5-10.	Copied from transmission direct descriptor except when de-asserted by checksum handling function. Refer to section 5.2.2.3(1) for transmission direct descriptor and refer to 5.2.3.4.
L4CC	1	Refer to Table 5-10.	Copied from transmission direct descriptor except when de-asserted by checksum handling function. Refer to section 5.2.2.3(1) for transmission direct descriptor and refer to 5.2.3.4.
IPV	3	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
FW	1	Refer to Table 5-10.	Copied from transmission direct descriptor except when set by checksum handling function. Refer to section 5.2.2.3(1) for transmission direct descriptor and refer to 5.2.3.4.
TPL	FRM_TPL_W	Total payload length	Payload length of frame. It only includes the data saved in the data RAM [FAB].
CSD	AXI_CHAIN_W	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
DV	PORT_N	Refer to Table 5-10.	Copied from transmission direct descriptor. Refer to section 5.2.2.3(1).
L3CKS	16	Layer 3 Checksum	Set to All0 except when set by checksum handling function, Refer to section 5.2.3.4.
L4CKS	16	Layer 4 Checksum	Set to All0 except when set by checksum handling function, Refer to section 5.2.3.4.
RSV	- -	Reserved area	Set to 0

(2) Ethernet local descriptor

Ethernet local descriptor format is described in follow.

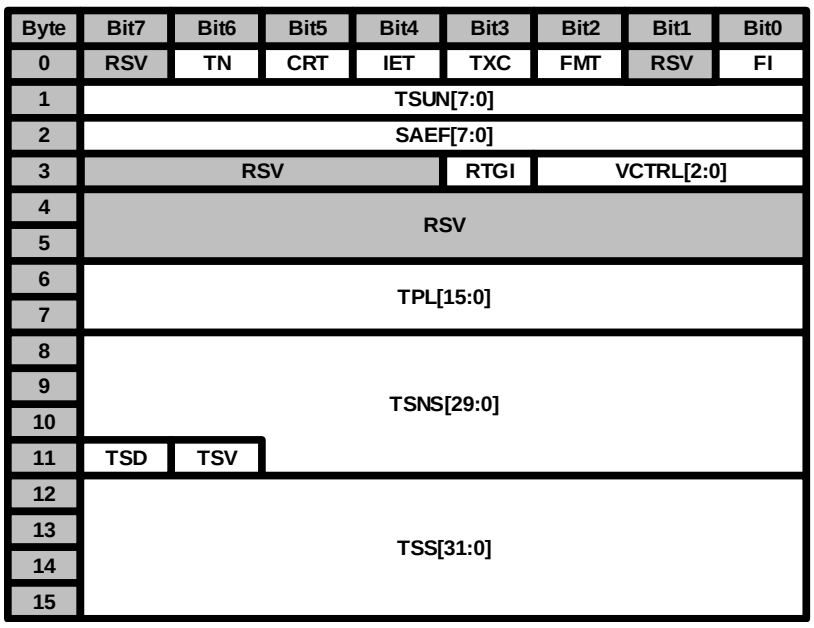


Fig 5.27: Ethernet local descriptor format

Table 5-14: Ethernet local descriptor field description

Field name	Bit width	Description	Value for GWCA
FI	1	Refer to Table 5-11.	Copied from transmission ethernet descriptor. Refer to section 5.2.2.3(2).
FMT	1	Refer to Table 5-11.	Copied from transmission ethernet descriptor. Refer to section 5.2.2.3(2).
TXC	1	Refer to Table 5-11.	Copied from transmission ethernet descriptor. Refer to section 5.2.2.3(2).
IET	1	Refer to Table 5-11.	Copied from transmission ethernet descriptor. Refer to section 5.2.2.3(2).
CRT	1	Refer to Table 5-11.	Copied from transmission ethernet descriptor. Refer to section 5.2.2.3(2).
TN	PTP_TN_W	Refer to Table 5-11.	Copied from transmission ethernet descriptor. Refer to section 5.2.2.3(2).
TSUN	8	Refer to Table 5-11.	Copied from transmission ethernet descriptor. Refer to section 5.2.2.3(2).
SAEF	8	Source agent error flags	Refer to 5.2.3.5(3).
VCTRL	FRM_VCTRL_W	VLAN control	VCTRL[2] values: - Set to **GWVCC.VIM.** VCTRL[1:0] values - 2'b00: For No TAG, R-TAG and Unknow TAG frames, refer to section 5.2.3.1. - 2'b01: For C-TAG and CR-TAG frames, refer to section 5.2.3.1. - 2'b10: For SC-TAG and SCR-TAG frames, refer to section 5.2.3.1. - 2'b11: For CoS-TAG and CoSR-TAG frames, refer to section 5.2.3.1.
RTGI	1	R-TAG in	Values: - 1'b0: For No TAG, CoS-TAG, C-TAG, SC-TAG and Unknow TAG frames, refer to section 5.2.3.1. - 1'b1: For R-TAG, CoSR-TAG, CR-TAG and SCR-TAG frames, refer to section 5.2.3.1.
TPL	FRM_TPL_W	Total payload length	Payload length of frame. It only includes the data saved in the data RAM [FAB].
TSV	1	Timestamp valid	Set to 1'b0. Unused in GWCA.

Field name	Bit width	Description	Value for GWCA
TSD	1	Timestamp default	Set to 1'b0. Unused in GWCA.
TSNS	30	Timestamp nanosecond [PTP]	Set to 30'b0. Unused in GWCA.
TSS	32	Timestamp second [PTP] Restrictions: - HW: The 16-upper bits of the gPTP [PTP] second part are truncated.	Set to 32'b0. Unused in GWCA.
RSV	- -	Reserved area	Set to 0

(3) SAEF format

Source agent error flag format is described in Fig 5.28 and Table 5-15.

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	RSV	DNE	RSV	SEQE	AXE	AREE	CKSE	TFE

Fig 5.28: Source agent error flag format

Table 5-15: 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 **GWTTFC** register explanation.
CKSE	1	ChecKSum Error Set conditions: - HW: One of the conditions detailed in Table 5-12.
AREE	1	AXI RAM ECC Error Set conditions: - HW: When an ECC error happened during a split frame, the frame is sent incomplete.
AXE	1	AXI Error Set conditions: - HW: When an AXIE error happened during a split frame, the frame is sent incomplete.
SEQE	1	SEQuence Error Set conditions: - HW: When a FSINGLE, FSTART, EOS or unknown descriptor is received during a split frame, the frame is sent incomplete.
DNE	1	Descriptor Number Error - HW: When a number **GWMDNC.TXDMN**+1 of descriptor has been already read to fetch a frame during a spit frame and the frame is not finished. The frame is sent incomplete.

Field name	Bit width	Description
RSV	- -	Reserved area Set to 0.

5.3 Data reception

GWCA allows data reception through the GWCA RX data path, The RX data path is described in Fig 5.29.

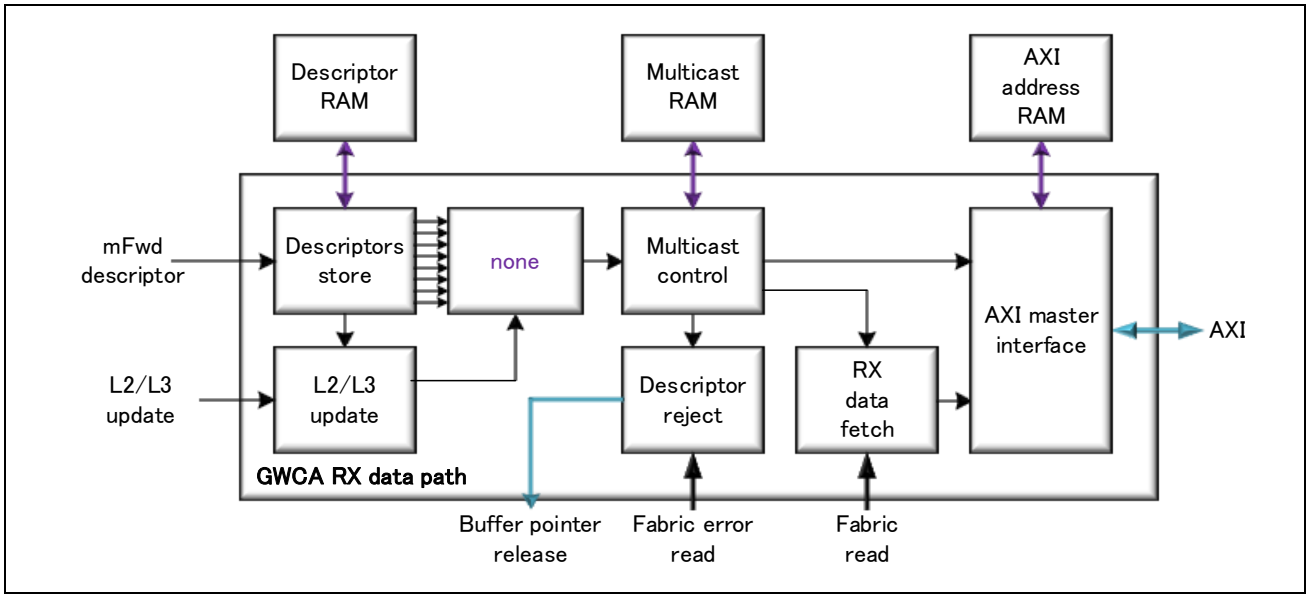


Fig 5.29: GWCA RX data path block diagram

The RX data path is separated in seven blocks:

- Descriptor store: This block aims at storing the descriptor received from the Forwarding Engine [FWD] in the Descriptor RAM and controls the arbitration between descriptors.
- L2/L3 update: This block aims at fetching the L2/L3 update rules from the Forwarding Engine to update the frame while sending it.
- Multicast and DCNC control: This block aims at checking the frame size and security level and decide to forward or discard frames and at changing/copying the descriptors to be able to send the data at several CPU sub-destinations.
- Descriptors 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.
- RX data fetch: This block fetches frames data from the local RAM, update frames depending on the information obtained by L2/L3 update module, on VLAN control information and the R-TAG information, can remove frame FCSs, and release 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.
- AXI master interface: This block handles the data/AXI descriptor exchange with the CPU.

5.3.1 Descriptor store

Descriptor store has two functions. Storing the descriptors coming from the descriptor bus [FWD] in the descriptor RAM and arbitrating descriptor read from the descriptor RAM to decide in which order the descriptors will be sent to the CPU.

5.3.1.1 Descriptor storage

Descriptor storage controls the descriptor storage using **GWIRC.IPVRi**, **GWRDQSC.RDQSL**, **GWRDQC.RDQD**, **GWRDQC.RDQP** and **GWRDQDCq.DQDq** registers and can be monitored using **GWRDQMq.DNQq** and **GWRDQMLMq.DMLQq** registers and **GWEIS0.DSECCES**, **GWEIS1.DQOES** and **GWEIS1.DQSES** interrupt registers. Fig 5.30 describes an example for Descriptor storage setting. And Fig 5.31 shows Descriptor storage method example of **GWRDRC.RDRM**=1.

Functions:

- **GWRDRC.RDRM** selects Descriptor RAM mode. By setting 0, Descriptor RAM is separated for each queue. And by setting 1, all area of Descriptor RAM is shared by all queues.
- **GWIRC.IPVRi** register is used to map descriptor received from forwarding engine with a given **FDESCR.IPV** [FWD] value to a determined descriptor queue. For example, by setting **GWIRC.IPVR6** to 1 all the descriptors received with **FDESCR.IPV** equal to 6 will be stored in descriptor queue number 1.
- **GWRDQSC.RDQSL[q]** is used to set a queues security level. If a queue is secure (**GWRDQSC.RDQSL[q]** is set) and a non-secure descriptor is received (**FDESCR.SEC** [FWD] is not set) for this queue, the descriptor will not be stored and **GWEIS1.DQSES[q]** flag will be set.
- **GWRDQC.RDQP** register is used to pause descriptor queues. For paused descriptor queues, the descriptor transmission to CPU will stop but the descriptor storage in the descriptor RAM will continue.
- **GWRDQC.RDQD** register is used to disable descriptor queues. For disabled descriptor queues and no descriptor will be stored in the Descriptor RAM.
- **GWRDQDCq.DQDq** registers are used to set the maximum number of descriptors that can be stored in each descriptor queue. If a descriptor is received for descriptor q while queue q is full (**GWRDQDCq.DQDq** == **GWRDQMq.DNQq**), the descriptor will not be stored and **GWEIS1.DQOES[q]** flag will be set.
- **GWRDQMq.DNQq** registers are used to monitor the current number of descriptors in each descriptor queue.
- **GWRDQMLMq.DMLQq** registers are used to monitor the maximum number of descriptors that has been held in each queue since the previous reset or the previous time corresponding register has been read.
- **GWEIS0.DSECCES** interrupt register signals that descriptors have been lost because of a descriptor RAM ECC error. **GWEIS1.DIECCES** register signals that descriptors have been lost because of a descriptor Info RAM ECC error. They will be asserted on same time.
- **GWEIS1.DQOES** register signals that descriptors have been discarded because of a descriptor queue overflow. **GWEIS1.DQSES** register signals that descriptors have been discarded because of a descriptor queue security error. They will be asserted on same time.
- **GWLDCN.LDN** counts total number of lost descriptors by ECC Error of descriptor Info RAM.
- **GWCSDRC** will change CSD value of descriptor. (This function is implemented only by [GWCA])

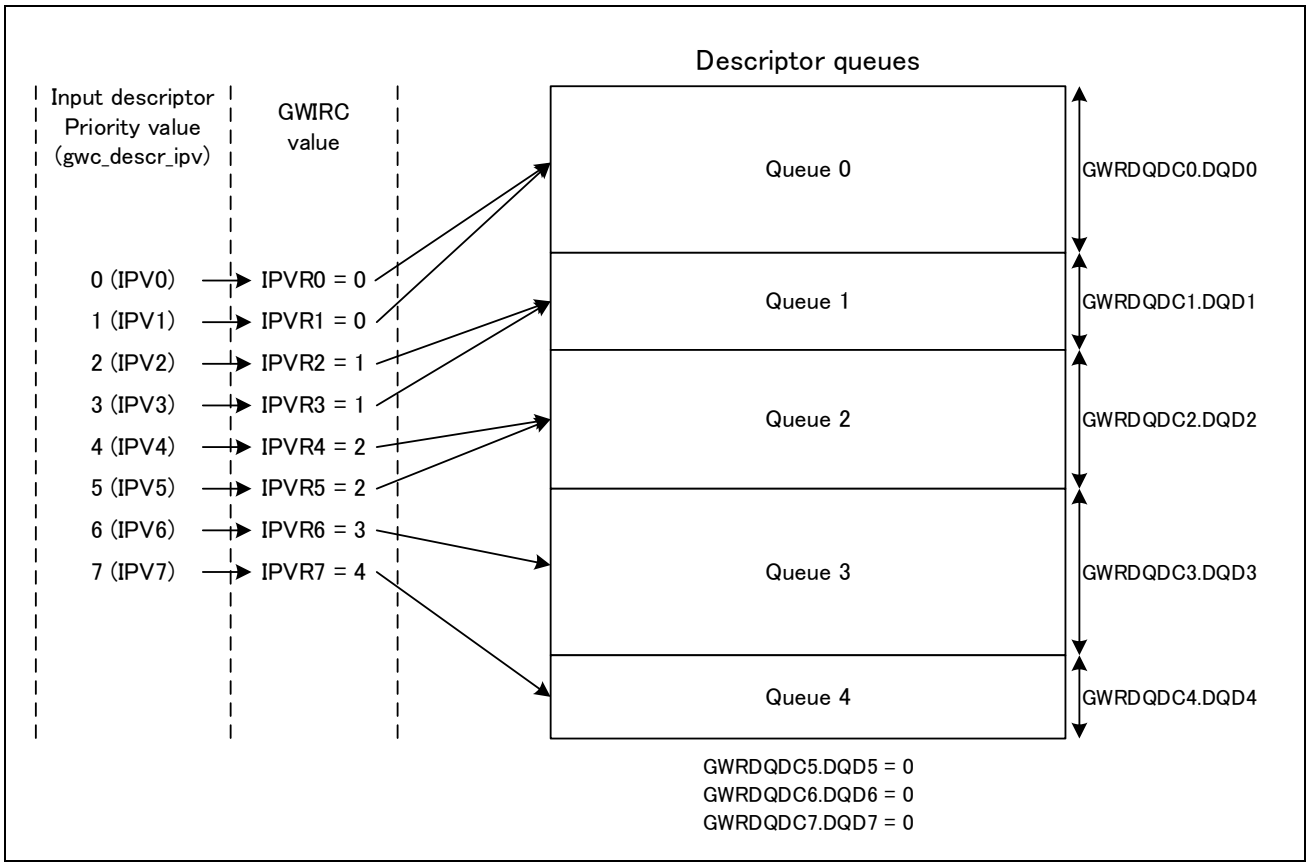


Fig 5.30: Descriptor storage setting example (FRM_PRIO_N == 8)

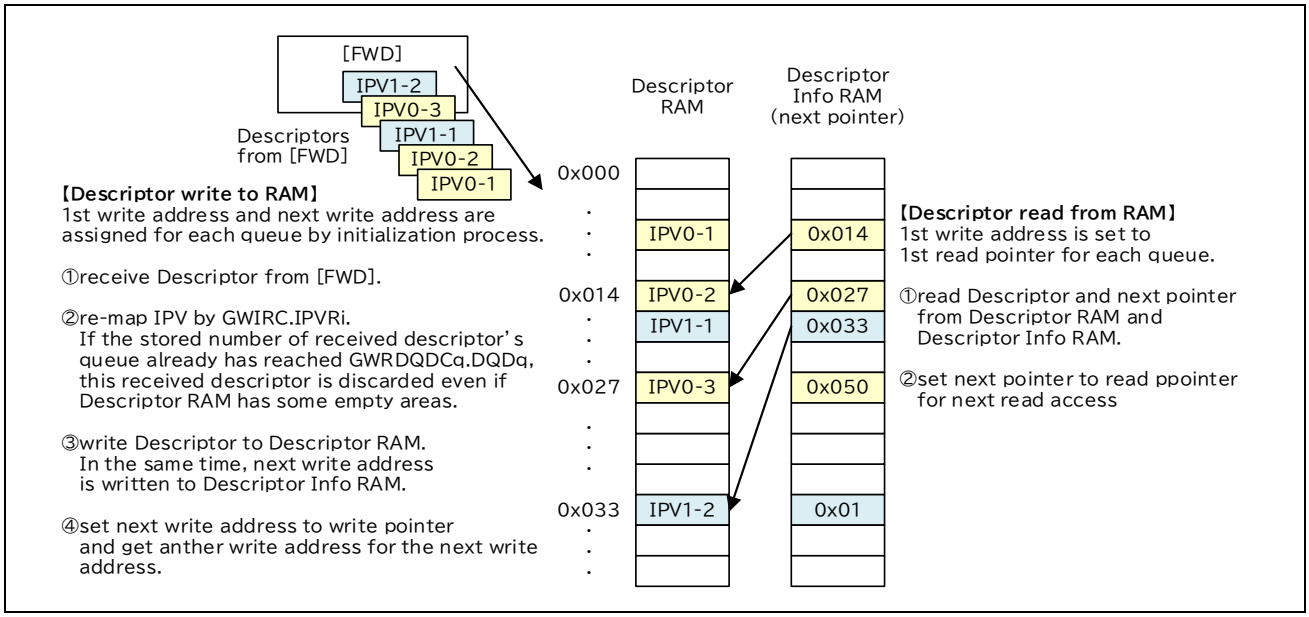


Fig 5.31: Descriptor storage method example (GWRDRC.RDRM=1)

5.3.1.2 Descriptor arbitration

Descriptor arbitration controls the descriptor reading order for reception using **GWRDQAC** register. It has 4 modes. The Strict Priority, the Round Robin, the Weight Round Robin and the Hybrid mode. This function is used to ensure QoS when the AXI access allows a throughput which is too small to allow all the data to be stored in the user RAM.

(a) Strict priority (SP)

To set the descriptor arbitration in strict priority mode, all the **GWRDQAC.RDQAq** (q=0.. FRM_PRIO_N-1) registers should be set to 4'b0. In this case a strict priority algorithm will be applied for descriptor arbitration. The strict priority algorithm is described through an example in Fig 5.32.

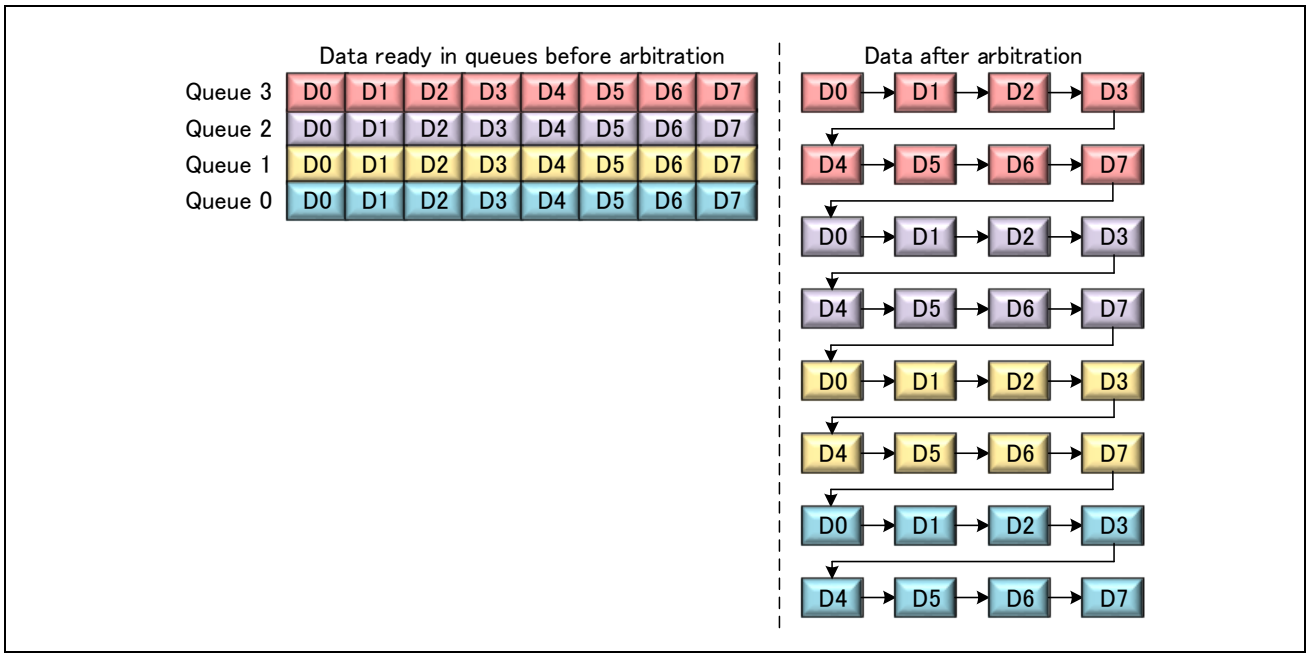


Fig 5.32: Strict priority arbitration example

(b) Round Robin (RR)

To set the descriptor arbitration in round robin mode, all the **GWRDQAC.RDQAq** (q=0.. FRM_PRIO_N-1) registers should be set to 4'b1. In this case a round robin algorithm will be applied for descriptor arbitration. The round robin algorithm is described through an example in Fig 5.33.

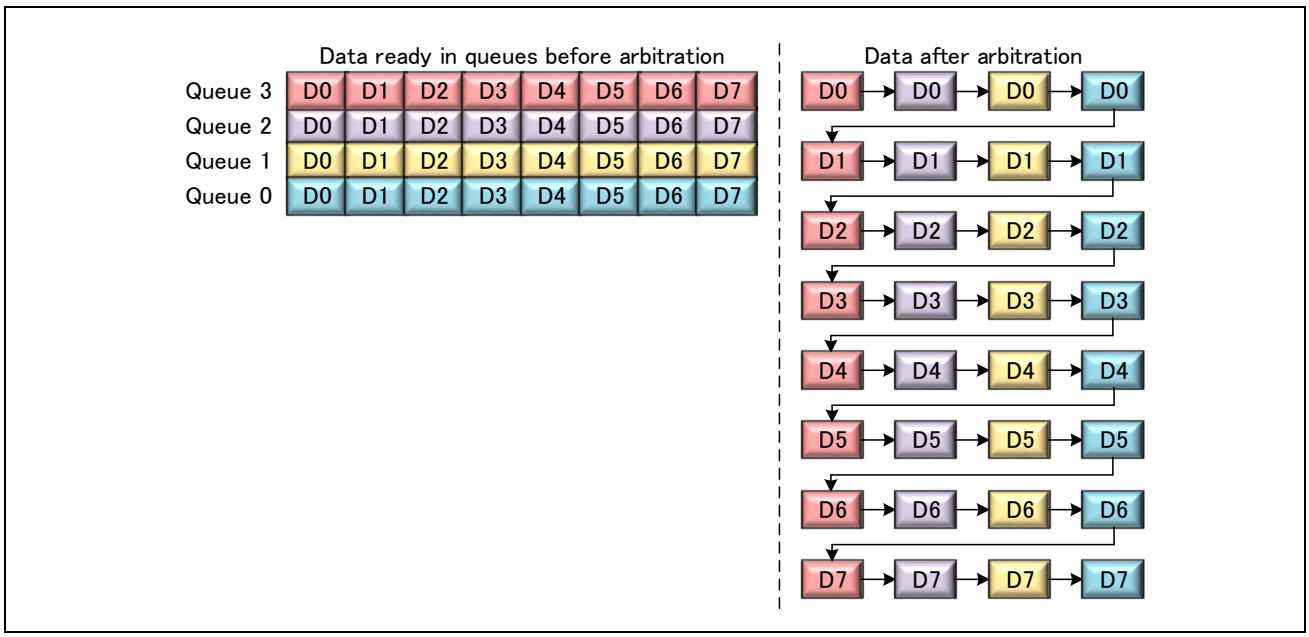


Fig 5.33: Round robin arbitration example

(c) Weight Round Robin (WRR)

To set the descriptor arbitration in weight round robin mode, all the **GWRDQAC.RDQAq** (q=0.. FRM_PRIO_N-1) registers should be set to a value different than 4'd0 with at least one queue set to a value different than 4'd1. In this case a weight round robin algorithm will be applied for descriptor arbitration. The weight round robin algorithm is described through an example in Fig 5.33 (RDQA0 = 2, RDQA1 = 1, RDQA2 = 3 and RDQA3 = 4). In this arbitration **GWRDQAC.RDQAq** register represents the weight of queue q. the heavier a queue is the more chance it will have to transmit data.

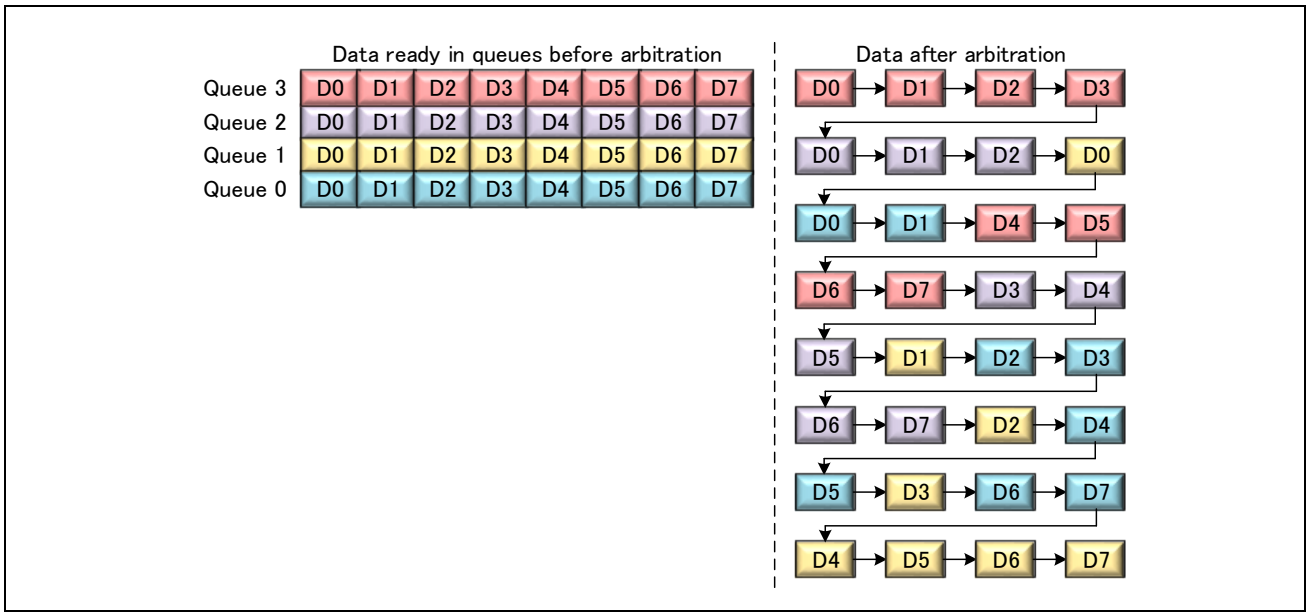


Fig 5.34: Weight round robin arbitration example

(d) Hybrid arbitration

It is allowed to use at the same time SP and RR or SP and WRR. This mode is called the hybrid arbitration mode. To set the descriptor arbitration in hybrid arbitration mode, some of the **GWRDQAC.RDQAq** (q=0.. FRM_PRIO_N-1) registers should be set to 4'd0 and some others to a value different than 4'd0 (For restrictions refer to **GWRDQAC.RDQAq** register explanation). The hybrid arbitration algorithm is described through an example in Fig 5.33 (RDQA0 = 0, RDQA1 = 1, RDQA2 = 2 and RDQA3 = 0).

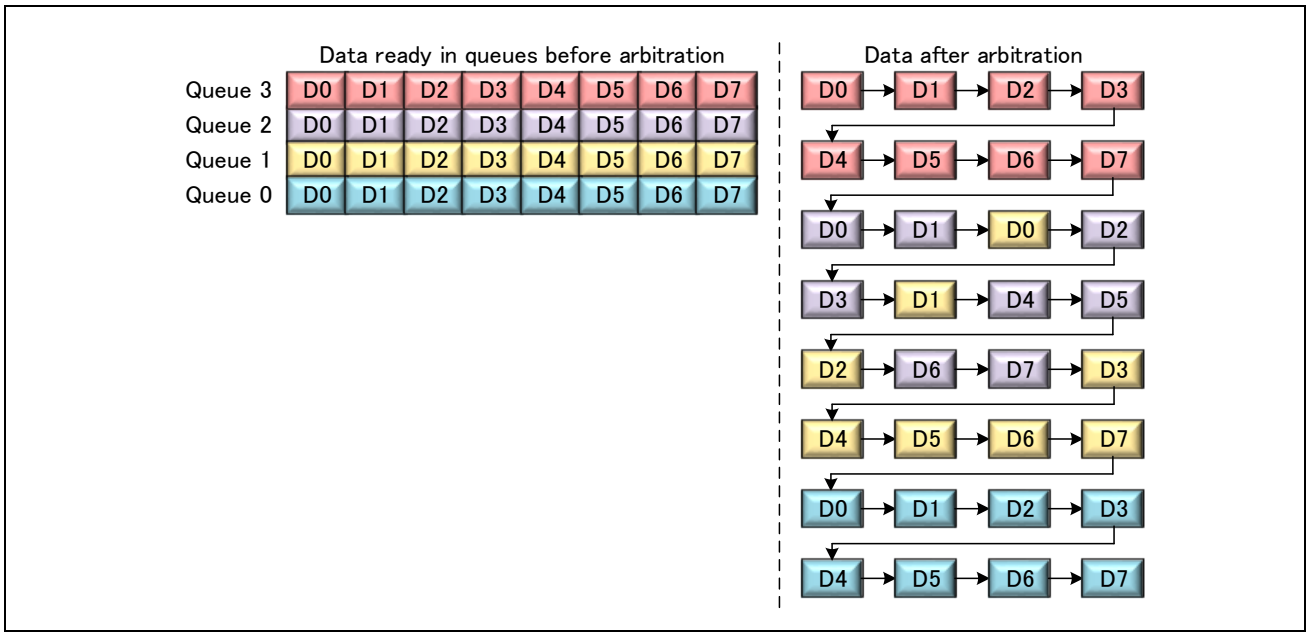


Fig 5.35: Hybrid arbitration example

5.3.2 L2/L3 update

L2/L3 update aims at fetching the update rules thanks to L2/L3 update bus [FWD]. This function can be monitored using **GWEIS0.L23UECCES** interrupt register.

The L2/L3 update rules will be fetched for any descriptor received with **FDESCR.RV** set [FWD]. Routing number **FDESCR.RN** [FWD] will be used for L2/L3 update.

Functions:

- For rule fetching setting details refer to Forwarding Engine specification document [FWD].
- For frame update information refer to RX data fetch block explanation in section 5.3.4.
- **GWEIS0.L23UECCES** aim at monitoring ECC errors in Layer2/Layer3 update table [FWD]. Refer to register explanation for more details.

5.3.3 Multicast and DCNC (Descriptor Chain Number Change) control

DCNC function aims at changing descriptor changing number (CSD) from [MFWD] from by source port. it allows different CPU to process a frames per source port. This feature is processed before the Multicast function.

Multicast function aims at copying the same frame to several AXI descriptor queue. It allows different CPU to process same frames at the same time without interfering with each other. If nothing is set, all the descriptor queues are initialized as unicast queues. Multicast control has four functions. Checking the frame size, security level and queue type (TX or RX), copying the descriptor coming from L2/L3 update using the multicast and DCNC table, modifying the multicast and DCNC table using multicast learning and reading the multicast and DCNC table using multicast searching.

5.3.3.1 Frame check

Frame check aims at checking if a frame has the right properties to be received by CPU using **GWRMFSCq**, **GWDCCi**.**SLi**, **GWDCCi.DQTi** registers and can be monitored using **GWEIS.FSES**, **GWEIS4.DSSES**, **GWEIS4.DSSEIOS**, **GWEIS4.DSSECN**, **GWEIS5.DCTES**, **GWEIS5.DCTEIOS**, **GWEIS5.DCTECN** interrupt registers.

Functions:

- **GWRMFSCq** 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 **GWRMFSCq.MFSq** will be discarded and **GWEIS0.FSES** flag will be set. The frame size considered is the size after VLAN update and with FCS. In this case multicast won't happen and the frame will totally be discarded.
- **GWDCCi**.**SLi** registers are used to set the security level of a CPU sub destination. Any frame received unsecure (**FDESCR.SEC** [FWD] is not set) for a secure CPU sub destination (**GWDCCi**.**SLi** is set) discarded and **GWEIS4.DSSES** flag will be set (during multicast, for all CPU sub destination for which **GWDCCi**.**SLi** is set frame will be discarded and for all CPU sub destination for which **GWDCCi**.**SLi** is not set, frame will be received).
- **GWDCCi.DQTi** is used to control is the received frame is going to a reception descriptor queue. Any frame received for a transmission descriptor queue will be discarded and **GWEIS5.DCTES** flag will be set. In case of multicast, only the targeted transmission queue will see their data lost.
- **GWEIS.FSES** interrupt register signals that a descriptor has been rejected because the corresponding frame where oversized.
- **GWEIS4.DSSES**, **GWEIS4.DSSEIOS** and **GWEIS4.DSSECN** interrupt registers signal that a descriptor has been rejected because of an AXI descriptor queue security error.
- **GWEIS5.DCTES**, **GWEIS5.DCTEIOS**, **GWEIS5.DCTECN** interrupt registers signal that a descriptor has been rejected because of a descriptor chain type error.

5.3.3.2 DCNC (Descriptor Chain Number Change)

Descriptor chain number change aims at changing descriptor to send next (Multicast function) using the multicast and DCNC table.

The (multicast and) DCNC table is divided in three fields, **MUL.DCNCE** which is enabling this function, **MUL.DCNCSPN**[PORT_W-1:0] which is the target source port number and **MUL.DCNCEN**[AXI_CHAIN_W-1:0] which is change to descriptor chain number.

First, find the DCNC table entry number that matches the CSD number. Henceforth, DCNCEN points to the next entry number (and continue). This link will be terminated under any of the following conditions.

- DCNCE == 1'b0
- Source port number == DCNCSPN
- The link exceeds (PORT_N) times. (The maximum number of accesses to this RAM (Entry) is PORT_N times per one descriptor.)

[Restrictions] SW: Stimulus configurations for the Chain remapping need to ensure that it finds either DCNCE 1'b0 or a match termination scenario i.e. source port number matches with the DCNCSPN value (when DCNCE 1'b1).

The (multicast and) DCNC table utilization is described in Fig 5.37 through an example.

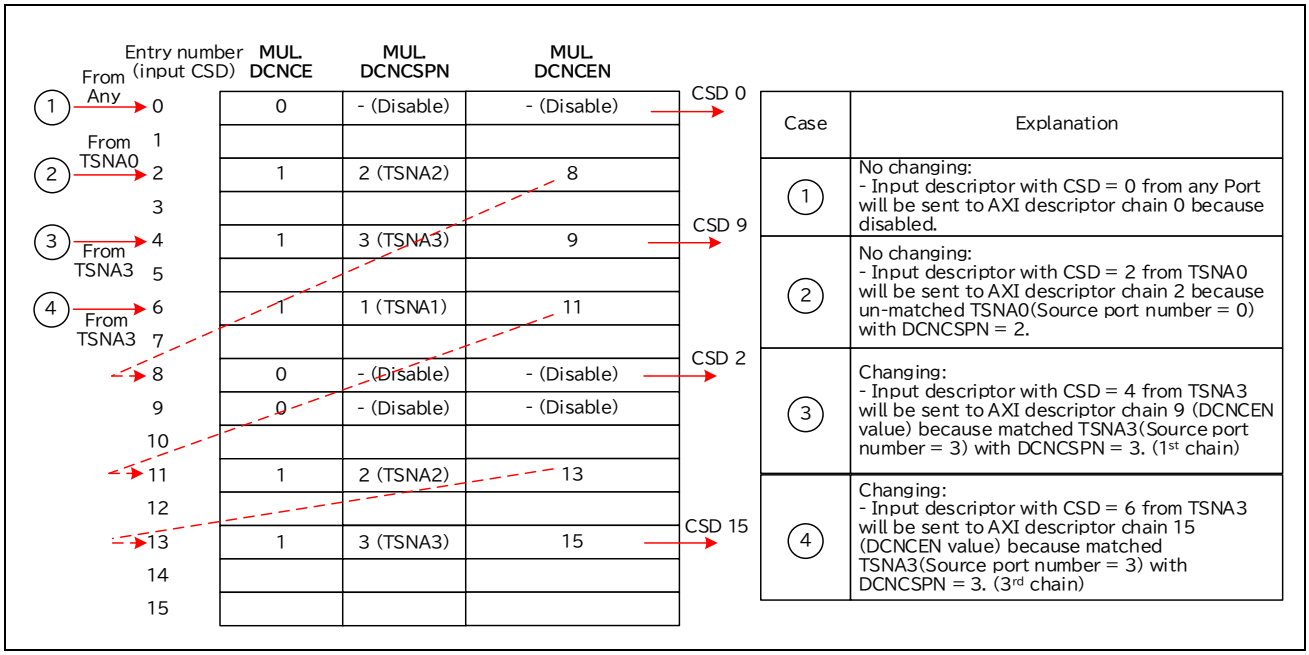


Fig 5.36: (Multicast and) DCNC table utilization example

5.3.3.3 Descriptor copy

Descriptor copy aims at multicasting descriptor to send them to the AXI master interface and the RX data fetch using the multicast (and DCNC) table.

The multicast (and DCNC) table is divided in two fields, **MUL.MN**[2:0] which is the multicast number and **MUL.MNRCN**[AXI_CHAIN_W-1:0] which is the next descriptor chain and is used to link AXI descriptor chains together. AXI descriptor chains linked together form a multicast chain to which a frame will be multicast.
Because the multicast (and DCNC) table is based on links all the multicast patterns are not possible.

The multicast (and DCNC) table utilization is described in Fig 5.37 through an example.

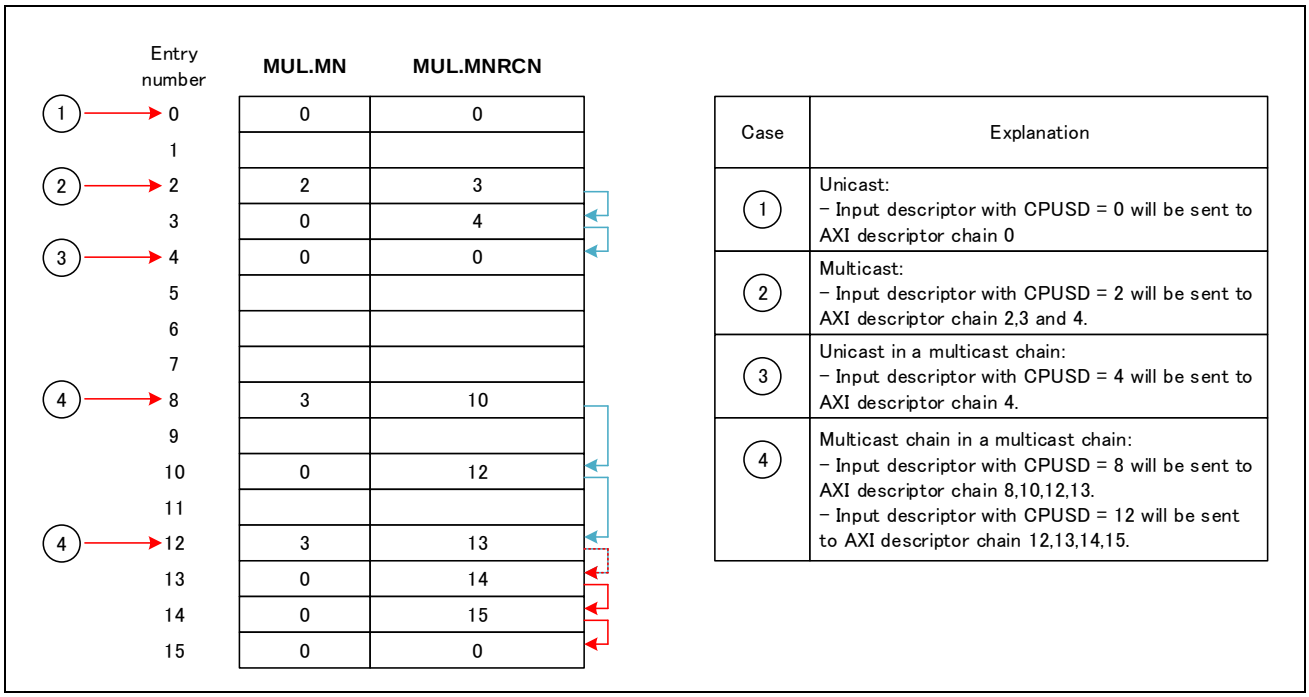


Fig 5.37: Multicast (and DCNC) table utilization example

5.3.3.4 Multicast and DCNC learning

Multicast and DCNC learning aims at updating entries in the multicast and DCNC table using **GWMSTLS** and **GWMSTLR** registers. Table 5-16 describes mapping between learning registers and fields in multicast and DCNC table. Table 5-17 describes the learn results, their mapping with the multicast and DCNC table and how to handle them. It is possible to multicast a frame to 8 AXI descriptor chains.

Functions:

- **GWMSTLS** register is used to set the information needed to overwrite an entry.
- **GWMSTLR** register is used to check if the previously set entry has been overwritten.
- To set entries in the multicast and DCNC table, refer to software flow in section 4.2.7.
- If GWAC.AMP = 1'b1 (the AXI transaction is pending), learning will not complete. Release the AXI master pose.

Table 5-16: Learn registers/Multicast and DCNC table mapping registers

Register name	Field name in L3 table
GWMSTLS.MSENL	Not present in multicast and DCNC table, used to indicate which entry should be overwritten.
GWMSTLS.DCNCENL	**MUL.DCNCEN**, refer to section 5.3.3.2
GWMSTLS.DCNCSPNL	**MUL.DCNCSPN**, refer to section 5.3.3.2
GWMSTLS.DCNCEL	**MUL.DCNCE**, refer to section 5.3.3.2
GWMSTLS.MNL	**MUL.MN**, refer to section 5.3.3.3
GWMSTLS.MNRCNL	**MUL.MNRCN**, refer to section 5.3.3.3

Table 5-17: Learn result

Register name	Field name	Field explanation
GWMSTLR.MTLF	Learning fail	Learning fail will happen in one of the following conditions: - When an APB tries to learn a multicast entry and the multicast and DCNC table is not ready (**GWMTIRM.MTR** is not set).

(1) Static learning

When a system aims at using the multicast and DCNC table in a static way (set in CONFIG mode only) there is no restriction regarding the entry format, the entry link pointers and the learning order of entries.

(2) Dynamic learning

When a system aims at using the multicast and DCNC table in a dynamic way (set in OPERATION mode), the restrictions described in Fig 5.38 should be respected.

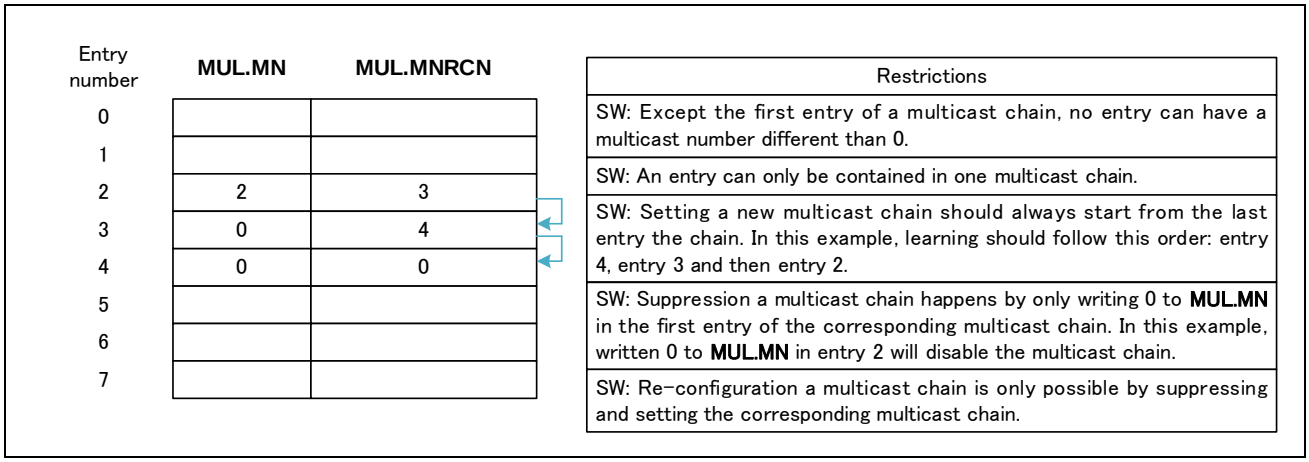


Fig 5.38: Multicast and DCNC table dynamic utilization restrictions

5.3.3.5 Multicast searching

Multicast searching aims at reading entries in the multicast and DCNC table using **GWMSTSS** and **GWMSTSR** registers. Table 5-18 describes mapping between learning registers and fields in multicast and DCNC table. Table 5-19

Functions:

- **GWMSTSS** register is used to set the entry that should be read.
- **GWMSTSR** register is used to read the read result.
- To read an entry in the multicast and DCNC table, refer to software flow in section 4.2.9.

Table 5-18: Search registers/Multicast and DCNC table mapping registers

Register name	Field name in L3 table
GWMSTSS.MSENS	Not present in multicast and DCNC table, used to indicate which entry should be read.

Table 5-19: Search result

Register name	Field name/ corresponding field in multicast and DCNC table	Field explanation
GWMSTSR.DCNCENR	**MUL.DCNCEN**	Multicast and DCNC table read value.
GWMSTSR.DCNCSPNR	**MUL.DCNCSPN**	Multicast and DCNC table read value.
GWMSTSR.DCNCER	**MUL.DCNCE**	Multicast and DCNC table read value.
GWMSTSR.MNR	**MUL.MN**	Multicast and DCNC table read value.
GWMSTSR.MNRCNR	**MUL.MNRCN**	Multicast and DCNC table read value.
GWMSTSR.MTSEF	Searching ECC fail	Leaning failed because of an ECC error. During an ECC error, **MUL.MN** and **MUL.MNRCN** are not valid.

5.3.4 RX data fetch

RX data fetch has follow functions. Fetching the data from the Local RAM, un-tagging/ VLANs depending on VLAN settings, inserting an R-TAG, inserting checksums received by local descriptors, updating the data depending on L2/L3 update information fetched in L2/L3 update, handling FCS, sending the data the AXI master interface and releasing the buffer pointer of the read frames.

The data fetching from Local RAM, the data sending to the AXI master interface and the pointer release are here for switch hardware purpose and its description is not required for switch utilization so it will not be described.

5.3.4.1 VLAN(SC-TAG)

Fetching the data from the Local RAM, updating the data depending on “L23 update” and “TAG function registers”. Updating VLANs data path is described in Fig 5.39: VLAN data path.

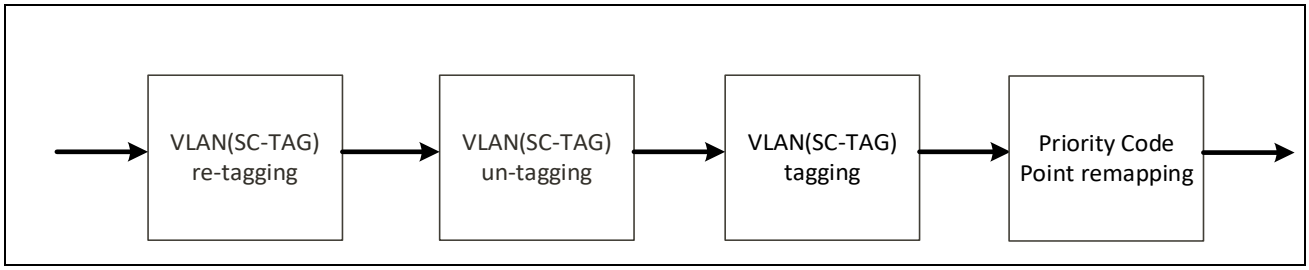


Fig 5.39: VLAN data path

(1) VLAN(SC-TAG) re-tagging

Layer 2 update can happen for the following fields:

- Destination MAC address updated to **L23U.MDA** when **L23U.MDAU** set.
- Source MAC address is updated to {**GWMAC0.MAUP**, **GWMAC1.MADP**} when **L23U.MSAU** set.
- C-TAG VID is updated to **L23U.CVID** when **L23U.CVIDU** set.
- C-TAG PCP is updated to **L23U.CPCP** when **L23U.CPCPU** set.
- C-TAG DEI is updated to **L23U.CDEI** when **L23U.CDEIU** set.
- S-TAG VID is updated to **L23U.SVID** when **L23U.SVIDU** set.
- S-TAG PCP is updated to **L23U.SPCP** when **L23U.SPCPU** set.
- S-TAG DEI is updated to **L23U.SDEI** when **L23U.SDEIU** set.

Restrictions:

- HW: To update the C-TAG/S-TAG fields, the C-TAG/S-TAG should be present in the frame. If not preset, the update will not happen (e.g. L2 update does not change the frame format and does not permit TAG insertion).

(2) VLAN(SC-TAG) un-tagging

VLAN update aims at modifying VLAN frame format depending on switch VLAN mode (**FWGC.SVM** register in forwarding engine [FWD]), the VLAN control information given by the forwarding engine descriptor (FDESCR.VCTRL [FWD]) and **GWVCC.VEM** register.

Functions:

- **GWVCC.VEM** register sets the VLAN reception mode.

The frames format after VLAN un-tagging are described in Fig 5.40, Fig 5.41 and Fig 5.42 Switch VLAN mode (**FWGC.SVM** register in forwarding engine [FWD]) and VLAN control information. The conversion between the frame stored in the Local RAM and the frame format after VLAN un-tagging is described in Fig 5.43.

In description of follow C-TAG/SC-TAG/CoS-TAG are updated value by placed before VLAN(SC-TAG) re-tagging. (CoS-TAG is updated to C-TAG via Layer 2 update.)

	GWVCC.VEM				
VCTRL	3'b000	3'b001	3'b010	3'b011	3'b100
0	No TAG	No TAG	No TAG	No TAG	No TAG
1	No TAG	C-TAG	C-TAG	C-TAG	C-TAG
2	No TAG	C-TAG	C-TAG	SC-TAG	SC-TAG
3	No TAG	CoS-TAG	CoS-TAG	CoS-TAG	CoS-TAG
4	NA	NA	NA	NA	NA
5	NA	NA	NA	NA	NA
6	NA	NA	NA	NA	NA
7	NA	NA	NA	NA	NA

Fig 5.40: Frame format after VLAN un-tagging: Switch in No-VLAN mode [FWD]

	GWVCC.VEM				
VCTRL	3'b000	3'b001	3'b010	3'b011	3'b100
0	No TAG	No TAG	C-TAG	No TAG	C-TAG
1	No TAG	C-TAG	C-TAG	C-TAG	C-TAG
2	No TAG	C-TAG	C-TAG	SC-TAG	SC-TAG
3	No TAG	CoS-TAG	C-TAG	CoS-TAG	C-TAG
4	No TAG	No TAG	C-TAG	No TAG	C-TAG
5	No TAG	No TAG	C-TAG	No TAG	C-TAG
6	No TAG	No TAG	C-TAG	No TAG	C-TAG
7	No TAG	CoS-TAG	C-TAG	CoS-TAG	C-TAG

Fig 5.41: Frame format after VLAN un-tagging: Switch in C-VLAN mode [FWD]

	GWVCC.VEM				
VCTRL	3'b000	3'b001	3'b010	3'b011	3'b100
0	No TAG	No TAG	C-TAG	No TAG	SC-TAG
1	No TAG	C-TAG	C-TAG	C-TAG	SC-TAG
2	No TAG	C-TAG	C-TAG	SC-TAG	SC-TAG
3	No TAG	CoS-TAG	C-TAG	CoS-TAG	SC-TAG
4	No TAG	No TAG	C-TAG	No TAG	SC-TAG
5	No TAG	C-TAG	C-TAG	C-TAG	SC-TAG
6	No TAG	C-TAG	C-TAG	C-TAG	SC-TAG
7	No TAG	CoS-TAG	C-TAG	CoS-TAG	SC-TAG

Fig 5.42: Frame format after VLAN un-tagging: Switch in SC-VLAN mode [FWD]

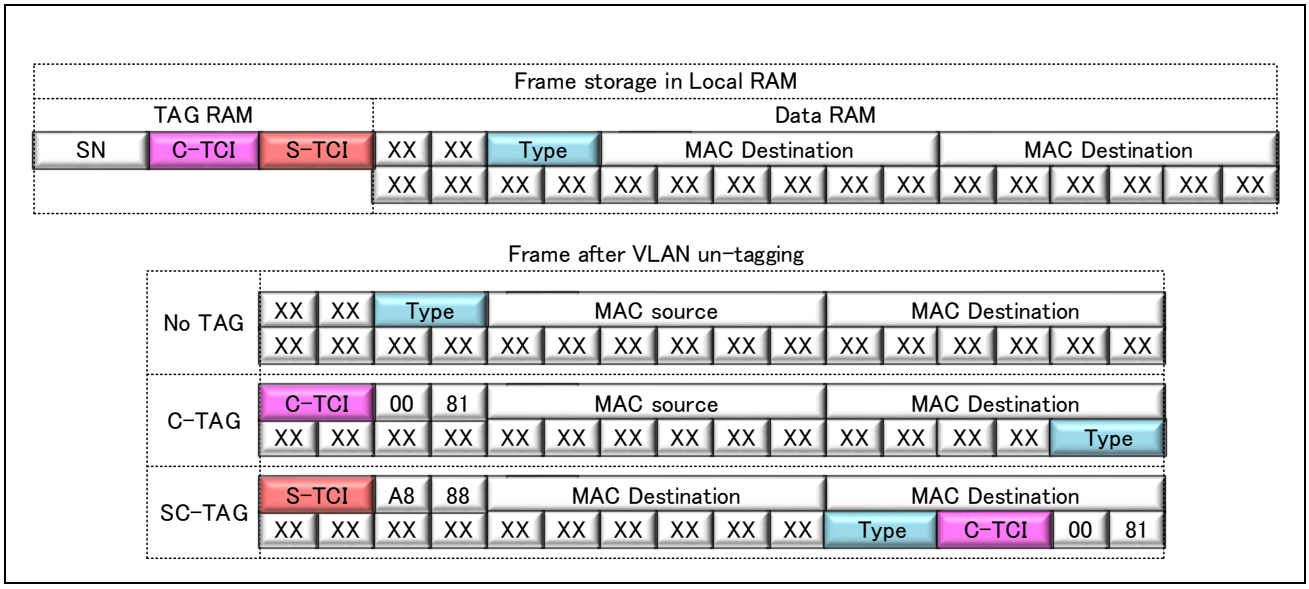


Fig 5.43: Local RAM frame to frame format after VLAN un-tagging conversion

Notes

- The Local RAM frame to frame format after VLAN un-tagging conversion for CoS-TAG is not described because it is the same than for C-TAG except that the VID in the C-TCI will be fixed to 0.
- The R-TAG insertion is not described here because part of the L2/L3 update function, refer to section 5.3.4.4.
- H8100 corresponds to C-TAG TPID, C-TCI corresponds to a C-TAG TCI with a non-null VID, CoS-TCI corresponds to a C-TAG TCI with a null VID and CCs-TCI corresponds to C-TCI or CoS-TCI [802.1Q].
- H88A8 corresponds to S-TAG TPID and S-TCI corresponds to a S-TAG TCI [802.1Q].
- Type corresponds to any Ethernet type value which is not HF1C1, H88A8 or H8100.
- TPID values can be changed in Forwarding Engine [FWD]. Values for TPIDs can be set in forwarding engine (in **FWTTC0** and **FWTTC1** registers).

(3) VLAN(SC-TAG) tagging

VLAN tagging sets present TAGs in a frame using the format given by [MFWD] tagconf input pin after L2 update and VLAN un-tagging.

Functions:

- **[MFWD] FWL23UR.L23UTC** sets the VLAN egress mode.
- **GWVTC** register or L2 update bus (L2/L3 update C-TAG and S-TAG) is used to set the TAGs.
- VLAN tagging does not overwrite present TAGs, adds TAGs only if it does not exist.

The frames format after VLAN tagging are described in Fig 5.44.

	[MFWD] FWL23UR*.L23UTC						
Frame format	3'b000	3'b001	3'b010	3'b011	3'b100	3'b101	3'b110
	no-tagging	L2/L3 update		GWVTC update		un-tagging	
NO-TAG	-	**C-TAG**	**C-TAG**	**C-TAG**	**C-TAG**	-	-
	-	**-**	**S-TAG**	**-**	**S-TAG**	-	-
C-TAG	C-TAG	C-TAG	C-TAG	C-TAG	C-TAG	**-**	C-TAG
	-	-	**S-TAG**	-	**S-TAG**	**-**	**-**
SC-TAG	C-TAG	C-TAG	C-TAG	C-TAG	C-TAG	**-**	C-TAG
	S-TAG	S-TAG	S-TAG	S-TAG	S-TAG	**-**	**-**
CoS-TAG	CoS-TAG	CoS-TAG	CoS-TAG	CoS-TAG	CoS-TAG	**-**	CoS-TAG
	-	-	**S-TAG**	-	**S-TAG**	**-**	-

Fig 5.44: Frame format after VLAN tagging

Notes

- When gwc_l23_tagconf equal to 3'b000, frame format remains the same.
- When gwc_l23_tagconf equal to 3'b001, L2/L3 update adds C-TAG **(L23U.CVID, L23U.CPCP, L23U.CDEI)** only if it does not exist.
- When gwc_l23_tagconf equal to 3'b010, L2/L3 update adds C-TAG **(L23U.CVID, L23U.CPCP, L23U.CDEI)** and S-TAG **(L23U.SVID, L23U.SPCP, L23U.SDEI)** only if it does not exist.
- When gwc_l23_tagconf equal to 3'b011, **GWVTC** register adds C-TAG only if it does not exist.
- When gwc_l23_tagconf equal to 3'b100, **GWVTC** register adds C-TAG and S-TAG only if it does not exist.
- When gwc_l23_tagconf equal to 3'b101, frame format changes No-TAG mode.
- When gwc_l23_tagconf equal to 3'b110, only S-TAG is removed.
- When gwc_l23_tagconf equal to 3'b111, frame format remains the same.

(4) Priority Code Point remapping

This function is recognized as disable by default values.

This function can replace PCP and DEI for TAGs int the egress frames. This function is processed last after all subsequent frame conversions.

Functions:

- The PCP and DEI in C-TAG is replaced by the value in the corresponding register **GWECD0/1RC**. When DEI = 1, **GWECD1RC** register replaces the PCP and DEI in C-TAG. When DEI = 0, **GWECD0RC** register replaces the PCP and DEI in C-TAG. For example, by setting {**GWECD0RC.ECD0PR6**, **GWECD0RC**.**ECD0DR6}** to {2, 1}, the C-TAG frame of {PCP,DEI}=={6, 0} will be remapped to {PCP, DEI}=={2, 1}.
- The PCP and DEI in S-TAG is replaced by the value in the corresponding register **GWESD0/1RC**. When DEI = 1, **GWESD1RC** register replaces the PCP and DEI in S-TAG. When DEI = 0, **GWESD0RC** register replaces the PCP and DEI in S-TAG. For example, by setting {**GWESD0RC.ESD0PR6**, **GWESD0RC**.**ESD0DR6}** to {2, 1}, the S-TAG frame of {PCP,DEI}=={6, 0} will be remapped to {PCP, DEI}=={2, 1}.

5.3.4.2 R-TAG insertion

The R-TAG insertion happens for the two following cases:

- For frames received with an R-TAG (**LDESCR.RTGI** was set during reception, refer to section 5.2.3.5, so, in descriptor received for forwarding, **FDESCR.RTGI** was also set) and for which routing is not requested (**FDESCR.RV** was not set in received descriptor [FWD]) or for which routing is requested but the R-TAG stripping is not request by routing (**L23U.RTU** [FWD] was not set to 2'10 in routing number **FDESCR.RN** [FWD] during routing rule fetching by L2/L3 update block, refer to section 5.3.2).
- For frames received without an R-TAG and for which routing and the R-TAG insertion is requested(**L23U.RTU** [FWD] was set to 2'01 in routing number **FDESCR.RN** [FWD] during routing rule fetching by L2/L3 update block, refer to section 5.3.2).

In both cases, the R-TAG sequence number (SN in Fig 5.45) will be set to the sequence number **FDESCR.SEQN** [FWD] received in received descriptor.

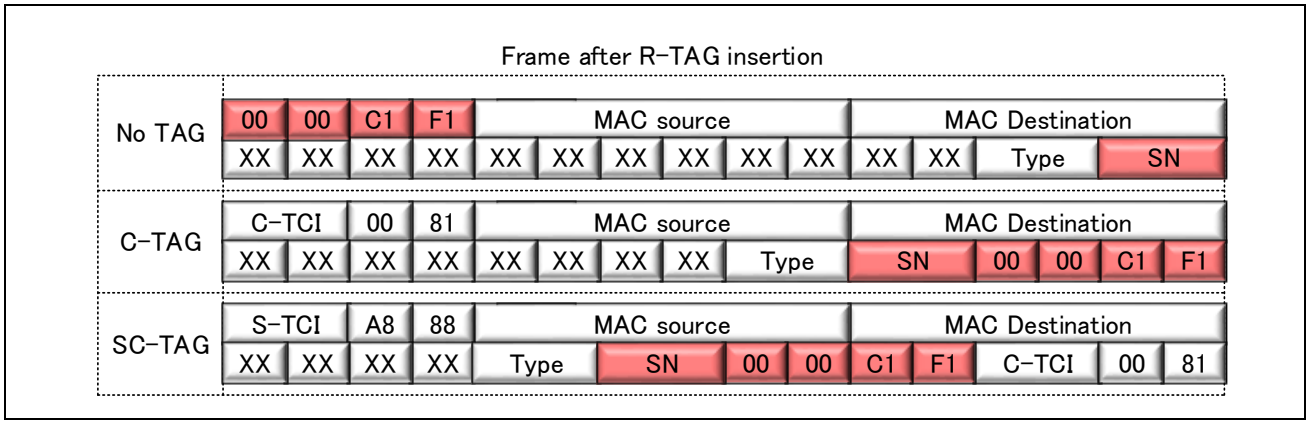


Fig 5.45: R-TAG inserted frame format

Notes:

- HF1C1 corresponds to R-TAG TPID and SN correspond to R-TAG sequence number[802.1CB].
- H8100 corresponds to C-TAG TPID, C-TCI corresponds to a C-TAG TCI with a non-null VID, CoS-TCI corresponds to a C-TAG TCI with a null VID and CCs-TCI corresponds to C-TCI or CoS-TCI [802.1Q].
- H88A8 corresponds to S-TAG TPID and S-TCI corresponds to a S-TAG TCI [802.1Q].
- Type corresponds to any Ethernet type value which is not HF1C1, H88A8 or H8100.
- TPID values can be changed in Forwarding Engine [FWD]. Values for TPIDs can be set in forwarding engine (in **FWTTC0** and **FWTTC1** registers).

5.3.4.3 Checksum insertion

Checksum insertion aims at inserting checksums calculated at ingress port (refer to section 5.2.3.4) and stored inside the local descriptor in the egress frame.

Check sum insertion can happen for the following fields:

- Layer 3 checksum is updated to **LDESCR.L3CKS** when a direct local descriptor (**LDESCR.FMT** is set) is received with **LDESCR.L3CC** bit set. Layer 3 checksum includes IPv4 header checksum.
- Layer 4 checksum is updated to **LDESCR.L4CKS** when a direct local descriptor (**LDESCR.FMT** is set) is received with **LDESCR.L4CC** bit set. Layer 4 checksum includes IPv4/IPv6 frames TCP header checksum, IPv4/IPv6 frames UDP header checksum and IPv4/IPv6 frames ICMP header checksum.

5.3.4.4 L2/L3 update

L2/L3 updates is used to the layer 2 and the layer 3 of frames. L2/L3 update happens for frames for which routing is requested (received descriptor with **FDESCR.RV** set [FWD]) and depending on the update information fetched by the L2/L3 update block (refer to section 5.3.2).

(1) L2 update

Refer to section 5.3.4.1(1).

(2) L3 update

Layer 3 updates happens when **L23U.TTLU** was set. L3 update will happen in two steps, the frame decoding and the L3 update.

The decoder decodes the type field (refer to section 5.2.3.1) of the Layer 3 update frame and separate frames in the following formats:

- IPv4: When type field is equal to H0800.
- IPv6: When type field is equal to H86DD.
- Unknown: When type field is different than H0800 and H86DD.

Depending on the decoding result, the L3 update will happen has following:

- For IPv4: TimeToLive will be decremented by 1 if it is different than 0 and HeaderChecksum will be corrected using a relative equation (If the checksum were wrong in the input frame, the checksum will also be wrong for the output L3 updated frame).
- For IPv6: HopLimit will be decremented by 1 if it is different than 0.
- For Unknown: L3 update will be ignored.

5.3.4.5 FCS handling

Ethernet frame FCS can be forwarded to the CPU using **GWRGC.RCPT** register.

Functions:

- **GWRGC.RCPT** is used to pass frame FCSs to the CPU.

However, following conditions should be met for GWCA to forward a frame FCS to the CPU:

- For Ethernet agent [TSN], RMAC shouldn't have removed the FCS [RMAC] at the input port.
- For GWCA the FCS should be present in the ingress frame and valid (FI set and FW not set in input descriptor, refer to section 5.2.2.3).
- The TAGs should be the same at the egress than at the ingress port.
  This condition will be affected by "VLAN replacement by HW-TAG (by re-tagging only)" or "PCP/DEI remapping". If these features are enabled, even the cases "re-tagging HW TAGs same value in ingress/egress port", "mask the re-tagging HW TAGs in ingress port" or "Priority Code Point Remapping is enabled but incoming PCP/DEI will be remapped same value in ingress/egress port" will **not** be recognized as "same" on egress port.
- The frame shouldn't be a routed frame (received descriptor **FDESCR.RV** field not set [FWD]).

Notes:

- If one of the above conditions is not met by a frame, the FCS will be removed from it even if **GWRGC.RCPT** register is set to 1'b1.
- If a frame is received by CPU with an FCS, FI flag will be set in the reception descriptor. If not, it will not be set.

5.3.5 AXI master interface

The AXI master interface handles the data/AXI descriptor exchange with the CPU. It will receive the updated data and the information related to the data for RX data fetch block and store them in the CPU RAM. To do so, it uses **GWDCCi.EDEi**, **GWDCCi.ETSi**, **GWDCCi.SMi**, and can be monitored using **GWAARSS**, **GWAARSR0** and **GWAARSR1**, **GWIDAUASi**, **GWIDASMi**, **GWIDASAM0i**, **GWIDASAM1i**, **GWIDACAM0i**, **GWIDACAM1i** registers and **GWEIS0.AECCES**, **GWEIS2i.DFESt**, **GWEIS3.IAOESi**, **GWEIS4.DSES**, **GWEIS4.DSEIOS**, **GWEIS4.DSECN** interrupt registers.

Functions:

- **GWDCCi.EDEi** and **GWDCCi.ETSi** registers set the descriptor format that will be used for descriptor queue number i refer to section 5.1.2.1.
- **GWDCCi.SMi** register sets how GWCA will write back descriptors. In this section all explanation will use **GWDCCi.SMi** equal to 2'b00. For **GWDCCi.SMi** equal to 2'b01 or 2'b10 refer to the register explanation.
- For **GWAARSS**, **GWAARSR0** and **GWAARSR1** registers refer to section 5.1.6.
- **GWIDAUASi**, **GWIDASMi**, **GWIDASAM0i**, **GWIDASAM1i**, **GWIDACAM0i**, **GWIDACAM1i** registers and **GWEIS3.IAOESi** interrupt register are used for incremental area monitoring, refer to section 5.3.5.3.
- **GWEIS0.AECCES** interrupt register should that some frames may have been lost because of and AXI address RAM ECC error.
- **GWEIS2i.DFESt** interrupt register signals that frames have been lost because a full descriptor queue.
- **GWEIS4.DSES**, **GWEIS4.DSEIOS**, **GWEIS4.DSECN** interrupt registers signal that frames have been received with an unexpected size, refer to 5.3.5.2.

Data reception can be done using linear or cyclic descriptor queues (refer to section 5.1.5). Data reception process descriptors can be set six different ways, through the basic data reception descriptor chain (section 5.3.5.1), the Size-controlled data reception descriptor chain (section 5.3.5.2), the Single page incremental data reception descriptor chain (section 5.3.5.3), the Interrupt based multi-page incremental data reception descriptor chain (section 5.3.5.4), the Header removal incremental data reception descriptor chain (section 5.3.5.5).

5.3.5.1 Basic data reception

Basic data reception process descriptor utilization is described in Fig 5.46. Basic data reception process descriptor format is described in Fig 5.47 and Table 5-20.

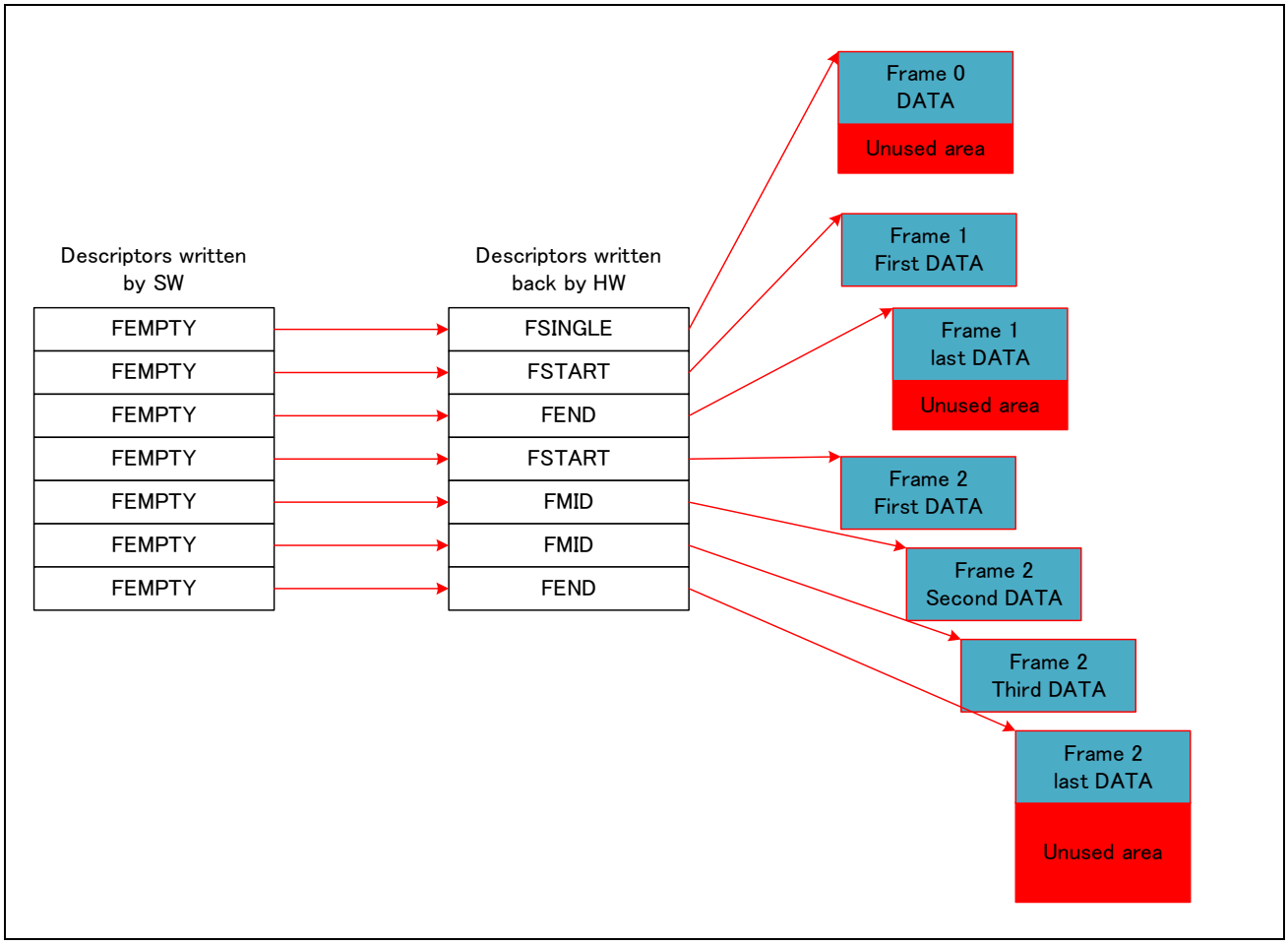


Fig 5.46: Basic data reception process descriptor utilization

Note:

- If a frame is too big to fit in an area reserved by one descriptor, it will be automatically split. The first descriptor will be FSTART, the last descriptor will be FEND and all other descriptors will be FMID.
- When a frame is split the hardware will write back the FSTART descriptor last. Even if the queue is set in keep DT mode, this order is kept.

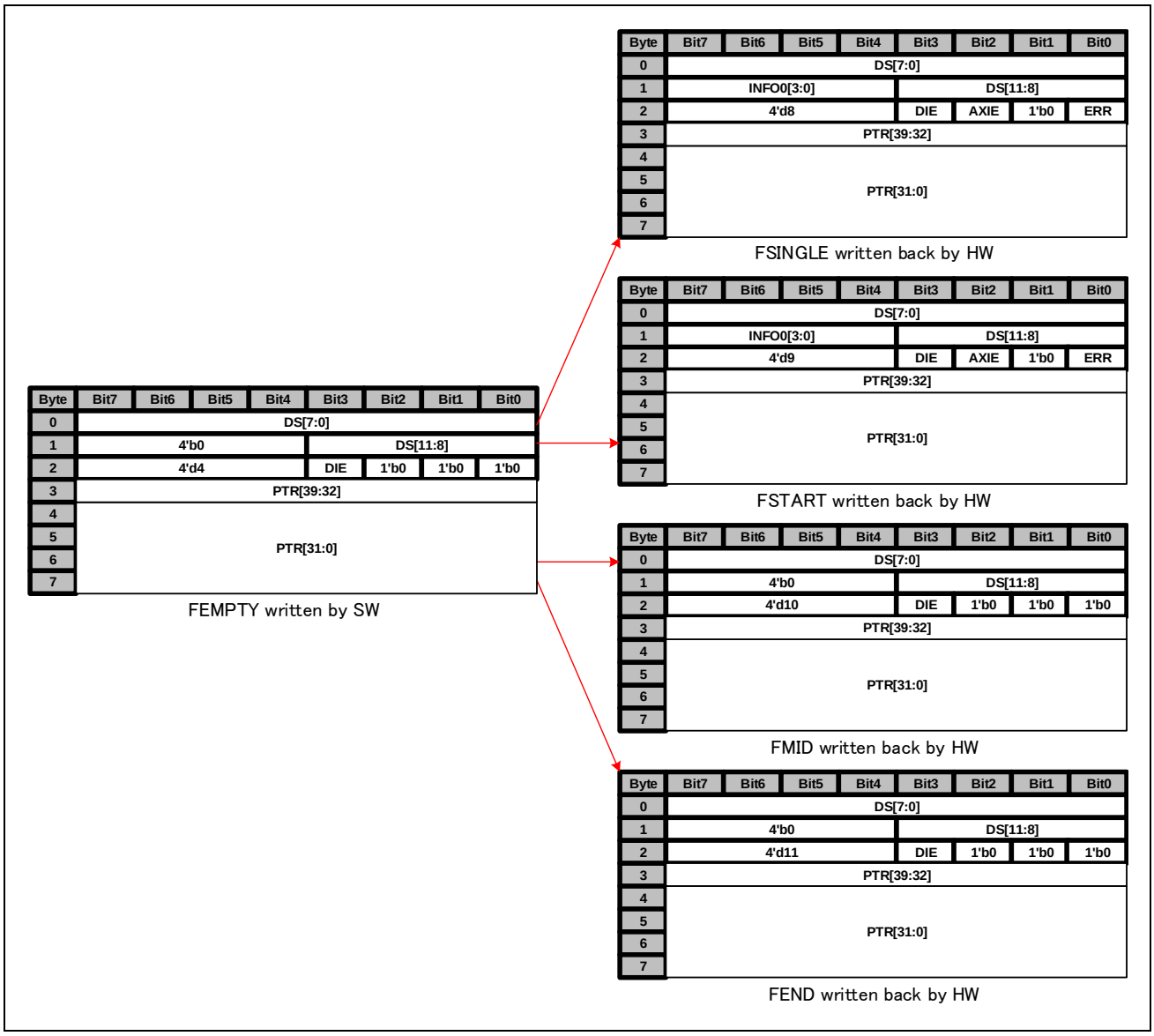


Fig 5.47: Basic data reception process descriptor utilization format (for basic descriptor, section 5.1.2.1)

Table 5-20: Basic data reception process descriptor utilization field description

Field name	Field description before write-back	Field description after write-back
DS	Size of available URAM memory associated to the pointer for data storage	Actual size that has been written by the AXIBMI in the URAM.
INFO0	4'b0	4'b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.

Field name	Field description before write-back	Field description after write-back
ERR	1'b0	Set conditions: - HW: An ECC error happened while reading the data from Local RAM and a FSINGLE or a FSTART descriptor written back. - HW: An AXI address ECC error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. - HW: A descriptor queue full error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. - HW: A descriptor number error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame will not be written) and a FSINGLE or a FSTART descriptor written back.
DSE	1'b0	1'b0
AXIE	1'b0	Set conditions: - HW: An AXI error happened while writing a data (BRESP != 2'b00) and a FSINGLE or a FSTART descriptor written back. - HW: An AXI error happened while reading a descriptor during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back.
DIE	DIE	DIE (Value is copied from read descriptor).
DT	4'd4 for FEMPTY	4'd8 for FSINGLE 4'd9 for FSTART 4'd10 for FMID 4'd11 for FEND
PTR	Address where data should be written in the URAM	PTR (Value is copied from read descriptor).
INFO1 (GWDCCi.EDEi == 1'b1)	64'b0	64'b0 for FEND and FMID, For FSTART and FSINGLE, refer to section 5.3.5.6.
TS (GWDCCi.ETSi == 1'b1)	64'b0	64'b0 for FEND and FMID, For FSTART and FSINGLE refer to section 5.3.5.6.

5.3.5.2 Size-controlled data reception

Size-controlled data reception aims at controlling the frame size at reception while purposely splitting the header and the payload of a frame. By size-controlled data reception the SW ensure that descriptor order is maintained even if a too short or too long frame is received. Size-controlled data reception process descriptor utilization is described in Fig 5.48. Size-controlled data reception process descriptor format is described Fig 5.49 in and Table 5-21. Size-controlled data reception can be monitored using **GWEIS4.DSES**, **GWEIS4.DSEIOS**, **GWEIS4.DSECN** interrupt registers.

Function:

- **GWEIS4.DSES**, **GWEIS4.DSEIOS**, **GWEIS4.DSECN** interrupt registers signal a frame received with un unexpected size.

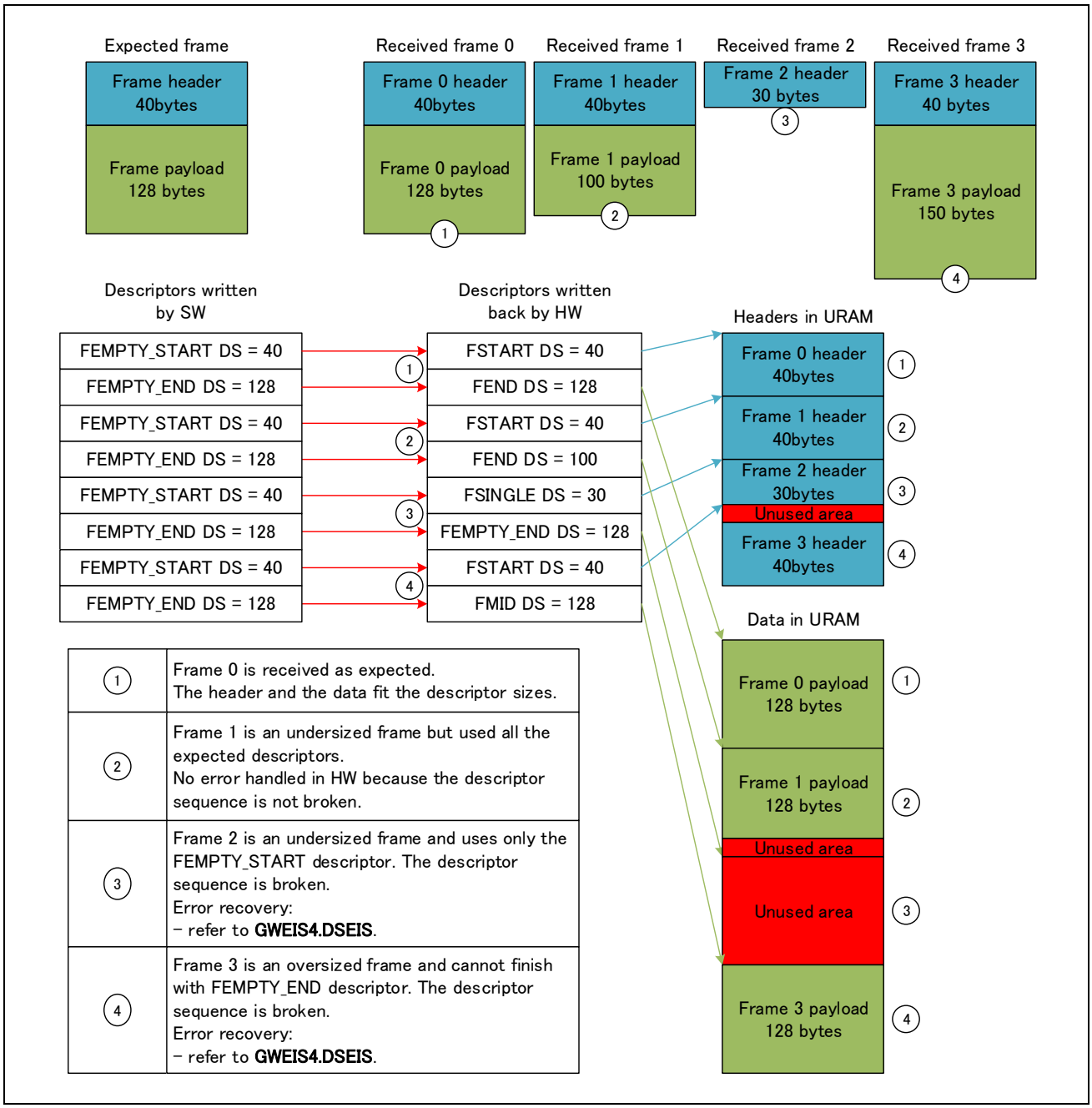


Fig 5.48: Size-controlled data reception process descriptor utilization

Notes:

- When a frame is split the hardware will write back the FSTART descriptor last. Even if the queue is set in keep DT mode, this order is kept.
- A frame can be split in more than two descriptors by adding FEMPTY_MID descriptors between FEMPTY_START and FEMPTY_END descriptors.

Restrictions:

- SW: The process descriptor queue should always repeat the pattern of the same descriptors from the first descriptor to the descriptor before the termination descriptor. This pattern should start with an FEMPTY_START descriptor, finish by a FEMPTY_END descriptor and in between be only composed by FEMPTY_MID descriptors. If this is violated (for example, FEMPTY_START is detected in the middle), the frame will be discarded/rejected without notification.

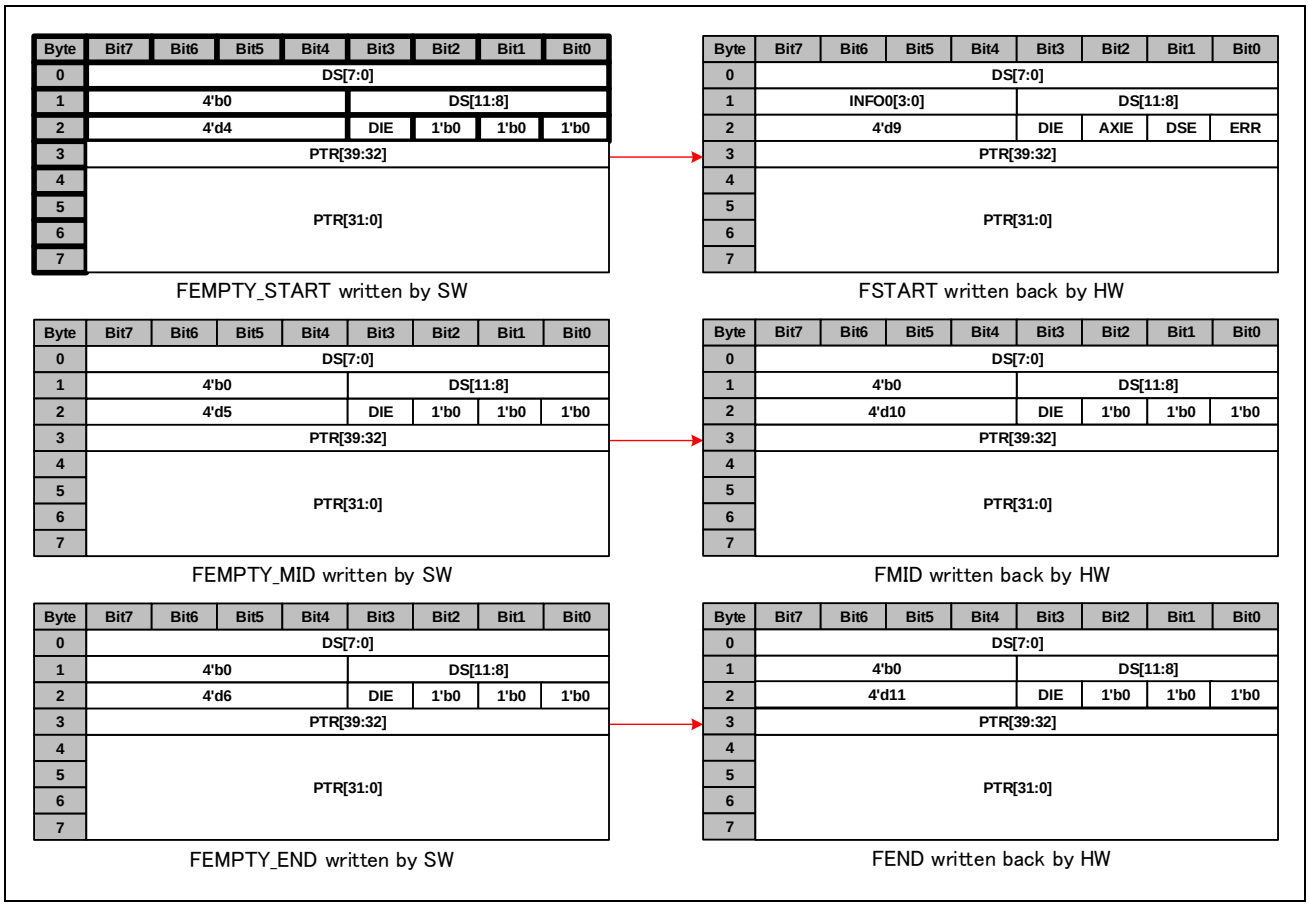


Fig 5.49: Size-controlled data reception process descriptor utilization format

(for basic descriptor, section 5.1.2.1)

Table 5-21: Size-controlled data reception process descriptor utilization field description

Field name	Field description before write-back	Field description after write-back
DS	Size of available URAM memory associated to the pointer for data storage	Actual size that has been written by the AXIBMI in the URAM.
INFO0	4'b0	4'b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.

Field name	Field description before write-back	Field description after write-back
ERR	1'b0	Set condition: - HW: An ECC error happened while reading the data from Local RAM and a FSINGLE or a FSTART descriptor written back. - HW: An AXI address ECC error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. - HW: A descriptor queue full error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. (This case shouldn't happen due to the process descriptor order) - HW: A descriptor number error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame will not be written) and a FSINGLE or a FSTART descriptor written back.
DSE	1'b0	Set conditions: - HW: The frame has been finished with a FEMPTY_START or a FEMPTY_MID descriptor. (undersized frame) - HW: The frame is too big to finish with a FEMPTY_END descriptor. (oversized frame)
AXIE	1'b0	Set conditions: - HW: An AXI error happened while writing a data (BRESP != 2'b00) and a FSINGLE or a FSTART descriptor written back. - HW: An AXI error happened while reading a descriptor during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back.
DIE	DIE	DIE (Value is copied from read descriptor).
DT	4'd5 for FEMPTY_START 4'd6 for FEMPTY_MID 4'd7 for FEMPTY_END	4'd9 for FSTART 4'd10 for FMID 4'd11 for FEND
PTR	Address where data should be written in the URAM	PTR (Value is copied from read descriptor).
INFO1 (GWDCCi.EDEi == 1'b1)	64'b0	64'b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.
TS (GWDCCi.ETSi == 1'b1)	64'b0	64'b0 for FEND and FMID, For FSTART and FSINGLE refer to section 5.3.5.6.

5.3.5.3 Single page incremental data reception

Single page incremental data reception aims at saving the received frames back to back in a given memory area. By single page incremental data reception, the SW avoid wasting CPU power to stack data together. Single page incremental data reception process descriptor utilization is described in Fig 5.50. Single page incremental data reception process descriptor format is described Fig 5.51, Table 5-22, Fig 5.52 and Table 5-23. Single page incremental data reception HW/SW synchronization should be done by **GWIDAUASi** register and can be monitored by **GWIDASMi**, **GWIDASAM0i**, **GWIDASAM1i**, **GWIDACAM0i**, **GWIDACAM1i** registers and **GWEIS3.IAOESi** interrupt register.

Functions:

- **GWIDAUASi** register shows the number of bytes that has been written in the incremental area. After reading the data from the CPU RAM, SW should write to this register the quantity of data that has been read. The CPU data handling can be done, either based on the written back descriptor in the corresponding descriptor queue, or on the byte value contained in the increment area than can be read in **GWIDAUASi** register.
- **GWIDASMi** register is used to monitor the total size of the incremental area.
- {**GWIDASAM0i.IDASAUPi**, **GWIDASAM1i.IDASADPi**} register array is used to monitor the start address of the incremental area.
- {**GWIDACAM0i.IDACAUPi**, **GWIDACAM1i.IDACADPi**} register array is used to monitor the next used address in the incremental area.
- **GWEIS3.IAOESi** interrupt register signals data loss due to incremental area overflow.

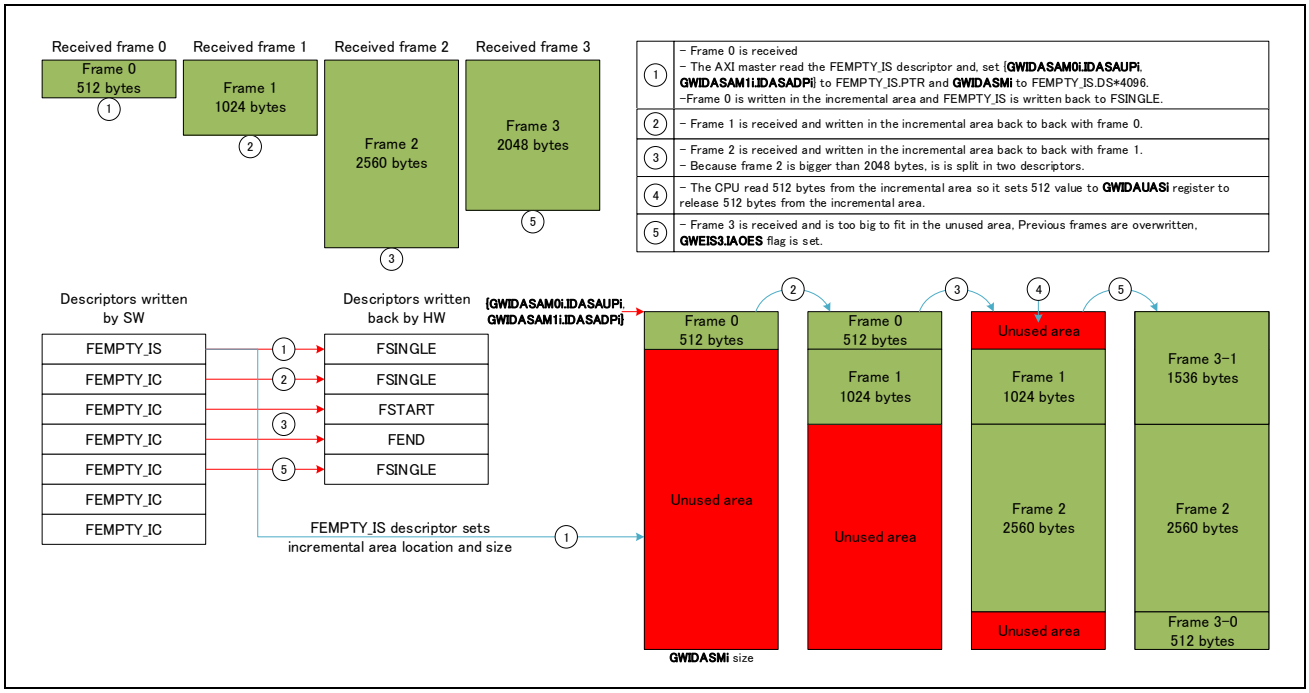


Fig 5.50: Single page incremental data reception process descriptor utilization

(for basic descriptor, section 5.1.2.1)

Restrictions:

- SW: This reception type can only be used for the first AXI_RINC_N RX chains, refer to section 5.1.1.

Notes:

- If a frame is too big to fit in an area reserved by one descriptor, it will be automatically split. The first descriptor will be FSTART, the last descriptor will be FEND and all other descriptors will be FMID.
- When a frame is split the hardware will write back the FSTART descriptor last. Even if the queue is set in keep DT mode, this order is kept.
- When GWCA reset has been applied (GWMS.OPS set to 2'b00), all the incremental registers (i.e incremental page start address, current address, page size and used page size) were cleared. Therefore, when the GWCA has moved back again to OPERATION mode (GWMS.OPS set to 2'b11) and a frame has been received by GWCA incremental chain, it's not be able to write frame using FEMPTY_IC descriptor. Descriptor of AXI current address have to be(set up) a FEMPTY_IS.

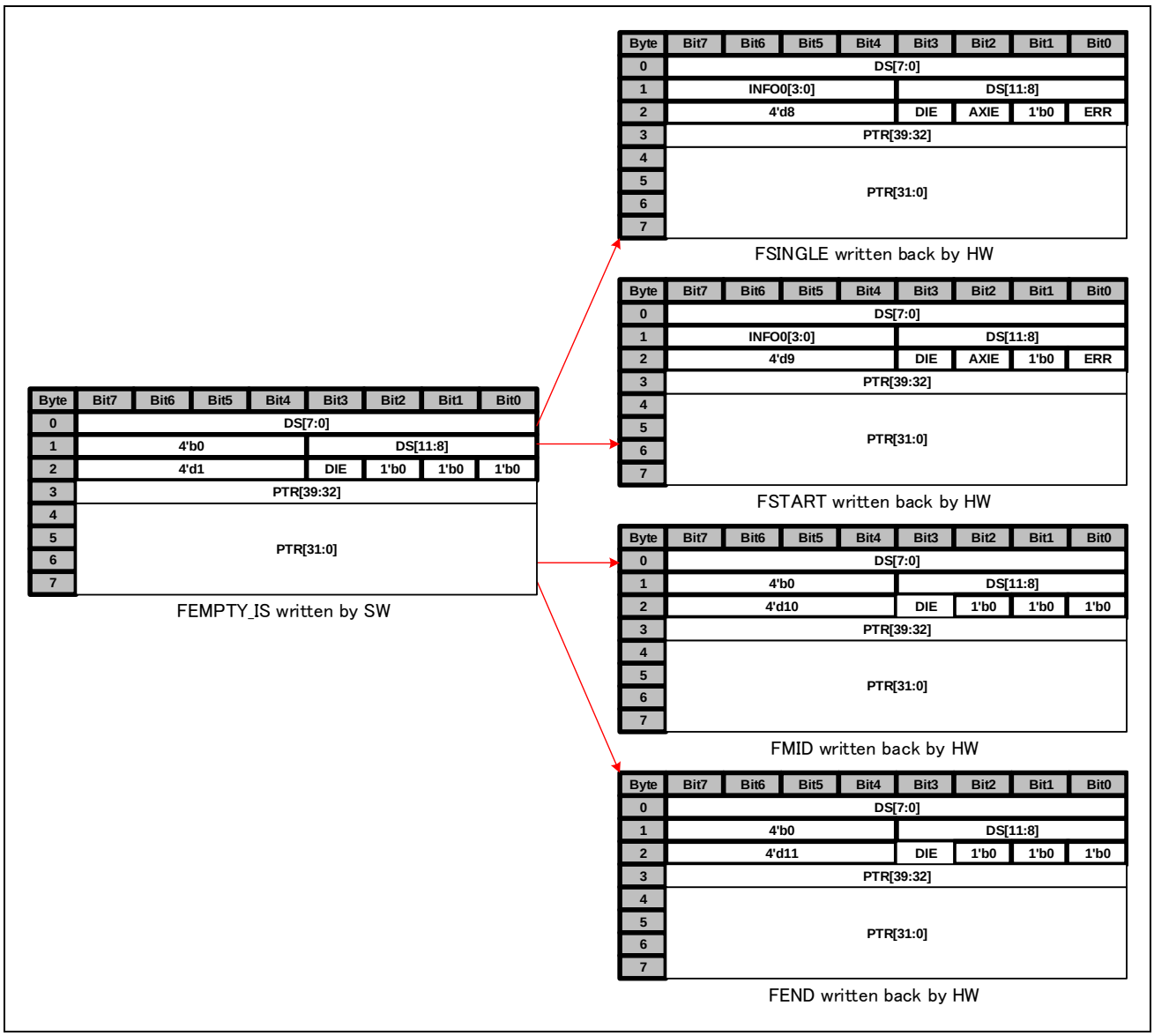


Fig 5.51: Single page incremental data reception FEMPTY_IS process descriptor utilization format

Table 5-22: Single page incremental data reception FEMPTY_IS process descriptor utilization field description

Field name	Field description before write-back	Field description after write-back
DS	Size of the incremental area in 4Kbytes unity.	Size of the data stored in the URAM associated tot the descriptor.
INFO0	4'b0	4'b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.
ERR	1'b0	Set conditions: - HW: An ECC error happened while reading the data from Local RAM and a FSINGLE or a FSTART descriptor written back. - HW: An AXI address ECC error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. - HW: A descriptor queue full error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. - HW: A descriptor number error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame will not be written) and a FSINGLE or a FSTART descriptor written back.
DSE	1'b0	1'b0
AXIE	1'b0	Set conditions: - HW: An AXI error happened while writing a data (BRESP != 2'b00) and a FSINGLE or a FSTART descriptor written back. - HW: An AXI error happened while reading a descriptor during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back.
DIE	DIE	DIE (Value is copied from read descriptor).
DT	4'd1 for FEMPTY_IS	4'd8 for FSINGLE 4'd9 for FSTART 4'd10 for FMID 4'd11 for FEND
PTR	Address where the incremental area starts in the URAM.	PTR (Value is copied from read descriptor).
INFO1 (GWDCCi.EDEi == 1'b1)	64'b0	64'b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.
TS (GWDCCi.ETSi == 1'b1)	64'b0	64'b0 for FEND and FMID, For FSTART and FSINGLE, refer to section 5.3.5.6.

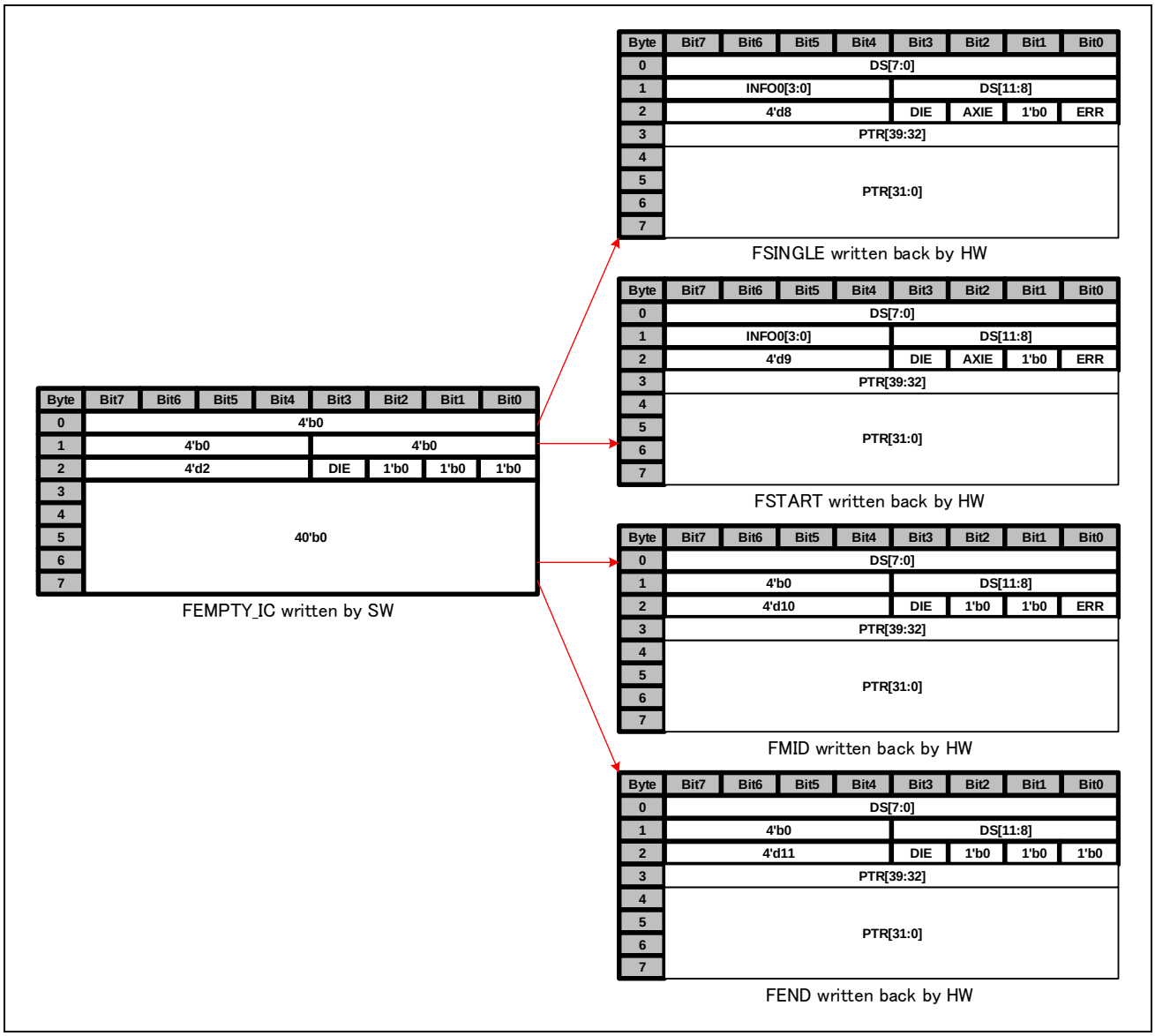


Fig 5.52: Single page incremental data reception FEMPTY_IC process descriptor utilization format

Table 5-23: Single page incremental data reception FEMPTY_IC process descriptor utilization field description

Field name	Field description before write-back	Field description after write-back
DS	12'b0	Size of the data stored in the URAM associated tot the descriptor.
INFO0	4'b0	4'b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.
ERR	1'b0	Set condition: - HW: An ECC error happened while reading the data from Local RAM and a FSINGLE or a FSTART descriptor written back.
DSE	1'b0	1'b0

Field name	Field description before write-back	Field description after write-back
AXIE	1’b0	Set conditions: - HW: An AXI error happened while writing a data (BRESP != 2’b00) and a FSINGLE or a FSTART descriptor written back. - HW: An AXI error happened while reading a descriptor for a split frame and the descriptor was not the first.
DIE	DIE	DIE (Value is copied from read descriptor).
DT	4’d2 for FEMPTY_IC	4’d8 for FSINGLE 4’d9 for FSTART 4’d10 for FMID 4’d11 for FEND
PTR	40’b0	Address where the data corresponding to the descriptor has been written in the URAM.
INFO1 (GWDCCi.EDEi == 1’b1)	64’b0	64’b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.
TS (GWDCCi.ETSi == 1’b1)	64’b0	64’b0 for FEND and FMID, For FSTART and FSINGLE, refer to section 5.3.5.6.

5.3.5.4 Interrupt based multi-page incremental data reception

Interrupt based multi-page incremental data reception aims at saving the received frames back to back in a given memory area. By Interrupt based multi-page incremental data reception, the SW avoid wasting CPU power to stack data together. Interrupt based multi-page incremental data reception process descriptor utilization is described in Fig 5.53. Interrupt based multi-page incremental data reception process descriptor format are the same than for single page incremental data reception, refer to section 5.3.5.3.

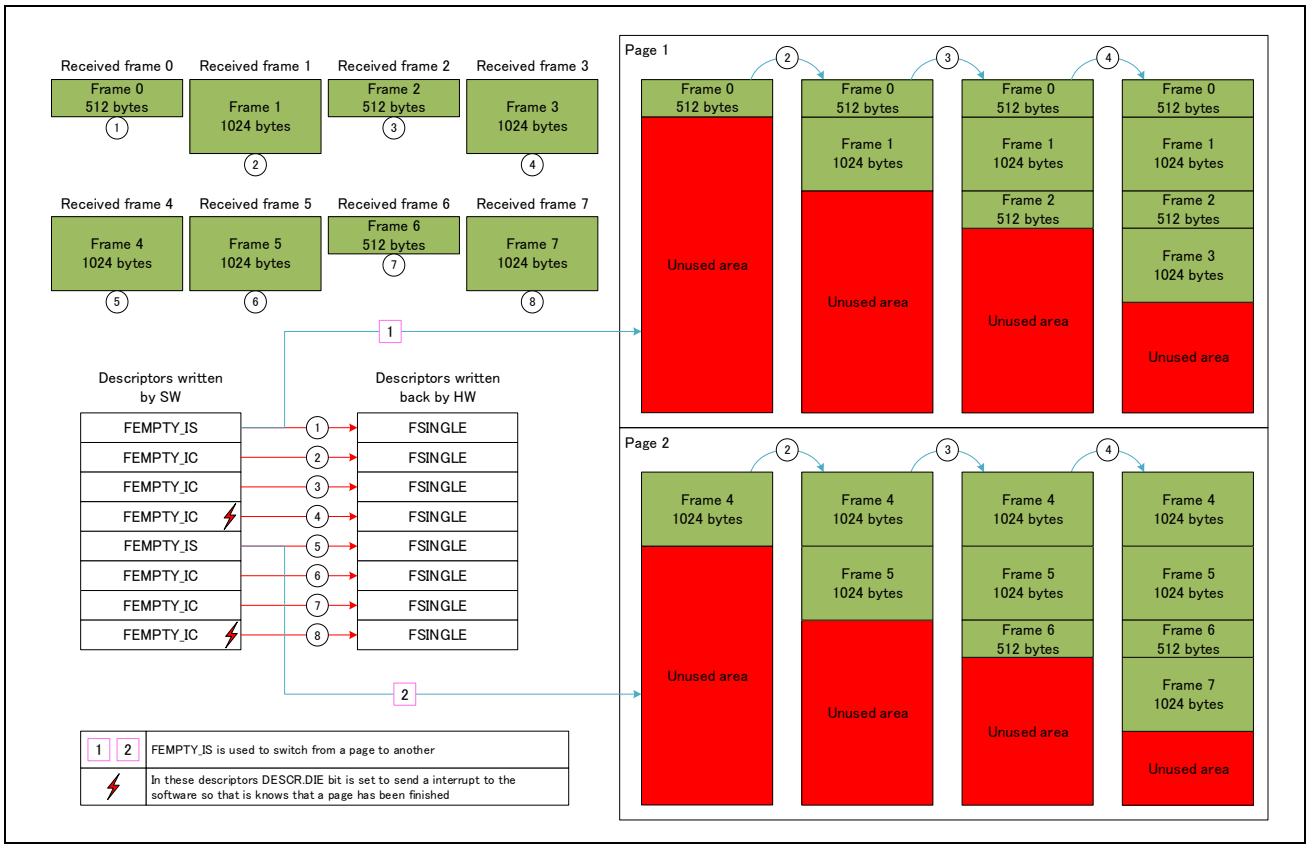


Fig 5.53: Interrupt based multi-page incremental data reception process descriptor utilization

Restrictions:

- SW: It should be ensured that, knowing the expected data and/or controlling page sizes and the number of descriptors for a page, a page cannot overflow.
- SW: This reception type can only be used for the first AXI_RINC_N RX chains.

Notes:

- If a frame is too big to fit in an area reserved by one descriptor, it will be automatically split. The first descriptor will be FSTART, the last descriptor will be FEND and all other descriptors will be FMID.
- When a frame is split the hardware will write back the FSTART descriptor last. Even if the queue is set in keep DT mode, this order is kept.
- Interrupt based multi-page incremental data reception allows to use any number of pages.
- When GWCA reset has been applied (GWMS.OPS set to 2'b00), all the incremental registers (i.e incremental page start address, current address, page size and used page size) were cleared. Therefore, when the GWCA has moved back again to OPERATION mode (GWMS.OPS set to 2'b11) and a frame has been received by GWCA incremental chain, it is not be able to write frame using FEMPTY_IC descriptor. Descriptor of AXI current address have to be(set up) a FEMPTY_IS.

5.3.5.5 Header removal incremental data reception

Header removal incremental data reception aims at suppressing header from incremental data. It is an update of single page incremental data reception (section 5.3.5.3) and interrupt based multi-page incremental data reception (section 5.3.5.4) by adding a FEMPTY_ND between each incremental descriptor. Header removal incremental data reception process descriptor utilization is described in Fig 5.54. Header removal process descriptor format is described in Fig 5.55 and Table 5-24.

Restrictions:

- SW: A FEMPTY_IS/FEMPTY_IC descriptor can only contain 2048bytes. It should be taken in account when setting the descriptor in the descriptor queue because the HW doesn't ensure that the descriptor sequence is correct.

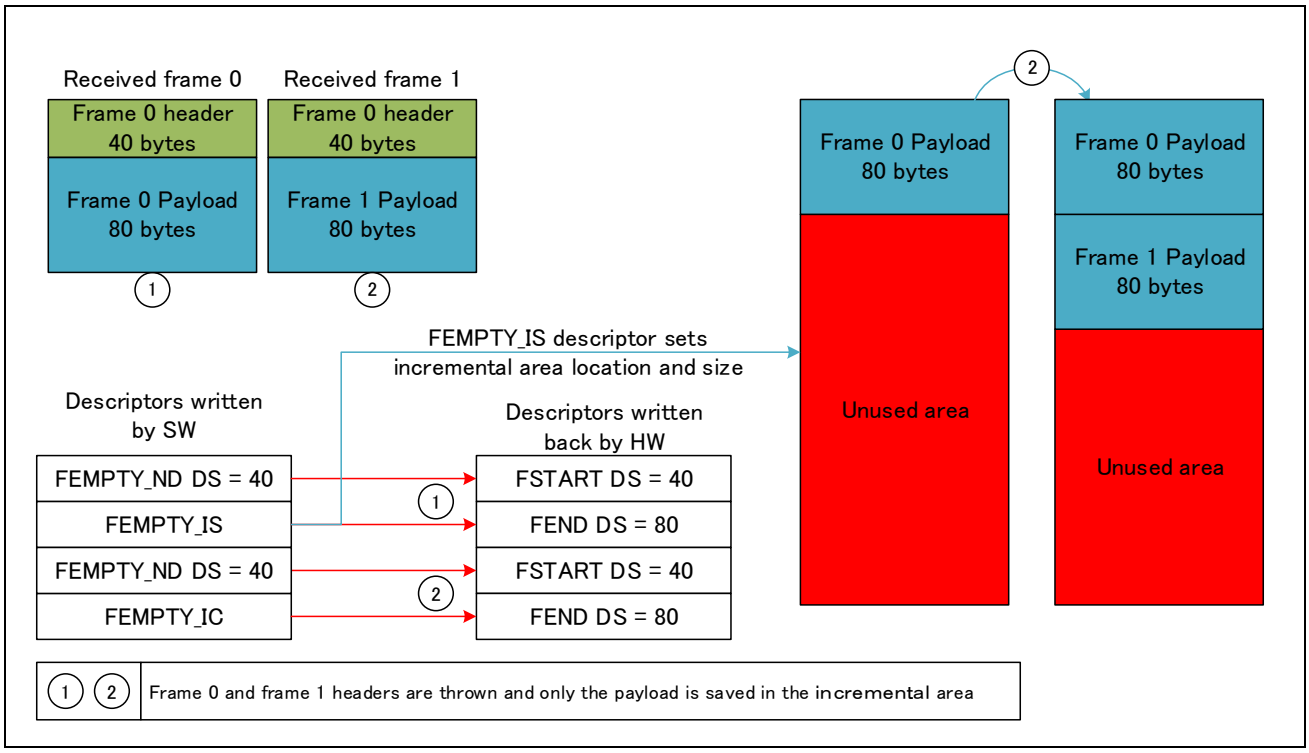


Fig 5.54: Header removal incremental data reception process descriptor utilization

Notes:

- When using Size controlled data reception, to get rid of the header part of a frame and still keep the descriptor order, it is possible to start every frame with a FEMPTY_START descriptor always pointing toward the same memory address (by setting always the same pointer).
- When GWCA reset has been applied (GWMS.OPS set to 2'b00), all the incremental registers (i.e incremental page start address, current address, page size and used page size) were cleared. Therefore, when the GWCA has moved back again to OPERATION mode (GWMS.OPS set to 2'b11) and a frame has been received by GWCA incremental chain, it is not be able to write frame using FEMPTY_IC descriptor. Descriptor of AXI current address have to be(set up) a FEMPTY_IS.

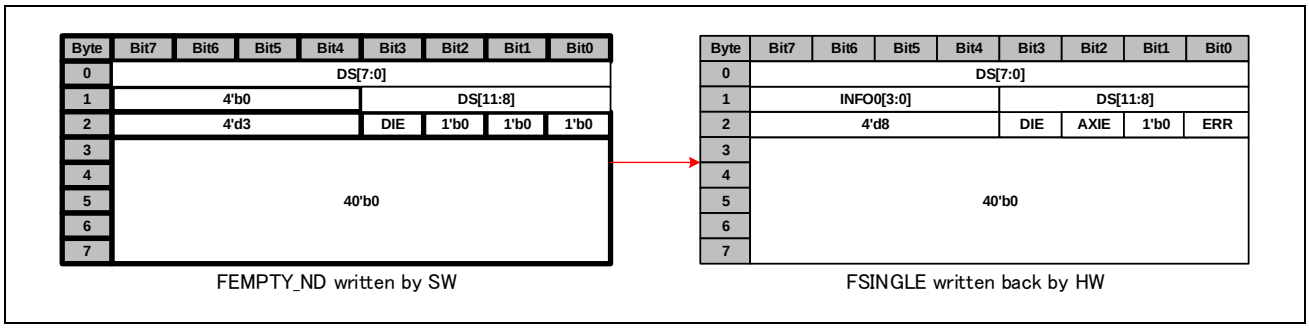


Fig 5.55: Header removal utilization format (for basic descriptor, section 5.1.2.1)

Table 5-24: Header removal process descriptor utilization field description

Field name	Field description before write-back	Field description after write-back
DS	Size to be rejected	Actual size that has been rejected (it should be the same as DS before write-back).
INFO0	4'b0	4'b0 for FEND and FMID. For FSTART and FSINGLE, refer to section 5.3.5.6.
ERR	1'b0	Set conditions: - HW: An ECC error happened while reading the data from Local RAM and a FSINGLE or a FSTART descriptor written back. - HW: An AXI address ECC error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. - HW: A descriptor queue full error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back. - HW: A descriptor number error happened while reading an AXI address during a split frame (The first descriptor has been processed normally but the last part of the frame will not be written) and a FSINGLE or a FSTART descriptor written back.
DSE	1'b0	1'b0
AXIE	1'b0	Set conditions: - HW: An AXI error happened while writing a data (BRESP != 2'b00) and a FSINGLE or a FSTART descriptor written back. - HW: An AXI error happened while reading a descriptor during a split frame (The first descriptor has been processed normally but the last part of the frame has no descriptor to be written) and a FSINGLE or a FSTART descriptor written back.
DIE	DIE	DIE (Value is copied from read descriptor).

Field name	Field description before write-back	Field description after write-back
DT	4'd3 for FEMPTY_ND	4'd8 for FSINGLE 4'd9 for FSTART 4'd10 for FMID 4'd11 for FEND Restrictions: - SW: Because the HW turns a FEMPTY_ND descriptor in a FSINGLE, FSTART, FMID or FEND descriptor, by reading the descriptor it is impossible to know that is contains no data. Consequently, SW should remember where the FEMPTY_ND descriptors have been written.
PTR	32'b0	32'b0
INFO1 (GWDCCi.EDEi == 1'b1)	64'b0	64'b0 for FEND and FMID, For FSTART and FSINGLE, refer to section 5.3.5.6.
TS (GWDCCi.ETSi == 1'b1)	64'b0	64'b0 for FEND and FMID, For FSTART and FSINGLE refer to section 5.3.5.6.

5.3.5.6 Data reception common descriptor format

In this section are described the fields which are common for any data transmission (INFO0, INFO1 and TS). There are two types of formats, the reception direct descriptor and the reception ethernet descriptor formats. They are respectively created from direct local descriptors and ethernet local descriptors, refer to section 5.2.3.5. All the fields contained in AXI descriptors, if quoted, will be written **ADESCR.{Field name}**.

(1) Reception direct descriptor

Reception direct descriptor corresponds to frames sent by a CPU and by deciding their target directly (Forwarding engine processing has been by-passed [FWD]). Reception direct descriptor INFO and TS formats are described in follow.

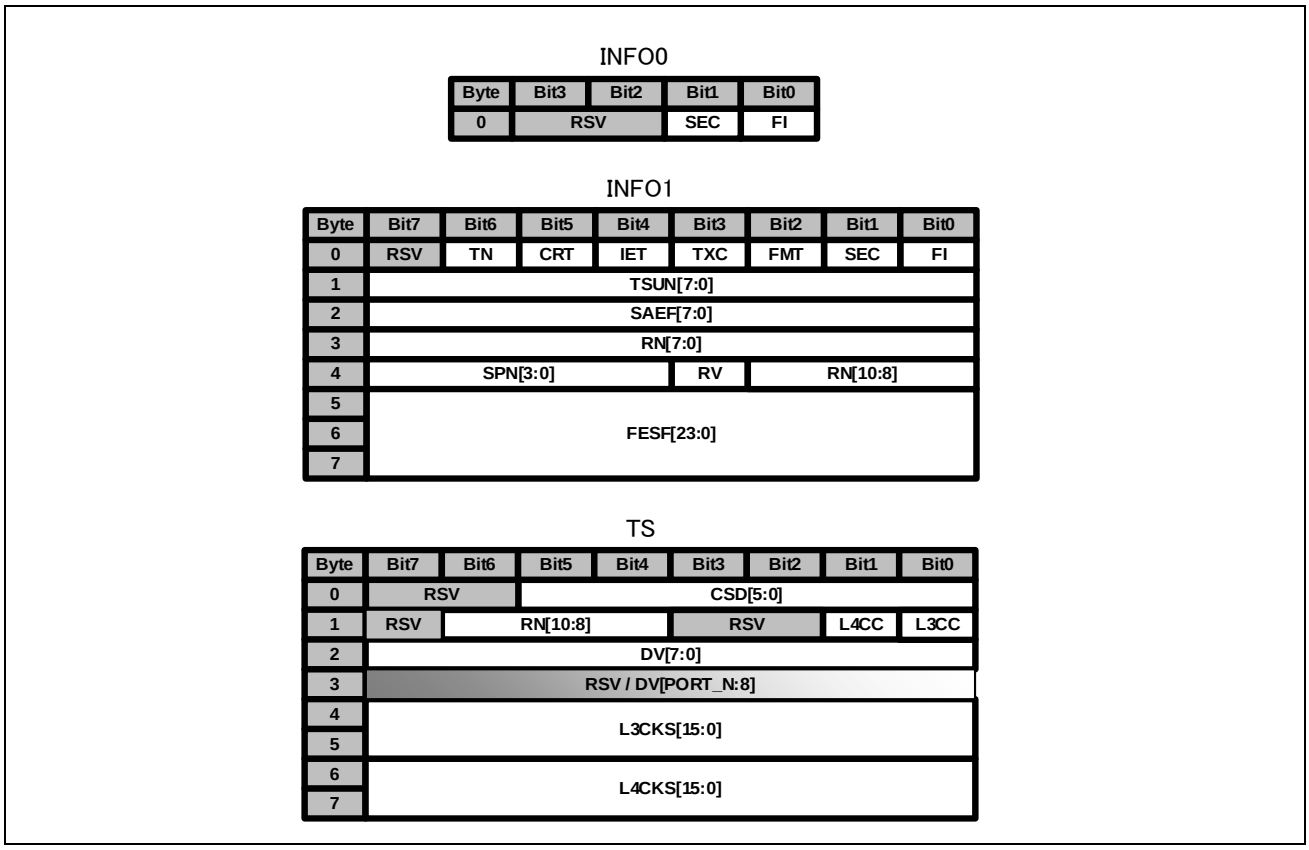


Fig 5.56: Reception direct descriptor INFO and TS formats

Table 5-25: Reception direct descriptor INFO and TS field descriptions

Field name	Bit width	Description	Value for GWCA
FI	1	Refer to Table 5-10.	Refer to section 5.3.4.5.
SEC	1	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
FMT	1	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
TXC	1	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
IET	1	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
CRT	1	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
TN	PTP_TN_W	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
TSUN	8	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).

Field name	Bit width	Description	Value for GWCA
SAEF	8	Refer to Table 5-13.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
L3CC	1	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
L4CC	1	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
RV	1	Refer to Table 5-10.	Received from Descriptor Bus gwc_descr_rv pin [FWD]. Copied from received descriptor **FDESCR.RV** field [FWD].
RN	LTH_RRULE_W	Refer to Table 5-10.	Received from Descriptor Bus gwc_descr_rn pins [FWD]. Copied from received descriptor **FDESCR.RN** field [FWD].
SPN	PORT_W	Source port number	Port number from which the frame entered the switch [FAB]. Received from Descriptor Bus gwc_descr_spn pins [FWD]. Copied from received descriptor **FDESCR.SPN** field [FWD].
FESF	GWC_META_INFO_W	Forwarding engine status flags	Received from Descriptor Bus gwc_descr_minfo pins [FWD]. Copied from received descriptor **FDESCR.MINFO** field [FWD].
CSD	AXI_CHAIN_W	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
DV	PORT_N	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
L3CKS	16	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
L4CKS	16	Refer to Table 5-10.	Copied from direct local descriptor. Refer to section 5.2.3.5(1).
RSV	- -	Reserved area	Set to 0.

(2) Reception ethernet descriptor

Reception ethernet descriptor corresponds to frames that has been by the Forwarding engine [FWD]. Reception ethernet descriptor INFO and TS formats are described in Fig 5.57 and Table 5-26.

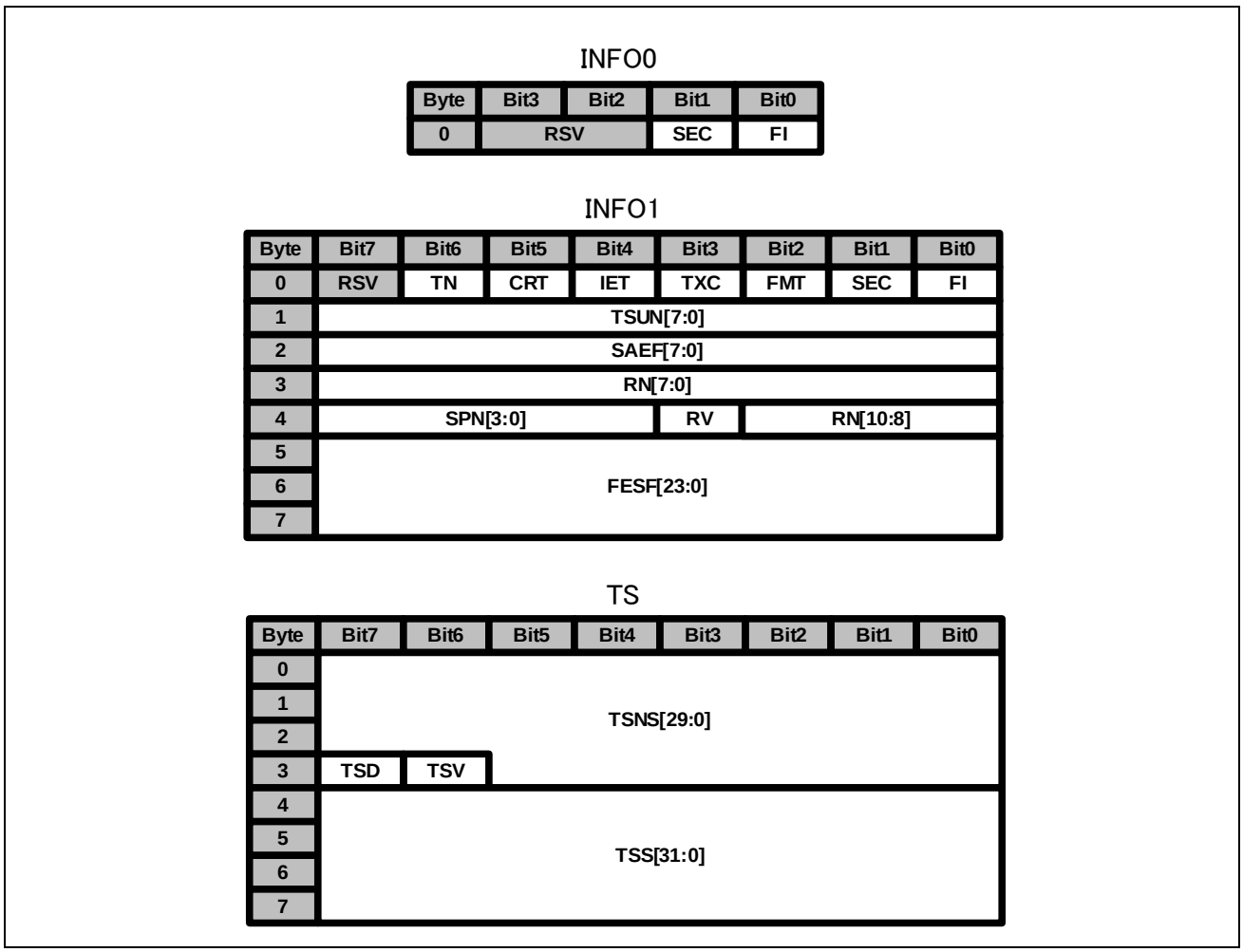


Fig 5.57: Reception ethernet descriptor format INFO and TS formats

Table 5-26: Reception ethernet descriptor format INFO and TS field descriptions

Field name	Bit width	Description	Value for GWCA
FI	1	Refer to Table 5-11.	Refer to section 5.3.4.5.
SEC	1	Refer to Table 5-11.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
FMT	1	Refer to Table 5-11.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
TXC	1	Refer to Table 5-11.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
IET	1	Refer to Table 5-11.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
CRT	1	Refer to Table 5-11.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
TN	PTP_TN_W	Refer to Table 5-11.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
TSUN	8	Refer to Table 5-11.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
SAEF	8	Refer to Table 5-14.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
RV	1	Refer to Table 5-11.	Copied from received descriptor **FDESCR.RV** field [FWD].
RN	LTH_RRULE_W	Refer to Table 5-11.	Copied from received descriptor **FDESCR.RN** field [FWD].
SPN	PORT_W	Source port number	Port number from which the frame entered the switch [FAB]. Copied from received descriptor **FDESCR.SPN** field [FWD].
FESF	GWC_META_INFO_W	Forwarding engine status flags	Copied from received descriptor **FDESCR.MINFO** field [FWD].
TSV	1	Timestamp valid	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
TSD	1	Timestamp default	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
TSNS	30	Timestamp nanosecond [PTP] PCH header timestamp [29:0]	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
TSS	32	Timestamp second [PTP] PCH header timestamp [31:30] Restrictions: - HW: The 16-upper bits of the gPTP [PTP] second part are truncated.	Copied from ethernet local descriptor from [GWCA Data transmission] or [TSNA Data reception].
RSV	- -	Reserved area	Set to 0.

5.4 Timestamp reception [PTP]

GWCA allows Timestamp reception through the GWCA TS path, The TS path is described in Fig 5.58.

Timestamps received through this function are only Ethernet agent TX timestamps. Ethernet agent RX timestamps are stored in the local RAM along with their corresponding data and can only be received by CPU through RX data path with the corresponding data.

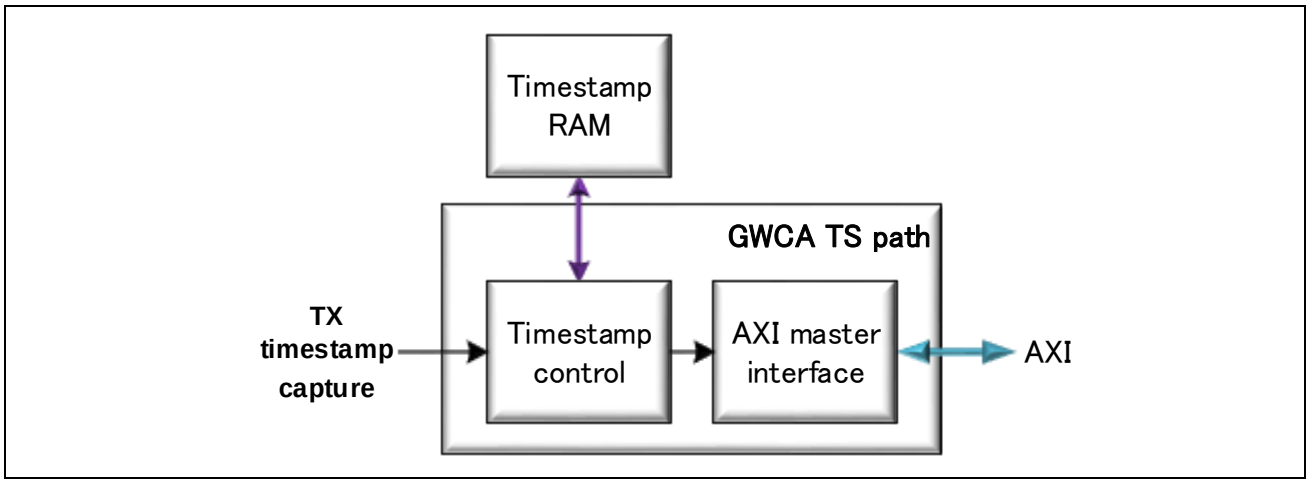


Fig 5.58: GWCA TS path block diagram

The TS path is separated in two blocks:

- Timestamp control: This block aims are receiving the timestamps from TX Timestamp Capture interface [RMAC], storing them in the Timestamp RAM and sending them to the AXI master interface.
- AXI master interface: This block handles the timestamp/AXI descriptor exchange with the CPU.

5.4.1 Timestamp control

Timestamp control controls the descriptor storage/fetching to/from the Timestamp RAM using **GWTSDCCs.TEs** register and can be monitored using **GWTSNM** and **GWTSMNM** registers and **GWEIS0.TSECCES** and **GWEIS0.TSOVFES** interrupt registers.

Functions:

- **GWTSDCCs.TEs** is used to enable timestamp reception for timer s. If enabled, all timestamps received from TX Timestamp Capture interfaces [RMAC] for timer s will be stored in the timestamp RAM and later given to the AXI master interface.
- **GWTSNM** register is used to monitor the current number of timestamps present in the timestamp RAM.
- **GWTSMNM** register is used to monitor the maximum number of timestamps that has been held in the timestamp RAM since the previous reset or the previous time corresponding register has been read.
- **GWEIS0.TSECCES** interrupt register is used to flag a timestamp loss because of an ECC error happened while reading the Timestamps RAM.
- **GWEIS0.TSOVFES** interrupt register is used to flag a timestamp loss because of Timestamp RAM overflow.

5.4.2 AXI master interface

The AXI master interface handles the timestamp/AXI descriptor exchange with the CPU. It will receive the timestamps and the information related to the timestamps and store them in the CPU RAM. To do so, it uses **GWTDCAC0s**, **GWTDCAC1s** and **GWTSDCCs.DCSs** registers and it can be monitored using **GWTSDIS.TSDISs** and **GWEIS0.TDFES** interrupt registers.

Functions:

- **GWTDCAC0s** and **GWTDCAC1s** registers form the AXI address {**GWTDCAC0s.TSCCAUPs**, **GWTDCAC1s.TSCCADPs** with correspond to the current address being processed by the GWCA for TS descriptor queue s. It can be used to set the first address of the descriptor queue during GWCA initialization, move an under-use TS descriptor queue to another location in the CPU RAM or monitor the currently processed descriptor address.
- **GWTSDCCs.DCSs** register is used to map timer s timestamps to a TS descriptor queue number.
- **GWTSDIS.TSDISs** interrupt register is used to flag some timestamps are ready in the CPU RAM.
- **GWEIS0.TDFES** interrupt register is used to flag a timestamp loss because the timestamp descriptor queue is full.

Timestamp reception can be done using linear or cyclic descriptor queues (refer to section 5.1.5). Timestamp reception process descriptor utilization is described in Fig 5.59. Timestamp reception process descriptor format is described in Fig 5.60 and Table 5-27.

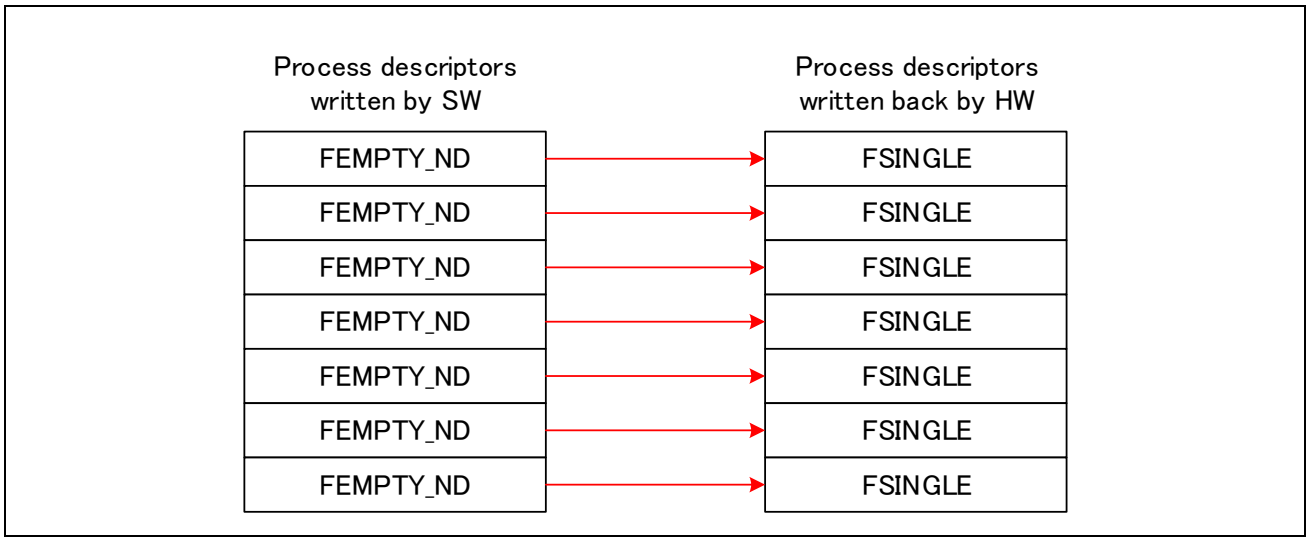


Fig 5.59: TS reception process descriptor utilization

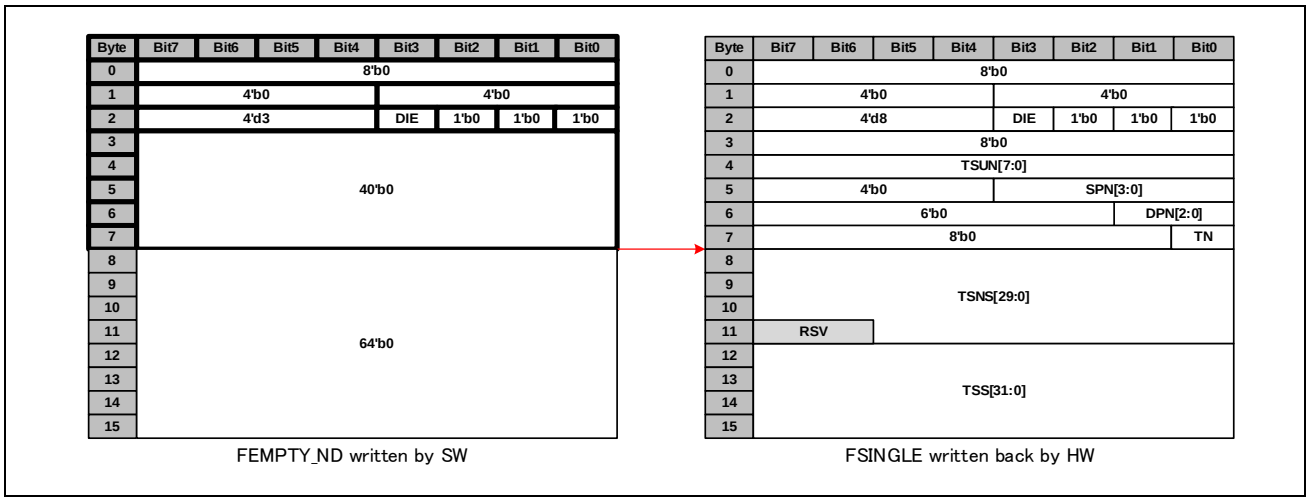


Fig 5.60: TS reception process descriptor utilization format (only Timestamp descriptor, section 5.1.2.1)

Table 5-27: TS process descriptor utilization field description

Field name	Field description written by SW	Field description after HW write-back
DS	12’b0	12’b0
INFO0	4’b0	4’b0
ERR	1’b0	1’b0
DSE	1’b0	1’b0
AXIE	1’b0	1’b0
DIE	DIE	DIE (Value is copied from read descriptor).
DT	4’d3 for FEMPTY_ND	4’d8 for FSINGLE
PTR	40’b0	Used as timestamp related information field. Refer to Table 5-28
INFO1	NA: Info1 is never present for TS reception	NA: Info1 is never present for TS reception.
TS	64’b0	Refer to Table 5-29

Table 5-28: TS reception process descriptor PTR field explanation

Field name	Bit width	Description
TSUN	8	Timestamp unique number [RMAC]
SPN	PORT_W	Port number from which the timestamp corresponding frame entered the switch [FAB]
DPN	PORT_TSNA_W	Port number by which the timestamp has been taken
TN	PTP_TN_W	Timer number

Table 5-29: TS reception process descriptor TS field explanation

Field name	Bit width	Description
TSNS	30	Timestamp nanosecond [PTP]
TSS	32	Timestamp second [PTP] Restrictions: - HW: The 16-upper bits of the gPTP [PTP] second part are truncated.

5.5 gPTP triggered transmission

It is possible to use gPTP timer to trig a periodic transmission start using time-controlled data transmission (refer to section Table 5-9). To do so, **CYCLIC_COMP[i][c]** (Where i is 0 for GWCA0 and 1 for GWCA1) functional interface is linked to **INTER_TX[AXI_TLIM_N+c]** (refer to Fig 5.1).

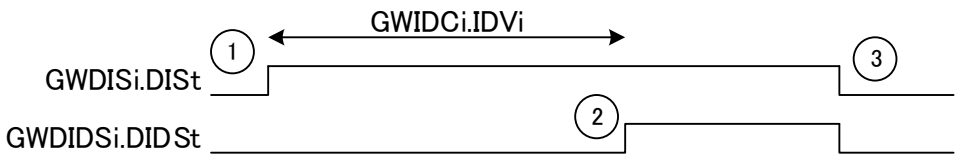
For CYC_COMP_N parameter explanation, refer to gPTP specification [gPTP].

5.6 Interrupt delay function

Interrupt delay function aims at delaying data interrupts (**GWDISi.DISt**) to accumulate data during a certain among of time before requesting software to process the data. As a result, software load is diminished, and data latency is still guaranteed. To operate, interrupt delay function uses **GWIDPC** and **GWIDCi** registers and can be monitored using **GWDIDSi** interrupt register. Interrupt delay function is described in Fig 5.61.

Functions:

- **GWIDPC** register sets an internal clock of period *clk_delay_period* used as base clock for interrupt delay function.
- **GWIDCi** register sets the delay time for descriptor queue i in *clk_delay_period*.
- **GWDIDSi.DIDSt** register is the delayed **GWDISi.DISt** interrupt register and trigs the interrupt gwc_gwdis_int[t].
- For interrupt handling, software can decide to read **GWDIDSi** registers and to process only interrupts that has set until their corresponding delay time is archived (allows to be sure a certain amount of data came before processing) or can decide to read **GWDISi** to process all the interrupts even the ones which didn't reached their delay time (allows to be sure a certain amount of data came before processing at least for one queue can diminish the interrupt load).



1	GWDISi.DISt flag is set
2	After GWIDCi.IDVi*_clk_delay_period_ GWDIDSi.DIDSt is asserted with the interrupt if GWDIEi.DIEt is set to 1
3	SW clears GWDISi.DISt flag, it de-asserts the interrupt

Fig 5.61: Interrupt delay function

Restrictions:

- HW: The maximum delay achievable by the interrupt delay function is clk_period*4194304. For example, it will be around 10ms for a clock clk with a frequency of 400MHz.

6. Precautions

6.1 Precautions

NA.

6.2 Restrictions (Including known problems)

NA.

Published by Renesas Electronics Corporation