

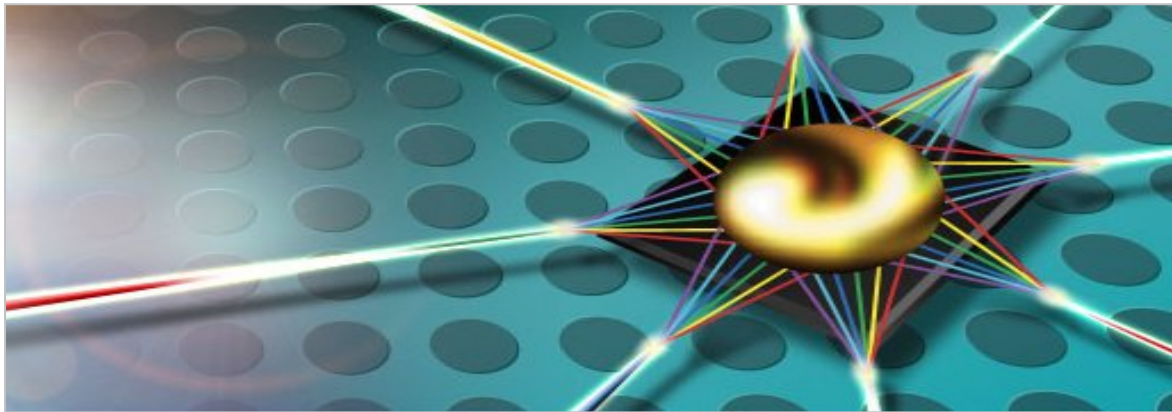


# NPS-400

## EZdp Reference Manual

### Data Plane Environment Library for NPS-400 Network Processors

Document Version 1.9



**Document Number: 27-8153-10**

The information contained is proprietary and confidential.

## Preface

Copyright 2015 EZchip Semiconductor Ltd. EZchip is a registered trademark of EZchip Semiconductor Ltd. Linux is a registered trademark of Linus Torvalds. Brand and product names are trademarks or registered trademarks of their respective holders.

This document contains information proprietary to EZchip and may not be reproduced in any form without prior written consent from EZchip Semiconductor Ltd.

This document is provided on an “as is” basis. While the information contained herein is believed to be accurate, in no event will EZchip be liable for damages arising directly or indirectly from any use of the information contained in this document. All specifications are subject to change without notice.

**EZchip Semiconductor Inc.**, 2700 Zanker Road, Suite 150, San Jose, CA 95134, USA, Tel: (408) 520-3700, Fax: (408) 520-3701

**EZchip Semiconductor Ltd.**, 1 Hatamar Street, PO Box 527, Yokneam 20692, Israel, Tel: +972-4-959-6666, Fax: +972-4-959-4166

Email: [info@ezchip.com](mailto:info@ezchip.com), Web: <http://www.ezchip.com>

EZchip welcomes your comments on this publication. Please address them to: [supportNP@ezchip.com](mailto:supportNP@ezchip.com).

## About this Manual

This document describes the EZchip Data Plane Services library (EZdp) and its related APIs. The EZdp library provides an application programming interface (API) for data-plane applications running on NPS network processors, abstracting the complexities of the underlying CTOP core instruction set and various hardware accelerators.

This manual is intended for software developers who plan to develop data-plane applications for products using the EZchip NPS-400 network processor. To use this manual, you should be familiar with the network processor architecture.

The *EZdk Release Notes* may contain information supplemental to this document.

## Related Documents

For additional information refer to:

DOCUMENT	CONTENT
<i>NPS-400 Architectural Specifications</i>	Overview of the architecture, feature set and functionality of the NPS-400 network processor.
<i>NPS-400 Programming Manual</i>	Overview of the software programming model and concepts for the EZchip NPS-400 network processor.

## This Document

The following is a brief description of the contents of each section:

CHAPTER	NAME	DESCRIPTION
Section 1	<b>Overview</b>	Provides overview of the data-plane services library (EZdp).
Section 2	<b>Directory Structure</b>	Describes the directory structure of EZdp.
Section 3	<b>API Organization</b>	Lists the groups of data-plane API routines.
Section 4	<b>API Overview</b>	Provides an overview of each of the API routines/commands in the groups.
Section 5	<b>Reference</b>	Lists each API routine in the group followed by its structures and enumerations in alphabetical order.

## Revision History

REVISION	DATE	DESCRIPTION OF MODIFICATION
1.9	Sept. 6, 2015	<p>Updated throughout. Refers to EZdk version 1.9a.</p> <p><b>ezdp_atomic.h</b></p> <ul style="list-style-type: none"> <li>Removed ezdp_atomic_xchg8_ext_addr and ezdp_atomic_xchg16_ext_addr.</li> <li>Added ezdp_get_mem_section_info and ezdp_mem_section_info_str.</li> </ul> <p><b>ezdp_job.h</b></p> <ul style="list-style-type: none"> <li>Added ezdp_valid_tm_queue_depth_handle.</li> </ul> <p><b>ezdp_pci.h</b></p> <ul style="list-style-type: none"> <li>ezdp_set_pci_msgq_read_index replaced ezdp_update_pci_msgq_read_index and ezdp_update_pci_msgq_read_index_async.</li> </ul> <p><b>ezdp_processor.h</b></p> <ul style="list-style-type: none"> <li>ezdp_get_hw_thread_id renamed ezdp_get_thread_id.</li> <li>ezdp_get_hw_core_id renamed ezdp_get_core_id.</li> <li>ezdp_get_hw_cluster_id renamed ezdp_get_cluster_id.</li> </ul> <p><b>ezdp_search_prm.h</b></p> <ul style="list-style-type: none"> <li>Added ezdp_prm_lookup_alg_tcam.</li> </ul> <p><b>ezdp_security.h</b></p> <ul style="list-style-type: none"> <li>Added ezdp_security_block_size.</li> </ul>
1.8a	July 6, 2015	<p>Refers to EZdk version 1.8a release.</p> <p><b>ezdp.h</b></p> <ul style="list-style-type: none"> <li>Section 5.2.1.4 Function Documentation was missing from the document.</li> </ul> <p><b>ezdp_math.h</b></p> <ul style="list-style-type: none"> <li>ezdp_add_checksum, ezdp_sub_checksum: Note corrected: The checksum calculation assumes an even (2 byte aligned) offset.</li> </ul> <p><b>ezdp_memory.h</b></p> <ul style="list-style-type: none"> <li>ezdp_calc_checksum_ext_addr, ezdp_calc_checksum: Note corrected: The checksum calculation assumes an even (2 byte aligned) offset. If the pointer is odd, the checksum result should be swapped.</li> </ul> <p><b>ezdp_search.h</b></p> <ul style="list-style-type: none"> <li>ezdp_modify_hash_entry, ezdp_scan_hash_slot: Hash is full indication removed.</li> <li>ezdp_lookup_alg_tcam: Input parameter added: <i>priority_ptr</i> - pointer to returned priority.</li> </ul> <p><b>ezdp_search_prm.h</b></p> <ul style="list-style-type: none"> <li>ezdp_prm_lookup_table_entry: <i>table_base_addr</i> replaced <i>prm_lookup_desc</i> input parameter.</li> <li>ezdp_prm_lookup_hash_entry, ezdp_prm_locate_hash_entry: <i>hash_base_addr</i> replaced <i>prm_lookup_desc</i> input parameter.</li> <li>ezdp_prm_add_hash_entry: bool replaced with uint32_t. Return is 0 (success), ENOMEM (hash is full). Use ezdp_get_err_msg() API to get the detailed error message of the failure.</li> <li>ezdp_prm_modify_hash_entry: bool replaced with void. No return.</li> <li>ezdp_prm_lookup_ultra_ip_entry: <i>uip_base_addr</i> replaced <i>prm_lookup_desc</i> input parameter.</li> </ul>
1.8	Mar. 29, 2015	Updated throughout. Refers to EZdk version 1.8a beta.
1.7	Nov. 9, 2014	Updated throughout. Refers to EZdk version 1.7a.
1.6	July 22, 2014	Updated throughout. Refers to EZdk version 1.6a.
1.5	Mar. 3, 2014	Updated throughout. Refers to EZdk version 1.5a.
1.4	Oct. 31, 2013	Updated throughout. Refers to EZdk version 1.4a.
1.3	July 10, 2013	Updated throughout. Refers to EZdk version 1.3a.
1.2	Apr. 18, 2013	Updated throughout. Refers to EZdk version 1.2a.
1.1	Jan. 17, 2013	Initial release. Refers to EZdk version 1.1a.

# Contents

Preface.....	2
About this Manual.....	2
Related Documents .....	2
This Document.....	2
Revision History .....	3
<b>1. Overview .....</b>	<b>6</b>
<b>2. Directory Structure .....</b>	<b>7</b>
<b>3. API Organization .....</b>	<b>8</b>
<b>4. API Overview .....</b>	<b>9</b>
4.1 EZdp Library Management (ezdp.h) .....	9
4.2 Atomic Operations (ezdp_atomic.h).....	10
4.3 Counter Operations (ezdp_counter.h).....	14
4.3.1 On-demand Counter Operations.....	14
4.3.2 Posted Counter Operations .....	16
4.4 Frame Data Decoding (ezdp_decode.h) .....	17
4.5 DMA Operations (ezdp_dma.h) .....	17
4.6 Frame Buffer Management (ezdp_frame.h) .....	18
4.6.1 Resource Management .....	18
4.6.2 DMA Operations .....	18
4.6.3 Multicast Reference Counters .....	19
4.6.4 Additional Operations.....	19
4.6.5 Frame Iterator Operations.....	19
4.6.6 TM Internal Memory Buffer Management (TM Data Cache).....	19
4.7 Job Management (ezdp_job.h) .....	20
4.7.1 Resource Management .....	20
4.7.2 DMA Operations .....	20
4.7.3 Receiving Jobs.....	20
4.7.4 Transmitting Jobs .....	21
4.7.5 Moving Job to Another Queue .....	21
4.7.6 Discarding Jobs.....	21
4.7.7 Job Containers .....	21
4.7.8 Inter Process Communication.....	22
4.7.9 System Congestion Status .....	22
4.8 Lock Operations (ezdp_lock.h) .....	23
4.9 ALU Operations (ezdp_math.h) .....	24
4.9.1 Arithmetic and Logical Operations .....	24
4.9.2 Bit Manipulation Operations .....	25
4.9.3 Hash Operations .....	26
4.10 Memory Operations (ezdp_memory.h) .....	27
4.11 PCI Interface Operations (ezdp_pci.h) .....	28
4.11.1 PCI Message Queue Operations.....	28
4.11.2 Copy Operations.....	28
4.11.3 Configuration Space Operations.....	28
4.12 Pool Operations (ezdp_pool.h).....	29
4.12.1 Index Pool Operations .....	29
4.12.2 Memory Pool Operations.....	29
4.13 Processor Control Operations (ezdp_processor.h) .....	30
4.13.1 Identification Operations .....	30
4.13.2 Synchronization and Scheduling Operations.....	30
4.14 Queue Operations (ezdp_queue.h) .....	31
4.14.1 Ring (Array Queue) Operations .....	31

4.14.2	List Queue Operations .....	31
4.15	Search Structure Operations (ezdp_search.h).....	32
4.15.1	Direct Table Structures .....	32
4.15.2	Hash Structures.....	32
4.15.3	UltraIP Structures .....	32
4.15.4	TCAM Structures .....	33
4.15.5	Algorithmic TCAM Structures.....	33
4.16	Primitive Search Structure Operations (ezdp_search_prm.h) .....	34
4.16.1	Direct Table Structures .....	34
4.16.2	Hash Structures.....	34
4.16.3	UltraIP Structures .....	34
4.16.4	Algorithmic TCAM Structures.....	35
4.17	Security Operations (ezdp_security.h) .....	36
4.18	String Operations (ezdp_string.h).....	37
4.19	Time Operations (ezdp_time.h).....	37
<b>5.</b>	<b>Reference.....</b>	<b>38</b>

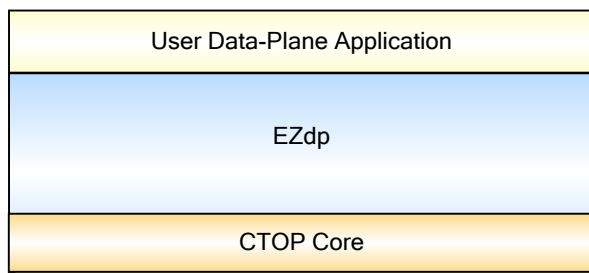
# 1. Overview

The EZchip data-plane services library (EZdp) provides an application programming interface (API) for data-plane applications running on NPS network processors, abstracting the complexities of the underlying CTOP core instruction set and various hardware accelerators.

Using the API routines outlined in this document, programmers can write C (ANSI) code for data-plane applications, including tasks such as packet processing, modification and forwarding; network protocol decoding; classification; etc.


The data-plane APIs are provided as a static library (EZdp) which is linked with the user's data-plane application. The data-plane APIs are performed in the context of the calling user-application tasks.

**Figure 1. DPE overview**



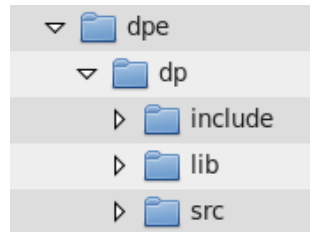
The data-plane services library provides APIs for utilizing the NPS CTOP core optimized instruction set. These provide convenient C-based wrappers for invoking the optimized instructions from C-based application code.

In addition, the data-plane services library provides both synchronous and asynchronous APIs for invoking various hardware accelerations external to the CTOP core. Synchronous commands internally perform implicit hardware scheduling, allowing the CTOP to process another hardware thread while the original request is being serviced. This provides a simple and convenient programming model, while still being efficient. Asynchronous commands are also provided for more advanced scenarios, allowing fine-grained control of the hardware scheduling. As a convention, the asynchronous API routine names end with the suffix “\_async”.

 For more information on the software programming model and concepts for the EZchip NPS-400 network processor, see the *NPS-400 Programming Manual*.

## 2. Directory Structure

The figure and table below show the directory structure after installing the EZdp library.



FOLDER	CONTENTS
/EZdk	
/dpe	Data plane environment libraries
/dp	Data-plane application services library
/include	Include files
/lib	Binary files
/src	Source files

## 3. API Organization

The data-plane APIs are arranged in groups of routines with common functionality, with each group is defined in a separate header file.

Following is a list of the main header files and their functionality:

Files	Description
<a href="#">ezdp.h</a>	EZdp library initialization and version information.
<a href="#">ezdp_atomic.h</a>	API routines for performing atomic operations.
<a href="#">ezdp_counter.h</a>	API routines for counter operations.
<a href="#">ezdp_decode.h</a>	API routines for performing decoding of frame data for standard protocol headers.
<a href="#">ezdp_dma.h</a>	API routines for performing DMA operations.
<a href="#">ezdp_frame.h</a>	API routines for operating with frame buffers.
<a href="#">ezdp_job.h</a>	API routines for operating with jobs.
<a href="#">ezdp_lock.h</a>	API routines for lock operations.
<a href="#">ezdp_math.h</a>	API routines for advanced ALU operations, including arithmetic operations, logical operations, and bit-manipulation operations.
<a href="#">ezdp_memory.h</a>	API routines for operating with memory addresses.
<a href="#">ezdp_pci.h</a>	API routines for operating with the PCI Express interface.
<a href="#">ezdp_pool.h</a>	API routines for operating with pools (user-defined index pool, memory pool).
<a href="#">ezdp_processor.h</a>	API routines for controlling the CTOP processors.
<a href="#">ezdp_queue.h</a>	API routines for queue operations.
<a href="#">ezdp_search.h</a>	API routines for operating on search structures.
<a href="#">ezdp_search_prm.h</a>	API routines for advanced low-level operations on search structures.
<a href="#">ezdp_security.h</a>	API routines for operating security accelerators.
<a href="#">ezdp_string.h</a>	API routines for manipulating character arrays.
<a href="#">ezdp_time.h</a>	API routines for retrieving network and system time.



## 4. API Overview

### 4.1 EZdp Library Management (ezdp.h)

The EZdp API provides API routines for initialization and running of the data-plane application.

API Routine	Async	Description
ezdp_sync_cp	--	Sync until CP configuration is completed.
ezdp_init_global	--	Initialize the data-plane application; should be called once per data-plane executable.
ezdp_init_local	--	Initialize the data-plane process; should be called once per data-plane process (e.g. after each fork).
ezdp_run	--	Run a data plane application.


The **Async** column indicates which of the synchronous commands also have matching asynchronous commands. As a convention, the asynchronous API routine names end with the suffix “\_async”.

In addition, the EZdp API provides the following general services.

API Routine	Async	Description
ezdp_get_version	--	Return the version information of the ezdp library.
ezdp_get_err_msg	--	Return specific error message.
ezdp_get_mem_section_info	--	Return sizes of the memory sections.
ezdp_mem_section_info_str	--	Return printable string of the memory sizes.

## 4.2 Atomic Operations (ezdp\_atomic.h)

EZdp provides API routines for performing atomic operations. These operations function on either extended addresses or summarized addresses.

 For more information on extended and summarized addresses, see the *NPS-400 Programming Manual*.

API Routine	Async	Description
ezdp_atomic_read8_ext_addr	--	Atomically read an 8-bit value from an extended address.
ezdp_atomic_read16_ext_addr	--	Atomically read a 16-bit value from an extended address.
ezdp_atomic_read32_ext_addr	--	Atomically read a 32-bit value from an extended address.
ezdp_atomic_read32_sum_addr	+	Atomically read a 32-bit value from a summarized address.
ezdp_atomic_read64_sum_addr	+	Atomically read a 64-bit value from a summarized address.
ezdp_atomic_write8_ext_addr	+	Atomically write an 8-bit value to an extended address.
ezdp_atomic_write16_ext_addr	+	Atomically write a 16-bit value to an extended address.
ezdp_atomic_write32_ext_addr	+	Atomically write a 32-bit value to an extended address.
ezdp_atomic_write32_sum_addr	+	Atomically write a 32-bit value to a summarized address.
ezdp_atomic_write64_sum_addr	+	Atomically write a 64-bit value to a summarized address.
ezdp_atomic_xchg32_ext_addr	--	Atomically exchange a 32-bit value in an extended address.
ezdp_atomic_xchg32_sum_addr	--	Atomically exchange a 32-bit value in a summarized address.
ezdp_atomic_cmpxchg8_ext_addr	--	Atomically compare and exchange an 8-bit value in an extended address.
ezdp_atomic_cmpxchg16_ext_addr	--	Atomically compare and exchange a 16-bit value in an extended address.
ezdp_atomic_cmpxchg32_ext_addr	--	Atomically compare and exchange a 32-bit value in an extended address.
ezdp_atomic_cmpxchg32_sum_addr	--	Atomic test and set a 32-bit value in a summarized address.
ezdp_atomic_read_and_tst8_ext_addr	--	Atomic read, test and set an 8-bit value in an extended address.
ezdp_atomic_read_and_tst16_ext_addr	--	Atomic read, test and set a 16-bit value in an extended address.
ezdp_atomic_read_and_tst32_ext_addr	--	Atomic read, test and set a 32-bit value in an extended address.
ezdp_atomic_read_and_tst32_sum_addr	--	Atomic read, test and set a 32-bit value in a summarized address.
ezdp_atomic_read_and_clear8_ext_addr	--	Atomically read and clear an 8-bit value in an extended address.

API Routine	Async	Description
ezdp_atomic_read_and_clear16_ext_addr	--	Atomically read and clear a 16-bit value in an extended address.
ezdp_atomic_read_and_clear32_ext_addr	--	Atomically read and clear a 32-bit value in an extended address.
ezdp_atomic_read_and_clear32_sum_addr	--	Atomically read and clear a 32-bit value in a summarized address.
ezdp_atomic_read_and_clear64_sum_addr	--	Atomically read and clear a 64-bit value in a summarized address.
ezdp_atomic_add8_ext_addr	+	Atomically perform an 8-bit logical ADD operation on an extended address.
ezdp_atomic_add16_ext_addr	+	Atomically perform a 16-bit logical ADD operation on an extended address.
ezdp_atomic_add32_ext_addr	+	Atomically perform a 32-bit logical ADD operation on an extended address.
ezdp_atomic_read_and_add8_ext_addr	--	Atomically read and perform an 8-bit logical ADD operation on an extended address.
ezdp_atomic_read_and_add16_ext_addr	--	Atomically read and perform a 16-bit logical ADD operation on an extended address.
ezdp_atomic_read_and_add32_ext_addr	--	Atomically read and perform a 32-bit logical ADD operation on an extended address.
ezdp_atomic_add32_sum_addr	+	Atomically perform a 32-bit logical ADD operation on a summarized address.
ezdp_atomic_add64_sum_addr	+	Atomically perform a 64-bit logical ADD operation on a summarized address.
ezdp_atomic_read_and_add8_sum_addr	--	Atomically read and perform an 8-bit logical ADD operation on a summarized address.
ezdp_atomic_read_and_add16_sum_addr	--	Atomically read and perform a 16-bit logical ADD operation on a summarized address.
ezdp_atomic_read_and_add32_sum_addr	--	Atomically read and perform a 32-bit logical ADD operation on a summarized address.
ezdp_atomic_read_and_add64_sum_addr	--	Atomically read and perform a 64-bit logical ADD operation on a summarized address.
ezdp_atomic_dual_add32_ext_addr	+	Atomically perform a dual ADD operation to two 32-bit variables pointed to by the extended address.
ezdp_atomic_dual_add32_sum_addr	+	Atomically perform a dual ADD operation to two 32-bit variables pointed to by the summarized address.
ezdp_atomic_dual_add64_sum_addr	+	Atomically perform a dual ADD operation to two 64-bit variables pointed to by the summarized address.
ezdp_atomic_read_and_dual_add32_sum_addr	--	Atomically read and perform a dual ADD operation to two 32-bit variables pointed to by the summarized address.
ezdp_atomic_read_and_dual_add64_sum_addr	--	Atomically read and perform a dual ADD operation to two 64-bit variables pointed to by the summarized address.
ezdp_atomic_read_and_inc8_ext_addr	--	Atomically read and increment an 8-bit value in an extended address.
ezdp_atomic_read_and_inc16_ext_addr	--	Atomically read and increment a 16-bit value in an extended address.
ezdp_atomic_read_and_inc32_ext_addr	--	Atomically read and increment a 32-bit value in an extended address.

API Routine	Async	Description
ezdp_atomic_read_and_inc32_sum_addr	--	Atomically read and increment a 32-bit value in a summarized address.
ezdp_atomic_read_and_inc64_sum_addr	--	Atomically read and increment a 64-bit value in a summarized address.
ezdp_atomic_read_and_dec8_ext_addr	--	Atomically read and decrement conditionally by one 8-bit value in an extended address (zero value does not underflow).
ezdp_atomic_read_and_dec16_ext_addr	--	Atomically read and decrement conditionally by one 16-bit value in an extended address (zero value does not underflow).
ezdp_atomic_read_and_dec32_ext_addr	--	Atomically read and decrement conditionally by one 32-bit value in an extended address (zero value does not underflow).
ezdp_atomic_read_and_dec32_sum_addr	--	Atomically read and decrement conditionally by one 32-bit value in a summarized address (zero value does not underflow).
ezdp_atomic_read_and_dec64_sum_addr	--	Atomically read and decrement conditionally by one 64-bit value in a summarized address (zero value does not underflow).
ezdp_atomic_read_and_inc32_cond_ext_addr	--	Atomically read and increment conditionally a 32-bit value in an extended address.
ezdp_atomic_read_and_inc32_cond_sum_addr	--	Atomically read and increment conditionally a 32-bit value in a summarized address.
ezdp_atomic_and8_ext_addr	+	Atomically perform an 8-bit logical AND operation on an extended address.
ezdp_atomic_and16_ext_addr	+	Atomically perform a 16-bit logical AND operation on an extended address.
ezdp_atomic_and32_ext_addr	+	Atomically perform a 32-bit logical AND operation on an extended address.
ezdp_atomic_read_and_and8_ext_addr	--	Atomically read and perform an 8-bit logical AND operation on an extended address.
ezdp_atomic_read_and_and16_ext_addr	--	Atomically read and perform a 16-bit logical AND operation on an extended address.
ezdp_atomic_read_and_and32_ext_addr	--	Atomically read and perform a 32-bit logical AND operation on an extended address.
ezdp_atomic_and32_sum_addr	+	Atomically perform a 32-bit logical AND operation on a summarized address.
ezdp_atomic_read_and_and32_sum_addr	--	Atomically read and perform a 32-bit logical AND operation on a summarized address.
ezdp_atomic_or8_ext_addr	+	Atomically perform an 8-bit logical OR operation on an extended address.
ezdp_atomic_or16_ext_addr	+	Atomically perform a 16-bit logical OR operation on an extended address.
ezdp_atomic_or32_ext_addr	+	Atomically perform a 32-bit logical OR operation on an extended address.
ezdp_atomic_read_and_or8_ext_addr	--	Atomically read and perform an 8-bit logical OR operation on an extended address.
ezdp_atomic_read_and_or16_ext_addr	--	Atomically read and perform a 16-bit logical OR operation on an extended address.
ezdp_atomic_read_and_or32_ext_addr	--	Atomically read and perform a 32-bit logical OR operation on an extended address.

API Routine	Async	Description
ezdp_atomic_or8_sum_addr	+	Atomically perform an 8-bit logical OR operation on a summarized address.
ezdp_atomic_or16_sum_addr	+	Atomically perform a 16-bit logical OR operation on a summarized address.
ezdp_atomic_or32_sum_addr	+	Atomically perform a 32-bit logical OR operation on a summarized address.
ezdp_atomic_read_and_or32_sum_addr	--	Atomically read and perform a 32-bit logical OR operation on a summarized address.
ezdp_atomic_xor8_ext_addr	+	Atomically perform an 8-bit logical XOR operation on an extended address.
ezdp_atomic_xor16_ext_addr	+	Atomically perform a 16-bit logical XOR operation on an extended address.
ezdp_atomic_xor32_ext_addr	+	Atomically perform a 32-bit logical XOR operation on an extended address.
ezdp_atomic_read_and_xor8_ext_addr	--	Atomically read and perform an 8-bit logical XOR operation on an extended address.
ezdp_atomic_read_and_xor16_ext_addr	--	Atomically read and perform a 16-bit logical XOR operation on an extended address.
ezdp_atomic_read_and_xor32_ext_addr	--	Atomically read and perform a 32-bit logical XOR operation on an extended address.
ezdp_atomic_xor32_sum_addr	+	Atomically perform a 32-bit logical XOR operation on a summarized address.
ezdp_atomic_read_and_xor32_sum_addr	--	Atomically read and perform a 32-bit logical XOR operation on a summarized address.

## 4.3 Counter Operations (ezdp\_counter.h)

EZdp provides API routines for operating on counters. Separate routines are provided for each counter type: single, double, bitwise, token bucket, watchdog and posted counters. In addition, routines for reading counter messages from the counter message queues are provided.

### 4.3.1 On-demand Counter Operations

EZdp provides the following API routines for operating with on-demand counters.

API Routine	Async	Description
ezdp_write_single_ctr_cfg	+	Configure single counter and its initial value.
ezdp_read_single_ctr_cfg	+	Read single counter configuration.
ezdp_write_single_ctr	+	Initialize single counter with the value specified.
ezdp_xchg_single_ctr	--	Write single counter with the value specified and read previous counter value.
ezdp_read_single_ctr	--	Read single counter value.
ezdp_inc_single_ctr	+	Increment single counter by the value specified.
ezdp_read_and_inc_single_ctr	--	Increment single counter by the value specified and read previous counter value.
ezdp_dec_single_ctr	+	Decrement single counter by the value specified.
ezdp_read_and_dec_single_ctr	--	Decrement single counter by the value specified and read previous counter value.
ezdp_reset_single_ctr	+	Reset single counter to zero.
ezdp_read_and_reset_single_ctr	--	Reset single counter to zero and read previous counter value.
ezdp_cond_dec_single_ctr	+	Conditionally decrement single counter by the value specified.
ezdp_read_and_cond_dec_single_ctr	--	Conditionally decrement single counter by the value specified and read previous counter value.
ezdp_prefetch_single_ctr	+	Prefetch single counter into the local cache.
ezdp_write_dual_ctr_cfg	+	Configure dual counter and its initial values (byte and event).
ezdp_read_dual_ctr_cfg	+	Read dual counter configuration.
ezdp_read_dual_ctr	--	Read dual counter values (byte and event).
ezdp_inc_dual_ctr	+	Increment dual counter with the values specified (byte and event).
ezdp_read_and_inc_dual_ctr	--	Increment dual counter with the values specified (byte and event) and read previous counter values.
ezdp_dec_dual_ctr	+	Decrement dual counter's byte value by the value specified and event value by 1.
ezdp_read_and_dec_dual_ctr	--	Decrement dual counter's byte value by the value specified and event value by 1, and read previous counter values.
ezdp_reset_dual_ctr	+	Reset dual counter values (byte and event) to zero.
ezdp_read_and_reset_dual_ctr	--	Reset dual counter values (byte and event) to zero and read previous counter values.
ezdp_prefetch_dual_ctr	+	Prefetch dual counter into the local cache.

API Routine	Async	Description
ezdp_write_bitwise_ctr_cfg	+	Configure bitwise counter and its initial value.
ezdp_read_bitwise_ctr_cfg	+	Read bitwise counter configuration.
ezdp_write_bits_bitwise_ctr	+	Write the value to the selected bits in the bitwise counter.
ezdp_xchg_bits_bitwise_ctr	--	Write the value to the selected bits in the bitwise counter and read previous counter value.
ezdp_read_bitwise_ctr	--	Read bitwise counter value.
ezdp_read_bits_bitwise_ctr	--	Read the selected bits from the bitwise counter.
ezdp_inc_bits_bitwise_ctr	+	Increment the selected bits in the bitwise counter by the value specified.
ezdp_read_and_inc_bits_bitwise_ctr	--	Increment the selected bits in the bitwise counter by the value specified and read previous counter value.
ezdp_dec_bits_bitwise_ctr	+	Decrement the selected bits in the bitwise counter by the value specified.
ezdp_read_and_dec_bits_bitwise_ctr	--	Decrement the selected bits in the bitwise counter by the value specified and read previous counter value.
ezdp_reset_bitwise_ctr	+	Reset bitwise counter value to zero.
ezdp_read_and_reset_bitwise_ctr	--	Reset bitwise counter value to zero and read previous counter value.
ezdp_set_bits_bitwise_ctr	+	Set the selected bits in the bitwise counter according to the value specified.
ezdp_read_and_set_bits_bitwise_ctr	--	Set the selected bits in the bitwise counter according to the value specified and read previous counter value.
ezdp_clear_bits_bitwise_ctr	+	Clear the selected bits in the bitwise counter according to the value specified.
ezdp_read_and_clear_bits_bitwise_ctr	--	Clear the selected bits in the bitwise counter according to the value specified and read previous counter value.
ezdp_read_and_cond_write_bits_bitwise_ctr	--	Read, compare and conditionally set the specified bits in the bitwise counter with the value specified.
ezdp_prefetch_bitwise_ctr	+	Prefetch bitwise counter into the local cache
ezdp_write_tb_ctr_cfg	+	Configure token bucket counter.
ezdp_read_tb_ctr_cfg	--	Read token bucket counter configuration.
ezdp_update_tb_ctr	+	Update a Token Bucket with the specified value (e.g. packet length).
ezdp_read_tb_ctr	+	Get the resulting color after updating a Token Bucket with the specified value (e.g. packet length).
ezdp_check_tb_ctr	+	Get the resulting color after updating a Token Bucket with the specified value (e.g. packet length), without updating it.
ezdp_inc_tb_ctr	+	Force increment token buckets with the specified value (e.g. packet length).
ezdp_read_and_inc_tb_ctr	--	Force increment token buckets with the specified value (e.g. packet length) and get resulting color and bucket states.
ezdp_dec_tb_ctr	+	Force decrement token buckets with the specified value (e.g. packet length).
ezdp_read_and_dec_tb_ctr	--	Force decrement token buckets with the specified value (e.g. packet length) and get resulting color and bucket states.

API Routine	Async	Description
ezdp_prefetch_tb_ctr	+	Prefetch token bucket counter into the local cache.
ezdp_write_hier_tb_ctr_cfg	+	Configure hierarchical token bucket counter.
ezdp_read_hier_tb_ctr_cfg	--	Read hierarchical token bucket counter configuration
ezdp_inc_hier_tb_ctr	+	Increment hierarchical token bucket counter accumulator(s) by the value specified.
ezdp_read_and_inc_hier_tb_ctr	--	Increment hierarchical token bucket counter accumulator(s) by the value specified and read previous counter value.
ezdp_update_hier_tb_ctr	+	Update hierarchical token bucket counter with state, app bits or clear accumulators.
ezdp_read_and_update_hier_tb_ctr	--	Update hierarchical token bucket counter with state, app bits or clear accumulators and read previous counter value.
ezdp_change_state_hier_tb_ctr	--	Change hierarchical token bucket state to Ph1
ezdp_write_watchdog_ctr_cfg	+	Configure watchdog counter.
ezdp_read_watchdog_ctr_cfg	--	Read watchdog counter configuration.
ezdp_start_watchdog_ctr	+	Start the watchdog counter.
ezdp_inc_watchdog_ctr	+	Increment the watchdog counter events by one.
ezdp_check_watchdog_ctr	+	Check the number of events in the watchdog counter.
ezdp_prefetch_watchdog_ctr	+	Prefetch watchdog counter into the local cache.
ezdp_init_ctr_msg_queue_des	--	Initialize counter message queue descriptor.
ezdp_read_ctr_msg	--	Read counter message from message queue.

### 4.3.2 Posted Counter Operations

EZdp provides the following API routines for operating with posted counters.

API Routine	Async	Description
ezdp_write_posted_ctr	+	Initialize posted counter with the value specified.
ezdp_dual_write_posted_ctr	+	Initialize two successive posted counters with the value specified.
ezdp_add_posted_ctr	+	Add signed value to posted counter.
ezdp_dual_add_posted_ctr	+	Add signed values to two successive posted counters.
ezdp_report_posted_ctr	--	Generate posted counter value report.
ezdp_report_and_clear_posted_ctr	--	Generate posted counter value report and reset the counter to zero.
ezdp_dual_report_posted_ctr	--	Generate posted counter value report for two successive counters.
ezdp_dual_report_and_clear_posted_ctr	--	Generate posted counter value report for two successive counters and reset both counter values to zero.
ezdp_reset_posted_ctr	+	Reset posted counter.
ezdp_dual_reset_posted_ctr	+	Reset two successive posted counters
ezdp_init_posted_ctr_msg_queue_desc	--	Initialize posted counter message queue descriptor.
ezdp_read_posted_ctr_msg	--	Read posted counter message from message queue.



## 4.4 Frame Data Decoding (ezdp\_decode.h)

EZdp provides API routines for performing decoding of frame data for standard protocol headers.

The decoding operations are split into two types. The first type parses and decodes multi-byte frame data stored in the core's local memory (CMEM). These operations receiving a pointer to the start of the data in the CMEM and the size/length of the header to parse, and may take several cycles.


API Routine	Async	Description
ezdp_decode_mac	+	Parse and decode an Ethernet header.
ezdp_decode_ipv4	+	Parse and decode an IPv4 header.
ezdp_decode_ipv6	+	Parse and decode an IPv6 header.
ezdp_decode_mpls	+	Parse and decode an MPLS header.
ezdp_decode_mpls_label	+	Parse and decode an MPLS label.
ezdp_decode_tcp	--	Parse and decode a TCP header.

The second type does not parse data in the CMEM, but instead decodes a single, fixed-size value stored in a variable/register.

API Routine	Async	Description
ezdp_decode_ip_protocol	--	Decode an IP protocol value.
ezdp_decode_eth_type	--	Decode an Ethernet type value.

## 4.5 DMA Operations (ezdp\_dma.h)

EZdp provides API routines for performing DMA operations. These operations function on either extended addresses or summarized addresses.

 For more information on extended and summarized addresses, see the *NPS-400 Programming Manual*.

API Routine	Async	Description
ezdp_copy_data_by_ext_addr	+	Copy data between two extended addresses.
ezdp_load_data_from_ext_addr	+	Copy data from an extended address to CMEM.
ezdp_load_16_byte_data_from_ext_addr	+	Copy 16 bytes from an extended address to CMEM.
ezdp_load_32_byte_data_from_ext_addr	+	Copy 32 bytes from an extended address to CMEM.
ezdp_store_data_to_ext_addr	+	Copy data from CMEM to an extended address.
ezdp_store_16_byte_data_to_ext_addr	+	Copy 16 bytes from CMEM to an extended address.
ezdp_store_32_byte_data_to_ext_addr	+	Copy 32 bytes from CMEM to an extended address.
ezdp_load_data_from_sum_addr	+	Load data from a summarized address to CMEM.
ezdp_load_16_byte_data_from_sum_addr	+	Load 16 bytes from a summarized address to CMEM
ezdp_load_32_byte_data_from_sum_addr	+	Load 32 bytes from a summarized address to CMEM
ezdp_store_data_to_sum_addr	+	Store data from CMEM to summarized address.
ezdp_store_16_byte_data_to_sum_addr	+	Store 16 bytes from CMEM to a summarized address
ezdp_store_32_byte_data_to_sum_addr	+	Store 32 bytes from CMEM to a summarized address

## 4.6 Frame Buffer Management (ezdp\_frame.h)

EZdp provides API routines for operating with frame buffers. All frame buffer API routines operate on a buffer descriptor data structure (as defined in `ezdp_frame_defs.h`), which represents a handle to a frame buffer in either internal or external memory.

### 4.6.1 Resource Management

EZdp provides API routines for allocating/freeing frame buffers from the hardware buffer pools. The API supports simple and optimized operations for the common case of allocating/freeing a single buffer, and in addition also supports more advanced operations for allocating/freeing multiple buffers in a single command (up to 8 buffers per command).

API Routine	Async	Description
<code>ezdp_alloc_buf</code>	--	Allocate a single frame buffer.
<code>ezdp_free_buf</code>	+	Free a single frame buffer.
<code>ezdp_alloc_multi_buf</code>	+	Allocate multiple frame buffers.
<code>ezdp_buf_alloc_failed</code>	--	Check if allocation of the buffer failed.
<code>ezdp_free_multi_buf</code>	+	Free multiple frame buffers.
<code>ezdp_read_free_buf</code>	--	The number of buffers available to be obtained.
<code>ezdp_rebudget_buf</code>	+	Update the budget to which buffers are credited.

### 4.6.2 DMA Operations

EZdp provides API routines for performing DMA operations on frame data buffers. The API supports operations to load data from a frame buffer to the core's local memory (CMEM), store data from the CMEM to the frame buffer, and copy data between two frame buffers or between a frame buffer and an extended memory address.

API Routine	Async	Description
<code>ezdp_copy_frame_data</code>	+	Copy data between two frame buffers.
<code>ezdp_copy_frame_data_to_ext_addr</code>	+	Copy data from a frame buffer to an extended address.
<code>ezdp_copy_frame_data_from_ext_addr</code>	+	Copy data from an extended address to a frame buffer.
<code>ezdp_clone_frame_data</code>	+	Copy data between two frame buffers, with the same source and destination offset (optimized).
<code>ezdp_load_frame_data</code>	+	Copy data from a frame buffer to the CMEM.
<code>ezdp_store_frame_data</code>	+	Copy data from CMEM to a frame buffer.

Similarly, EZdp provides APIs to perform DMA operations on link-buffer-descriptor (LBD) data residing in frame buffers. These require separate/dedicated APIs as they use different memory error protection mechanisms.

API Routine	Async	Description
<code>ezdp_copy_frame_lbd</code>	+	Copy LBD data between two frame buffers.
<code>ezdp_copy_frame_lbd_to_ext_addr</code>	+	Copy LBD data from a frame buffer to an extended address.
<code>ezdp_copy_frame_lbd_from_ext_addr</code>	+	Copy LBD data from an extended address to a frame buffer.
<code>ezdp_clone_frame_lbd</code>	+	Copy LBD between two frame buffers, with the same source and destination offset (optimized).
<code>ezdp_load_frame_lbd</code>	+	Copy LBD data from a frame buffer to CMEM.
<code>ezdp_store_frame_lbd</code>	+	Copy LBD data from CMEM to a frame buffer.

### 4.6.3 Multicast Reference Counters

A dedicated multicast reference counter is maintained for each frame buffer. EZdp provides API routines for operating with the frame buffer's multicast reference counters.

API Routine	Async	Description
ezdp_alloc_mc_buf	--	Allocate a single frame buffer and set its multicast reference counter.
ezdp_free_mc_buf	--	Free a multicast frame buffer.
ezdp_write_mc_buf_counter	+	Set a frame buffer's multicast reference counter.
ezdp_read_mc_buf_counter	--	Get a frame buffer's multicast reference counter.
ezdp_atomic_read_and_inc_mc_buf_counter	--	Atomically read and increment a frame buffer's multicast reference counter.
ezdp_atomic_read_and_dec_mc_buf_counter	--	Atomically read and conditionally decrement a frame buffer's multicast reference counter.

### 4.6.4 Additional Operations

The following services are provided for operating with frame data structures:

API Routine	Async	Description
ezdp_calc_frame_data_checksum	--	Calculate checksum of frame data buffer.
ezdp_buf_data_len	--	Calculate the length of the buffer based on header offset and free bytes.
ezdp_lbd_length	--	Calculate LBD buffer length according to BD count.
ezdp_calc_header_offset	--	Calculate optimized frame header offset.

### 4.6.5 Frame Iterator Operations

The following services are provided for frame iterator operations:

API Routine	Async	Description
ezdp_get_first_buf	--	Gets first buffer from frame.
ezdp_get_next_buf	--	Get next buffer from frame.
ezdp_init_frame	--	Create a new frame by setting its frame descriptor params and init frame iterator.
ezdp_append_buf	--	Add newly allocated (by user) buffer to frame pointed by iterator.
ezdp_sync_frame	--	Store last LBD line to memory, if required.

### 4.6.6 TM Internal Memory Buffer Management (TM Data Cache)

The following additional services are provided for TM IMEM buffer management:

API Routine	Async	Description
ezdp_inc_tm_imem_buf_ctr	+	Increment TM IMEM buffer counter.
ezdp_dec_tm_imem_buf_ctr	+	Decrement TM IMEM buffer counter.
ezdp_read_tm_imem_buf_ctr	--	Number of IMEM buffers used by frame in TM.

## 4.7 Job Management (ezdp\_job.h)

EZdp provides API routines for operating with jobs. The job management API routines operate on a job ID which represents a handle to a job descriptor residing in the internal memory. In addition, some of the APIs operate on the job descriptor data structure itself (as defined in ezdp\_job\_defs.h), which represents an active job.

The job management API routines support various levels of abstraction and control. The API provides both low-level routines to enable the data-plane application fine-grained control of the job management, and higher-level API routines that are suitable for most use-cases and provide a more convenient API to perform the required operations.

### 4.7.1 Resource Management

EZdp provides API routines for allocating/freeing jobs from the hardware job pool in the Processor Manager Unit (PMU). Note that in most cases, the data-plane application does not create or free jobs, but rather only receives existing jobs which are created by the NPS hardware, performs operations on these jobs, and then sends the jobs back to the NPS hardware which later frees the job resources.

API Routine	Async	Description
ezdp_alloc_job_id	+	Allocate a new job from the PMU.
ezdp_alloc_multi_job_id	+	Allocate multiple new jobs from the PMU.
ezdp_job_alloc_failed	--	Check if allocation of the job failed.
ezdp_free_job_id	+	Recycle a job to the PMU.
ezdp_read_free_job	--	The number of jobs available to be obtained.
ezdp_rebudget_job	+	Update the budget to which jobs are credited.

### 4.7.2 DMA Operations

EZdp provides API routines for performing DMA operations on job descriptors.

API Routine	Async	Description
ezdp_load_job	+	Copy the job descriptor from IMEM to CMEM.
ezdp_store_job	+	Copy the job descriptor from CMEM to IMEM.
ezdp_store_job_container	+	Copy the job container descriptor from CMEM to IMEM.

These include operations to load job descriptor data from the internal memory to the core's local memory (CMEM), and store data from the CMEM to the internal memory. These APIs receive a job id representing the job descriptor in internal memory and a pointer to the job descriptor data in the CMEM.

### 4.7.3 Receiving Jobs

EZdp provides API routines for receiving jobs from the NPS hardware Processor Management Unit (PMU).

API Routine	Async	Description
ezdp_request_job_id	--	Request a new job from the PMU.
ezdp_wait_for_job_id	--	Suspend execution until a job request completes.
ezdp_cancel_job_request	--	Cancel a job request from the PMU.
ezdp_receive_job	--	Request a new job from the PMU and load it to CMEM.

### 4.7.4 Transmitting Jobs

EZdp provides API routines for transmitting jobs to the network interfaces.

API Routine	Async	Description
ezdp_send_job_id_to_tm	+	Transmit the job via the Traffic Manager (TM).
ezdp_send_job_to_tm	--	Store the job descriptor and transmit the job via the Traffic Manager (TM).
ezdp_send_job_id_to_interface	+	Transmit the job directly to an output queue channel, bypassing the TM.
ezdp_send_job_to_interface	--	Store the job descriptor and transmit the job directly to an output queue channel, bypassing the TM.

### 4.7.5 Moving Job to Another Queue

EZdp provides API routines for moving jobs to another queue in the Processor Management Unit (PMU).

API Routine	Async	Description
ezdp_send_job_id_to_queue	+	Dispatch the job to another queue in the PMU.
ezdp_send_job_to_queue	--	Store the job descriptor and dispatch the job to another queue in the PMU.
ezdp_update_job_id_queue	--	Move the job to another PMU queue without dispatching it.
ezdp_update_job_queue	--	Store the job descriptor and move the job to another PMU queue without dispatching it.

### 4.7.6 Discarding Jobs

EZdp provides API routines for discarding a job, including all its associated frame buffer resources.

API Routine	Async	Description
ezdp_discard_job_id	+	Discard a job and all its associated frame resources.
ezdp_discard_job	--	Store the job descriptor and discard the job and all its associated frame resources.

### 4.7.7 Job Containers

EZdp provides API routines for operating with job containers.

API Routine	Async	Description
ezdp_send_job_id_container	+	Send request to PMU to distribute the job container.
ezdp_send_job_container	--	Store the job container and send request to PMU to distribute it.
ezdp_container_job_count	--	Return the number of the jobs that are in the job container.
ezdp_container_info	--	Set the info according to the number of jobs in container.

## 4.7.8 Inter Process Communication

EZdp provides API routines for Inter Process Communication (IPC).

API Routine	Async	Description
ezdp_notify_cpu	--	Notify target CPU.
ezdp_notice_pending	--	Check if there is a new notice.
ezdp_clear_notice	--	Clear notice indication.
ezdp_wait_for_notice	--	Suspend execution until receiving new notification for CPU.
ezdp_check_notice	--	Check if there is a new notice and clear new notice indication.
ezdp_wait_for_event	--	Suspend execution until job request completed or new notice received.
ezdp_notifier	--	Register notifier function.
ezdp_handle_notice	--	Handle notice.

## 4.7.9 System Congestion Status

EZdp provides API routines for reading system congestion status from various sources.

API Routine	Async	Description
ezdp_read_congestion_status	--	Read priority drop congestion status.
ezdp_read_flow_control_status	--	Read flow control status.
ezdp_read_pmu_input_queue_congestion	--	Read PMU input queue congestion level.
ezdp_read_global_budget	--	Read global budget counter value.
ezdp_read_pmu_input_queue_status	--	Read PMU input queue status from system info.
ezdp_read_pmu_tm_output_queue_status	--	Read PMU TM output queue status from system info.
ezdp_read_pmu_discard_output_queue_status	--	Read PMU discard output queue status from system info.
ezdp_read_pmu_tm_bypass_output_queue_status	--	Read PMU TM bypass output queue status from system info.
ezdp_read_pmu_app_schlr_status	--	Read PMU application scheduler status from system info.
ezdp_read_pmu_group_schlr_status	--	Read PMU group scheduler status from system info.
ezdp_init_tm_reporting_desc	--	Initialize TM queue depth descriptor.
ezdp_calc_tm_queue_depth_handle	--	Calculate TM queue depth handle.
ezdp_get_tm_queue_depth	--	Get entity queue depth.
ezdp_valid_tm_queue_depth_handle	--	Validate tm_handle received from CP/DP API.

## 4.8 Lock Operations (ezdp\_lock.h)

EZdp provides API routines for spin lock functions.

API Routine	Async	Description
ezdp_init_spinlock_ext_addr	--	Initialize resources required for a spin lock.
ezdp_init_spinlock_sum_addr	--	Initialize resources required for a spin lock.
ezdp_lock_spinlock	--	Lock a spin lock.
ezdp_try_lock_spinlock	--	Lock a spin lock with limited number of attempts.
ezdp_unlock_spinlock	--	Release a spin lock which was locked.

EZdp provides API routines for qlock functions.

API Routine	Async	Description
ezdp_init_qlock	--	Initialize queue lock structure.
ezdp_destroy_qlock	--	Get queue lock address.
ezdp_alloc_qlock_slot	--	Allocate queue lock slot.
ezdp_free_qlock_slot	--	Free queue lock slot.
ezdp_lock_qlock	--	Try to lock queue lock.
ezdp_order_lock_qlock	--	Try to lock queue lock (with order).
ezdp_enqueue_qlock	--	Enqueue data to queue lock.
ezdp_dequeue_qlock	--	Dequeue data from queue lock.
ezdp_try_unlock_qlock	--	Try to unlock queue lock.

## 4.9 ALU Operations (ezdp\_math.h)

EZdp provides API routines for advanced ALU operations, including arithmetic operations, logical operations, and bit-manipulation operations. All bit manipulation APIs operate on 32-bit variables.

### 4.9.1 Arithmetic and Logical Operations

EZdp provides the following API routines for performing advanced mathematical operations. These APIs support mathematical operations on specific bits within 32-bit values in an optimized manner.

API Routine	Async	Description
ezdp_add	--	Add selected bits of src1 to selected bits of src2.
ezdp_sub	--	Subtract selected bits of src2 from selected bits of src1.
ezdp_and	--	Perform logical 'AND' between selected bits of src1 and src2.
ezdp_or	--	Perform logical 'OR' between selected bits of src1 and src2.
ezdp_not	--	Perform logical 'NOT' between selected bits of src1 and src2.
ezdp_xor	--	Perform logical 'XOR' between selected bits of src1 and src2.
ezdp_fxor8	--	Apply an 8 bit 'folded xor' operation on selected bits of src1 and src2.
ezdp_fxor16	--	Apply a 16 bit 'folded xor' operation on selected bits of src1 and selected bits of src2.
ezdp_shift_left	--	Perform a 'shift left' operation on a set of bits in src1, with shift size selected by 5 adjacent bits of src2.
ezdp_shift_right	--	Perform a 'shift right' operation on a set of bits in src1, with shift size selected by 5 adjacent bits of src2.
ezdp_count_bits	--	Count the number of bits with value of '1' in a set of bits in src.
ezdp_div	--	Divide 8 selected bits of src1 by 4 selected bits of src2.
ezdp_mod	--	Modulus 8 selected bits of src1 by 4 selected bits of src2.
ezdp_pow_of_2	--	Calculate the value of $2^{\text{exp}}$ and merge into any position in dst.
ezdp_merge_pow_of_2	--	Calculate the value of $2^{\text{exp}}$ and merge into any position in src.



## 4.9.2 Bit Manipulation Operations

EZdp provides the following API routines for performing advanced bit-manipulation operations. These APIs support extracting, inserting, and merging of multiple bits within 32 bit values in an optimized manner.

API Routine	Async	Description
ezdp_set_bit	--	Set a single bit in src to 'one'.
ezdp_clear_bit	--	Clear a single bit in src (sets to 'zero')
ezdp_find_first_one	--	Find the position of the first 'one' in the range src[src_pos+size-1 : src_pos] (from lsb to msb).
ezdp_find_first_zero	--	Find the position of the first 'zero' in the range src[src_pos+size-1 : src_pos] (from lsb to msb).
ezdp_get_bitfield	--	Get a set of adjacent bits from src and place into any position in dst.
ezdp_merge_bitfield	--	Get a set of adjacent bits from src2 and merge into any position in src1.
ezdp_get_2_bitfields	--	Get 2 sets of adjacent bits from src1 and src2 and place into two locations in dst.
ezdp_merge_2_bitfields	--	Get 2 sets of adjacent bits from src1 and src2 and merge into two locations in src1.
ezdp_get_bit	--	Get any bit from src and place in any position in dst.
ezdp_merge_bit	--	Get any bit from src2 and merge it any position in src1.
ezdp_get_2_bits	--	Get two separate bits from src and place in two separate locations in dst.
ezdp_merge_2_bits	--	Get two separate bits from src2 and merge into two separate locations in src1.
ezdp_get_3_bits	--	Get three separate bits from src and place in three separate locations in dst.
ezdp_merge_3_bits	--	Get three separate bits from src2 and merge into three separate locations in src1.
ezdp_get_4_bits	--	Get four separate bits from src and place in four separate locations in dst.
ezdp_merge_4_bits	--	Get four separate bits from src2 and merge into four separate locations in src1.
ezdp_combine_4_bits	--	Get four separate bits from src and place into 4 adjacent bits in dst.
ezdp_combine_merge_4_bits	--	Get four separate bits from src2 to merge into 4 adjacent bits in src1.
ezdp_split_4_bits	--	Get four adjacent bits from src and place in four separate positions in destination.
ezdp_split_merge_4_bits	--	Get four adjacent bits from src2 and merge into four separate positions in src1.
ezdp_get_4_bytes	--	Extract any four bytes from src1 and src2.
ezdp_reflect_bits	--	Perform bit swap in resolution of 1, 2 or 4 bytes.

### 4.9.3 Hash Operations

EZdp provides the following API routines for calculating advanced hash values.

API Routine	Async	Description
ezdp_hash	--	General purpose hash function.
ezdp_hash32	--	General purpose hash function for 32-bit input.
ezdp_hash64	--	General purpose hash function for 64-bit input.
ezdp_bulk_hash	--	General purpose hash function for up to 64-byte input.
ezdp_calc_crc16	--	Perform CRC16 calculation
ezdp_calc_crc32	--	Perform CRC32 calculation
ezdp_add_checksum	--	Add value to checksum.
ezdp_sub_checksum	--	Subtract value from checksum.

## 4.10 Memory Operations (ezdp\_memory.h)

EZdp provides API routines for operating with memory addresses.

API Routine	Async	Description
ezdp_calc_checksum_ext_addr	--	Calculate checksum of data on extended address.
ezdp_calc_checksum	--	Calculate checksum on a block of memory in CMEM.
ezdp_is_null_sum_addr	--	Check if a summarized address is null.
ezdp_calc_sum_addr	--	Calculate summarized address from address descriptor and key.
ezdp_sum_addr_to_ext_addr	--	Calculate extended address from summarized address.
ezdp_ext_addr_to_sum_addr	--	Calculate summarized address from extended address.
ezdp_calc_sum_addr_offset	--	Calculate offset of a summarized address from extended address.
ezdp_scramble_sum_addr	--	Scramble given summarized address.
ezdp_scramble_ext_addr	--	Scramble given extended address.

## 4.11 PCI Interface Operations (ezdp\_pci.h)

EZdp provides API routines for operating with the PCI Express interface.

**Note:** The PCI Interface APIs are preliminary and not supported in the current EZdk release. They are provided here as reference only.

### 4.11.1 PCI Message Queue Operations

API Routine	Async	Description
ezdp_init_pci_queue_desc	--	Initialize and get PCI message queue description.
ezdp_get_pci_msg	--	Get message from PCI queue according to given index.
ezdp_set_pci_msgq_read_index	+	Set read index of PCI message queue.
ezdp_get_pci_msgq_write_index	--	Get write index of PCI message queue.
ezdp_get_pci_msgq_read_index	--	Get read index of PCI message queue.

### 4.11.2 Copy Operations

API Routine	Async	Description
ezdp_copy_frame_data_to_pci	+	Copy data from a frame buffer to PCI address.
ezdp_copy_frame_data_from_pci	+	Copy data from PCI address to a frame buffer.
ezdp_load_data_from_pci	+	Copy data from a PCI address to CMEM.
ezdp_store_data_to_pci	+	Copy data from CMEM to PCI address.
ezdp_copy_pci_data_to_ext_addr	+	Copy PCI data to extended addresses.
ezdp_copy_pci_data_from_ext_addr	+	Copy extended address data to PCI.
ezdp_translate_pci_addr	+	PCI address translation request; result will be saved in CMEM.
ezdp_translate_pci_addr_to_ext_addr	+	Translate PCI address to extended addresses.
ezdp_send_message_to_pci	+	Copy data from CMEM to PCI address.
ezdp_send_interrupt_to_pci	+	Send interrupt message to PCI.

### 4.11.3 Configuration Space Operations

API Routine	Async	Description
ezdp_get_pci_ctrl_reg	--	Get the value of the PCIe vendor specific configuration space register.
ezdp_set_pci_ctrl_reg	--	Set the value of the PCIe vendor specific configuration space register.

## 4.12 Pool Operations (ezdp\_pool.h)

EZdp provides the following API routines for operating with pools, including user-defined index pool and memory pool. The API supports simple and optimized operations for the common case of allocating/freeing a single index, and in addition also supports more advanced operations for allocating/freeing multiple indexes in a single command (up to 8 indexes per command).

### 4.12.1 Index Pool Operations

API Routine	Async	Description
ezdp_alloc_index	--	Allocate a single index from an index pool.
ezdp_free_index	+	Free a single index from an index pool.
ezdp_alloc_multi_index	+	Allocate multiple indexes from an index pool.
ezdp_free_multi_index	+	Free multiple indexes from an index pool.
ezdp_read_free_indexes	--	The number of indexes available to be obtained.

### 4.12.2 Memory Pool Operations

API Routine	Async	Description
ezdp_init_memory_pool	--	Initialize a memory pool.
ezdp_alloc_obj	--	Allocate a single object from a memory pool.
ezdp_free_obj	--	Free a single object from a memory pool.
ezdp_get_obj	--	Get object based on object id.
ezdp_read_free_objs	--	The number of objects available to be obtained.

## 4.13 Processor Control Operations (ezdp\_processor.h)

EZdp provides API routines for controlling the CTOP processors.

### 4.13.1 Identification Operations

EZdp provides the following API routines for CTOP processor identification.

API Routine	Async	Description
ezdp_get_cpu_id	--	Get the logical ID of the processor that the process is running on (0-4095).
ezdp_get_thread_id	--	Get the ID of the thread (within the core) that the process is running on (0-15).
ezdp_get_core_id	--	Get the ID of the core (within the cluster) that the process is running on (0-15).
ezdp_get_cluster_id	--	Get the ID of the cluster that the process is running on (0-15).
ezdp_calc_cpu_id	--	Calculate the logical ID of a processor.

### 4.13.2 Synchronization and Scheduling Operations

EZdp provides the following API routines for CTOP processor synchronization and scheduling.

API Routine	Async	Description
ezdp_sync	--	Relinquish the execution unit until all outstanding transactions complete.
ezdp_rsync	--	Relinquish the execution unit until all outstanding read transactions complete.
ezdp_mb	--	Wait until all outstanding memory accesses complete.
ezdp_rmb	--	Wait until all outstanding memory read accesses complete.
ezdp_wmb	--	Wait until all outstanding memory write accesses complete.

## 4.14 Queue Operations (ezdp\_queue.h)

EZdp provides the following API routines for operating with queues: ring (array queue) and list queue.

### 4.14.1 Ring (Array Queue) Operations

API Routine	Async	Description
ezdp_init_ring	--	Initialize ring.
ezdp_ring_empty	--	Check if ring is empty.
ezdp_ring_full	--	Check if array_queue is full.
ezdp_ring_length	--	Return the number of entries in ring.
ezdp_enqueue_ring	--	Insert new entry to ring.
ezdp_dequeue_ring	--	Remove a head entry from ring.

### 4.14.2 List Queue Operations

API Routine	Async	Description
ezdp_init_list	--	Initialize list
ezdp_list_empty	--	Check if a list is empty.
ezdp_enqueue_list	--	Insert a new entry into a list.
ezdp_dequeue_list	--	Remove a head entry from a list.
ezdp_peek_list	--	Peek at head entry of a list.
ezdp_destroy_list	--	Destroy a list.

## 4.15 Search Structure Operations (ezdp\_search.h)

EZdp provides API routines for operating on search structures such as lookup operations and entry management operations (e.g. adding, updating and deleting entries). Separate routines are provided for each of the supported search structure types (direct table, hash, UltraIP and TCAM).

### 4.15.1 Direct Table Structures

EZdp provides the following API routines for operating with direct table structures.

API Routine	Async	Description
ezdp_init_table_struct_desc	--	Initialize the structure descriptor for a table structure.
ezdp_validate_table_struct_desc	--	Validate the table structure parameters.
ezdp_lookup_table_entry	--	Lookup an entry in a table structure.
ezdp_add_table_entry	--	Add an entry in a table structure.
ezdp_modify_table_entry	--	Modify an existing entry in a table structure.
ezdp_update_table_entry	--	Update an entry in a table structure. Add the entry if it does not exist, otherwise modify it.
ezdp_delete_table_entry	--	Delete an entry from a table structure.

### 4.15.2 Hash Structures

EZdp provides the following API routines for operating with hash structures.

API Routine	Async	Description
ezdp_init_hash_struct_desc	--	Initialize the structure descriptor for a hash structure.
ezdp_validate_hash_struct_desc	--	Validate the hash structure parameters
ezdp_lookup_hash_entry	--	Lookup an entry in a hash structure.
ezdp_lookup_hash_entry_ctx	--	Lookup an entry in a hash structure while maintaining operation context.
ezdp_add_hash_entry	--	Add an entry in a hash structure.
ezdp_modify_hash_entry	--	Modify an existing entry in a hash structure.
ezdp_update_hash_entry	--	Update an entry in a hash structure. Add the entry if it does not exist, otherwise modify it.
ezdp_delete_hash_entry	--	Delete an entry from a hash structure.
ezdp_scan_hash_slot	--	Scan a hash slot.
get_hash_entry_key	--	Get hash key from entry.

### 4.15.3 UltraIP Structures

EZdp provides the following API routines for operating with UltraIP structures.

API Routine	Async	Description
ezdp_init_ultra_ip_struct_desc	--	Initialize the structure descriptor for an UltraIP structure.
ezdp_validate_ultra_ip_struct_desc	--	Validate the UltraIP structure parameters.
ezdp_lookup_ultra_ip_entry	--	Lookup an entry in an UltraIP structure.



### 4.15.4 TCAM Structures

EZdp provides the following API routines for operating with the ternary CAMs (TCAMs).

API Routine	Async	Description
ezdp_lookup_int_tcam	+	Lookup an entry in an internal TCAM.
ezdp_lookup_ext_tcam	+	Lookup an entry in an external TCAM.

### 4.15.5 Algorithmic TCAM Structures

EZdp provides the following API routines for operating with algorithmic TCAM search structures.

API Routine	Async	Description
ezdp_init_alg_tcam_struct_desc	--	Initialize the structure descriptor for an algorithmic TCAM structure.
ezdp_validate_alg_tcam_struct_desc	--	Validate the algorithmic TCAM structure parameters.
ezdp_lookup_alg_tcam	--	Lookup an entry in an algorithmic TCAM structure.

## 4.16 Primitive Search Structure Operations (ezdp\_search\_prm.h)

EZdp provides API routines for advanced low-level operations on search structures.

### 4.16.1 Direct Table Structures

EZdp provides the following API routines for operating with direct table structures.

API Routine	Async	Description
ezdp_prm_lock_table_line	--	Lock a table entry
ezdp_prm_trylock_table_line	--	Try to lock a table entry
ezdp_prm_unlock_table_line	--	Release a table entry lock
ezdp_prm_get_table_base_addr	--	Get search base address (used for lookup) from table struct descriptor
ezdp_prm_lookup_table_entry	--	Lookup an entry in a table structure.
ezdp_prm_update_table_entry	--	Add a new entry in a table structure. Override the existing entry.
ezdp_prm_delete_table_entry	--	Delete an entry from a table structure.

### 4.16.2 Hash Structures

EZdp provides the following API routines for operating with direct table structures.

API Routine	Async	Description
ezdp_prm_hash_key32	--	Calculate hash value for keys of up to 32 bits.
ezdp_prm_hash_key64	--	Calculate hash value for keys of up to 64 bits.
ezdp_prm_hash_bulk_key	--	Calculate hash value for keys > 8 bytes.
ezdp_prm_lock_hash_slot	--	Lock a hash slot according to the hashed key.
ezdp_prm_trylock_hash_slot	--	Try to lock a hash slot according to the hashed key.
ezdp_prm_unlock_hash_slot	--	Release a hash slot lock.
ezdp_prm_get_hash_base_addr	--	Get search base address (used for lookup) from hash struct descriptor.
ezdp_prm_lookup_hash_entry	--	Lookup an entry in a hash structure.
ezdp_prm_locate_hash_entry	--	Lookup an entry location in a hash structure.
ezdp_prm_add_hash_entry	--	Add a new entry in a hash structure.
ezdp_prm_modify_hash_entry	--	Modify an existing entry in a hash structure. entry_ptr should be updated with the new result.
ezdp_prm_delete_hash_entry	--	Delete an existing entry from a hash structure.
ezdp_prm_get_hash_first_entry	--	Get first entry of a hash slot.
ezdp_prm_get_hash_next_entry	--	Get next entry of a hash slot.
ezdp_prm_compress_hash_entry	--	Compress a hash entry with other entries in the same hash slot, if possible.

### 4.16.3 UltraIP Structures

EZdp provides the following API routines for operating with UltraIP structures.

API Routine	Async	Description
ezdp_prm_get_ultra_ip_base_addr	--	Get search base address (used for lookup) from UltraIP struct descriptor
ezdp_prm_lookup_ultra_ip_entry	--	Lookup an entry in an UltraIP structure.

#### 4.16.4 Algorithmic TCAM Structures

EZdp provides the following API routines for operating with algorithmic TCAM search structures.

API Routine	Async	Description
ezdp_prm_lookup_alg_tcam	--	Lookup an entry in an algorithmic TCAM structure.

## 4.17 Security Operations (ezdp\_security.h)

EZdp provides API routines for operating NPS security accelerators.

API Routine	Async	Description
ezdp_encrypt	+	Encrypt a data segment.
ezdp_decrypt	+	Decrypt a data segment.
ezdp_mac_calculation	+	Calculate the message authentication code (MAC) on a data segment.
ezdp_start_hmac_calculation	+	Start a hash-based message authentication code (MAC) calculation.
ezdp_end_hmac_calculation	+	Complete a hash-based message authentication code (MAC) calculation.
ezdp_generate_security_initial_vector	+	Generate security initial vector.
ezdp_end_gcm_mac_calculation	+	Complete a GCM hash-based message authentication code (MAC) calculation.
ezdp_expand_security_key	+	Expands the key in the security context memory.
ezdp_write_security_state	+	Copy the security state data from CMEM to the security context memory.
ezdp_read_security_state	+	Copy the security state data from the security context memory to CMEM.
ezdp_security_state_size	--	Return the state size.
ezdp_write_security_key	+	Copy the security key from CMEM to the security context memory.
ezdp_read_security_key	+	Copy the security key from the security context memory to CMEM.
ezdp_security_key_size	--	Return the key size.
ezdp_write_security_mac	+	Copy the security mac from CMEM to the security context memory.
ezdp_read_security_mac	+	Copy the security mac from the security context memory to CMEM.
ezdp_security_mac_size	--	Return the MAC size.
ezdp_write_security_initial_vector	+	Copy the security initial vector from CMEM to the security context memory.
ezdp_read_security_initial_vector	+	Copy the security initial vector from the security context memory to CMEM.
ezdp_security_initial_vector_size	--	Return the initial vector size.
ezdp_write_security_context	+	Copy the security context from CMEM to the security context memory.
ezdp_read_security_context	+	Copy the security context from the security context memory to CMEM.
ezdp_security_block_size	--	Return the algorithmic engine minimal block size.

## 4.18 String Operations (ezdp\_string.h)

EZdp provides API routines for manipulating character arrays.

API Routine	Async	Description
ezdp_mem_copy	--	Copy a block of memory.
ezdp_mem_set	--	Set a block of memory to the specified value.
ezdp_mem_cmp	--	Compare two blocks of memory in CMEM.
ezdp_mem_cmp_byte_skip	--	Compare two blocks of memory in CMEM, skipping intermediate bytes.

## 4.19 Time Operations (ezdp\_time.h)

EZdp provides API routines for retrieving network and system time.

API Routine	Async	Description
ezdp_get_system_tick	+	Get system tick (in core cycles).
ezdp_get_real_time_clock	+	Get real time clock.

## 5. Reference

The pages that follow list the API routines as well as their structures and enumerations.

# Table of Contents

Data Structure Index.....	4
File Index.....	7
Data Structure Documentation .....	8
ezdp_1588_header.....	8
ezdp_1step_1588_header .....	9
ezdp_2step_1588_header .....	11
ezdp_app_schlr_status.....	13
ezdp_bitwise_ctr_cfg.....	14
ezdp_buffer_desc.....	15
ezdp_buffer_info .....	16
ezdp_congestion_status .....	17
ezdp_ctr_msg.....	19
ezdp_decode_eth_type_retval .....	21
ezdp_decode_ip_next_protocol .....	24
ezdp_decode_ip_protocol_retval.....	26
ezdp_decode_ipv4_control.....	29
ezdp_decode_ipv4_errors.....	31
ezdp_decode_ipv4_result .....	33
ezdp_decode_ipv4_retval.....	35
ezdp_decode_ipv6_control.....	37
ezdp_decode_ipv6_errors.....	39
ezdp_decode_ipv6_result .....	41
ezdp_decode_ipv6_retval .....	43
ezdp_decode_mac_control .....	45
ezdp_decode_mac_errors .....	48
ezdp_decode_mac_protocol_type .....	50
ezdp_decode_mac_result.....	53
ezdp_decode_mac_retval .....	55
ezdp_decode_mpls_label_result.....	57
ezdp_decode_mpls_label_retval.....	59
ezdp_decode_mpls_result.....	61
ezdp_decode_mpls_retval .....	64
ezdp_decode_tcp_errors .....	67
ezdp_decode_tcp_retval .....	68
ezdp_driver_desc.....	69
ezdp_driver_desc_flags .....	70
ezdp_dual_add32_result .....	71
ezdp_dual_add64_result.....	72
ezdp_dual_ctr .....	73
ezdp_dual_ctr_cfg .....	74
ezdp_dual_ctr_result.....	76
ezdp_ext_addr .....	77
ezdp_ext_linked_buffers_desc .....	79
ezdp_flow_control_status.....	80
ezdp_frame_desc .....	81
ezdp_group_schlr_status .....	85
ezdp_hier_tb_ctr_cfg.....	86
ezdp_hier_tb_result .....	88
ezdp_hier_tb_ug_app_bits.....	90
ezdp_hier_tb_update.....	92
ezdp_input_queue_status.....	94
ezdp_job_container_cmd_desc.....	96
ezdp_job_container_desc .....	98
ezdp_job_desc .....	100
ezdp_job_discard_cmd_info.....	101
ezdp_job_queue_cmd_info.....	102
ezdp_job_rx_confirmation_info .....	103

ezdp_job_rx_info.....	104
ezdp_job_rx_interface_info.....	107
ezdp_job_rx_loopback_info.....	110
ezdp_job_rx_timer_info.....	111
ezdp_job_rx_user_info.....	112
ezdp_job_transmit_cmd_info.....	113
ezdp_job_tx_info.....	114
ezdp_large_linked_buffers_desc.....	119
ezdp_linked_buffers_desc.....	120
ezdp_linked_buffers_desc_line.....	121
ezdp_list_cfg.....	122
ezdp_lookup_ext_tcaml6B_data_result_element.....	123
ezdp_lookup_ext_tcaml32B_data_result_element.....	125
ezdp_lookup_ext_tcaml4B_data_result_element.....	127
ezdp_lookup_ext_tcaml8B_data_result_element.....	129
ezdp_lookup_ext_tcamlindex_l6B_data_result_element.....	131
ezdp_lookup_ext_tcamlindex_l32B_data_result_element.....	133
ezdp_lookup_ext_tcamlindex_l4B_data_result_element.....	135
ezdp_lookup_ext_tcamlindex_l8B_data_result_element.....	137
ezdp_lookup_ext_tcamlindex_result_element.....	139
ezdp_lookup_ext_tcamlretval.....	141
ezdp_lookup_int_tcaml12B_data_result.....	143
ezdp_lookup_int_tcaml16B_data_result.....	144
ezdp_lookup_int_tcaml4B_data_result.....	145
ezdp_lookup_int_tcaml8B_data_result.....	146
ezdp_lookup_int_tcamlresult.....	147
ezdp_lookup_int_tcamlretval.....	148
ezdp_lookup_int_tcamlstandard_result.....	149
ezdp_lookup_retval.....	150
ezdp_mem_pool_config.....	151
ezdp_mem_section_info.....	152
ezdp_output_queue_status.....	155
ezdp_pci_addr.....	156
ezdp_pci_info.....	158
ezdp_pci_msg.....	159
ezdp_pci_msg_ctrl.....	160
ezdp_pci_msg_payload_ats.....	161
ezdp_pci_msg_payload_elbi.....	162
ezdp_pci_msg_payload_msix.....	163
ezdp_posted_ctr_msg.....	164
ezdp_ring_cfg.....	166
ezdp_rtc.....	167
ezdp_security_handle.....	168
ezdp_single_ctr_cfg.....	169
ezdp_small_linked_buffers_desc.....	171
ezdp_sum_addr.....	172
ezdp_sum_addr_table_desc.....	173
ezdp_tb_ctr_cfg.....	175
ezdp_tb_ctr_result.....	177
ezdp_version.....	179
ezdp_watchdog_accumulative_window_cfg.....	181
ezdp_watchdog_ctr_cfg.....	183
ezdp_watchdog_ctr_check_result.....	185
ezdp_watchdog_ctr_start_result.....	187
ezdp_watchdog_sliding_window_cfg.....	188
File Documentation.....	190
dpe/dp/include/ezdp.h.....	190
dpe/dp/include/ezdp_atomic.h.....	195
dpe/dp/include/ezdp_counter.h.....	228
dpe/dp/include/ezdp_counter_defs.h.....	265



dpe/dp/include/ezdp_decode.h .....	295
dpe/dp/include/ezdp_decode_defs.h .....	300
dpe/dp/include/ezdp_defs.h .....	345
dpe/dp/include/ezdp_dma.h .....	348
dpe/dp/include/ezdp_frame.h .....	359
dpe/dp/include/ezdp_frame_defs.h .....	378
dpe/dp/include/ezdp_job.h .....	391
dpe/dp/include/ezdp_job_defs.h .....	409
dpe/dp/include/ezdp_lock.h .....	435
dpe/dp/include/ezdp_lock_defs.h .....	440
dpe/dp/include/ezdp_math.h .....	441
dpe/dp/include/ezdp_memory.h .....	459
dpe/dp/include/ezdp_memory_defs.h .....	462
dpe/dp/include/ezdp_pci.h .....	472
dpe/dp/include/ezdp_pci_defs.h .....	484
dpe/dp/include/ezdp_pool.h .....	493
dpe/dp/include/ezdp_pool_defs.h .....	497
dpe/dp/include/ezdp_processor.h .....	498
dpe/dp/include/ezdp_queue.h .....	501
dpe/dp/include/ezdp_queue_defs.h .....	505
dpe/dp/include/ezdp_search.h .....	506
dpe/dp/include/ezdp_search_defs.h .....	518
dpe/dp/include/ezdp_search_prm.h .....	554
dpe/dp/include/ezdp_security.h .....	563
dpe/dp/include/ezdp_security_defs.h .....	577
dpe/dp/include/ezdp_string.h .....	584
dpe/dp/include/ezdp_time.h .....	586
dpe/dp/include/ezdp_time_defs.h .....	588
dpe/dp/include/ezdp_version.h .....	589
Index .....	590

# Data Structure Index

## Data Structures

Here are the data structures with brief descriptions:

<a href="#"><u>ezdp_1588_header</u></a> (1588 header format definition )	8
<a href="#"><u>ezdp_1step_1588_header</u></a> (1-step 1588 header format definition )	9
<a href="#"><u>ezdp_2step_1588_header</u></a> (2-step 1588 header format definition )	11
<a href="#"><u>ezdp_app_schlr_status</u></a> (PMU application scheduler status (based on PMU system info) )	13
<a href="#"><u>ezdp_bitwise_ctr_cfg</u></a> (On-demand bitwise counter configuration definition )	14
<a href="#"><u>ezdp_buffer_desc</u></a> (Buffer descriptor (BD) data structure )	15
<a href="#"><u>ezdp_buffer_info</u></a> (Buffer descriptor info )	16
<a href="#"><u>ezdp_congestion_status</u></a> (System priority drop congestion status )	17
<a href="#"><u>ezdp_ctr_msg</u></a> (Counter message queue definition )	19
<a href="#"><u>ezdp_decode_eth_type_retval</u></a> (Decode ip protocol return value struct definition )	21
<a href="#"><u>ezdp_decode_ip_next_protocol</u></a> (IP protocol type flags )	24
<a href="#"><u>ezdp_decode_ip_protocol_retval</u></a> (Decode ip protocol return value struct definition )	26
<a href="#"><u>ezdp_decode_ipv4_control</u></a> (IPv4 addresses decoding result )	29
<a href="#"><u>ezdp_decode_ipv4_errors</u></a> (IPv4 header decode error flags )	31
<a href="#"><u>ezdp_decode_ipv4_result</u></a> (Decode IPv4 result )	33
<a href="#"><u>ezdp_decode_ipv4_retval</u></a> (Decode IPv4 return value struct definition )	35
<a href="#"><u>ezdp_decode_ipv6_control</u></a> (IPv6 addresses decoding result )	37
<a href="#"><u>ezdp_decode_ipv6_errors</u></a> (IPv6 header decode error flags )	39
<a href="#"><u>ezdp_decode_ipv6_result</u></a> (Decode IPv6 result )	41
<a href="#"><u>ezdp_decode_ipv6_retval</u></a> (Decode IPv4 return value struct definition )	43
<a href="#"><u>ezdp_decode_mac_control</u></a> (MAC addresses decoding result )	45
<a href="#"><u>ezdp_decode_mac_errors</u></a> (MAC header decode error flags )	48
<a href="#"><u>ezdp_decode_mac_protocol_type</u></a> (Ethernet type definition )	50
<a href="#"><u>ezdp_decode_mac_result</u></a> (Decode MAC result )	53
<a href="#"><u>ezdp_decode_mac_retval</u></a> (Decode MAC return value struct definition )	55
<a href="#"><u>ezdp_decode_mpls_label_result</u></a> (Ezdp_decode_mpls_label_result struct for ezdp )	57
<a href="#"><u>ezdp_decode_mpls_label_retval</u></a> (Decode MPLS label return value struct definition )	59
<a href="#"><u>ezdp_decode_mpls_result</u></a> (Ezdp_decode_mpls_result struct for ezdp )	61
<a href="#"><u>ezdp_decode_mpls_retval</u></a> (Decode MPLS return value struct definition )	64
<a href="#"><u>ezdp_decode_tcp_errors</u></a> (TCP header decode error flags )	67
<a href="#"><u>ezdp_decode_tcp_retval</u></a> (Decode TCP return value struct definition )	68
<a href="#"><u>ezdp_driver_desc</u></a> (TX/RX descriptor )	69
<a href="#"><u>ezdp_driver_desc_flags</u></a> (TX/RX descriptor flags structure )	70
<a href="#"><u>ezdp_dual_add32_result</u></a> (The result of the atomic dual add32 instruction )	71
<a href="#"><u>ezdp_dual_add64_result</u></a> (The result of the atomic dual add64 instruction )	72
<a href="#"><u>ezdp_dual_ctr</u></a> (On-demand dual counter value )	73
<a href="#"><u>ezdp_dual_ctr_cfg</u></a> (On-demand dual counter configuration definition )	74
<a href="#"><u>ezdp_dual_ctr_result</u></a> (On-demand dual value counter result value )	76
<a href="#"><u>ezdp_ext_addr</u></a> (Extended address definition )	77
<a href="#"><u>ezdp_ext_linked_buffers_desc</u></a> (Extended linked buffers descriptor )	79
<a href="#"><u>ezdp_flow_control_status</u></a> (Flow control status )	80
<a href="#"><u>ezdp_frame_desc</u></a> (Frame descriptor data structure )	81
<a href="#"><u>ezdp_group_schlr_status</u></a> (PMU group scheduler status (based on PMU system info) )	85

<a href="#"><u>ezdp_hier_tb_ctr_cfg</u></a> (Statistic hierarchical token bucket counter config structure (write cfg usage) )	86
<a href="#"><u>ezdp_hier_tb_result</u></a> (Hierarchical token bucket counter result value definition )	88
<a href="#"><u>ezdp_hier_tb_ug_app_bits</u></a> (Application bits of Hierarchical token bucket for ultra green feature )	90
<a href="#"><u>ezdp_hier_tb_update</u></a> (Hierarchical token bucket update counter definition )	92
<a href="#"><u>ezdp_input_queue_status</u></a> (PMU physical input queue status definition (based on PMU system info) )	94
<a href="#"><u>ezdp_job_container_cmd_desc</u></a> (Job container request )	96
<a href="#"><u>ezdp_job_container_desc</u></a> (Job container descriptor )	98
<a href="#"><u>ezdp_job_desc</u></a> (Job descriptor data structure )	100
<a href="#"><u>ezdp_job_discard_cmd_info</u></a> (Job container discard request info )	101
<a href="#"><u>ezdp_job_queue_cmd_info</u></a> (Job container send to queue request info )	102
<a href="#"><u>ezdp_job_rx_confirmation_info</u></a> (Info field for incoming job from TX confirmation ports )	103
<a href="#"><u>ezdp_job_rx_info</u></a> (Job receive info )	104
<a href="#"><u>ezdp_job_rx_interface_info</u></a> (Info field for incoming job from external RX interfaces )	107
<a href="#"><u>ezdp_job_rx_loopback_info</u></a> (Info field for incoming job from loopback ports )	110
<a href="#"><u>ezdp_job_rx_timer_info</u></a> (Info field for incoming timer job (PMU Timer) )	111
<a href="#"><u>ezdp_job_rx_user_info</u></a> (Info field for incoming frame job from generic user forwarding )	112
<a href="#"><u>ezdp_job_transmit_cmd_info</u></a> (Job container send out request info )	113
<a href="#"><u>ezdp_job_tx_info</u></a> (Info field for transmitting frame job (TM mode is full or tm qos bypass) )	114
<a href="#"><u>ezdp_large_linked_buffers_desc</u></a> (Large linked buffers descriptor )	119
<a href="#"><u>ezdp_linked_buffers_desc</u></a> (A generic linked buffers descriptor )	120
<a href="#"><u>ezdp_linked_buffers_desc_line</u></a> (LBD Line data structure )	121
<a href="#"><u>ezdp_list_cfg</u></a> (List queue configuration data structure )	122
<a href="#"><u>ezdp_lookup_ext_tcaml6B_data_result_element</u></a> (Lookup external tcaml6 Byte associated data only result )	123
<a href="#"><u>ezdp_lookup_ext_tcaml32B_data_result_element</u></a> (Lookup external tcaml32 Byte associated data only result )	125
<a href="#"><u>ezdp_lookup_ext_tcaml4B_data_result_element</u></a> (Lookup external tcaml4 Byte associated data only result )	127
<a href="#"><u>ezdp_lookup_ext_tcaml8B_data_result_element</u></a> (Lookup external tcaml8 Byte associated data only result )	129
<a href="#"><u>ezdp_lookup_ext_tcaml_index_16B_data_result_element</u></a> (Lookup external tcamlindex result with 16 Byte associated data )	131
<a href="#"><u>ezdp_lookup_ext_tcaml_index_32B_data_result_element</u></a> (Lookup external tcamlindex result with 32 Byte associated data )	133
<a href="#"><u>ezdp_lookup_ext_tcaml_index_4B_data_result_element</u></a> (Lookup external tcamlindex result with 4 Byte associated data )	135
<a href="#"><u>ezdp_lookup_ext_tcaml_index_8B_data_result_element</u></a> (Lookup external tcamlindex result with 8 Byte associated data )	137
<a href="#"><u>ezdp_lookup_ext_tcaml_index_result_element</u></a> (Lookup external tcamlindex result element )	139
<a href="#"><u>ezdp_lookup_ext_tcaml_retval</u></a> (Lookup external tcamlreturn value )	141
<a href="#"><u>ezdp_lookup_int_tcaml12B_data_result</u></a> (Lookup internal tcaml12 byte associated data result )	143
<a href="#"><u>ezdp_lookup_int_tcaml16B_data_result</u></a> (Lookup internal tcaml16 byte associated data result )	144
<a href="#"><u>ezdp_lookup_int_tcaml4B_data_result</u></a> (Lookup internal tcaml4 byte associated data result )	145
<a href="#"><u>ezdp_lookup_int_tcaml8B_data_result</u></a> (Lookup internal tcaml8 byte associated data result )	146
<a href="#"><u>ezdp_lookup_int_tcaml_result</u></a> (Lookup ITCAM result definition )	147
<a href="#"><u>ezdp_lookup_int_tcaml_retval</u></a> (Lookup ITCAM retval definition )	148
<a href="#"><u>ezdp_lookup_int_tcaml_standard_result</u></a> (Lookup internal tcamlstandard result )	149
<a href="#"><u>ezdp_lookup_retval</u></a> (Lookup return value )	150

<a href="#"><u>ezdp_mem_pool_config</u></a> (Memory pool configuration data structure )	151
<a href="#"><u>ezdp_mem_section_info</u></a>	152
<a href="#"><u>ezdp_output_queue_status</u></a> (PMU output queue status definition (based on PMU system info) )	155
<a href="#"><u>ezdp_pci_addr</u></a> (PCI Address data structure )	156
<a href="#"><u>ezdp_pci_info</u></a> (PCI info for describing to which endpoint, physical function, virtual function and queue the frame is to be sent )	158
<a href="#"><u>ezdp_pci_msg</u></a> (Message from PCI queue )	159
<a href="#"><u>ezdp_pci_msg_ctrl</u></a> (PCI message control )	160
<a href="#"><u>ezdp_pci_msg_payload_ats</u></a> (PCI ATS message payload )	161
<a href="#"><u>ezdp_pci_msg_payload_elbi</u></a> (PCI ELBI message payload )	162
<a href="#"><u>ezdp_pci_msg_payload_msix</u></a> (PCI MSIX message payload )	163
<a href="#"><u>ezdp_posted_ctr_msg</u></a> (Posted counter message queue definition )	164
<a href="#"><u>ezdp_ring_cfg</u></a> (Ring (array queue) configuration data structure )	166
<a href="#"><u>ezdp_rtc</u></a> (Ezdp_rtc struct for ezdp )	167
<a href="#"><u>ezdp_security_handle</u></a> (Security handle configuration data structure )	168
<a href="#"><u>ezdp_single_ctr_cfg</u></a> (On-demand single value counter configuration definition )	169
<a href="#"><u>ezdp_small_linked_buffers_desc</u></a> (Small linked buffers descriptor.May hold up to 3 buffs )	171
<a href="#"><u>ezdp_sum_addr</u></a> (Summarized Address data structure )	172
<a href="#"><u>ezdp_sum_addr_table_desc</u></a> (Structure definition table entry data structure )	173
<a href="#"><u>ezdp_tb_ctr_cfg</u></a> (Token bucket counter configuration definition )	175
<a href="#"><u>ezdp_tb_ctr_result</u></a> (Token bucket counter result value definition )	177
<a href="#"><u>ezdp_version</u></a> (Version info data structure )	179
<a href="#"><u>ezdp_watchdog_accumulative_window_cfg</u></a> (Watchdog accumulative window configuration definition )	181
<a href="#"><u>ezdp_watchdog_ctr_cfg</u></a> (Watchdog counter configuration definition )	183
<a href="#"><u>ezdp_watchdog_ctr_check_result</u></a> (Watchdog counter check result definition )	185
<a href="#"><u>ezdp_watchdog_ctr_start_result</u></a> (Watchdog counter check result definition )	187
<a href="#"><u>ezdp_watchdog_sliding_window_cfg</u></a> (Watchdog sliding window configuration definition )	188

# File Index

## File List

Here is a list of all files with brief descriptions:

dpe/dp/include/ <a href="#">ezdp.h</a>	190
dpe/dp/include/ <a href="#">ezdp_atomic.h</a>	195
dpe/dp/include/ <a href="#">ezdp_counter.h</a>	228
dpe/dp/include/ <a href="#">ezdp_counter_defs.h</a>	265
dpe/dp/include/ <a href="#">ezdp_decode.h</a>	295
dpe/dp/include/ <a href="#">ezdp_decode_defs.h</a>	300
dpe/dp/include/ <a href="#">ezdp_defs.h</a>	345
dpe/dp/include/ <a href="#">ezdp_dma.h</a>	348
dpe/dp/include/ <a href="#">ezdp_frame.h</a>	359
dpe/dp/include/ <a href="#">ezdp_frame_defs.h</a>	378
dpe/dp/include/ <a href="#">ezdp_job.h</a>	391
dpe/dp/include/ <a href="#">ezdp_job_defs.h</a>	409
dpe/dp/include/ <a href="#">ezdp_lock.h</a>	435
dpe/dp/include/ <a href="#">ezdp_lock_defs.h</a>	440
dpe/dp/include/ <a href="#">ezdp_math.h</a>	441
dpe/dp/include/ <a href="#">ezdp_memory.h</a>	459
dpe/dp/include/ <a href="#">ezdp_memory_defs.h</a>	462
dpe/dp/include/ <a href="#">ezdp_pci.h</a>	472
dpe/dp/include/ <a href="#">ezdp_pci_defs.h</a>	484
dpe/dp/include/ <a href="#">ezdp_pool.h</a>	493
dpe/dp/include/ <a href="#">ezdp_pool_defs.h</a>	497
dpe/dp/include/ <a href="#">ezdp_processor.h</a>	498
dpe/dp/include/ <a href="#">ezdp_queue.h</a>	501
dpe/dp/include/ <a href="#">ezdp_queue_defs.h</a>	505
dpe/dp/include/ <a href="#">ezdp_search.h</a>	506
dpe/dp/include/ <a href="#">ezdp_search_defs.h</a>	518
dpe/dp/include/ <a href="#">ezdp_search_prm.h</a>	554
dpe/dp/include/ <a href="#">ezdp_security.h</a>	563
dpe/dp/include/ <a href="#">ezdp_security_defs.h</a>	577
dpe/dp/include/ <a href="#">ezdp_string.h</a>	584
dpe/dp/include/ <a href="#">ezdp_time.h</a>	586
dpe/dp/include/ <a href="#">ezdp_time_defs.h</a>	588
dpe/dp/include/ <a href="#">ezdp_version.h</a>	589

# Data Structure Documentation

## ezdp\_1588\_header Struct Reference

1588 header format definition

### Data Fields

- union {
- struct {
- unsigned [pad0](#) : 7
- *Reserved bits 25 to 31.*    unsigned [pad1](#) : 24
- *< Determine the action to process for an expanded job    >*
- } u
- union {
- struct [ezdp\\_1step\\_1588\\_header\\_one\\_step](#)
- *Job transmit request info.*    struct [ezdp\\_2step\\_1588\\_header\\_two\\_step](#)
- *Job dispatch request info.*    } [u](#)
- };

### Detailed Description

1588 header format definition

### Field Documentation

unsigned [ezdp\\_1588\\_header::pad0](#)

Reserved bits 25 to 31.

unsigned [ezdp\\_1588\\_header::pad1](#)

Determine the action to process for an expanded job

Reserved bits 0 to 23

struct [ezdp\\_1step\\_1588\\_header ezdp\\_1588\\_header::one\\_step](#) [read]

Job transmit request info.

struct [ezdp\\_2step\\_1588\\_header ezdp\\_1588\\_header::two\\_step](#) [read]

Job dispatch request info.

union { ... } [ezdp\\_1588\\_header::u](#)

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_1step\_1588\_header Struct Reference

1-step 1588 header format definition

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_1STEP\_1588\_HEADER\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_1STEP\_1588\_HEADER\_RESERVED28\_31\_SIZE
- *Reserved bits 28 to 31.*   unsigned [correction\\_odd\\_start](#):  
EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_ODD\_START\_SIZE
- *Correction field start from odd byte.*   unsigned [wrap\\_around\\_condition](#):  
EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_SIZE
- *Wrap-around condition detected.*   unsigned [inject\\_checksum\\_flag](#):  
EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_SIZE
- *Checksum injection required.*   unsigned [pad1](#) : EZDP\_1STEP\_1588\_HEADER\_RESERVED24\_SIZE
- *Reserved bit 24.*   unsigned [pad2](#) : EZDP\_1STEP\_1588\_HEADER\_RESERVED16\_23\_SIZE
- *Reserved bits 16 to 23.*   uint16\_t [checksum](#)
- *Intermediate/Partial checksum field.*   uint16\_t [checksum\\_offset](#)
- *Offset of checksum field inside PTP frame.*   uint16\_t [correction\\_offset](#)
- *Offset of correction field inside the PTP frame.*   uint64\_t [correction](#)
- *Intermediate/Partial correction field.*   }
- };

### Detailed Description

1-step 1588 header format definition

### Field Documentation

uint32\_t [ezdp\\_1step\\_1588\\_header::raw\\_data](#)[EZDP\_1STEP\_1588\_HEADER\_WORD\_COUNT]

unsigned [ezdp\\_1step\\_1588\\_header::pad0](#)

Reserved bits 28 to 31.

unsigned [ezdp\\_1step\\_1588\\_header::correction\\_odd\\_start](#)

Correction field start from odd byte.

unsigned [ezdp\\_1step\\_1588\\_header::wrap\\_around\\_condition](#)

Wrap-around condition detected.

unsigned [ezdp\\_1step\\_1588\\_header::inject\\_checksum\\_flag](#)

Checksum injection required.

unsigned [ezdp\\_1step\\_1588\\_header::pad1](#)

Reserved bit 24.

unsigned [ezdp\\_1step\\_1588\\_header::pad2](#)

Reserved bits 16 to 23.

uint16\_t [ezdp\\_1step\\_1588\\_header::checksum](#)

Intermediate/Partial checksum field.

uint16\_t [ezdp\\_1step\\_1588\\_header::checksum\\_offset](#)

Offset of checksum field inside PTP frame.

uint16\_t [ezdp\\_1step\\_1588\\_header::correction\\_offset](#)

Offset of correction field inside the PTP frame.

uint64\_t [ezdp\\_1step\\_1588\\_header::correction](#)

Intermediate/Partial correction field.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)



## ezdp\_2step\_1588\_header Struct Reference

2-step 1588 header format definition

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_2STEP\_1588\_HEADER\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_2STEP\_1588\_HEADER\_RESERVED24\_31\_SIZE
- Reserved bits 24 to 31.    unsigned [\\_\\_pad1](#) : EZDP\_2STEP\_1588\_HEADER\_RESERVED24\_SIZE
- Reserved for TS mode flag.   unsigned [\\_\\_pad2](#) : EZDP\_2STEP\_1588\_HEADER\_RESERVED0\_23\_SIZE
- Reserved bits 0 to 23.    unsigned [\\_\\_pad3](#) : EZDP\_2STEP\_1588\_HEADER\_RESERVED32\_63\_SIZE
- Reserved bits 32 to 63.    uint8\_t [free\\_bytes](#)
- This field indicates how many free bytes are left at the end of the frame header data buffer.    uint8\_t [header\\_offset](#)
- This is the frame header starting point in the first frame data buffer.    unsigned [\\_\\_pad4](#) : EZDP\_2STEP\_1588\_HEADER\_RESERVED76\_77\_SIZE
- Reserved bits 76 to 77.    unsigned [class\\_of\\_service](#): EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_SIZE
- Frame class of service grade.    unsigned [\\_\\_pad5](#) : EZDP\_2STEP\_1588\_HEADER\_RESERVED74\_75\_SIZE
- Reserved bits 74 to 75.    unsigned [buf\\_budget\\_id](#): EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_SIZE
- Budget group ID.    struct [ezdp\\_buffer\\_desc](#) [buf\\_desc](#)
- Pointer to a 256B buffer located either in IMEM or EMEM.    }
- };

### Detailed Description

2-step 1588 header format definition

### Field Documentation

uint32\_t [ezdp\\_2step\\_1588\\_header::raw\\_data](#)[EZDP\_2STEP\_1588\_HEADER\_WORD\_COUNT]

unsigned [ezdp\\_2step\\_1588\\_header:: \\_\\_pad0](#)

Reserved bits 24 to 31.

unsigned [ezdp\\_2step\\_1588\\_header:: \\_\\_pad1](#)

Reserved for TS mode flag.

unsigned [ezdp\\_2step\\_1588\\_header:: \\_\\_pad2](#)

Reserved bits 0 to 23.

unsigned [ezdp\\_2step\\_1588\\_header:: \\_\\_pad3](#)

Reserved bits 32 to 63.

**uint8\_t [ezdp\\_2step\\_1588\\_header::free\\_bytes](#)**

This field indicates how many free bytes are left at the end of the frame header data buffer.

This field is valid only for STANDARD frames.

**uint8\_t [ezdp\\_2step\\_1588\\_header::header\\_offset](#)**

This is the frame header starting point in the first frame data buffer.

To include embedded LBD in the same buffer of the first frame data (EMBEDDED\_BD frame type), the header\_offset must be at least the max configured embedded LBD size (16B or 32B). The gaps from beginning of the buffer or from max embedded LBD size to header\_offset can be utilized by frame context or can be used for optimized header modification (increasing or decreasing).

**unsigned [ezdp\\_2step\\_1588\\_header::pad4](#)**

Reserved bits 76 to 77.

**unsigned [ezdp\\_2step\\_1588\\_header::class\\_of\\_service](#)**

Frame class of service grade.

**unsigned [ezdp\\_2step\\_1588\\_header::pad5](#)**

Reserved bits 74 to 75.

**unsigned [ezdp\\_2step\\_1588\\_header::buf\\_budget\\_id](#)**

Budget group ID.

Budget identifies an allocated IMEM or EMEM buffer resource control operation. By default a budget group ID is associated with an Rx port ID from which the frame was received.

**struct [ezdp\\_buffer\\_desc ezdp\\_2step\\_1588\\_header::buf\\_desc](#) [read]**

Pointer to a 256B buffer located either in IMEM or EMEM.

The type to which the buffer points is determined by frame type.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_app\_schlr\_status Struct Reference

PMU application scheduler status (based on PMU system info).

### Data Fields

- union {
- [ezdp\\_app\\_schlr\\_status\\_t raw\\_data](#)
- struct {
- unsigned [enable](#): EZDP\_APP\_SCHLR\_STATUS\_ENABLE\_SIZE
- *The application scheduler is enabled.* unsigned [busy](#): EZDP\_APP\_SCHLR\_STATUS\_BUSY\_SIZE
- *The application scheduler has pending requests (busy).* unsigned [pad0](#) : EZDP\_APP\_SCHLR\_STATUS\_RESERVED13\_SIZE
- *Reserved bit 13.* unsigned [dispatched\\_job](#): EZDP\_APP\_SCHLR\_STATUS\_DISPATCHED\_JOB\_SIZE
- *The number of jobs dispatched from the application scheduler.* }
- };

### Detailed Description

PMU application scheduler status (based on PMU system info).  
There are 8 application schedulers in each PMU side.

### Field Documentation

[ezdp\\_app\\_schlr\\_status\\_t ezdp\\_app\\_schlr\\_status::raw\\_data](#)

unsigned [ezdp\\_app\\_schlr\\_status::enable](#)

The application scheduler is enabled.

unsigned [ezdp\\_app\\_schlr\\_status::busy](#)

The application scheduler has pending requests (busy).

unsigned [ezdp\\_app\\_schlr\\_status:: pad0](#)

Reserved bit 13.

unsigned [ezdp\\_app\\_schlr\\_status::dispatched\\_job](#)

The number of jobs dispatched from the application scheduler.

Defined as number of jobs that were dispatched from the application scheduler for processing and are waiting for "job done".

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_bitwise\_ctr\_cfg Struct Reference

On-demand bitwise counter configuration definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_BITWISE\_CTR\_CFG\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_BITWISE\_CTR\_CFG\_ECC\_SIZE
- ECC.   unsigned [pad1](#) : EZDP\_BITWISE\_CTR\_CFG\_SUB\_TYPE\_SIZE
- Counter sub type (bitwise=9).   unsigned [pad2](#) : EZDP\_BITWISE\_CTR\_CFG\_RESERVED0\_18\_SIZE
- Reserved bits 0 to 18.   unsigned [pad3](#) : EZDP\_BITWISE\_CTR\_CFG\_RESERVED32\_63\_SIZE
- Reserved bits 32 to 63.   uint64\_t [data](#)
- Counter data value.   }
- };

### Detailed Description

On-demand bitwise counter configuration definition.

### Field Documentation

uint32\_t [ezdp\\_bitwise\\_ctr\\_cfg::raw\\_data](#)[EZDP\_BITWISE\_CTR\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_bitwise\\_ctr\\_cfg:: pad0](#)  
ECC.

unsigned [ezdp\\_bitwise\\_ctr\\_cfg:: pad1](#)  
Counter sub type (bitwise=9).

unsigned [ezdp\\_bitwise\\_ctr\\_cfg:: pad2](#)  
Reserved bits 0 to 18.

unsigned [ezdp\\_bitwise\\_ctr\\_cfg:: pad3](#)  
Reserved bits 32 to 63.

uint64\_t [ezdp\\_bitwise\\_ctr\\_cfg::data](#)  
Counter data value.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_buffer\_desc Struct Reference

Buffer descriptor (BD) data structure.

### Data Fields

- union {
- [ezdp\\_buffer\\_desc\\_t raw\\_data](#)
- struct {
- unsigned [valid\\_data\\_buf](#): EZDP\_BUFFER\_DESC\_VALID\_DATA\_BUF\_SIZE
- *The data buffer is valid.*     unsigned [pad0](#): EZDP\_BUFFER\_DESC\_RESERVED28\_29\_SIZE
- < Indicate whether index points to buffer in IMEM or EMEM address space.     unsigned [id](#): EZDP\_BUFFER\_DESC\_ID\_SIZE
- *This ID provides an index pointer to a 256B buffer.*     }
- };

---

### Detailed Description

Buffer descriptor (BD) data structure.

---

### Field Documentation

#### [ezdp\\_buffer\\_desc\\_t ezdp\\_buffer\\_desc::raw\\_data](#)

#### unsigned [ezdp\\_buffer\\_desc::valid\\_data\\_buf](#)

The data buffer is valid.

NOTE: FD with non valid data buffers can't be send to tm or interface or discarded.

#### unsigned [ezdp\\_buffer\\_desc::pad0](#)

Indicate whether index points to buffer in IMEM or EMEM address space.

Reserved bits 28 to 29

#### unsigned [ezdp\\_buffer\\_desc::id](#)

This ID provides an index pointer to a 256B buffer.

The buffer is located either in IMEM or in EMEM address space and it is interpreted according to mem\_type.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_buffer\_info Struct Reference

Buffer descriptor info.

### Data Fields

- union {
- uint8\_t [free\\_bytes](#)
- };

---

### Detailed Description

Buffer descriptor info.

---

### Field Documentation

uint8\_t [ezdp\\_buffer\\_info::free\\_bytes](#)

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_congestion\_status Struct Reference

System priority drop congestion status.

### Data Fields

- union {
- [ezdp\\_congestion\\_status\\_t raw\\_data](#)
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_CONGESTION\_STATUS\_RESERVED14\_15\_SIZE
- *Reserved bits 14-15.* unsigned [\\_\\_pad1](#) : EZDP\_CONGESTION\_STATUS\_RESERVED11\_SIZE
- *< Indicate the port RX congestion level received from RxIF.* unsigned [job\\_guarantee](#) : EZDP\_CONGESTION\_STATUS\_JOB\_GUARANTEE\_SIZE
- *Indicate that job are still guarantee level.* unsigned [\\_\\_pad2](#) : EZDP\_CONGESTION\_STATUS\_RESERVED7\_SIZE
- *< Indicate the job budget congestion level received from FCU.* unsigned [emem\\_buf\\_guarantee](#) : EZDP\_CONGESTION\_STATUS\_EMEM\_BUF\_GUARANTEE\_SIZE
- *Indicate that EMEM buffers are still guarantee level.* unsigned [\\_\\_pad3](#) : EZDP\_CONGESTION\_STATUS\_RESERVED3\_SIZE
- *< Indicate the EMEM buffer budget congestion level received from FCU.* unsigned [imem\\_buf\\_guarantee](#) : EZDP\_CONGESTION\_STATUS\_IMEM\_BUF\_GUARANTEE\_SIZE
- *Indicate that IMEM buffers are still guarantee level.* }
- };

### Detailed Description

System priority drop congestion status.

### Field Documentation

[ezdp\\_congestion\\_status\\_t ezdp\\_congestion\\_status::raw\\_data](#)

unsigned [ezdp\\_congestion\\_status::\\_\\_pad0](#)

Reserved bits 14-15.

unsigned [ezdp\\_congestion\\_status::\\_\\_pad1](#)

< Indicate the port RX congestion level received from RxIF.

Not applicable to channel or group. Applicable only for physical ports. Reserved bit 11.

unsigned [ezdp\\_congestion\\_status::job\\_guarantee](#)

Indicate that job are still guarantee level.

**unsigned [ezdp\\_congestion\\_status::pad2](#)**

< Indicate the job budget congestion level received from FCU.  
Reserved bit 7.

**unsigned [ezdp\\_congestion\\_status::emem\\_buf\\_guarantee](#)**

Indicate that EMEM buffers are still guarantee level.

**unsigned [ezdp\\_congestion\\_status::pad3](#)**

< Indicate the EMEM buffer budget congestion level received from FCU.  
Reserved bit 3.

**unsigned [ezdp\\_congestion\\_status::imem\\_buf\\_guarantee](#)**

Indicate that IMEM buffers are still guarantee level.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)



## ezdp\_ctr\_msg Struct Reference

Counter message queue definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_CTR\_MSG\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0\\_\\_](#): EZDP\_CTR\_MSG\_ECC\_SIZE
- ECC.    unsigned [\\_\\_pad1\\_\\_](#): EZDP\_CTR\_MSG\_RESERVED8\_23\_SIZE
- *reserved bits 8-23*    unsigned [overflow\\_error\\_condition](#):  
EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_SIZE
- *Queue was overrun and old messages are lost.*    unsigned [\\_\\_pad2\\_\\_](#):  
EZDP\_CTR\_MSG\_RESERVED6\_SIZE
- *reserved bit 6*    unsigned [overflow](#): EZDP\_CTR\_MSG\_OVERFLOW\_SIZE
- Counter overflow.    struct [ezdp\\_sum\\_addr sum\\_addr](#)
- < Counter message type    union {
- uint64\_t [single\\_ctr\\_value](#)
- On-demand counter value.    struct [ezdp\\_dual\\_ctr\\_dual\\_ctr\\_value](#)
- On-demand dual counter value.    }
- }
- };

### Detailed Description

Counter message queue definition.

### Field Documentation

uint32\_t [ezdp\\_ctr\\_msg::raw\\_data](#)[EZDP\_CTR\_MSG\_WORD\_COUNT]

unsigned [ezdp\\_ctr\\_msg::\\_\\_pad0\\_\\_](#)

ECC.

unsigned [ezdp\\_ctr\\_msg::\\_\\_pad1\\_\\_](#)

reserved bits 8-23

unsigned [ezdp\\_ctr\\_msg::overflow\\_error\\_condition](#)

Queue was overrun and old messages are lost.

unsigned [ezdp\\_ctr\\_msg::\\_\\_pad2\\_\\_](#)

reserved bit 6

unsigned [ezdp\\_ctr\\_msg::overflow](#)

Counter overflow.

The counter has lost coherency

**struct** [ezdp\\_sum\\_addr ezdp\\_ctr\\_msg::sum\\_addr](#) [read]

< Counter message type

< Counter message type counter summarize address

**uint64\_t** [ezdp\\_ctr\\_msg::single\\_ctr\\_value](#)

On-demand counter value.

**struct** [ezdp\\_dual\\_ctr ezdp\\_ctr\\_msg::dual\\_ctr\\_value](#) [read]

On-demand dual counter value.

**union** { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_decode\_eth\_type\_retval Struct Reference

Decode ip protocol return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_eth\\_type\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_RESERVED13\_31\_SIZE
- *Reserved bits 13 to 31.* unsigned [other](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_SIZE
- *Ethernet type is not one of the decoded types.* unsigned [pppoe\\_discovery](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DISCOVERY\_SIZE
- *Ethernet type is PPPoE Discovery Stage, value 0x8863.* unsigned [pppoe\\_session](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_SIZE
- *Ethernet type is PPPoE Session Stage, value 0x8864.* unsigned [user\\_def1](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_SIZE
- *Ethernet type is equal to user defined value 1.* unsigned [user\\_def0](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_SIZE
- *Ethernet type is equal to user defined value 0.* unsigned [length](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_SIZE
- *Ethernet type is less than or equal to 0x0600 and indicates that this field is length.* unsigned [ipv6](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_SIZE
- *Ethernet type is Internet Protocol, Version 6 (IPv6), value 0x86dd.* unsigned [mpls\\_multicast](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MULTICAST\_SIZE
- *Ethernet type is MPLS multicast, value 0x8848.* unsigned [mpls\\_unicast](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNICAST\_SIZE
- *Ethernet type is MPLS unicast, value 0x8847.* unsigned [arp](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_SIZE
- *Ethernet type is Address Resolution Protocol (ARP), value 0x0806.* unsigned [eth\\_88a8](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8\_SIZE
- *Ethernet type is Provider Bridging (IEEE 802.1ad) and Shortest Path Bridging IEEE 802.1aq, value 0x88a8.* unsigned [eth\\_8100](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_SIZE
- *Ethernet type is VLAN-tagged frame (IEEE 802.1Q) and Shortest Path Bridging IEEE 802.1aq, value 0x8100.* unsigned [ipv4](#): EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_SIZE
- *Ethernet type is Internet Protocol Version 4 (IPv4), value 0x0800.* }
- };

### Detailed Description

Decode ip protocol return value struct definition.

### Field Documentation

[ezdp\\_decode\\_eth\\_type\\_retval\\_t ezdp\\_decode\\_eth\\_type\\_retval::raw\\_data](#)

unsigned [ezdp\\_decode\\_eth\\_type\\_retval::pad0](#)

Reserved bits 13 to 31.

unsigned [ezdp\\_decode\\_eth\\_type\\_retval::other](#)

Ethernet type is not one of the decoded types.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::pppoe\\_discovery](#)**

Ethernet type is PPPoE Discovery Stage, value 0x8863.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::pppoe\\_session](#)**

Ethernet type is PPPoE Session Stage, value 0x8864.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::user\\_def1](#)**

Ethernet type is equal to user defined value 1.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::user\\_def0](#)**

Ethernet type is equal to user defined value 0.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::length](#)**

Ethernet type is less than or equal to 0x0600 and indicates that this field is length.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::ipv6](#)**

Ethernet type is Internet Protocol, Version 6 (IPv6), value 0x86dd.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::mpls\\_multicast](#)**

Ethernet type is MPLS multicast, value 0x8848.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::mpls\\_unicast](#)**

Ethernet type is MPLS unicast, value 0x8847.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::arp](#)**

Ethernet type is Address Resolution Protocol (ARP), value 0x0806.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::eth\\_88a8](#)**

Ethernet type is Provider Bridging (IEEE 802.1ad) and Shortest Path Bridging IEEE 802.1aq, value 0x88a8.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::eth\\_8100](#)**

Ethernet type is VLAN-tagged frame (IEEE 802.1Q) and Shortest Path Bridging IEEE 802.1aq, value 0x8100.

**unsigned [ezdp\\_decode\\_eth\\_type\\_retval::ipv4](#)**

Ethernet type is Internet Protocol Version 4 (IPv4), value 0x0800.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ip\_next\_protocol Struct Reference

IP protocol type flags.

### Data Fields

- union {
- [ezdp\\_decode\\_ip\\_next\\_protocol\\_t raw\\_data](#)
- struct {
- unsigned [other](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_SIZE
- *Protocol is not one of the decode types.* unsigned [icmp\\_igmp](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_IGMP\_SIZE
- *Control frame - ICMP/IGMP.* unsigned [ipv6](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_SIZE
- *IPv6 protocol.* unsigned [ipv4](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_SIZE
- *IPv4 protocol.* unsigned [gre](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_SIZE
- *GRE protocol.* unsigned [mpls](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_SIZE
- *MPLS protocol.* unsigned [udp](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_SIZE
- *UDP protocol.* unsigned [tcp](#): EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_SIZE
- *TCP protocol.* }
- };

### Detailed Description

IP protocol type flags.

### Field Documentation

[ezdp\\_decode\\_ip\\_next\\_protocol\\_t ezdp\\_decode\\_ip\\_next\\_protocol::raw\\_data](#)

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::other](#)

Protocol is not one of the decode types.

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::icmp\\_igmp](#)

Control frame - ICMP/IGMP.

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::ipv6](#)

IPv6 protocol.

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::ipv4](#)

IPv4 protocol.

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::gre](#)

GRE protocol.

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::mpls](#)

MPLS protocol.

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::udp](#)

UDP protocol.

unsigned [ezdp\\_decode\\_ip\\_next\\_protocol::tcp](#)

TCP protocol.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ip\_protocol\_retval Struct Reference

Decode ip protocol return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_ip\\_protocol\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_RESERVED14\_31\_SIZE
- *Reserved bits 14 to 31.* unsigned [other](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_OTHER\_SIZE
- *Protocol is other.* unsigned [ah\\_prot](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_AH\_PROT\_SIZE
- *AH (Authentication Header) protocol.* unsigned [esp\\_prot](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_ESP\_PROT\_SIZE
- *ESP (Encapsulating Security Payload) protocol.* unsigned [def\\_ip\\_prot\\_3](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_3\_SIZE
- *User defined protocol 3.* unsigned [def\\_ip\\_prot\\_2](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_2\_SIZE
- *User defined protocol 2.* unsigned [def\\_ip\\_prot\\_1](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_1\_SIZE
- *User defined protocol 1.* unsigned [def\\_ip\\_prot\\_0](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_0\_SIZE
- *User defined protocol 0.* unsigned [icmp\\_igmp](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_ICMP\_IGMP\_SIZE
- *Control frame - ICMP/IGMP.* unsigned [ipv6](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV6\_SIZE
- *Protocol is IPv6.* unsigned [ipv4](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV4\_SIZE
- *Protocol is IPv4.* unsigned [gre](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_GRE\_SIZE
- *Protocol is GRE.* unsigned [mpls](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_MPLS\_SIZE
- *Protocol is MPLS.* unsigned [udp](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_UDP\_SIZE
- *Protocol is UDP.* unsigned [tcp](#): EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_TCP\_SIZE
- *Protocol is TCP.* }
- };

---

### Detailed Description

Decode ip protocol return value struct definition.

---

### Field Documentation

[ezdp\\_decode\\_ip\\_protocol\\_retval\\_t ezdp\\_decode\\_ip\\_protocol\\_retval::raw\\_data](#)

unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::pad0](#)

Reserved bits 14 to 31.

unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::other](#)

Protocol is other.

unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::ah\\_prot](#)



AH (Authentication Header) protocol.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::esp\\_prot](#)**

ESP (Encapsulating Security Payload) protocol.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::def\\_ip\\_prot\\_3](#)**

User defined protocol 3.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::def\\_ip\\_prot\\_2](#)**

User defined protocol 2.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::def\\_ip\\_prot\\_1](#)**

User defined protocol 1.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::def\\_ip\\_prot\\_0](#)**

User defined protocol 0.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::icmp\\_igmp](#)**

Control frame - ICMP/IGMP.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::ipv6](#)**

Protocol is IPv6.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::ipv4](#)**

Protocol is IPv4.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::gre](#)**

Protocol is GRE.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::mpls](#)**

Protocol is MPLS.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::udp](#)**

Protocol is UDP.

**unsigned [ezdp\\_decode\\_ip\\_protocol\\_retval::tcp](#)**

Protocol is TCP.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ipv4\_control Struct Reference

IPv4 addresses decoding result.

### Data Fields

- union {
- [ezdp\\_decode\\_ipv4\\_control\\_t raw\\_data](#)
- struct {
- unsigned [user\\_config2](#): EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_SIZE
- *Destination IP is equal to user configured DIP 2 masked with user configured DIP mask 2.* unsigned [user\\_config1](#): EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_SIZE
- *Destination IP is equal to user configured DIP 1 masked with user configured DIP mask 1.* unsigned [user\\_config0](#): EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG0\_SIZE
- *Destination IP is equal to user configured DIP 0 masked with user configured DIP mask 0.* unsigned [igmp](#): EZDP\_DECODE\_IPV4\_CONTROL\_IGMP\_SIZE
- *Destination IP is IGMP.* unsigned [icmp](#): EZDP\_DECODE\_IPV4\_CONTROL\_ICMP\_SIZE
- *Destination IP is ICMP.* unsigned [pad0](#): EZDP\_DECODE\_IPV4\_CONTROL\_RESERVED\_2\_SIZE
- *Reserved bit 2.* unsigned [internetwork\\_multicast\\_range](#): EZDP\_DECODE\_IPV4\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_SIZE
- *Destination IP is 0xE0-00-01-xx.* unsigned [link\\_local\\_multicast\\_range](#): EZDP\_DECODE\_IPV4\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_SIZE
- *Destination IP is 0xE0-00-00-xx.* }
- };

---

### Detailed Description

IPv4 addresses decoding result.

---

### Field Documentation

[ezdp\\_decode\\_ipv4\\_control\\_t ezdp\\_decode\\_ipv4\\_control::raw\\_data](#)

unsigned [ezdp\\_decode\\_ipv4\\_control::user\\_config2](#)

Destination IP is equal to user configured DIP 2 masked with user configured DIP mask 2.

unsigned [ezdp\\_decode\\_ipv4\\_control::user\\_config1](#)

Destination IP is equal to user configured DIP 1 masked with user configured DIP mask 1.

unsigned [ezdp\\_decode\\_ipv4\\_control::user\\_config0](#)

Destination IP is equal to user configured DIP 0 masked with user configured DIP mask 0.

unsigned [ezdp\\_decode\\_ipv4\\_control::igmp](#)

Destination IP is IGMP.

unsigned [ezdp\\_decode\\_ipv4\\_control::icmp](#)

Destination IP is ICMP.

unsigned [ezdp\\_decode\\_ipv4\\_control::pad0](#)

Reserved bit 2.

unsigned [ezdp\\_decode\\_ipv4\\_control::internetwork\\_multicast\\_range](#)

Destination IP is 0xE0-00-01-xx.

unsigned [ezdp\\_decode\\_ipv4\\_control::link\\_local\\_multicast\\_range](#)

Destination IP is 0xE0-00-00-xx.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ipv4\_errors Struct Reference

IPv4 header decode error flags.

### Data Fields

- union {
- [ezdp\\_decode\\_ipv4\\_errors\\_t raw\\_data](#)
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_DECODE\_IPV4\_ERRORS\_RESERVED9\_15\_SIZE
- Reserved bits 9 to 15.   unsigned [decode\\_error](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_SIZE
- Aggregated decode error flag.   unsigned [sip\\_equal\\_dip](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_SIZE
- Destination IP equals source IP.   unsigned [checksum\\_error](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_SIZE
- Header checksum error.   unsigned [not\\_ipv4\\_version](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_SIZE
- Incorrect version.   unsigned [header\\_length\\_gt\\_frame\\_length](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_SIZE
- IPv4 header length field value is greater than frame length.   unsigned [total\\_length\\_gt\\_frame\\_length](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_SIZE
- IPv4 header total length field value is greater than frame length.   unsigned [header\\_length\\_lt\\_5](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_SIZE
- IPv4 header length is less than 5.   unsigned [sip\\_is\\_zero](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_SIZE
- Source IP is zero.   unsigned [sip\\_is\\_multicast](#):  
EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_SIZE
- Source IP is multicast (SIP is greater than 0xE0-00-00-00).   }
- };

### Detailed Description

IPv4 header decode error flags.

### Field Documentation

[ezdp\\_decode\\_ipv4\\_errors\\_t ezdp\\_decode\\_ipv4\\_errors::raw\\_data](#)

unsigned [ezdp\\_decode\\_ipv4\\_errors::\\_\\_pad0](#)

Reserved bits 9 to 15.

unsigned [ezdp\\_decode\\_ipv4\\_errors::decode\\_error](#)

Aggregated decode error flag.

Indicate at least one decode error

unsigned [ezdp\\_decode\\_ipv4\\_errors::sip\\_equal\\_dip](#)

Destination IP equals source IP.

**unsigned [ezdp\\_decode\\_ipv4\\_errors::checksum\\_error](#)**

Header checksum error.

**unsigned [ezdp\\_decode\\_ipv4\\_errors::not\\_ipv4\\_version](#)**

Incorrect version.

**unsigned [ezdp\\_decode\\_ipv4\\_errors::header\\_length\\_gt\\_frame\\_length](#)**

IPv4 header length field value is greater than frame length.

**unsigned [ezdp\\_decode\\_ipv4\\_errors::total\\_length\\_gt\\_frame\\_length](#)**

IPv4 header total length field value is greater than frame length.

**unsigned [ezdp\\_decode\\_ipv4\\_errors::header\\_length\\_lt\\_5](#)**

IPv4 header length is less than 5.

**unsigned [ezdp\\_decode\\_ipv4\\_errors::sip\\_is\\_zero](#)**

Source IP is zero.

**unsigned [ezdp\\_decode\\_ipv4\\_errors::sip\\_is\\_multicast](#)**

Source IP is multicast (SIP is greater than 0xE0-00-00-00).

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ipv4\_result Struct Reference

Decode IPv4 result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DECODE\_IPV4\_RESULT\_WORD\_COUNT]
- struct {
- struct [ezdp\\_decode\\_ipv4\\_control control](#)
- Decode control.   struct [ezdp\\_decode\\_ipv4\\_errors error\\_codes](#)
- Decode error codes.   unsigned [first\\_fragment](#):  
EZDP\_DECODE\_IPV4\_RESULT\_FIRST\_FRAGMENT\_SIZE
- First fragment flag.   unsigned [\\_pad0](#) : EZDP\_DECODE\_IPV4\_RESULT\_RESERVED\_2\_6\_SIZE
- Reserved bits 2 to 6.   unsigned [user\\_config\\_sip](#):  
EZDP\_DECODE\_IPV4\_RESULT\_USER\_CONFIG\_SIP\_SIZE
- Source IP (SIP) is equal to user configured SIP masked with user configured SIP mask.   unsigned  
[option\\_exist](#): EZDP\_DECODE\_IPV4\_RESULT\_OPTION\_EXIST\_SIZE
- IPv4 header length is greater than 5 - options exist.   unsigned [\\_pad1](#) :  
EZDP\_DECODE\_IPV4\_RESULT\_RESERVED\_56\_63\_SIZE
- Reserved bits 56 to 63.   struct [ezdp\\_decode\\_ip\\_next\\_protocol next\\_protocol](#)
- Next protocol struct.   uint16\_t [sip\\_dip\\_hash](#)
- sip+dip hash   }
- };

---

### Detailed Description

Decode IPv4 result.

---

### Field Documentation

uint32\_t [ezdp\\_decode\\_ipv4\\_result::raw\\_data](#) [EZDP\_DECODE\_IPV4\_RESULT\_WORD\_COUNT]

struct [ezdp\\_decode\\_ipv4\\_control](#) [ezdp\\_decode\\_ipv4\\_result::control](#) [read]

Decode control.

struct [ezdp\\_decode\\_ipv4\\_errors](#) [ezdp\\_decode\\_ipv4\\_result::error\\_codes](#) [read]

Decode error codes.

unsigned [ezdp\\_decode\\_ipv4\\_result::first\\_fragment](#)

First fragment flag.

unsigned [ezdp\\_decode\\_ipv4\\_result::\\_pad0](#)

Reserved bits 2 to 6.

**unsigned [ezdp\\_decode\\_ipv4\\_result::user\\_config\\_sip](#)**

Source IP (SIP) is equal to user configured SIP masked with user configured SIP mask.

**unsigned [ezdp\\_decode\\_ipv4\\_result::option\\_exist](#)**

IPv4 header length is greater than 5 - options exist.

**unsigned [ezdp\\_decode\\_ipv4\\_result::pad1](#)**

Reserved bits 56 to 63.

**struct [ezdp\\_decode\\_ip\\_next\\_protocol](#) [ezdp\\_decode\\_ipv4\\_result::next\\_protocol](#) [read]**

Next protocol struct.

**uint16\_t [ezdp\\_decode\\_ipv4\\_result::sip\\_dip\\_hash](#)**

sip+dip hash

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)



## ezdp\_decode\_ipv4\_retval Struct Reference

Decode IPv4 return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_ipv4\\_retval\\_t raw\\_data](#)
- struct {
- struct [ezdp\\_decode\\_ipv4\\_control control](#)
- Decode control. struct [ezdp\\_decode\\_ipv4\\_errors\\_error\\_codes](#)
- Decode error codes. unsigned [first\\_fragment](#):  
EZDP\_DECODE\_IPV4\_RETVAL\_FIRST\_FRAGMENT\_SIZE
- First fragment flag. unsigned [pad0](#) : EZDP\_DECODE\_IPV4\_RETVAL\_RESERVED\_2\_6\_SIZE
- Reserved bits 2 to 6. unsigned [user\\_config\\_sip](#):  
EZDP\_DECODE\_IPV4\_RETVAL\_USER\_CONFIG\_SIP\_SIZE
- Source IP (SIP) is equal to user configured SIP masked with user configured SIP mask. unsigned  
[option\\_exist](#): EZDP\_DECODE\_IPV4\_RETVAL\_OPTION\_EXIST\_SIZE
- IPv4 header length is greater than 5 - options exist. }
- };

### Detailed Description

Decode IPv4 return value struct definition.

### Field Documentation

[ezdp\\_decode\\_ipv4\\_retval\\_t ezdp\\_decode\\_ipv4\\_retval::raw\\_data](#)

struct [ezdp\\_decode\\_ipv4\\_control ezdp\\_decode\\_ipv4\\_retval::control](#) [read]

Decode control.

struct [ezdp\\_decode\\_ipv4\\_errors ezdp\\_decode\\_ipv4\\_retval::error\\_codes](#) [read]

Decode error codes.

unsigned [ezdp\\_decode\\_ipv4\\_retval::first\\_fragment](#)

First fragment flag.

unsigned [ezdp\\_decode\\_ipv4\\_retval:: pad0](#)

Reserved bits 2 to 6.

unsigned [ezdp\\_decode\\_ipv4\\_retval::user\\_config\\_sip](#)

Source IP (SIP) is equal to user configured SIP masked with user configured SIP mask.

unsigned [ezdp\\_decode\\_ipv4\\_retval::option\\_exist](#)

IPv4 header length is greater than 5 - options exist.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ipv6\_control Struct Reference

IPv6 addresses decoding result.

### Data Fields

- union {
- [ezdp\\_decode\\_ipv6\\_control\\_t raw\\_data](#)
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_DECODE\_IPV6\_CONTROL\_RESERVED7\_8\_SIZE
- *Reserved bits 7 to 8.* unsigned [dip\\_is\\_wellknown\\_multicast](#):  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_WELLKNOWN\_MULTICAST\_SIZE
- *Destination IP is FF\*...dip.byte[1].bit[4]=0.* unsigned [dip\\_is\\_multicast](#):  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_MULTICAST\_SIZE
- *Destination IP is FF\*...dip.byte[1].bit[4]=1.* unsigned [\\_\\_pad1](#) :  
EZDP\_DECODE\_IPV6\_CONTROL\_RESERVED\_3\_SIZE
- *Reserved bit 3.* unsigned [solicited\\_node\\_multicast\\_range](#):  
EZDP\_DECODE\_IPV6\_CONTROL\_SOLICITED\_NODE\_MULTICAST\_RANGE\_SIZE
- *Destination IP is ff02::01:ffxx:xxxx.* unsigned [internetwork\\_multicast\\_range](#):  
EZDP\_DECODE\_IPV6\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_SIZE
- *Destination IP is ff02::01xx.* unsigned [link\\_local\\_multicast\\_range](#):  
EZDP\_DECODE\_IPV6\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_SIZE
- *Destination IP is ff02::xx.* }
- };

### Detailed Description

IPv6 addresses decoding result.

### Field Documentation

[ezdp\\_decode\\_ipv6\\_control\\_t ezdp\\_decode\\_ipv6\\_control::raw\\_data](#)

unsigned [ezdp\\_decode\\_ipv6\\_control::pad0](#)

Reserved bits 7 to 8.

unsigned [ezdp\\_decode\\_ipv6\\_control::dip\\_is\\_wellknown\\_multicast](#)

Destination IP is FF\*...dip.byte[1].bit[4]=0.

unsigned [ezdp\\_decode\\_ipv6\\_control::dip\\_is\\_multicast](#)

Destination IP is FF\*...dip.byte[1].bit[4]=1.

unsigned [ezdp\\_decode\\_ipv6\\_control::pad1](#)

Reserved bit 3.

**unsigned** [ezdp\\_decode\\_ipv6\\_control::solicited\\_node\\_multicast\\_range](#)

Destination IP is ff02::01:ffxx:xxxx.

**unsigned** [ezdp\\_decode\\_ipv6\\_control::internetwork\\_multicast\\_range](#)

Destination IP is ff02::01xx.

**unsigned** [ezdp\\_decode\\_ipv6\\_control::link\\_local\\_multicast\\_range](#)

Destination IP is ff02::xx.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ipv6\_errors Struct Reference

IPv6 header decode error flags.

### Data Fields

- union {
- [ezdp\\_decode\\_ipv6\\_errors\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#): EZDP\_DECODE\_IPV6\_ERRORS\_RESERVED10\_15\_SIZE
- *Reserved bits 10 to 15.* unsigned [sip\\_is\\_multicast](#): EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_MULTICAST\_SIZE
- *SIP is multicast (first octet is 0xFF).* unsigned [payload\\_missing](#): EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_MISSING\_SIZE
- *Payload Length is zero when next header field is not hop-by-hop (0x00).* unsigned [decode\\_error](#): EZDP\_DECODE\_IPV6\_ERRORS\_DECODE\_ERROR\_SIZE
- *Aggregated decode error flag.* unsigned [sip\\_equal\\_dip](#): EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_EQUAL\_DIP\_SIZE
- *Destination IP equals source IP.* unsigned [dip\\_is\\_one](#): EZDP\_DECODE\_IPV6\_ERRORS\_DIP\_IS\_ONE\_SIZE
- *Destination IP is one.* unsigned [dip\\_is\\_zero](#): EZDP\_DECODE\_IPV6\_ERRORS\_DIP\_IS\_ZERO\_SIZE
- *Destination IP is zero.* unsigned [sip\\_is\\_one](#): EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ONE\_SIZE
- *Source IP is one.* unsigned [sip\\_is\\_zero](#): EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ZERO\_SIZE
- *Source IP is zero.* unsigned [not\\_ipv6\\_version](#): EZDP\_DECODE\_IPV6\_ERRORS\_NOT\_IPV6\_VERSION\_SIZE
- *Incorrect version.* unsigned [payload\\_gt\\_frame\\_length](#): EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_GT\_FRAME\_LENGTH\_SIZE
- *Payload length greater than frame length.* }
- };

### Detailed Description

IPv6 header decode error flags.

### Field Documentation

[ezdp\\_decode\\_ipv6\\_errors\\_t ezdp\\_decode\\_ipv6\\_errors::raw\\_data](#)

unsigned [ezdp\\_decode\\_ipv6\\_errors::pad0](#)

Reserved bits 10 to 15.

unsigned [ezdp\\_decode\\_ipv6\\_errors::sip\\_is\\_multicast](#)

SIP is multicast (first octet is 0xFF).

unsigned [ezdp\\_decode\\_ipv6\\_errors::payload\\_missing](#)

Payload Length is zero when next header field is not hop-by-hop (0x00).

**unsigned [ezdp\\_decode\\_ipv6\\_errors::decode\\_error](#)**

Aggregated decode error flag.

Indicate at least one decode error

**unsigned [ezdp\\_decode\\_ipv6\\_errors::sip\\_equal\\_dip](#)**

Destination IP equals source IP.

**unsigned [ezdp\\_decode\\_ipv6\\_errors::dip\\_is\\_one](#)**

Destination IP is one.

**unsigned [ezdp\\_decode\\_ipv6\\_errors::dip\\_is\\_zero](#)**

Destination IP is zero.

**unsigned [ezdp\\_decode\\_ipv6\\_errors::sip\\_is\\_one](#)**

Source IP is one.

**unsigned [ezdp\\_decode\\_ipv6\\_errors::sip\\_is\\_zero](#)**

Source IP is zero.

**unsigned [ezdp\\_decode\\_ipv6\\_errors::not\\_ipv6\\_version](#)**

Incorrect version.

**unsigned [ezdp\\_decode\\_ipv6\\_errors::payload\\_gt\\_frame\\_length](#)**

Payload length greater than frame length.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_ipv6\_result Struct Reference

Decode IPv6 result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DECODE\_IPV6\_RESULT\_WORD\_COUNT]
- struct {
- struct [ezdp\\_decode\\_ipv6\\_errors error\\_codes](#)
- Error codes.   unsigned [\\_\\_pad0\\_\\_](#): EZDP\_DECODE\_IPV6\_RESULT\_RESERVED\_15\_SIZE
- Reserved bit 15.   unsigned [global\\_addresses](#):  
EZDP\_DECODE\_IPV6\_RESULT\_GLOBAL\_ADDRESSES\_SIZE
- Both addresses scope are global.   unsigned [site\\_local\\_address](#):  
EZDP\_DECODE\_IPV6\_RESULT\_SITE\_LOCAL\_ADDRESS\_SIZE
- One of addresses scope is site local.   unsigned [link\\_local\\_address](#):  
EZDP\_DECODE\_IPV6\_RESULT\_LINK\_LOCAL\_ADDRESS\_SIZE
- One of addresses scope is link local.   unsigned [\\_\\_pad1\\_\\_](#):  
EZDP\_DECODE\_IPV6\_RESULT\_RESERVED9\_11\_SIZE
- Reserved bits 9 to 11.   unsigned [options\\_exist](#): EZDP\_DECODE\_IPV6\_RESULT\_OPTIONS\_EXIST\_SIZE
- Options exist.   struct [ezdp\\_decode\\_ipv6\\_control control](#)
- Control struct.   unsigned [\\_\\_pad2\\_\\_](#): EZDP\_DECODE\_IPV6\_RESULT\_RESERVED\_56\_63\_SIZE
- Reserved bits 56 to 63.   struct [ezdp\\_decode\\_ip\\_next\\_protocol next\\_protocol](#)
- Next protocol struct.   uint16\_t [sip\\_dip\\_hash](#)
- sip+dip hash   }
- };

### Detailed Description

Decode IPv6 result.

### Field Documentation

uint32\_t [ezdp\\_decode\\_ipv6\\_result::raw\\_data](#) [EZDP\_DECODE\_IPV6\_RESULT\_WORD\_COUNT]

struct [ezdp\\_decode\\_ipv6\\_errors ezdp\\_decode\\_ipv6\\_result::error\\_codes](#) [read]

Error codes.

unsigned [ezdp\\_decode\\_ipv6\\_result::\\_\\_pad0\\_\\_](#)

Reserved bit 15.

unsigned [ezdp\\_decode\\_ipv6\\_result::global\\_addresses](#)

Both addresses scope are global.

unsigned [ezdp\\_decode\\_ipv6\\_result::site\\_local\\_address](#)

One of addresses scope is site local.

unsigned [ezdp\\_decode\\_ipv6\\_result::link\\_local\\_address](#)

One of addresses scope is link local.

unsigned [ezdp\\_decode\\_ipv6\\_result::pad1](#)

Reserved bits 9 to 11.

unsigned [ezdp\\_decode\\_ipv6\\_result::options\\_exist](#)

Options exist.

struct [ezdp\\_decode\\_ipv6\\_control](#) [ezdp\\_decode\\_ipv6\\_result::control](#) [read]

Control struct.

unsigned [ezdp\\_decode\\_ipv6\\_result::pad2](#)

Reserved bits 56 to 63.

struct [ezdp\\_decode\\_ip\\_next\\_protocol](#) [ezdp\\_decode\\_ipv6\\_result::next\\_protocol](#) [read]

Next protocol struct.

uint16\_t [ezdp\\_decode\\_ipv6\\_result::sip\\_dip\\_hash](#)

sip+dip hash

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)



## ezdp\_decode\_ipv6\_retval Struct Reference

Decode IPv4 return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_ipv6\\_retval\\_t raw\\_data](#)
- struct {
- struct [ezdp\\_decode\\_ipv6\\_errors error\\_codes](#)
- *Error codes.* unsigned [\\_\\_pad0\\_\\_](#): EZDP\_DECODE\_IPV6\_RETVAL\_RESERVED\_15\_SIZE
- *Reserved bit 15.* unsigned [global\\_addresses](#): EZDP\_DECODE\_IPV6\_RETVAL\_GLOBAL\_ADDRESSES\_SIZE
- *Both addresses scope are global.* unsigned [site\\_local\\_address](#): EZDP\_DECODE\_IPV6\_RETVAL\_SITE\_LOCAL\_ADDRESS\_SIZE
- *One of addresses scope is site local.* unsigned [link\\_local\\_address](#): EZDP\_DECODE\_IPV6\_RETVAL\_LINK\_LOCAL\_ADDRESS\_SIZE
- *One of addresses scope is link local.* unsigned [\\_\\_pad1\\_\\_](#): EZDP\_DECODE\_IPV6\_RETVAL\_RESERVED9\_11\_SIZE
- *Reserved bits 9 to 11.* unsigned [options\\_exist](#): EZDP\_DECODE\_IPV6\_RETVAL\_OPTIONS\_EXIST\_SIZE
- *Options exist.* struct [ezdp\\_decode\\_ipv6\\_control control](#)
- *Control struct.* }
- };

### Detailed Description

Decode IPv4 return value struct definition.

### Field Documentation

[ezdp\\_decode\\_ipv6\\_retval\\_t ezdp\\_decode\\_ipv6\\_retval::raw\\_data](#)

struct [ezdp\\_decode\\_ipv6\\_errors ezdp\\_decode\\_ipv6\\_retval::error\\_codes](#) [read]

Error codes.

unsigned [ezdp\\_decode\\_ipv6\\_retval::\\_\\_pad0\\_\\_](#)

Reserved bit 15.

unsigned [ezdp\\_decode\\_ipv6\\_retval::global\\_addresses](#)

Both addresses scope are global.

unsigned [ezdp\\_decode\\_ipv6\\_retval::site\\_local\\_address](#)

One of addresses scope is site local.

unsigned [ezdp\\_decode\\_ipv6\\_retval::link\\_local\\_address](#)

One of addresses scope is link local.

unsigned [ezdp\\_decode\\_ipv6\\_retval::pad1](#)

Reserved bits 9 to 11.

unsigned [ezdp\\_decode\\_ipv6\\_retval::options\\_exist](#)

Options exist.

struct [ezdp\\_decode\\_ipv6\\_control](#) [ezdp\\_decode\\_ipv6\\_retval::control](#) [read]

Control struct.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mac\_control Struct Reference

MAC addresses decoding result.

### Data Fields

- union {
- [ezdp\\_decode\\_mac\\_control\\_t raw\\_data](#)
- struct {
- unsigned [\\_pad0](#); EZDP\_DECODE\_MAC\_CONTROL\_RESERVED13\_15\_SIZE
- *Reserved bits 13 to 15.* unsigned [smac\\_equals\\_dmac](#):  
EZDP\_DECODE\_MAC\_CONTROL\_SMAC\_EQUALS\_DMACE\_SIZE
- *Source MAC equals destination MAC.* unsigned [user\\_config3](#):  
EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG3\_SIZE
- *Destination MAC (DMAC) is equal to user configured DMAC 3 masked with user configured DMAC mask 3.* unsigned [user\\_config2](#): EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG2\_SIZE
- *Destination MAC (DMAC) is equal to user configured DMAC 2 masked with user configured DMAC mask 2.* unsigned [user\\_config1](#): EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG1\_SIZE
- *Destination MAC (DMAC) is equal to user configured DMAC 1 masked with user configured DMAC mask 1.* unsigned [user\\_config0](#): EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG0\_SIZE
- *Destination MAC (DMAC) is equal to user configured DMAC 0 masked with user configured DMAC mask 0.* unsigned [ipv6\\_multicast](#): EZDP\_DECODE\_MAC\_CONTROL\_IPV6\_MULTICAST\_SIZE
- *DMAC is 0x33-33-xx-xx-xx-xx.* unsigned [ipv4\\_multicast](#):  
EZDP\_DECODE\_MAC\_CONTROL\_IPV4\_MULTICAST\_SIZE
- *DMAC is 0x01-00-5E-xx-xx-xx.* unsigned [vrrp\\_mac](#):  
EZDP\_DECODE\_MAC\_CONTROL\_VRRP\_MAC\_SIZE
- *DMAC is 0x00-00-5E-00-01-xx.* unsigned [mac\\_control\\_other](#):  
EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_OTHER\_SIZE
- *DMAC is 0x01-80-C2-[01-FF]-[01-FF]-[01-FF].* unsigned [mac\\_control\\_lsb\\_2x](#):  
EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_LSB\_2X\_SIZE
- *DMAC is 0x01-80-C2-00-00-2x.* unsigned [mac\\_control\\_lsb\\_1x](#):  
EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_LSB\_1X\_SIZE
- *DMAC is 0x01-80-C2-00-00-1x.* unsigned [mac\\_control\\_lsb\\_0x](#):  
EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_LSB\_0X\_SIZE
- *DMAC is 0x01-80-C2-00-00-0x.* unsigned [my\\_mac](#):  
EZDP\_DECODE\_MAC\_CONTROL\_MY\_MAC\_SIZE
- *Destination MAC is my MAC ( DMAC is less than or equal to Low MAC and greater than or equal to High MAC, where Low and High MACs are user configured). }*
- };

### Detailed Description

MAC addresses decoding result.

### Field Documentation

[ezdp\\_decode\\_mac\\_control\\_t ezdp\\_decode\\_mac\\_control::raw\\_data](#)

unsigned [ezdp\\_decode\\_mac\\_control::\\_pad0](#)

Reserved bits 13 to 15.

**unsigned [ezdp\\_decode\\_mac\\_control::smac\\_equals\\_dmac](#)**

Source MAC equals destination MAC.

**unsigned [ezdp\\_decode\\_mac\\_control::user\\_config3](#)**

Destination MAC (DMAC) is equal to user configured DMAC 3 masked with user configured DMAC mask 3.

**unsigned [ezdp\\_decode\\_mac\\_control::user\\_config2](#)**

Destination MAC (DMAC) is equal to user configured DMAC 2 masked with user configured DMAC mask 2.

**unsigned [ezdp\\_decode\\_mac\\_control::user\\_config1](#)**

Destination MAC (DMAC) is equal to user configured DMAC 1 masked with user configured DMAC mask 1.

**unsigned [ezdp\\_decode\\_mac\\_control::user\\_config0](#)**

Destination MAC (DMAC) is equal to user configured DMAC 0 masked with user configured DMAC mask 0.

**unsigned [ezdp\\_decode\\_mac\\_control::ipv6\\_multicast](#)**

DMAC is 0x33-33-xx-xx-xx-xx.

**unsigned [ezdp\\_decode\\_mac\\_control::ipv4\\_multicast](#)**

DMAC is 0x01-00-5E-xx-xx-xx.

**unsigned [ezdp\\_decode\\_mac\\_control::vrrp\\_mac](#)**

DMAC is 0x00-00-5E-00-01-xx.

**unsigned [ezdp\\_decode\\_mac\\_control::mac\\_control\\_other](#)**

DMAC is 0x01-80-C2-[01-FF]-[01-FF]-[01-FF].

**unsigned [ezdp\\_decode\\_mac\\_control::mac\\_control\\_lsb\\_2x](#)**

DMAC is 0x01-80-C2-00-00-2x.

**unsigned [ezdp\\_decode\\_mac\\_control::mac\\_control\\_lsb\\_1x](#)**

DMAC is 0x01-80-C2-00-00-1x.

**unsigned [ezdp\\_decode\\_mac\\_control::mac\\_control\\_lsb\\_0x](#)**

DMAC is 0x01-80-C2-00-00-0x.

unsigned [ezdp\\_decode\\_mac\\_control::my\\_mac](#)

Destination MAC is my MAC ( DMAC is less than or equal to Low MAC and greater than or equal to High MAC, where Low and High MACs are user configured).

union { ... }

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mac\_errors Struct Reference

MAC header decode error flags.

### Data Fields

- union {
- [ezdp\\_decode\\_mac\\_errors\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_DECODE\_MAC\_ERRORS\_RESERVED5\_7\_SIZE
- *Reserved bits 5 to 7.* unsigned [decode\\_error](#): EZDP\_DECODE\_MAC\_ERRORS\_DECODE\_ERROR\_SIZE
- *Aggregated decode error flag.* unsigned [ip\\_version\\_mismatch\\_in\\_pppoe](#): EZDP\_DECODE\_MAC\_ERRORS\_IP\_VERSION\_MISMATCH\_IN\_PPPOE\_SIZE
- *IP version mismatch in PPPoE.* unsigned [dmac\\_is\\_zero](#): EZDP\_DECODE\_MAC\_ERRORS\_DMAC\_IS\_ZERO\_SIZE
- *Destination MAC is zero.* unsigned [smac\\_is\\_zero](#): EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_ZERO\_SIZE
- *Source MAC is zero.* unsigned [smac\\_is\\_not\\_unicast](#): EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_NOT\_UNICAST\_SIZE
- *Source MAC is not unicast.* }
- };

### Detailed Description

MAC header decode error flags.

### Field Documentation

[ezdp\\_decode\\_mac\\_errors\\_t ezdp\\_decode\\_mac\\_errors::raw\\_data](#)

unsigned [ezdp\\_decode\\_mac\\_errors::pad0](#)

Reserved bits 5 to 7.

unsigned [ezdp\\_decode\\_mac\\_errors::decode\\_error](#)

Aggregated decode error flag.

Indicate at least one decode error

unsigned [ezdp\\_decode\\_mac\\_errors::ip\\_version\\_mismatch\\_in\\_pppoe](#)

IP version mismatch in PPPoE.

unsigned [ezdp\\_decode\\_mac\\_errors::dmac\\_is\\_zero](#)

Destination MAC is zero.

unsigned [ezdp\\_decode\\_mac\\_errors::smac\\_is\\_zero](#)

Source MAC is zero.

unsigned [ezdp\\_decode\\_mac\\_errors::smac\\_is\\_not\\_unicast](#)

Source MAC is not unicast.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mac\_protocol\_type Struct Reference

Ethernet type definition.

### Data Fields

- union {
- [ezdp\\_decode\\_mac\\_protocol\\_type\\_t raw\\_data](#)
- struct {
- unsigned [\\_pad0](#) : EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_RESERVED\_15\_SIZE
- *Reserved bit 15.* unsigned [user\\_config\\_vlan2](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG\_VLAN2\_SIZE
- *Ethernet type is equal to user configured vlan type 2.* unsigned [pppoe\\_discovery](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE\_DISCOVERY\_SIZE
- *Ethernet type is PPPoE Discovery Stage, value 0x8863.* unsigned [pppoe\\_session](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE\_SESSION\_SIZE
- *Ethernet type is PPPoE Session Stage, value 0x8864.* unsigned [user\\_config3](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG3\_SIZE
- *Ethernet type is equal to user configured type 3.* unsigned [user\\_config2](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG2\_SIZE
- *Ethernet type is equal to user configured type 2.* unsigned [user\\_config1](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG1\_SIZE
- *Ethernet type is equal to user configured type 1.* unsigned [user\\_config0](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG0\_SIZE
- *Ethernet type is equal to user configured type 0.* unsigned [length](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_LENGTH\_SIZE
- *Ethernet type is less or equal 0x0600 and indicate that this field is length.* unsigned [ipv6](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV6\_SIZE
- *Ethernet type is Internet Protocol, Version 6 (IPv6), value 0x86dd.* unsigned [mpls\\_multicast](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_MULTICAST\_SIZE
- *Ethernet type is MPLS multicast, value 0x8848.* unsigned [mpls\\_unicast](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_UNICAST\_SIZE
- *Ethernet type is MPLS unicast, value 0x8847.* unsigned [arp](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_ARP\_SIZE
- *Ethernet type is Address Resolution Protocol (ARP), value 0x0806.* unsigned [user\\_config\\_vlan1](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG\_VLAN1\_SIZE
- *Ethernet type is equal to user configured vlan type 1.* unsigned [user\\_config\\_vlan0](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG\_VLAN0\_SIZE
- *Ethernet type is equal to user configured vlan type 0.* unsigned [ipv4](#):  
EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV4\_SIZE
- *Ethernet type is Internet Protocol Version 4 (IPv4), value 0x0800.* }
- };

### Detailed Description

Ethernet type definition.

### Field Documentation

[ezdp\\_decode\\_mac\\_protocol\\_type\\_t ezdp\\_decode\\_mac\\_protocol\\_type::raw\\_data](#)

unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::\\_pad0](#)



Reserved bit 15.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::user\\_config\\_vlan2](#)**

Ethernet type is equal to user configured vlan type 2.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::pppoe\\_discovery](#)**

Ethernet type is PPPoE Discovery Stage, value 0x8863.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::pppoe\\_session](#)**

Ethernet type is PPPoE Session Stage, value 0x8864.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::user\\_config3](#)**

Ethernet type is equal to user configured type 3.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::user\\_config2](#)**

Ethernet type is equal to user configured type 2.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::user\\_config1](#)**

Ethernet type is equal to user configured type 1.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::user\\_config0](#)**

Ethernet type is equal to user configured type 0.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::length](#)**

Ethernet type is less or equal 0x0600 and indicate that this field is length.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::ipv6](#)**

Ethernet type is Internet Protocol, Version 6 (IPv6), value 0x86dd.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::mpls\\_multicast](#)**

Ethernet type is MPLS multicast, value 0x8848.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::mpls\\_unicast](#)**

Ethernet type is MPLS unicast, value 0x8847.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::arp](#)**

Ethernet type is Address Resolution Protocol (ARP), value 0x0806.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::user\\_config\\_vlan1](#)**

Ethernet type is equal to user configured vlan type 1.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::user\\_config\\_vlan0](#)**

Ethernet type is equal to user configured vlan type 0.

**unsigned [ezdp\\_decode\\_mac\\_protocol\\_type::ipv4](#)**

Ethernet type is Internet Protocol Version 4 (IPv4), value 0x0800.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mac\_result Struct Reference

Decode MAC result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DECODE\_MAC\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_DECODE\_MAC\_RESULT\_RESERVED\_31\_SIZE
- *Reserved bit 31.* unsigned [number\\_of\\_tags](#);  
EZDP\_DECODE\_MAC\_RESULT\_NUMBER\_OF\_TAGS\_SIZE
- *Number of tags.* unsigned [pad1](#) : EZDP\_DECODE\_MAC\_RESULT\_RESERVED26\_27\_SIZE
- *Reserved bits 26 to 27.* unsigned [ipv6\\_in\\_pppoe](#);  
EZDP\_DECODE\_MAC\_RESULT\_IPV6\_IN\_PPPOE\_SIZE
- *IPv6 in PPPoE.* unsigned [ipv4\\_in\\_pppoe](#); EZDP\_DECODE\_MAC\_RESULT\_IPV4\_IN\_PPPOE\_SIZE
- *IPv4 in PPPoE.* struct [ezdp\\_decode\\_mac\\_errors\\_error\\_codes](#)
- *Error codes.* struct [ezdp\\_decode\\_mac\\_control\\_control](#)
- *control struct* struct [ezdp\\_decode\\_mac\\_protocol\\_type\\_tag2\\_protocol\\_type](#)
- *Tag2 protocol type.* struct [ezdp\\_decode\\_mac\\_protocol\\_type\\_tag1\\_protocol\\_type](#)
- *Tag1 protocol type.* struct [ezdp\\_decode\\_mac\\_protocol\\_type\\_last\\_tag\\_protocol\\_type](#)
- *Last tag protocol type.* struct [ezdp\\_decode\\_mac\\_protocol\\_type\\_tag3\\_protocol\\_type](#)
- *Tag3 protocol type.* unsigned [pad2](#) : EZDP\_DECODE\_MAC\_RESULT\_RESERVED120\_127\_SIZE
- *Reserved bits 120 to 127.* uint8\_t [layer2\\_size](#)
- *Layer2 size.* uint16\_t [da\\_sa\\_hash](#)
- *SMAC+DMAC hash.* }
- };

### Detailed Description

Decode MAC result.

### Field Documentation

uint32\_t [ezdp\\_decode\\_mac\\_result::raw\\_data](#)[EZDP\_DECODE\_MAC\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_decode\\_mac\\_result::pad0](#)

Reserved bit 31.

unsigned [ezdp\\_decode\\_mac\\_result::number\\_of\\_tags](#)

Number of tags.

unsigned [ezdp\\_decode\\_mac\\_result::pad1](#)

Reserved bits 26 to 27.

unsigned [ezdp\\_decode\\_mac\\_result::ipv6\\_in\\_pppoe](#)

IPv6 in PPPoE.

unsigned [ezdp\\_decode\\_mac\\_result::ipv4\\_in\\_pppoe](#)

IPv4 in PPPoE.

struct [ezdp\\_decode\\_mac\\_errors](#) [ezdp\\_decode\\_mac\\_result::error\\_codes](#) [read]

Error codes.

struct [ezdp\\_decode\\_mac\\_control](#) [ezdp\\_decode\\_mac\\_result::control](#) [read]

control struct

struct [ezdp\\_decode\\_mac\\_protocol\\_type](#) [ezdp\\_decode\\_mac\\_result::tag2\\_protocol\\_type](#) [read]

Tag2 protocol type.

struct [ezdp\\_decode\\_mac\\_protocol\\_type](#) [ezdp\\_decode\\_mac\\_result::tag1\\_protocol\\_type](#) [read]

Tag1 protocol type.

struct [ezdp\\_decode\\_mac\\_protocol\\_type](#) [ezdp\\_decode\\_mac\\_result::last\\_tag\\_protocol\\_type](#) [read]

Last tag protocol type.

struct [ezdp\\_decode\\_mac\\_protocol\\_type](#) [ezdp\\_decode\\_mac\\_result::tag3\\_protocol\\_type](#) [read]

Tag3 protocol type.

unsigned [ezdp\\_decode\\_mac\\_result::pad2](#)

Reserved bits 120 to 127.

uint8\_t [ezdp\\_decode\\_mac\\_result::layer2\\_size](#)

Layer2 size.

uint16\_t [ezdp\\_decode\\_mac\\_result::da\\_sa\\_hash](#)

SMAC+DMAC hash.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mac\_retval Struct Reference

Decode MAC return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_mac\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_DECODE\_MAC\_RETVAL\_RESERVED\_31\_SIZE
- *Reserved bit 31.* unsigned [number\\_of\\_tags](#);  
EZDP\_DECODE\_MAC\_RETVAL\_NUMBER\_OF\_TAGS\_SIZE
- *Number of tags.* unsigned [pad1](#) : EZDP\_DECODE\_MAC\_RETVAL\_RESERVED26\_27\_SIZE
- *Reserved bits 26 to 27.* unsigned [ipv6\\_in\\_pppoe](#);  
EZDP\_DECODE\_MAC\_RETVAL\_IPV6\_IN\_PPPOE\_SIZE
- *IPv6 in PPPoE.* unsigned [ipv4\\_in\\_pppoe](#); EZDP\_DECODE\_MAC\_RETVAL\_IPV4\_IN\_PPPOE\_SIZE
- *IPv4 in PPPoE.* struct [ezdp\\_decode\\_mac\\_errors\\_error\\_codes](#)
- *Error codes.* struct [ezdp\\_decode\\_mac\\_control\\_control](#)
- *control struct* }
- };

### Detailed Description

Decode MAC return value struct definition.

### Field Documentation

[ezdp\\_decode\\_mac\\_retval\\_t ezdp\\_decode\\_mac\\_retval::raw\\_data](#)

unsigned [ezdp\\_decode\\_mac\\_retval::pad0](#)

Reserved bit 31.

unsigned [ezdp\\_decode\\_mac\\_retval::number\\_of\\_tags](#)

Number of tags.

unsigned [ezdp\\_decode\\_mac\\_retval::pad1](#)

Reserved bits 26 to 27.

unsigned [ezdp\\_decode\\_mac\\_retval::ipv6\\_in\\_pppoe](#)

IPv6 in PPPoE.

unsigned [ezdp\\_decode\\_mac\\_retval::ipv4\\_in\\_pppoe](#)

IPv4 in PPPoE.

struct [ezdp\\_decode\\_mac\\_errors](#) [ezdp\\_decode\\_mac\\_retval::error\\_codes](#) [read]

Error codes.

struct [ezdp\\_decode\\_mac\\_control](#) [ezdp\\_decode\\_mac\\_retval::control](#) [read]

control struct

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mpls\_label\_result Struct Reference

[ezdp\\_decode\\_mpls\\_label\\_result](#) struct for ezdp

### Data Fields

- union {
- [ezdp\\_decode\\_mpls\\_label\\_result t raw\\_data](#)
- struct {
- unsigned [pad0](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_RESERVED10\_31\_SIZE
- *Reserved bits 10 to 31.* unsigned [stop\\_bit](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_STOP\_BIT\_SIZE
- *Stop bit = exception\_bit | (~exception\_bit and ~reserved\_label).* unsigned [exception\\_bit](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_EXCEPTION\_BIT\_SIZE
- *Expection bit = Or\_reduce (((first 8 bits of retval) ^ inv\_from\_host) and mask\_from\_host).* unsigned [user\\_config3](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG3\_SIZE
- *Label is equal to user configured label 3.* unsigned [user\\_config2](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG2\_SIZE
- *Label is equal to user configured label 2.* unsigned [user\\_config1](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG1\_SIZE
- *Label is equal to user configured label 1.* unsigned [user\\_config0](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG0\_SIZE
- *Label is equal to user configured label 0.* unsigned [ttl\\_is\\_one](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_TTL\_IS\_ONE\_SIZE
- *TTL is one.* unsigned [ttl\\_is\\_zero](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_TTL\_IS\_ZERO\_SIZE
- *TTL is zero.* unsigned [reserved\\_label](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_RESERVED\_LABEL\_SIZE
- *Reserved label - label value is less than 16.* unsigned [end\\_of\\_stack](#): EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_END\_OF\_STACK\_SIZE
- *Last label on MPLS stack.* }
- };

### Detailed Description

[ezdp\\_decode\\_mpls\\_label\\_result](#) struct for ezdp

### Field Documentation

[ezdp\\_decode\\_mpls\\_label\\_result t ezdp\\_decode\\_mpls\\_label\\_result::raw\\_data](#)

unsigned [ezdp\\_decode\\_mpls\\_label\\_result::pad0](#)

Reserved bits 10 to 31.

unsigned [ezdp\\_decode\\_mpls\\_label\\_result::stop\\_bit](#)

Stop bit = exception\_bit | (~exception\_bit and ~reserved\_label).

unsigned [ezdp\\_decode\\_mpls\\_label\\_result::exception\\_bit](#)

Expection bit = Or\_reduce (((first 8 bits of retval) ^ inv\_from\_host) and mask\_from\_host).

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::user\\_config3](#)**

Label is equal to user configured label 3.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::user\\_config2](#)**

Label is equal to user configured label 2.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::user\\_config1](#)**

Label is equal to user configured label 1.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::user\\_config0](#)**

Label is equal to user configured label 0.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::ttl\\_is\\_one](#)**

TTL is one.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::ttl\\_is\\_zero](#)**

TTL is zero.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::reserved\\_label](#)**

Reserved label - label value is less than 16.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_result::end\\_of\\_stack](#)**

Last label on MPLS stack.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)



## ezdp\_decode\_mpls\_label\_retval Struct Reference

Decode MPLS label return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_mpls\\_label\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED10\_31\_SIZE
- *Reserved bits 10 to 31.* unsigned [stop\\_bit](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_STOP\_BIT\_SIZE
- *Stop bit = exception\_bit | (~exception\_bit and ~reserved\_label).* unsigned [exception\\_bit](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_EXCEPTION\_BIT\_SIZE
- *Expection bit = or\_reduce (((first 8 bits of retval) ^ inv\_from\_host) and mask\_from\_host).* unsigned [user\\_config3](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG3\_SIZE
- *Label is equal to user configured label 3.* unsigned [user\\_config2](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG2\_SIZE
- *Label is equal to user configured label 2.* unsigned [user\\_config1](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG1\_SIZE
- *Label is equal to user configured label 1.* unsigned [user\\_config0](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG0\_SIZE
- *Label is equal to user configured label 0.* unsigned [ttl\\_is\\_one](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ONE\_SIZE
- *TTL is one.* unsigned [ttl\\_is\\_zero](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ZERO\_SIZE
- *TTL is zero.* unsigned [reserved\\_label](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED\_LABEL\_SIZE
- *Reserved label - label value is less than 16.* unsigned [end\\_of\\_stack](#): EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_END\_OF\_STACK\_SIZE
- *Last label on MPLS stack.* }
- };

### Detailed Description

Decode MPLS label return value struct definition.

### Field Documentation

[ezdp\\_decode\\_mpls\\_label\\_retval\\_t ezdp\\_decode\\_mpls\\_label\\_retval::raw\\_data](#)

unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::pad0](#)

Reserved bits 10 to 31.

unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::stop\\_bit](#)

Stop bit = exception\_bit | (~exception\_bit and ~reserved\_label).

unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::exception\\_bit](#)

Expection bit = or\_reduce (((first 8 bits of retval) ^ inv\_from\_host) and mask\_from\_host).

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::user\\_config3](#)**

Label is equal to user configured label 3.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::user\\_config2](#)**

Label is equal to user configured label 2.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::user\\_config1](#)**

Label is equal to user configured label 1.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::user\\_config0](#)**

Label is equal to user configured label 0.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::ttl\\_is\\_one](#)**

TTL is one.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::ttl\\_is\\_zero](#)**

TTL is zero.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::reserved\\_label](#)**

Reserved label - label value is less than 16.

**unsigned [ezdp\\_decode\\_mpls\\_label\\_retval::end\\_of\\_stack](#)**

Last label on MPLS stack.

**union { ... }**

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mpls\_result Struct Reference

[ezdp\\_decode\\_mpls\\_result](#) struct for ezdp

### Data Fields

- union {
- [ezdp\\_decode\\_mpls\\_result\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#): EZDP\_DECODE\_MPLS\_RESULT\_RESERVED28\_31\_SIZE
- *Reserved bits 28 to 31.* unsigned [label4\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ONE\_SIZE
- *Label4 TTL is one.* unsigned [label3\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ONE\_SIZE
- *Label3 TTL is one.* unsigned [label2\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ONE\_SIZE
- *Label2 TTL is one.* unsigned [label1\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ONE\_SIZE
- *Label1 TTL is one.* unsigned [pad1](#): EZDP\_DECODE\_MPLS\_RESULT\_RESERVED20\_23\_SIZE
- *Reserved bits 20 to 23.* unsigned [label4\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ZERO\_SIZE
- *Label4 TTL is zero.* unsigned [label3\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ZERO\_SIZE
- *Label3 TTL is zero.* unsigned [label2\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ZERO\_SIZE
- *Label2 TTL is zero.* unsigned [label1\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ZERO\_SIZE
- *Label1 TTL is zero.* unsigned [pad2](#): EZDP\_DECODE\_MPLS\_RESULT\_RESERVED10\_15\_SIZE
- *Reserved bits 10 to 15.* unsigned [last\\_entry\\_in\\_stack](#):  
EZDP\_DECODE\_MPLS\_RESULT\_LAST\_ENTRY\_IN\_STACK\_SIZE
- *Last entry in stack.* unsigned [pad3](#): EZDP\_DECODE\_MPLS\_RESULT\_RESERVED1\_7\_SIZE
- *Reserved bits 1 to 7.* unsigned [decode\\_error](#): EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERROR\_SIZE
- *Error flag: decode failed.* }
- };

### Detailed Description

[ezdp\\_decode\\_mpls\\_result](#) struct for ezdp

### Field Documentation

[ezdp\\_decode\\_mpls\\_result\\_t ezdp\\_decode\\_mpls\\_result::raw\\_data](#)

unsigned [ezdp\\_decode\\_mpls\\_result::pad0](#)

Reserved bits 28 to 31.

unsigned [ezdp\\_decode\\_mpls\\_result::label4\\_ttl\\_is\\_one](#)

Label4 TTL is one.

**unsigned [ezdp\\_decode\\_mpls\\_result::label3 ttl is one](#)**

Label3 TTL is one.

**unsigned [ezdp\\_decode\\_mpls\\_result::label2 ttl is one](#)**

Label2 TTL is one.

**unsigned [ezdp\\_decode\\_mpls\\_result::label1 ttl is one](#)**

Label1 TTL is one.

**unsigned [ezdp\\_decode\\_mpls\\_result:: pad1](#)**

Reserved bits 20 to 23.

**unsigned [ezdp\\_decode\\_mpls\\_result::label4 ttl is zero](#)**

Label4 TTL is zero.

**unsigned [ezdp\\_decode\\_mpls\\_result::label3 ttl is zero](#)**

Label3 TTL is zero.

**unsigned [ezdp\\_decode\\_mpls\\_result::label2 ttl is zero](#)**

Label2 TTL is zero.

**unsigned [ezdp\\_decode\\_mpls\\_result::label1 ttl is zero](#)**

Label1 TTL is zero.

**unsigned [ezdp\\_decode\\_mpls\\_result:: pad2](#)**

Reserved bits 10 to 15.

**unsigned [ezdp\\_decode\\_mpls\\_result::last\\_entry\\_in\\_stack](#)**

Last entry in stack.

**unsigned [ezdp\\_decode\\_mpls\\_result:: pad3](#)**

Reserved bits 1 to 7.

**unsigned [ezdp\\_decode\\_mpls\\_result::decode\\_error](#)**

Error flag: decode failed.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_mpls\_retval Struct Reference

Decode MPLS return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_mpls\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED28\_31\_SIZE
- *Reserved bits 28 to 31.* unsigned [label4\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL\_IS\_ONE\_SIZE
- *Label4 TTL is one.* unsigned [label3\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL\_IS\_ONE\_SIZE
- *Label3 TTL is one.* unsigned [label2\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_IS\_ONE\_SIZE
- *Label2 TTL is one.* unsigned [label1\\_ttl\\_is\\_one](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_IS\_ONE\_SIZE
- *Label1 TTL is one.* unsigned [\\_\\_pad1](#) : EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED20\_23\_SIZE
- *Reserved bits 20 to 23.* unsigned [label4\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL\_IS\_ZERO\_SIZE
- *Label4 TTL is zero.* unsigned [label3\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL\_IS\_ZERO\_SIZE
- *Label3 TTL is zero.* unsigned [label2\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_IS\_ZERO\_SIZE
- *Label2 TTL is zero.* unsigned [label1\\_ttl\\_is\\_zero](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_IS\_ZERO\_SIZE
- *Label1 TTL is zero.* unsigned [\\_\\_pad2](#) : EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED10\_15\_SIZE
- *Reserved bits 10 to 15.* unsigned [last\\_entry\\_in\\_stack](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_LAST\_ENTRY\_IN\_STACK\_SIZE
- *Last entry in stack.* unsigned [\\_\\_pad3](#) : EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED1\_7\_SIZE
- *Reserved bits 1 to 7.* unsigned [decode\\_error](#):  
EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ERROR\_SIZE
- *Error flag: decode failed.* }
- };

---

### Detailed Description

Decode MPLS return value struct definition.

---

### Field Documentation

[ezdp\\_decode\\_mpls\\_retval\\_t ezdp\\_decode\\_mpls\\_retval::raw\\_data](#)

unsigned [ezdp\\_decode\\_mpls\\_retval::\\_\\_pad0](#)

Reserved bits 28 to 31.

unsigned [ezdp\\_decode\\_mpls\\_retval::label4\\_ttl\\_is\\_one](#)

Label4 TTL is one.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::label3 ttl is one](#)

Label3 TTL is one.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::label2 ttl is one](#)

Label2 TTL is one.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::label1 ttl is one](#)

Label1 TTL is one.

**unsigned** [ezdp\\_decode\\_mpls\\_retval:: pad1](#)

Reserved bits 20 to 23.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::label4 ttl is zero](#)

Label4 TTL is zero.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::label3 ttl is zero](#)

Label3 TTL is zero.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::label2 ttl is zero](#)

Label2 TTL is zero.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::label1 ttl is zero](#)

Label1 TTL is zero.

**unsigned** [ezdp\\_decode\\_mpls\\_retval:: pad2](#)

Reserved bits 10 to 15.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::last\\_entry\\_in\\_stack](#)

Last entry in stack.

**unsigned** [ezdp\\_decode\\_mpls\\_retval:: pad3](#)

Reserved bits 1 to 7.

**unsigned** [ezdp\\_decode\\_mpls\\_retval::decode\\_error](#)

Error flag: decode failed.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)



## ezdp\_decode\_tcp\_errors Struct Reference

TCP header decode error flags.

### Data Fields

- union {
- [ezdp\\_decode\\_tcp\\_errors\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_DECODE\_TCP\_ERRORS\_RESERVED3\_7\_SIZE
- Reserved bits 3 to 7.   unsigned [decode\\_error](#): EZDP\_DECODE\_TCP\_ERRORS\_DECODE\_ERROR\_SIZE
- Aggregated decode error flag.   unsigned [syn\\_and\\_fin\\_eq\\_1](#): EZDP\_DECODE\_TCP\_ERRORS\_SYN\_AND\_FIN\_EQ\_1\_SIZE
- syn==1 and fin==1   unsigned [data\\_offset\\_lt\\_5](#): EZDP\_DECODE\_TCP\_ERRORS\_DATA\_OFFSET\_LT\_5\_SIZE
- Data offset is less than 5.   }
- };

### Detailed Description

TCP header decode error flags.

### Field Documentation

[ezdp\\_decode\\_tcp\\_errors\\_t ezdp\\_decode\\_tcp\\_errors::raw\\_data](#)

unsigned [ezdp\\_decode\\_tcp\\_errors::pad0](#)

Reserved bits 3 to 7.

unsigned [ezdp\\_decode\\_tcp\\_errors::decode\\_error](#)

Aggregated decode error flag.

Indicate at least one decode error

unsigned [ezdp\\_decode\\_tcp\\_errors::syn\\_and\\_fin\\_eq\\_1](#)

syn==1 and fin==1

unsigned [ezdp\\_decode\\_tcp\\_errors::data\\_offset\\_lt\\_5](#)

Data offset is less than 5.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_decode\_tcp\_retval Struct Reference

Decode TCP return value struct definition.

### Data Fields

- union {
- [ezdp\\_decode\\_tcp\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [\\_\\_pad0](#); EZDP\_DECODE\_TCP\_RETVAL\_RESERVED24\_31\_SIZE
- Reserved bits 24 to 31.   unsigned [\\_\\_pad1](#); EZDP\_DECODE\_TCP\_RETVAL\_RESERVED22\_23\_SIZE
- Reserved bits 22 to 23.   unsigned [data\\_offset](#); EZDP\_DECODE\_TCP\_RETVAL\_DATA\_OFFSET\_SIZE
- Data offset.   unsigned [\\_\\_pad2](#); EZDP\_DECODE\_TCP\_RETVAL\_RESERVED9\_15\_SIZE
- Reserved bits 9 to 15.   unsigned [options\\_exist](#); EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIST\_SIZE
- Options exist.   struct [ezdp\\_decode\\_tcp\\_errors\\_error\\_codes](#)
- Decode error codes.   }
- };

### Detailed Description

Decode TCP return value struct definition.

### Field Documentation

[ezdp\\_decode\\_tcp\\_retval\\_t ezdp\\_decode\\_tcp\\_retval::raw\\_data](#)

unsigned [ezdp\\_decode\\_tcp\\_retval::\\_\\_pad0](#)

Reserved bits 24 to 31.

unsigned [ezdp\\_decode\\_tcp\\_retval::\\_\\_pad1](#)

Reserved bits 22 to 23.

unsigned [ezdp\\_decode\\_tcp\\_retval::data\\_offset](#)

Data offset.

unsigned [ezdp\\_decode\\_tcp\\_retval::\\_\\_pad2](#)

Reserved bits 9 to 15.

unsigned [ezdp\\_decode\\_tcp\\_retval::options\\_exist](#)

Options exist.

struct [ezdp\\_decode\\_tcp\\_errors ezdp\\_decode\\_tcp\\_retval::error\\_codes](#) [read]

Decode error codes.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_decode\\_defs.h](#)

## ezdp\_driver\_desc Struct Reference

TX/RX descriptor.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DRIVER\_DESC\_WORD\_COUNT]
- struct {
- uint64\_t [buf\\_data\\_addr](#)
- Address of Buffer data.   uint32\_t [len](#)
- Buffer data length.   uint16\_t [sub\\_type](#)
- TODO: define.   struct [ezdp\\_driver\\_desc\\_flags](#) [flags](#)
- marks ring descriptor/buffer pair state and type   uint8\_t [total](#)
- The total number of buffers in an buffer block.   }
- };

### Detailed Description

TX/RX descriptor.

### Field Documentation

uint32\_t [ezdp\\_driver\\_desc::raw\\_data](#)[EZDP\_DRIVER\_DESC\_WORD\_COUNT]

uint64\_t [ezdp\\_driver\\_desc::buf\\_data\\_addr](#)

Address of Buffer data.

uint32\_t [ezdp\\_driver\\_desc::len](#)

Buffer data length.

uint16\_t [ezdp\\_driver\\_desc::sub\\_type](#)

TODO: define.

struct [ezdp\\_driver\\_desc\\_flags](#) [ezdp\\_driver\\_desc::flags](#) [read]

marks ring descriptor/buffer pair state and type

uint8\_t [ezdp\\_driver\\_desc::total](#)

The total number of buffers in an buffer block.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)

## ezdp\_driver\_desc\_flags Struct Reference

TX/RX descriptor flags structure.

### Data Fields

- union {
- [ezdp\\_driver\\_desc\\_flags\\_t raw\\_data](#)
- struct {
- unsigned [type](#): EZDP\_DRIVER\_DESC\_FLAGS\_TYPE\_SIZE
- *TODO: define.* unsigned [error](#): EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_SIZE
- *TODO: define.* unsigned [owner](#): EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_SIZE
- *indicates whether the descriptor belongs to the driver (OWNER=1) or the NIC (OWNER=0)* unsigned [data](#): EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_SIZE
- *indicates whether the descriptor is a message (DATA=0) or a data (DATA=1)* }
- };

### Detailed Description

TX/RX descriptor flags structure.

### Field Documentation

[ezdp\\_driver\\_desc\\_flags\\_t ezdp\\_driver\\_desc\\_flags::raw\\_data](#)

unsigned [ezdp\\_driver\\_desc\\_flags::type](#)

*TODO: define.*

unsigned [ezdp\\_driver\\_desc\\_flags::error](#)

*TODO: define.*

unsigned [ezdp\\_driver\\_desc\\_flags::owner](#)

*indicates whether the descriptor belongs to the driver (OWNER=1) or the NIC (OWNER=0)*

unsigned [ezdp\\_driver\\_desc\\_flags::data](#)

*indicates whether the descriptor is a message (DATA=0) or a data (DATA=1)*

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)

## ezdp\_dual\_add32\_result Struct Reference

The result of the atomic dual add32 instruction.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DUAL\_ADD32\_RESULT\_WORD\_COUNT]
- struct {
- int32\_t [original\\_value2](#)
- *The original value of variable 2.*   int32\_t [original\\_value1](#)
- *The original value of variable 1.*   }
- };

### Detailed Description

The result of the atomic dual add32 instruction.

### Field Documentation

uint32\_t [ezdp\\_dual\\_add32\\_result::raw\\_data](#)[EZDP\_DUAL\_ADD32\_RESULT\_WORD\_COUNT]

int32\_t [ezdp\\_dual\\_add32\\_result::original\\_value2](#)

The original value of variable 2.

int32\_t [ezdp\\_dual\\_add32\\_result::original\\_value1](#)

The original value of variable 1.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_memory\\_defs.h](#)

## ezdp\_dual\_add64\_result Struct Reference

The result of the atomic dual add64 instruction.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DUAL\_ADD64\_RESULT\_WORD\_COUNT]
- struct {
- int64\_t [original\\_value2](#)
- *The original value of variable 2.*   int64\_t [original\\_value1](#)
- *The original value of variable 1.*   }
- };

---

### Detailed Description

The result of the atomic dual add64 instruction.

---

### Field Documentation

uint32\_t [ezdp\\_dual\\_add64\\_result::raw\\_data](#)[EZDP\_DUAL\_ADD64\_RESULT\_WORD\_COUNT]

int64\_t [ezdp\\_dual\\_add64\\_result::original\\_value2](#)

The original value of variable 2.

int64\_t [ezdp\\_dual\\_add64\\_result::original\\_value1](#)

The original value of variable 1.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_memory\\_defs.h](#)

## ezdp\_dual\_ctr Struct Reference

On-demand dual counter value.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DUAL\_CTR\_WORD\_COUNT]
- struct {
- uint64\_t [byte](#)
- *Byte counter value.*     uint64\_t [event](#)
- *Event counter value.*   }
- };

---

### Detailed Description

On-demand dual counter value.

---

### Field Documentation

uint32\_t [ezdp\\_dual\\_ctr::raw\\_data](#)[EZDP\_DUAL\_CTR\_WORD\_COUNT]

uint64\_t [ezdp\\_dual\\_ctr::byte](#)

Byte counter value.

uint64\_t [ezdp\\_dual\\_ctr::event](#)

Event counter value.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_dual\_ctr\_cfg Struct Reference

On-demand dual counter configuration definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DUAL\_CTR\_CFG\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0\\_\\_](#): EZDP\_DUAL\_CTR\_CFG\_ECC\_SIZE
- ECC.   unsigned [\\_\\_pad1\\_\\_](#): EZDP\_DUAL\_CTR\_CFG\_RESERVED19\_23\_SIZE
- Reserved bits 19 to 23 (sub type - double=14).   unsigned [byte\\_report\\_exceeded](#): EZDP\_DUAL\_CTR\_CFG\_BYTE\_REPORT\_EXCEEDED\_SIZE
- Number of bits threshold on byte counter.   unsigned [\\_\\_pad2\\_\\_](#):
- EZDP\_DUAL\_CTR\_CFG\_CLR\_ON\_GC\_SIZE
- Clear counter when generating garbage collection message.   unsigned [enable\\_exceed\\_message](#):
- EZDP\_DUAL\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_SIZE
- Enable threshold exceed message.   unsigned [event\\_report\\_exceeded](#):
- EZDP\_DUAL\_CTR\_CFG\_EVENT\_REPORT\_EXCEEDED\_SIZE
- Number of bits threshold on event counter.   unsigned [byte\\_value\\_size](#):
- EZDP\_DUAL\_CTR\_CFG\_BYTE\_VALUE\_SIZE\_SIZE
- Byte counter size.   unsigned [\\_\\_pad3\\_\\_](#): EZDP\_DUAL\_CTR\_CFG\_RESERVED0\_SIZE
- Reserved bit 0 (overflow flag).   struct [ezdp\\_dual\\_ctr\\_value](#)
- Counter value.   }
- };

### Detailed Description

On-demand dual counter configuration definition.

### Field Documentation

uint32\_t [ezdp\\_dual\\_ctr\\_cfg::raw\\_data](#)[EZDP\_DUAL\_CTR\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_dual\\_ctr\\_cfg::\\_\\_pad0\\_\\_](#)

ECC.

unsigned [ezdp\\_dual\\_ctr\\_cfg::\\_\\_pad1\\_\\_](#)

Reserved bits 19 to 23 (sub type - double=14).

unsigned [ezdp\\_dual\\_ctr\\_cfg::byte\\_report\\_exceeded](#)

Number of bits threshold on byte counter.

unsigned [ezdp\\_dual\\_ctr\\_cfg::\\_\\_pad2\\_\\_](#)

Clear counter when generating garbage collection message.

Should always be set to 1.



unsigned [ezdp\\_dual\\_ctr\\_cfg::enable\\_exceed\\_message](#)

Enable threshold exceed message.

unsigned [ezdp\\_dual\\_ctr\\_cfg::event\\_report\\_exceeded](#)

Number of bits threshold on event counter.

unsigned [ezdp\\_dual\\_ctr\\_cfg::byte\\_value\\_size](#)

Byte counter size.

unsigned [ezdp\\_dual\\_ctr\\_cfg::pad3](#)

Reserved bit 0 (overflow flag).

struct [ezdp\\_dual\\_ctr\\_cfg::value](#) [read]

Counter value.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_dual\_ctr\_result Struct Reference

On-demand dual value counter result value.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_DUAL\_CTR\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_DUAL\_CTR\_RESULT\_RESERVED31\_SIZE
- Reserved bit 31.    unsigned [byte\\_value\\_msb](#): EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_MSB\_SIZE
- MSB byte counter value.    unsigned [byte\\_value\\_lsb](#):  
EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_LSB\_SIZE
- LSB byte counter value.    unsigned [event\\_value](#): EZDP\_DUAL\_CTR\_RESULT\_EVENT\_VALUE\_SIZE
- Event counter value.    }
- };

### Detailed Description

On-demand dual value counter result value.

### Field Documentation

uint32\_t [ezdp\\_dual\\_ctr\\_result::raw\\_data](#)[EZDP\_DUAL\_CTR\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_dual\\_ctr\\_result::\\_\\_pad0](#)

Reserved bit 31.

unsigned [ezdp\\_dual\\_ctr\\_result::byte\\_value\\_msb](#)

MSB byte counter value.

unsigned [ezdp\\_dual\\_ctr\\_result::byte\\_value\\_lsb](#)

LSB byte counter value.

unsigned [ezdp\\_dual\\_ctr\\_result::event\\_value](#)

Event counter value.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_ext\_addr Struct Reference

Extended address definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_EXT\_ADDR\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_EXT\_ADDR\_RESERVED16\_31\_SIZE
- Reserved bits 16 to 31.   unsigned [\\_\\_pad1](#) : EZDP\_EXT\_ADDR\_RESERVED14\_15\_SIZE
- Reserved bits 14-15.   unsigned [msid](#): EZDP\_EXT\_ADDR\_MSID\_SIZE
- < Type of the MSID   unsigned [\\_\\_pad2](#) : EZDP\_EXT\_ADDR\_RESERVED4\_7\_SIZE
- Reserved bits 4-7.   unsigned [address\\_msb](#): EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_SIZE
- 4 msb of 36 bits extended address   uint32\_t [address](#)
- 32 lsb of 36 bit extended address   }
- };

### Detailed Description

Extended address definition.

### Field Documentation

uint32\_t [ezdp\\_ext\\_addr::raw\\_data](#)[EZDP\_EXT\_ADDR\_WORD\_COUNT]

unsigned [ezdp\\_ext\\_addr::\\_\\_pad0](#)

Reserved bits 16 to 31.

unsigned [ezdp\\_ext\\_addr::\\_\\_pad1](#)

Reserved bits 14-15.

unsigned [ezdp\\_ext\\_addr::msid](#)

Type of the MSID

MSID select

unsigned [ezdp\\_ext\\_addr::\\_\\_pad2](#)

Reserved bits 4-7.

unsigned [ezdp\\_ext\\_addr::address\\_msb](#)

4 msb of 36 bits extended address

uint32\_t [ezdp\\_ext\\_addr::address](#)

32 lsb of 36 bit extended address

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_memory\\_defs.h](#)

## ezdp\_ext\_linked\_buffers\_desc Struct Reference

Extended linked buffers descriptor.

### Data Fields

- struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) [line](#) [EZDP\_EXTENDED\_LBD]  
*Array of 16 linked buffers lines, which contains up to 48 buffers.*
- 

### Detailed Description

Extended linked buffers descriptor.

May hold up to 48 buffs Applicable only in EZDP\_EXT\_FRAME frame type

---

### Field Documentation

struct [ezdp\\_linked\\_buffers\\_desc\\_line](#)  
[ezdp\\_ext\\_linked\\_buffers\\_desc::line](#)[EZDP\_EXTENDED\_LBD] [read]

Array of 16 linked buffers lines, which contains up to 48 buffers.

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_flow\_control\_status Struct Reference

Flow control status.

### Data Fields

- union {
- [ezdp\\_flow\\_control\\_status\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_FLOW\_CONTROL\_STATUS\_RESERVED4\_7\_SIZE
- Reserved bits 4 to 7.   unsigned [enable](#): EZDP\_FLOW\_CONTROL\_STATUS\_ENABLE\_SIZE
- Indicate if flow control element is enabled.   }
- };

---

### Detailed Description

Flow control status.

---

### Field Documentation

[ezdp\\_flow\\_control\\_status\\_t ezdp\\_flow\\_control\\_status::raw\\_data](#)

unsigned [ezdp\\_flow\\_control\\_status::pad0](#)

Reserved bits 4 to 7.

unsigned [ezdp\\_flow\\_control\\_status::enable](#)

Indicate if flow control element is enabled.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_frame\_desc Struct Reference

Frame descriptor data structure.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_FRAME\_DESC\_WORD\_COUNT]
- struct {
- uint8\_t [ecc](#)
- Eight-bit ECC field protecting bytes 0x1-0xf frame descriptor data.   unsigned [transmit\\_confirmation\\_flag](#): EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATION\_FLAG\_SIZE
- < The frame type field defines buffer organization and identifies which buffer type is pointed by buffer (BD).   unsigned [timestamp\\_flag](#): EZDP\_FRAME\_DESC\_TIMESTAMP\_FLAG\_SIZE
- On receive path, this flag indicates presence of timestamp value in [ezdp\\_job\\_desc.rx\\_info](#).   unsigned [transmit\\_keep\\_buf\\_flag](#): EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUF\_FLAG\_SIZE
- On receive path, this flag is echoed on confirmation responses.   unsigned [gross\\_checksum\\_flag](#): EZDP\_FRAME\_DESC\_GROSS\_CHECKSUM\_FLAG\_SIZE
- On receive path, the checksum flag indicates the checksum has been calculated for the frame.   unsigned [pad0](#): EZDP\_FRAME\_DESC\_RESERVED0\_1\_SIZE
- Reserved bits 0 to 1.   unsigned [pad1](#): EZDP\_FRAME\_DESC\_RESERVED14\_15\_SIZE
- Reserved bits 14 to 15.   unsigned [class\\_of\\_service](#): EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_SIZE
- Frame class of service grade.   unsigned [pad2](#): EZDP\_FRAME\_DESC\_RESERVED10\_11\_SIZE
- Reserved bits 10 to 11.   unsigned [buf\\_budget\\_id](#): EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_SIZE
- Budget group ID.   uint16\_t [frame\\_length](#)
- Total frame data length in bytes.   uint8\_t [data\\_buf\\_count](#)
- This field indicates the number of buffers (BDs) that hold the frame (include empty and null buffers).   uint8\_t [header\\_offset](#)
- This is the frame header starting point in the first frame data buffer.   struct [ezdp\\_buffer\\_desc buf\\_desc](#)
- Pointer to a 256B buffer located either in IMEM or EMEM.   uint8\_t [free\\_bytes](#)
- This field indicates how many free bytes are left at the end of the frame header data buffer.   uint8\_t [logical\\_id](#)
- Logical user info assigned by configuration to interfaces, Can be used to identify interfaces or group of same interface type.   unsigned [pad3](#): EZDP\_FRAME\_DESC\_RESERVED106\_110\_SIZE
- < Multicast control   unsigned [job\\_budget\\_id](#): EZDP\_FRAME\_DESC\_JOB\_BUDGET\_ID\_SIZE
- Job budget group ID.   }
- };

### Detailed Description

Frame descriptor data structure.

### Field Documentation

uint32\_t [ezdp\\_frame\\_desc::raw\\_data](#)[EZDP\_FRAME\_DESC\_WORD\_COUNT]

uint8\_t [ezdp\\_frame\\_desc::ecc](#)

Eight-bit ECC field protecting bytes 0x1-0xf frame descriptor data.

This field can be used by the SW if frame descriptor is not saved in DDR.

**unsigned [ezdp\\_frame\\_desc::transmit\\_confirmation\\_flag](#)**

< The frame type field defines buffer organization and identifies which buffer type is pointed by buffer (BD).

On receive path, this flag indicates that received job is a confirmation response. On transmit path, this flag is used as a request to return job confirmation when transmission is done. NOTE: When timestamp flag is on in addition to transmit\_confirmation flag, the confirmation frame (FD) will contain timestamp timer value captured on the MAC. This option is used for two-step 1588.

**unsigned [ezdp\\_frame\\_desc::timestamp\\_flag](#)**

On receive path, this flag indicates presence of timestamp value in [ezdp\\_job\\_desc.rx\\_info](#).

For two-step 1588 confirmation frame this flag also indicate success of adding 1588 timestamp. Or in the other words, timestamp\_flag will be off if adding captured 1588 timestamp failed. On transmit path, this flag is used as a request to add captured 1588 timestamp to header by MAC. For two-step 1588 packet, where confirmation response is required from the MAC, the transmit\_confirmation flag should also be on.

**unsigned [ezdp\\_frame\\_desc::transmit\\_keep\\_buf\\_flag](#)**

On receive path, this flag is echoed on confirmation responses.

On transmit path, this flag indicates to keep the frame buffers. It is SW responsibility to free frame buffers to prevent memory leakage. When FD is sent to the PMU flush queue TKB flag is implied to be zero (usage of PMU flush queue forces buffer release operation regardless of FD[TKB] state).

**unsigned [ezdp\\_frame\\_desc::gross\\_checksum\\_flag](#)**

On receive path, the checksum flag indicates the checksum has been calculated for the frame.

On transmit path, this flag is reserved on NPS-400.

**unsigned [ezdp\\_frame\\_desc::pad0](#)**

Reserved bits 0 to 1.

**unsigned [ezdp\\_frame\\_desc::pad1](#)**

Reserved bits 14 to 15.

**unsigned [ezdp\\_frame\\_desc::class\\_of\\_service](#)**

Frame class of service grade.

**unsigned [ezdp\\_frame\\_desc::pad2](#)**

Reserved bits 10 to 11.



**unsigned [ezdp\\_frame\\_desc::buf\\_budget\\_id](#)**

Budget group ID.

Budget identifies an allocated IMEM or EMEM buffer resource control operation. By default a budget group ID is associated with an Rx port ID from which the frame was received.

**uint16\_t [ezdp\\_frame\\_desc::frame\\_length](#)**

Total frame data length in bytes.

Represents the actual frame data received/transmitted on the wire starting from L2 frame header. The four-layer 2-byte CRC may be included (if the MAC is configured for keeping CRC bytes on Rx and/or not adding CRC bytes on Tx) or excluded (if the MAC is configured to strip CRC on Rx and/or add CRC on Tx). `frame_length` excludes all overheads such as frame context and other bytes preceding the `header_offset` position. Value of 0x3FFF represents that frame data length is greater or equal to 16383 bytes, therefore max length that can be accurately represented by an exact value is 0x3FFE (16382 bytes).

**uint8\_t [ezdp\\_frame\\_desc::data\\_buf\\_count](#)**

This field indicates the number of buffers (BDs) that hold the frame (include empty and null buffers).

Note that for an Extended frame the value does not include the buffer where the linked buffers descriptors is stored.

**uint8\_t [ezdp\\_frame\\_desc::header\\_offset](#)**

This is the frame header starting point in the first frame data buffer.

To include embedded LBD in the same buffer of the first frame data (EMBEDDED\_BD frame type), the `header_offset` must be at least the max configured embedded LBD size (16B or 32B). The gaps from beginning of the buffer or from max embedded LBD size to `header_offset` can be utilized by frame context or can be used for optimized header modification (increasing or decreasing).

**struct [ezdp\\_buffer\\_desc ezdp\\_frame\\_desc::buf\\_desc](#) [read]**

Pointer to a 256B buffer located either in IMEM or EMEM.

The type to which the buffer points is determined by frame type.

**uint8\_t [ezdp\\_frame\\_desc::free\\_bytes](#)**

This field indicates how many free bytes are left at the end of the frame header data buffer.

This field is valid only for STANDARD frames.

**uint8\_t [ezdp\\_frame\\_desc::logical\\_id](#)**

Logical user info assigned by configuration to interfaces, Can be used to identify interfaces or group of same interface type.

unsigned [ezdp\\_frame\\_desc::pad3](#)

< Multicast control

Reserved bits 106 to 110

unsigned [ezdp\\_frame\\_desc::job\\_budget\\_id](#)

Job budget group ID.

Budget identifies an allocated job resource control operations. By default a budget group ID is associated with an Rx port ID from which the frame was received.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_group\_schlr\_status Struct Reference

PMU group scheduler status (based on PMU system info).

### Data Fields

- union {
- [ezdp\\_group\\_schlr\\_status\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#); EZDP\_GROUP\_SCHLR\_STATUS\_RESERVED13\_15\_SIZE
- *Reserved bits 13 to 15.*   unsigned [dispatched\\_job](#);
- EZDP\_GROUP\_SCHLR\_STATUS\_DISPATCHED\_JOB\_SIZE
- *The number of jobs dispatched from the group scheduler.*   }
- };

---

### Detailed Description

PMU group scheduler status (based on PMU system info).

There are 16 group schedulers in each PMU side.

---

### Field Documentation

[ezdp\\_group\\_schlr\\_status\\_t ezdp\\_group\\_schlr\\_status::raw\\_data](#)

unsigned [ezdp\\_group\\_schlr\\_status::pad0](#)

Reserved bits 13 to 15.

unsigned [ezdp\\_group\\_schlr\\_status::dispatched\\_job](#)

The number of jobs dispatched from the group scheduler.

Defined as number of jobs that were dispatched from the group scheduler for processing and are waiting for "job done".

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_hier\_tb\_ctr\_cfg Struct Reference

Statistic hierarchical token bucket counter config structure (write cfg usage).

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_HIER\_TB\_CTR\_CFG\_WORD\_COUNT]
- struct {
- unsigned [ctr0\\_fail\\_threshold](#): EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_FAIL\_THRESHOLD\_SIZE
- *Fail threshold for first front end counter/accumulator.* unsigned [pad0](#) : EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED22\_26\_SIZE
- *reserved bit 22-26.* unsigned [ctr\\_sum\\_fail\\_threshold](#): EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_FAIL\_THRESHOLD\_SIZE
- *Fail threshold for sum of first and second front end counters/accumulators.* unsigned [ctr\\_sum\\_updt\\_threshold](#): EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_UPDT\_THRESHOLD\_SIZE
- *Update threshold for sum of first and second front end counters/accumulators.* unsigned [ctr1\\_fail\\_threshold](#): EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_FAIL\_THRESHOLD\_SIZE
- *Fail threshold for second front end counter/accumulator.* unsigned [ctr1\\_updt\\_threshold](#): EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_UPDT\_THRESHOLD\_SIZE
- *Update threshold for second front end counter/accumulator.* unsigned [pad1](#) : EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED0\_1\_SIZE
- *reserved bits 0-1* unsigned [pad2](#) : EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED63\_SIZE
- *reserved bits 63.* unsigned [timestamp\\_threshold](#): EZDP\_HIER\_TB\_CTR\_CFG\_TIMESTAMP\_THRESHOLD\_SIZE
- *Selects one of 4 sets of timestamp thresholds for this counter.* unsigned [ctr0\\_updt\\_threshold](#): EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_UPDT\_THRESHOLD\_SIZE
- *Update threshold for second front end counter/accumulator.* unsigned [app\\_bits](#): EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_SIZE
- *The application specific bits of the counter.* }
- };

### Detailed Description

Statistic hierarchical token bucket counter config structure (write cfg usage).

### Field Documentation

uint32\_t [ezdp\\_hier\\_tb\\_ctr\\_cfg::raw\\_data](#)[EZDP\_HIER\_TB\_CTR\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::ctr0\\_fail\\_threshold](#)

Fail threshold for first front end counter/accumulator.

Possible values: 0 - 17.

unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::pad0](#)

reserved bit 22-26.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::ctr\\_sum\\_fail\\_threshold](#)**

Fail threshold for sum of first and second front end counters/accumulators.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::ctr\\_sum\\_updt\\_threshold](#)**

Update threshold for sum of first and second front end counters/accumulators.

Possible values: 0 - 17.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::ctr1\\_fail\\_threshold](#)**

Fail threshold for second front end counter/accumulator.

Possible values: 0 - 17.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::ctr1\\_updt\\_threshold](#)**

Update threshold for second front end counter/accumulator.

Possible values: 0 - 17.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::pad1](#)**

reserved bits 0-1

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::pad2](#)**

reserved bits 63.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::timestamp\\_threshold](#)**

Selects one of 4 sets of timestamp thresholds for this counter.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::ctr0\\_updt\\_threshold](#)**

Update threshold for second front end counter/accumulator.

Possible values: 0 - 17.

**unsigned [ezdp\\_hier\\_tb\\_ctr\\_cfg::app\\_bits](#)**

The application specific bits of the counter.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_hier\_tb\_result Struct Reference

Hierarchical token bucket counter result value definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_HIER\_TB\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [fail](#): EZDP\_HIER\_TB\_RESULT\_FAIL\_SIZE
- *Operation Failed.*     unsigned [update\\_task](#): EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_SIZE
- *BE update procedure is required.*     unsigned [ctr1](#): EZDP\_HIER\_TB\_RESULT\_CTR1\_SIZE
- *< The state of the counter.*     unsigned [pad0](#) : EZDP\_HIER\_TB\_RESULT\_RESERVED0\_9\_SIZE
- *reserved bits 0-9*     unsigned [pad1](#) : EZDP\_HIER\_TB\_RESULT\_RESERVED56\_63\_SIZE
- *reserved bits 56-63*     unsigned [app\\_bits](#): EZDP\_HIER\_TB\_RESULT\_APP\_BITS\_SIZE
- *The application specific bits of the counter, as was configured by config API.*     unsigned [pad2](#) : EZDP\_HIER\_TB\_RESULT\_RESERVED82\_95\_SIZE
- *reserved bits 82-95*     unsigned [ctr0](#): EZDP\_HIER\_TB\_RESULT\_CTR0\_SIZE
- *Value of first front end counter/accumulator (pre colored green).*     }
- };

### Detailed Description

Hierarchical token bucket counter result value definition.

### Field Documentation

uint32\_t [ezdp\\_hier\\_tb\\_result::raw\\_data](#)[EZDP\_HIER\_TB\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_hier\\_tb\\_result::fail](#)

Operation Failed.

unsigned [ezdp\\_hier\\_tb\\_result::update\\_task](#)

BE update procedure is required.

unsigned [ezdp\\_hier\\_tb\\_result::ctr1](#)

< The state of the counter.

Value of first front end counter/accumulator (pre colored yellow)

unsigned [ezdp\\_hier\\_tb\\_result::pad0](#)

reserved bits 0-9

unsigned [ezdp\\_hier\\_tb\\_result:: pad1](#)

reserved bits 56-63

unsigned [ezdp\\_hier\\_tb\\_result::app\\_bits](#)

The application specific bits of the counter, as was configured by config API.

unsigned [ezdp\\_hier\\_tb\\_result:: pad2](#)

reserved bits 82-95

unsigned [ezdp\\_hier\\_tb\\_result::ctr0](#)

Value of first front end counter/accumulator (pre colored green).

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_hier\_tb\_ug\_app\_bits Struct Reference

Application bits of Hierarchical token bucket for ultra green feature.

### Data Fields

- union {
- [ezdp\\_hier\\_tb\\_ug\\_app\\_bits\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_HIER\_TB\_UG\_APP\_BITS\_RESERVED24\_31\_SIZE
- *bits out of app\_bit size.* unsigned [app\\_bits](#): EZDP\_HIER\_TB\_UG\_APP\_BITS\_APP\_BITS\_SIZE
- *The application specific bits of the counter.* unsigned [eighth\\_mode\\_ret\\_bits](#): EZDP\_HIER\_TB\_UG\_APP\_BITS\_EIGHTH\_MODE\_RET\_BITS\_SIZE
- *Returned bits for 8 bit mode Note: Will be updated while updating app\_bits.* unsigned [color\\_state\\_y](#): EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STATE\_Y\_SIZE
- *State of the buckets for pre-color yellow.* unsigned [color\\_state\\_g](#): EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STATE\_G\_SIZE
- *State of the buckets for pre-color green.* }
- };

### Detailed Description

Application bits of Hierarchical token bucket for ultra green feature.

### Field Documentation

[ezdp\\_hier\\_tb\\_ug\\_app\\_bits\\_t ezdp\\_hier\\_tb\\_ug\\_app\\_bits::raw\\_data](#)

unsigned [ezdp\\_hier\\_tb\\_ug\\_app\\_bits::pad0](#)

bits out of app\_bit size.

reserved bits 24-31

unsigned [ezdp\\_hier\\_tb\\_ug\\_app\\_bits::app\\_bits](#)

The application specific bits of the counter.

This field is part of Hybrid hier\_tb (ultra green feature

unsigned [ezdp\\_hier\\_tb\\_ug\\_app\\_bits::eighth\\_mode\\_ret\\_bits](#)

Returned bits for 8 bit mode Note: Will be updated while updating app\_bits.

unsigned [ezdp\\_hier\\_tb\\_ug\\_app\\_bits::color\\_state\\_y](#)

State of the buckets for pre-color yellow.

0x3 mean ultra yellow Note: Will be updated while updating app\_bits

unsigned [ezdp\\_hier\\_tb\\_ug\\_app\\_bits::color\\_state\\_g](#)



State of the buckets for pre-color green.

0x3 mean ultra greenNote: Will be updated while updating app\_bits

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_hier\_tb\_update Struct Reference

Hierarchical token bucket update counter definition.

### Data Fields

- union {
- [ezdp\\_hier\\_tb\\_update\\_t raw\\_data](#)
- struct {
- unsigned [set\\_active\\_state](#): EZDP\_HIER\_TB\_UPDATE\_SET\_ACTIVE\_STATE\_SIZE
- Set active state (non conditional).   unsigned [clr\\_ctr](#): EZDP\_HIER\_TB\_UPDATE\_CLR\_CTR\_SIZE
- Clear the front end counters value.   unsigned [set\\_app\\_bits](#): EZDP\_HIER\_TB\_UPDATE\_SET\_APP\_BITS\_SIZE
- Set the app specific bits.   unsigned [cond\\_set\\_active\\_state](#): EZDP\_HIER\_TB\_UPDATE\_COND\_SET\_ACTIVE\_STATE\_SIZE
- Set active state only if both front end counters are 0.   unsigned [pad0](#): EZDP\_HIER\_TB\_UPDATE\_RESERVED24\_27\_SIZE
- reserved 24-27   unsigned [app\\_bits](#): EZDP\_HIER\_TB\_UPDATE\_APP\_BITS\_SIZE
- The new application specific bits of the counter.   }
- };

---

### Detailed Description

Hierarchical token bucket update counter definition.

---

### Field Documentation

[ezdp\\_hier\\_tb\\_update\\_t ezdp\\_hier\\_tb\\_update::raw\\_data](#)

unsigned [ezdp\\_hier\\_tb\\_update::set\\_active\\_state](#)

Set active state (non conditional).

unsigned [ezdp\\_hier\\_tb\\_update::clr\\_ctr](#)

Clear the front end counters value.

unsigned [ezdp\\_hier\\_tb\\_update::set\\_app\\_bits](#)

Set the app specific bits.

unsigned [ezdp\\_hier\\_tb\\_update::cond\\_set\\_active\\_state](#)

Set active state only if both front end counters are 0.

unsigned [ezdp\\_hier\\_tb\\_update::pad0](#)

reserved 24-27

unsigned [ezdp\\_hier\\_tb\\_update::app\\_bits](#)

The new application specific bits of the counter.

Applicable only when set\_app\_bits is 1.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_input\_queue\_status Struct Reference

PMU physical input queue status definition (based on PMU system info).

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_INPUT\_QUEUE\_STATUS\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_INPUT\_QUEUE\_STATUS\_RESERVED19\_31\_SIZE
- *Reserved bits 19 to 31.*    unsigned [ready](#): EZDP\_INPUT\_QUEUE\_STATUS\_READY\_SIZE
- *The queue is ready to accept traffic.*    uint16\_t [dispatched\\_job](#)
- < *Indicate the queue congestion level.*    uint16\_t [size](#)
- *The total number of jobs in this list.*    uint16\_t [outstanding\\_job](#)
- *The number of outstanding jobs.*    }
- };

### Detailed Description

PMU physical input queue status definition (based on PMU system info).

### Field Documentation

uint32\_t [ezdp\\_input\\_queue\\_status::raw\\_data](#) [EZDP\_INPUT\_QUEUE\_STATUS\_WORD\_COUNT]

unsigned [ezdp\\_input\\_queue\\_status::\\_\\_pad0](#)

Reserved bits 19 to 31.

unsigned [ezdp\\_input\\_queue\\_status::ready](#)

The queue is ready to accept traffic.

uint16\_t [ezdp\\_input\\_queue\\_status::dispatched\\_job](#)

< Indicate the queue congestion level.

The number of dispatched jobs from the queue. Defined as number of jobs that were dispatched from the queue for processing and are waiting for "job done".

uint16\_t [ezdp\\_input\\_queue\\_status::size](#)

The total number of jobs in this list.

This include jobs which not yet dispatched, jobs which are waiting for "job done" and jobs which are waiting to get to head of the queue. The number of not dispatched job can be calculated by size - outstanding\_job.

**uint16\_t [ezdp\\_input\\_queue\\_status::outstanding\\_job](#)**

The number of outstanding jobs.

Defined as number of jobs that were dispatched from the queue and were not yet dequeued (either waiting for "job done" or waiting to get to head of the queue).

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_container\_cmd\_desc Struct Reference

Job container request.

### Data Fields

- union {
- [ezdp\\_job\\_container\\_cmd\\_desc\\_t raw\\_data](#)
- struct {
- uint16\_t [job\\_id](#)
- Pointer to job that needs to be expanded by the container.     union {
- struct {
- unsigned [pad0](#) : EZDP\_JOB\_CONTAINER\_CMD\_DESC\_RESERVED0\_11\_SIZE
- < Determine the action to process for an expanded job     }
- union {
- struct [ezdp\\_job\\_transmit\\_cmd\\_info transmit\\_info](#)
- Job transmit request info.     struct [ezdp\\_job\\_queue\\_cmd\\_info queue\\_info](#)
- Job dispatch request info.     struct [ezdp\\_job\\_discard\\_cmd\\_info discard\\_info](#)
- Job discard request info.     } [u](#)
- }
- }
- };

### Detailed Description

Job container request.

### Field Documentation

[ezdp\\_job\\_container\\_cmd\\_desc\\_t ezdp\\_job\\_container\\_cmd\\_desc::raw\\_data](#)

uint16\_t [ezdp\\_job\\_container\\_cmd\\_desc::job\\_id](#)

Pointer to job that needs to be expanded by the container.

NOTE: It must never point to another job container

unsigned [ezdp\\_job\\_container\\_cmd\\_desc::pad0](#)

< Determine the action to process for an expanded job

Reserved bits 0 to 11

struct [ezdp\\_job\\_transmit\\_cmd\\_info ezdp\\_job\\_container\\_cmd\\_desc::transmit\\_info](#) [read]

Job transmit request info.

struct [ezdp\\_job\\_queue\\_cmd\\_info ezdp\\_job\\_container\\_cmd\\_desc::queue\\_info](#) [read]

Job dispatch request info.

struct [ezdp\\_job\\_discard\\_cmd\\_info](#) [ezdp\\_job\\_container\\_cmd\\_desc::discard\\_info](#) [read]

Job discard request info.

union { ... } [ezdp\\_job\\_container\\_cmd\\_desc::u](#)

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_container\_desc Struct Reference

Job container descriptor.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_CONTAINER\_DESC\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_JOB\_CONTAINER\_DESC\_RESERVED29\_31\_SIZE
- *Reserved bits 29 to 31.*   [ezdp\\_job\\_container\\_info\\_t info](#): EZDP\_JOB\_CONTAINER\_DESC\_INFO\_SIZE
- *Number of job requests in the job container.*   unsigned [job\\_budget\\_id](#): EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_SIZE
- *Job budget group ID.*   unsigned [pad1](#) : EZDP\_JOB\_CONTAINER\_DESC\_RESERVED0\_15\_SIZE
- *Reserved bits 0 to 15.*   struct [ezdp\\_job\\_container\\_cmd\\_desc job\\_commands](#) [EZDP\_JOB\_CONTAINER\_DESC\_MAX\_NUM\_OF\_JOBS]
- *Job requests array.*   }
- };

---

### Detailed Description

Job container descriptor.

---

### Field Documentation

uint32\_t [ezdp\\_job\\_container\\_desc::raw\\_data](#) [EZDP\_JOB\_CONTAINER\_DESC\_WORD\_COUNT]

unsigned [ezdp\\_job\\_container\\_desc::pad0](#)

Reserved bits 29 to 31.

[ezdp\\_job\\_container\\_info\\_t ezdp\\_job\\_container\\_desc::info](#)

Number of job requests in the job container.

unsigned [ezdp\\_job\\_container\\_desc::job\\_budget\\_id](#)

Job budget group ID.

Budget identifies an allocated job resource control operation.

unsigned [ezdp\\_job\\_container\\_desc::pad1](#)

Reserved bits 0 to 15.

struct [ezdp\\_job\\_container\\_cmd\\_desc ezdp\\_job\\_container\\_desc::job\\_commands](#) [EZDP\_JOB\_CONTAINER\_DESC\_MAX\_NUM\_OF\_JOBS] [read]

Job requests array.

union { ... }

---



The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_desc Struct Reference

job descriptor data structure

### Data Fields

- struct [ezdp\\_frame\\_desc](#) [frame\\_desc](#)
- *Frame descriptor associated with the job.* union {
- struct [ezdp\\_job\\_rx\\_info](#) [rx\\_info](#)
- *Job receive info.* struct [ezdp\\_job\\_tx\\_info](#) [tx\\_info](#)
- *Job transmit info.* };

---

### Detailed Description

job descriptor data structure

---

### Field Documentation

struct [ezdp\\_frame\\_desc](#) [ezdp\\_job\\_desc::frame\\_desc](#) [read]

Frame descriptor associated with the job.

struct [ezdp\\_job\\_rx\\_info](#) [ezdp\\_job\\_desc::rx\\_info](#) [read]

Job receive info.

Applicable when job is received by the process or when sending or updating PMU queue.

struct [ezdp\\_job\\_tx\\_info](#) [ezdp\\_job\\_desc::tx\\_info](#) [read]

Job transmit info.

Applicable when job is sent to TM.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_discard\_cmd\_info Struct Reference

Job container discard request info.

### Data Fields

- union {
- [ezdp\\_job\\_discard\\_cmd\\_info\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED11\_15\_SIZE
- *Reserved bits 11 to 15.* unsigned [side](#): EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_SIZE
- *PMU side.* unsigned [pad1](#) : EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED0\_9\_SIZE
- *Reserved bits 0 to 9.* }
- };

### Detailed Description

Job container discard request info.

### Field Documentation

[ezdp\\_job\\_discard\\_cmd\\_info\\_t ezdp\\_job\\_discard\\_cmd\\_info::raw\\_data](#)

unsigned [ezdp\\_job\\_discard\\_cmd\\_info::pad0](#)

Reserved bits 11 to 15.

unsigned [ezdp\\_job\\_discard\\_cmd\\_info::side](#)

PMU side.

Define which side should handle the job done request (PMU implements the queuing and JD read operation)

unsigned [ezdp\\_job\\_discard\\_cmd\\_info::pad1](#)

Reserved bits 0 to 9.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_queue\_cmd\_info Struct Reference

Job container send to queue request info.

### Data Fields

- union {
- [ezdp\\_job\\_queue\\_cmd\\_info\\_t raw\\_data](#)
- struct {
- unsigned [\\_pad0](#) : EZDP\_JOB\_QUEUE\_CMD\_INFO\_RESERVED8\_15\_SIZE
- *Reserved bits 8 to 15.* unsigned [side](#): EZDP\_JOB\_QUEUE\_CMD\_INFO\_SIDE\_SIZE
- *PMU target queue side.* unsigned [target\\_queue](#):
- EZDP\_JOB\_QUEUE\_CMD\_INFO\_TARGET\_QUEUE\_SIZE
- *PMU target queue.* }
- };

---

### Detailed Description

Job container send to queue request info.

---

### Field Documentation

[ezdp\\_job\\_queue\\_cmd\\_info\\_t ezdp\\_job\\_queue\\_cmd\\_info::raw\\_data](#)

unsigned [ezdp\\_job\\_queue\\_cmd\\_info::\\_pad0](#)

Reserved bits 8 to 15.

unsigned [ezdp\\_job\\_queue\\_cmd\\_info::side](#)

PMU target queue side.

unsigned [ezdp\\_job\\_queue\\_cmd\\_info::target\\_queue](#)

PMU target queue.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_rx\_confirmation\_info Struct Reference

Info field for incoming job from TX confirmation ports.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_RESERVED8\_31\_SIZE
- *Reserved bits 8 to 31.*   uint8\_t [timestamp\\_sec](#)
- *The timestamp seconds counter is sampled at one with the nanoseconds counter, providing together the standard 5B timestamp.*   uint32\_t [timestamp\\_nsec](#)
- *The timestamp nanoseconds counter.*   }
- };

### Detailed Description

Info field for incoming job from TX confirmation ports.

### Field Documentation

uint32\_t [ezdp\\_job\\_rx\\_confirmation\\_info::raw\\_data](#)[EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_WORD\_COUNT]

unsigned [ezdp\\_job\\_rx\\_confirmation\\_info::pad0](#)

Reserved bits 8 to 31.

uint8\_t [ezdp\\_job\\_rx\\_confirmation\\_info::timestamp\\_sec](#)

The timestamp seconds counter is sampled at one with the nanoseconds counter, providing together the standard 5B timestamp.

Applicable only when frame descriptor time\_stamp flag is on.

uint32\_t [ezdp\\_job\\_rx\\_confirmation\\_info::timestamp\\_nsec](#)

The timestamp nanoseconds counter.

Together with timestamp\_sec provide the standard 5B timestamp. Applicable only when frame descriptor time\_stamp flag is on. The Tx MAC samples the timestamp at the beginning of transmission time and embeds the measurement in the response job, that can be interpreted by SW to calculate 1588 timestamp.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_rx\_info Struct Reference

Job receive info.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_RX\_INFO\_WORD\_COUNT]
- struct {
- union {
- struct [ezdp\\_job\\_rx\\_interface\\_info](#) [interface\\_info](#)
- *External interface information.*   struct [ezdp\\_job\\_rx\\_loopback\\_info](#) [loopback\\_info](#)
- *Loopback information.*   struct [ezdp\\_job\\_rx\\_confirmation\\_info](#) [confirmation\\_info](#)
- *Confirmation information.*   struct [ezdp\\_job\\_rx\\_timer\\_info](#) [timer\\_info](#)
- *Timer information.*   struct [ezdp\\_job\\_rx\\_user\\_info](#) [user\\_info](#)
- *User information.*   }
- unsigned [\\_\\_pad0\\_\\_](#): EZDP\_JOB\_RX\_INFO\_RESERVED112\_127\_SIZE
- *Reserved bits 112 to 127.*   uint16\_t [gross\\_checksum](#)
- *Checksum value starts at header\_offset and continues to end of frame, excluding bytes stripped by the MAC (e.g.   uint16\_t [seq\\_number](#)*
- *Sequence number of the job in a physical queue.*   unsigned [is\\_service\\_ready](#): EZDP\_JOB\_RX\_INFO\_IS\_SERVICE\_READY\_SIZE
- *This flag indicate that service, requested in dispatch command, is ready.*   unsigned [seq\\_number\\_valid](#): EZDP\_JOB\_RX\_INFO\_SEQ\_NUMBER\_VALID\_SIZE
- *Indicates that the seq\_number field is valid.*   unsigned [\\_\\_pad1\\_\\_](#): EZDP\_JOB\_RX\_INFO\_RESERVED108\_109\_SIZE
- *Reserved bits 108 to 109.*   unsigned [\\_\\_pad2\\_\\_](#): EZDP\_JOB\_RX\_INFO\_RESERVED104\_107\_SIZE
- *Reserved bits 104 to 107.*   unsigned [side](#): EZDP\_JOB\_RX\_INFO\_SIDE\_SIZE
- *The side of the PMU.*   unsigned [source\\_queue](#): EZDP\_JOB\_RX\_INFO\_SOURCE\_QUEUE\_SIZE
- *The Processor Manager Unit queue identification, initialized by Processor Manager Unit for incoming job according to configuration or by SW on the dispatch, in order to inform the next pipeline processing stage.*   }
- };

### Detailed Description

Job receive info.

### Field Documentation

uint32\_t [ezdp\\_job\\_rx\\_info::raw\\_data](#) [EZDP\_JOB\_RX\_INFO\_WORD\_COUNT]

struct [ezdp\\_job\\_rx\\_interface\\_info](#) [ezdp\\_job\\_rx\\_info::interface\\_info](#) [read]

External interface information.

struct [ezdp\\_job\\_rx\\_loopback\\_info](#) [ezdp\\_job\\_rx\\_info::loopback\\_info](#) [read]

Loopback information.

struct [ezdp\\_job\\_rx\\_confirmation\\_info](#) [ezdp\\_job\\_rx\\_info::confirmation\\_info](#) [read]

Confirmation information.

**struct [ezdp\\_job\\_rx\\_timer\\_info ezdp\\_job\\_rx\\_info::timer\\_info](#) [read]**

Timer information.

**struct [ezdp\\_job\\_rx\\_user\\_info ezdp\\_job\\_rx\\_info::user\\_info](#) [read]**

User information.

**unsigned [ezdp\\_job\\_rx\\_info::pad0](#)**

Reserved bits 112 to 127.

**uint16\_t [ezdp\\_job\\_rx\\_info::gross\\_checksum](#)**

Checksum value starts at header\_offset and continues to end of frame, excluding bytes stripped by the MAC (e.g.

four CRC bytes at the end of the frame). NOTE: This field is valid only when the checksum flag is set.

**uint16\_t [ezdp\\_job\\_rx\\_info::seq\\_number](#)**

Sequence number of the job in a physical queue.

Valid only when seq\_number\_valid flag is ON.

**unsigned [ezdp\\_job\\_rx\\_info::is\\_service\\_ready](#)**

This flag indicate that service, requested in dispatch command, is ready.

**unsigned [ezdp\\_job\\_rx\\_info::seq\\_number\\_valid](#)**

Indicates that the seq\_number field is valid.

Sequence number is set by Processor Manager Unit when frame is received from port, or when it updates sequence number based on dispatch request with sequence numbering service.

**unsigned [ezdp\\_job\\_rx\\_info::pad1](#)**

Reserved bits 108 to 109.

**unsigned [ezdp\\_job\\_rx\\_info::pad2](#)**

Reserved bits 104 to 107.

**unsigned [ezdp\\_job\\_rx\\_info::side](#)**

The side of the PMU.

**unsigned [ezdp\\_job\\_rx\\_info::source\\_queue](#)**

The Processor Manager Unit queue identification, initialized by Processor Manager Unit for incoming job according to configuration or by SW on the dispatch, in order to inform the next pipeline processing stage.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)



## ezdp\_job\_rx\_interface\_info Struct Reference

Info field for incoming job from external RX interfaces.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_RX\_INTERFACE\_INFO\_WORD\_COUNT]
- struct {
- unsigned [imem\\_buf\\_count](#): EZDP\_JOB\_RX\_INTERFACE\_INFO\_IMEM\_BUF\_COUNT\_SIZE
- < Indicate the PMU queue depth from which the job was sent.   unsigned [pad0](#) :  
EZDP\_JOB\_RX\_INTERFACE\_INFO\_RESERVED15\_SIZE
- Reserved bit 15.   unsigned [pad1](#) : EZDP\_JOB\_RX\_INTERFACE\_INFO\_RESERVED14\_SIZE
- Reserved bit 14.   unsigned [icu\\_succ\\_parsing\\_flag](#):  
EZDP\_JOB\_RX\_INTERFACE\_INFO\_ICU\_SUCC\_PARSING\_FLAG\_SIZE
- If this flag is cleared the L2-L3 frame header could not be parsed by ICU, and therefore  
global\_congestion\_level is invalid and frame CoS contains default value.   unsigned [truncation\\_flag](#):  
EZDP\_JOB\_RX\_INTERFACE\_INFO\_TRUNCATION\_FLAG\_SIZE
- Indicating the frame data was truncated on the Rx path and the data is illegal.   unsigned [pad2](#) :  
EZDP\_JOB\_RX\_INTERFACE\_INFO\_RESERVED11\_SIZE
- Reserved bit 11.   unsigned [pad3](#) : EZDP\_JOB\_RX\_INTERFACE\_INFO\_RESERVED10\_SIZE
- Reserved bit 10.   unsigned [crc\\_ok\\_flag](#): EZDP\_JOB\_RX\_INTERFACE\_INFO\_CRC\_OK\_FLAG\_SIZE
- CRC was checked by MAC and was found OK.   unsigned [crc\\_checked\\_flag](#):  
EZDP\_JOB\_RX\_INTERFACE\_INFO\_CRC\_CHECKED\_FLAG\_SIZE
- CRC checked by the MAC.   uint8\_t [timestamp\\_sec](#)
- The timestamp seconds counter is sampled at one with the nanoseconds counter, providing together the  
standard 5B timestamp.   uint32\_t [timestamp\\_nsec](#)
- The timestamp nanoseconds counter.   }
- };

### Detailed Description

Info field for incoming job from external RX interfaces.

### Field Documentation

uint32\_t

[ezdp\\_job\\_rx\\_interface\\_info::raw\\_data](#)[EZDP\_JOB\_RX\_INTERFACE\_INFO\_WORD\_COUNT]

unsigned [ezdp\\_job\\_rx\\_interface\\_info::imem\\_buf\\_count](#)

< Indicate the PMU queue depth from which the job was sent.

< Indicate the job budget congestion level reported by flow control at reception time. < Indicate the EMEM buffer budget congestion level reported by flow control at reception time. < Indicate the IMEM buffer budget congestion level reported by flow control at reception time. < Indicate the global congestion level of the system. Based on configuration and congestion level of the IMEM buffers budget, EMEM buffers budget, job budget. This is the number of leading frame buffers allocated by Rx NDMA from IMEM. The remaining buffers (if exist) are allocated in EMEM.

unsigned [ezdp\\_job\\_rx\\_interface\\_info:: pad0](#)

Reserved bit 15.

**unsigned [ezdp\\_job\\_rx\\_interface\\_info::pad1](#)**

Reserved bit 14.

**unsigned [ezdp\\_job\\_rx\\_interface\\_info::icu\\_succ\\_parsing\\_flag](#)**

If this flag is cleared the L2-L3 frame header could not be parsed by ICU, and therefore global\_congestion\_level is invalid and frame CoS contains default value.

**unsigned [ezdp\\_job\\_rx\\_interface\\_info::truncation\\_flag](#)**

Indicating the frame data was truncated on the Rx path and the data is illegal.

**unsigned [ezdp\\_job\\_rx\\_interface\\_info::pad2](#)**

Reserved bit 11.

**unsigned [ezdp\\_job\\_rx\\_interface\\_info::pad3](#)**

Reserved bit 10.

**unsigned [ezdp\\_job\\_rx\\_interface\\_info::crc\\_ok\\_flag](#)**

CRC was checked by MAC and was found OK.

Applicable only if checked by MAC flag is on.

**unsigned [ezdp\\_job\\_rx\\_interface\\_info::crc\\_checked\\_flag](#)**

CRC checked by the MAC.

**uint8\_t [ezdp\\_job\\_rx\\_interface\\_info::timestamp\\_sec](#)**

The timestamp seconds counter is sampled at one with the nanoseconds counter, providing together the standard 5B timestamp.

Applicable only when frame descriptor time\_stamp flag is on. The timestamp is sampled by the Rx MAC when the frame is speculated to be within the PTP length range (based on frame length) to minimize overheads.

**uint32\_t [ezdp\\_job\\_rx\\_interface\\_info::timestamp\\_nsec](#)**

The timestamp nanoseconds counter.

Together with `timestamp_sec` provide the standard 5B timestamp. Applicable only when frame descriptor `time_stamp` flag is on. The timestamp is sampled by the Rx MAC when the frame is speculated to be within the PTP length range (based on frame length) to minimize overheads.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- `dpe/dp/include/ezdp\_job\_defs.h`

## ezdp\_job\_rx\_loopback\_info Struct Reference

Info field for incoming job from loopback ports.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_RX\_LOOPBACK\_INFO\_WORD\_COUNT]
- struct {
- uint16\_t [replication\\_id](#)
- *The multicast replication id is a running number from 0 to the replication\_count (include replication\_count), provided by a SW unicast frame sent to the TM loopback port.*   unsigned [pad0](#) :
- EZDP\_JOB\_RX\_LOOPBACK\_INFO\_RESERVED0\_15\_SIZE
- Reserved bits 0 to 15.   unsigned [pad1](#) : EZDP\_JOB\_RX\_LOOPBACK\_INFO\_RESERVED32\_63\_SIZE
- Reserved bits 32 to 63.   }
- };

### Detailed Description

Info field for incoming job from loopback ports.

### Field Documentation

uint32\_t  
[ezdp\\_job\\_rx\\_loopback\\_info::raw\\_data](#)[EZDP\_JOB\_RX\_LOOPBACK\_INFO\_WORD\_COUNT]

uint16\_t [ezdp\\_job\\_rx\\_loopback\\_info::replication\\_id](#)

The multicast replication id is a running number from 0 to the replication\_count (include replication\_count), provided by a SW unicast frame sent to the TM loopback port.

unsigned [ezdp\\_job\\_rx\\_loopback\\_info::pad0](#)

Reserved bits 0 to 15.

unsigned [ezdp\\_job\\_rx\\_loopback\\_info::pad1](#)

Reserved bits 32 to 63.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_rx\_timer\_info Struct Reference

Info field for incoming timer job (PMU Timer).

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_RX\_TIMER\_INFO\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_JOB\_RX\_TIMER\_INFO\_RESERVED16\_31\_SIZE
- *Reserved 16 to 31.*   uint8\_t [timer\\_id](#)
- *Job originating unique timer ID.*   unsigned [pad1](#) :
- EZDP\_JOB\_RX\_TIMER\_INFO\_RESERVED0\_8\_SIZE
- *Reserved bits 0 to 7.*   uint32\_t [event\\_id](#)
- *The timer event number represents a sequentially advancing event in the interval generated by the timer.*   }
- };

---

### Detailed Description

Info field for incoming timer job (PMU Timer).

---

### Field Documentation

uint32\_t [ezdp\\_job\\_rx\\_timer\\_info::raw\\_data](#)[EZDP\_JOB\_RX\_TIMER\_INFO\_WORD\_COUNT]

unsigned [ezdp\\_job\\_rx\\_timer\\_info::pad0](#)

Reserved 16 to 31.

uint8\_t [ezdp\\_job\\_rx\\_timer\\_info::timer\\_id](#)

Job originating unique timer ID.

unsigned [ezdp\\_job\\_rx\\_timer\\_info::pad1](#)

Reserved bits 0 to 7.

uint32\_t [ezdp\\_job\\_rx\\_timer\\_info::event\\_id](#)

The timer event number represents a sequentially advancing event in the interval generated by the timer.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_rx\_user\_info Struct Reference

Info field for incoming frame job from generic user forwarding.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_RX\_USER\_INFO\_WORD\_COUNT]
- struct {
- uint32\_t [user\\_data\\_info0](#)
- User data info0.   uint32\_t [user\\_data\\_info1](#)
- User data info1.   }
- };

---

### Detailed Description

Info field for incoming frame job from generic user forwarding.

---

### Field Documentation

uint32\_t [ezdp\\_job\\_rx\\_user\\_info::raw\\_data](#)[EZDP\_JOB\_RX\_USER\_INFO\_WORD\_COUNT]

uint32\_t [ezdp\\_job\\_rx\\_user\\_info::user\\_data\\_info0](#)

User data info0.

uint32\_t [ezdp\\_job\\_rx\\_user\\_info::user\\_data\\_info1](#)

User data info1.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_transmit\_cmd\_info Struct Reference

Job container send out request info.

### Data Fields

- union {
- [ezdp\\_job\\_transmit\\_cmd\\_info\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_RESERVED12\_15\_SIZE
- Reserved bits 12 to 15.   unsigned [side](#): EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_SIZE
- < Define the send mode   unsigned [output\\_channel](#):
- EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_OUTPUT\_CHANNEL\_SIZE
- Output channel id.    }
- };

### Detailed Description

Job container send out request info.

### Field Documentation

[ezdp\\_job\\_transmit\\_cmd\\_info\\_t ezdp\\_job\\_transmit\\_cmd\\_info::raw\\_data](#)

unsigned [ezdp\\_job\\_transmit\\_cmd\\_info:: pad0](#)

Reserved bits 12 to 15.

unsigned [ezdp\\_job\\_transmit\\_cmd\\_info::side](#)

< Define the send mode

The TM/PMU side. Define which side should handle the job done request (PMU implements the queuing and JD read operation)

unsigned [ezdp\\_job\\_transmit\\_cmd\\_info::output\\_channel](#)

Output channel id.

Define direct mapping to destination port. Also used to select (by configuration) to one of the 4 PMU TM bypass queues that will hold the packet. Applicable only for TM bypass mode

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_job\_tx\_info Struct Reference

Info field for transmitting frame job (TM mode is full or tm qos bypass).

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_JOB\_TX\_INFO\_WORD\_COUNT]
- struct {
- unsigned [wred\\_color](#): EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_SIZE
- *Policer WRED Color field.*    unsigned [pad0](#) : EZDP\_JOB\_TX\_INFO\_RESERVED25\_29\_SIZE
- *Reserved bits 25 to 29.*    unsigned [packet\\_switch\\_id\\_select](#): EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_ID\_SELECT\_SIZE
- *Selects an entry in packet switch table.*    union {
- uint16\_t [replication\\_count](#)
- *Defines the number of repeated job descriptors generated by the PMU replication logic.*    uint16\_t [user\\_info](#)
- *User information that gets echoed back to the job created by the loopback port.*    uint16\_t [explicit\\_packet\\_switch\\_id](#)
- *Select an entry from the PSID table.*    uint16\_t [dest\\_queue](#)
- *Define which PMU queue should be selected for the frame arriving from a loopback port.*    }
- unsigned [pad1](#) : EZDP\_JOB\_TX\_INFO\_RESERVED61\_SIZE
- *< Controls the outgoing packet drop policy.*    unsigned [qos\\_bypass](#): EZDP\_JOB\_TX\_INFO\_QOS\_BYPASS\_SIZE
- *TM QOS bypass.*    unsigned [pad2](#) : EZDP\_JOB\_TX\_INFO\_RESERVED59\_SIZE
- *Reserved bit 59.*    unsigned [pad3](#) : EZDP\_JOB\_TX\_INFO\_RESERVED52\_55\_SIZE
- *< Selects traffic manager output queue mapping mode.*    unsigned [side](#): EZDP\_JOB\_TX\_INFO\_SIDE\_SIZE
- *Side to forward the frame to.*    unsigned [flow\\_id](#): EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_SIZE
- *Flow ID field.*    unsigned [stat\\_code\\_profile1](#): EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE1\_SIZE
- *Statistics Code Profile 1 field.*    unsigned [stat\\_code\\_profile2](#): EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE2\_SIZE
- *Statistics Code Profile 2 field.*    unsigned [pad4](#) : EZDP\_JOB\_TX\_INFO\_RESERVED88\_90\_SIZE
- *Reserved bits 88 to 90.*    unsigned [stat\\_stream\\_id](#): EZDP\_JOB\_TX\_INFO\_STAT\_STREAM\_ID\_SIZE
- *Statistic stream id field.*    unsigned [wred\\_flow\\_template\\_profile](#): EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_SIZE
- *Policer WRED flow template profile id field.*    unsigned [wred\\_class\\_template\\_profile](#): EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_TEMPLATE\_PROFILE\_SIZE
- *Policer WRED class template profile id field.*    uint8\_t [wred\\_flow\\_scale\\_profile](#)
- *Policer WRED flow absolute scaling profile id field.*    uint8\_t [wred\\_class\\_scale\\_profile](#)
- *Policer WRED class absolute scaling profile id field.*    unsigned [pad5](#) : EZDP\_JOB\_TX\_INFO\_RESERVED102\_103\_SIZE
- *Reserved bits 102 to 103.*    unsigned [inter\\_packet\\_gap\\_control](#): EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CONTROL\_SIZE
- *Select the inter-packet gap emulation mode.*    unsigned [inter\\_packet\\_gap](#): EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_SIZE
- *The Inter Packet Gap emulation field provides per frame overhead for TM IPG emulation logic.*    }
- };

### Detailed Description

Info field for transmitting frame job (TM mode is full or tm qos bypass).



## Field Documentation

uint32\_t [ezdp\\_job\\_tx\\_info::raw\\_data](#)[EZDP\_JOB\_TX\_INFO\_WORD\_COUNT]

unsigned [ezdp\\_job\\_tx\\_info::wred\\_color](#)

Policer WRED Color field.

Defines the color of the packet, which is used at TM enqueue point for implementation of the WRED algorithm.

unsigned [ezdp\\_job\\_tx\\_info::pad0](#)

Reserved bits 25 to 29.

unsigned [ezdp\\_job\\_tx\\_info::packet\\_switch\\_id\\_select](#)

Selects an entry in packet switch table.

Used together with 9 bits from L1 entry to get the entry id in packet switch table.

uint16\_t [ezdp\\_job\\_tx\\_info::replication\\_count](#)

Defines the number of repeated job descriptors generated by the PMU replication logic.

Applicable only when destination port is loopback and multicast\_mode is REPLICATION. The replication logic counts from zero to replication\_count (meaning that replication\_count determines the extra jobs created in addition to the first loopback job). Single job with no further repetition is a legal scenario (identified by replication\_num=0 and multicast\_mode=REPLICATION).

uint16\_t [ezdp\\_job\\_tx\\_info::user\\_info](#)

User information that gets echoed back to the job created by the loopback port.

Applicable only when destination port is loopback and no multicast\_mode = REPLICATION port is not configured to override PMU queue selection. Applicable only when destination port is loopback.

uint16\_t [ezdp\\_job\\_tx\\_info::explicit\\_packet\\_switch\\_id](#)

Select an entry from the PSID table.

uint16\_t [ezdp\\_job\\_tx\\_info::dest\\_queue](#)

Define which PMU queue should be selected for the frame arriving from a loopback port.

Applicable only when destination port is loopback and configured to enable override PMU queue section from SW. NOTE: The job is inserted to the same PMU side to which loopback is block to.

unsigned [ezdp\\_job\\_tx\\_info::pad1](#)

< Controls the outgoing packet drop policy.

Reserved bit 61

**unsigned [ezdp\\_job\\_tx\\_info::qos\\_bypass](#)**

TM QOS bypass.

When flag is on, the packet runs through TM, bypassing the per flow queue, directly to the packet switching table selection. Only EXPLICIT and BASE packet switch modes are available when flag is ON. TM control fields flow\_id, WRED controls and WRED statistics are not applicable.

**unsigned [ezdp\\_job\\_tx\\_info::pad2](#)**

Reserved bit 59.

**unsigned [ezdp\\_job\\_tx\\_info::pad3](#)**

< Selects traffic manager output queue mapping mode.

The TM port switching table selects an output channel, and the output channel further selects the side of a Tx port it is assigned to. Reserved bits 52 to 55.

**unsigned [ezdp\\_job\\_tx\\_info::side](#)**

Side to forward the frame to.

**unsigned [ezdp\\_job\\_tx\\_info::flow\\_id](#)**

Flow ID field.

The flow ID uniquely selects an L4 entity in the target TM, and TM configuration topology further maps it through TM scheduling levels down to the target port.

**unsigned [ezdp\\_job\\_tx\\_info::stat\\_code\\_profile1](#)**

Statistics Code Profile 1 field.

Used in TM WRED statistics reporting, participates in generation of a 5-bit policer drop code in either topology based statistics or stream id based statistics.

**unsigned [ezdp\\_job\\_tx\\_info::stat\\_code\\_profile2](#)**

Statistics Code Profile 2 field.

Participates together with Statistics Code Profile 1 in generation of a 5-bit policer drop code in either topology based statistics or stream ID based statistics. It provides 0-3 bits replacing bits from the aggregated policer result when mapping stream ID based reporting. Additionally it participates in mapping the topology based reporting together with profile 1 code and pass/drop decision.

**unsigned [ezdp\\_job\\_tx\\_info::pad4](#)**

Reserved bits 88 to 90.

**unsigned [ezdp\\_job\\_tx\\_info::stat\\_stream\\_id](#)**

Statistic stream id field.

The stat\_stream\_id field select the block to be reported, where the reporting code selects the counter in the block. It points to the base location where a continuous block with statistic counters associated with the packet can be updated. On WRED reporting, together with policer codes, statistic code profile 1 and statistic code profile 2 stream based reporting code is generated, mapping one counter in a block of up to 32 counters for the reported stream.

**unsigned [ezdp\\_job\\_tx\\_info::wred\\_flow\\_template\\_profile](#)**

Policer WRED flow template profile id field.

This field selects one of sixteen RED behavior templates to be used by the WRED algorithm at the flow level (L4). Each template holds eight profiles (one per color), totaling 128 templates. A single entry in the template provides per priority the relative percentage of different ranges of the WRED graph for that priority.

**unsigned [ezdp\\_job\\_tx\\_info::wred\\_class\\_template\\_profile](#)**

Policer WRED class template profile id field.

This field selects one of sixteen RED behavior templates to be used by the WRED algorithm at the class level (L3). Each template holds eight profiles (one per color), totaling 128 templates. A single entry in the template provides per priority the relative percentage of different ranges of the WRED graph for that priority.

**uint8\_t [ezdp\\_job\\_tx\\_info::wred\\_flow\\_scale\\_profile](#)**

Policer WRED flow absolute scaling profile id field.

The policer\_flow\_scale\_index (PFAI) selects an absolute scaling factor index from a 256-entry table, used in combination with the selected policer\_flow\_template\_index to define WRED algorithm behavior at the flow level (L4). Normalized regions provided per priority by the template are scaled to describe the final WRED graphs.

**uint8\_t [ezdp\\_job\\_tx\\_info::wred\\_class\\_scale\\_profile](#)**

Policer WRED class absolute scaling profile id field.

This field selects an absolute scaling factor index from a 256-entry table, used in combination with the selected policer\_class\_template\_index to define WRED algorithm behavior at the class level (L3). Normalized regions provided per priority by the template are scaled to describe the final WRED graphs.

**unsigned [ezdp\\_job\\_tx\\_info::pad5](#)**

Reserved bits 102 to 103.

**unsigned [ezdp\\_job\\_tx\\_info::inter\\_packet\\_gap\\_control](#)**

Select the inter-packet gap emulation mode.

**unsigned [ezdp\\_job\\_tx\\_info::inter\\_packet\\_gap](#)**

The Inter Packet Gap emulation field provides per frame overhead for TM IPG emulation logic.

The `inter_packet_gap_enum` field comprises 5-bit mantissa (-16 to 15) and one bit for exponent selection mode. (TM configuration has global exponent configuration to scale the mantissa by x1, x4, ,16 or x128.) The covered range is: (-16..15) x (1/4/16/128) bytes. It affects packet length taking into account parts that are added or excluded from shaping and fairness algorithms as well as from statistics reporting. The TM can independently use `inter_packet_gap_enum` for altering frame length on shaper accounting and for statistics reporting. Additionally `inter_packet_gap_enum` can affect TM WFQ scheduling at each level independently. Besides emulating physical media characteristics (such as Ethernet inter-packet gap bytes), `inter_packet_gap_enum` may account for parts of a packet that are either ignored or added in TM algorithms (for example, it may consider only layer 3 IP traffic and use `inter_packet_gap_enum` to exclude layer 2 header as well as four CRC bytes from the packet length for shaping and scheduling). `inter_packet_gap_enum` can also be used to account for proprietary header bytes.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- `dpe/dp/include/ezdp\_job\_defs.h`

## ezdp\_large\_linked\_buffers\_desc Struct Reference

Large linked buffers descriptor.

### Data Fields

- struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) [line](#) [EZDP\_LARGE\_LBD]  
*Array of 2 linked buffers lines, which contains up to 6 buffers.*
- 

### Detailed Description

Large linked buffers descriptor.

May hold up to 6 buffs

---

### Field Documentation

struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) [ezdp\\_large\\_linked\\_buffers\\_desc::line](#)[EZDP\_LARGE\_LBD]  
[read]

Array of 2 linked buffers lines, which contains up to 6 buffers.

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_linked\_buffers\_desc Struct Reference

A generic linked buffers descriptor.

### Data Fields

- struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) [line](#) [0]

*Linked buffers lines.*

---

### Detailed Description

A generic linked buffers descriptor.

Good for pointers

---

### Field Documentation

struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) [ezdp\\_linked\\_buffers\\_desc::line](#)[0] [read]

Linked buffers lines.

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_linked\_buffers\_desc\_line Struct Reference

LBD Line data structure.

### Data Fields

- uint8\_t [ecc](#)
- ECC. struct [ezdp\\_buffer\\_info](#) [buf\\_info](#)  
[EZDP\_LINKED\_BUFFER\_DESC\_LINE\_NUMBER\_OF\_BUFFERS\_DESC]
- Array of 3 buffers info. struct [ezdp\\_buffer\\_desc](#) [buf\\_desc](#)  
[EZDP\_LINKED\_BUFFER\_DESC\_LINE\_NUMBER\_OF\_BUFFERS\_DESC]

Array of 3 buffers.

---

### Detailed Description

LBD Line data structure.

---

### Field Documentation

uint8\_t [ezdp\\_linked\\_buffers\\_desc\\_line::ecc](#)

ECC.

struct [ezdp\\_buffer\\_info](#)  
[ezdp\\_linked\\_buffers\\_desc\\_line::buf\\_info](#)[EZDP\_LINKED\_BUFFER\_DESC\_LINE\_NUMBER\_OF\_BUFFERS\_DESC] [read]

Array of 3 buffers info.

struct [ezdp\\_buffer\\_desc](#)  
[ezdp\\_linked\\_buffers\\_desc\\_line::buf\\_desc](#)[EZDP\_LINKED\_BUFFER\_DESC\_LINE\_NUMBER\_OF\_BUFFERS\_DESC] [read]

Array of 3 buffers.

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_list\_cfg Struct Reference

list queue configuration data structure

### Data Fields

- [ezdp\\_sum\\_addr\\_t head](#)
  - Head address of the list. [ezdp\\_sum\\_addr\\_t tail](#)
  - Tail address of the list. [ezdp\\_mem\\_pool\\_t queue\\_memory\\_pool](#)
- Memory pool for the queue elements.
- 

### Detailed Description

list queue configuration data structure

---

### Field Documentation

#### [ezdp\\_sum\\_addr\\_t ezdp\\_list\\_cfg::head](#)

Head address of the list.

#### [ezdp\\_sum\\_addr\\_t ezdp\\_list\\_cfg::tail](#)

Tail address of the list.

#### [ezdp\\_mem\\_pool\\_t ezdp\\_list\\_cfg::queue\\_memory\\_pool](#)

Memory pool for the queue elements.

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_queue\\_defs.h](#)



## ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element Struct Reference

Lookup external tcam 16 Byte associated data only result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]
- struct {
- unsigned [valid](#): EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.*   unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.*   unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.*   unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.*   unsigned  
   [pad0](#) : EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE
- *Reserved bit 24.*   uint8\_t [assoc\\_data](#)  
   [EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- *120 bit Associated data*   }
- };

### Detailed Description

Lookup external tcam 16 Byte associated data only result.

### Field Documentation

uint32\_t  
[ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element::raw\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element::valid](#)

Result valid.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element::match](#)

Match.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element::lookup\\_error](#)

< Result element type.

Must be set by user. Recommend to set to EZDP\_USER\_DEFINED\_ASSOC\_DATA1 or EZDP\_USER\_DEFINED\_ASSOC\_DATA2 or EZDP\_USER\_DEFINED\_ASSOC\_DATA3 to match other response types definition Lookup error. Will be on if any error occurred. Applicable for the first result element only.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element::truncated](#)

Truncated response indication - result returned was truncated due to being larger than result\_len.

Applicable for the first result element only

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element::pad0](#)

Reserved bit 24.

uint8\_t

[ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element::assoc\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

120 bit Associated data

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element Struct Reference

Lookup external tcam 32 Byte associated data only result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]
- struct {
- unsigned [valid](#): EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.*   unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.*   unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.*   unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.*   unsigned  
   [pad0](#) : EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE
- *Reserved bit 24.*   uint8\_t [assoc\\_data](#)  
   [EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- 248 bit Associated data   }
- };

### Detailed Description

Lookup external tcam 32 Byte associated data only result.

### Field Documentation

uint32\_t  
[ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element::raw\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element::valid](#)

Result valid.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element::match](#)

Match.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element::lookup\\_error](#)

< Result element type.

Must be set by user. Recommend to set to EZDP\_USER\_DEFINED\_ASSOC\_DATA1 or EZDP\_USER\_DEFINED\_ASSOC\_DATA2 or EZDP\_USER\_DEFINED\_ASSOC\_DATA3 to match other response types definition Lookup error. Will be on if any error occurred. Applicable for the first result element only.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element::truncated](#)

Truncated response indication - result returned was truncated due to being larger than result\_len.  
Applicable for the first result element only

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element::pad0](#)

Reserved bit 24.

uint8\_t

[ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element::assoc\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

248 bit Associated data

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element Struct Reference

Lookup external tcam 4 Byte associated data only result.

### Data Fields

- union {
- [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element\\_t raw\\_data](#)
- struct {
- unsigned [valid](#): EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.* unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.* unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.* unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.* unsigned  
[pad0](#): EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE
- *Reserved bit 24.* uint8\_t [assoc\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- *24 bit Associated data* }
- };

### Detailed Description

Lookup external tcam 4 Byte associated data only result.

### Field Documentation

[ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element\\_t](#)  
[ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element::raw\\_data](#)

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element::valid](#)

Result valid.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element::match](#)

Match.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element::lookup\\_error](#)

< Result element type.

Must be set by user. Recommend to set to EZDP\_USER\_DEFINED\_ASSOC\_DATA1 or EZDP\_USER\_DEFINED\_ASSOC\_DATA2 or EZDP\_USER\_DEFINED\_ASSOC\_DATA3 to match other response types definition Lookup error. Will be on if any error occurred. Applicable for the first result element only.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element::truncated](#)

Truncated response indication - result returned was truncated due to being larger than result\_len.

Applicable for the first result element only

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element::pad0](#)

Reserved bit 24.

uint8\_t

[ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element::assoc\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

24 bit Associated data

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element Struct Reference

Lookup external tcam 8 Byte associated data only result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]
- struct {
- unsigned [valid](#): EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.*   unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.*   unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.*   unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.*   unsigned  
   [pad0](#) : EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE
- *Reserved bit 24.*   uint8\_t [assoc\\_data](#)  
   [EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- 56 bit Associated data   }
- };

### Detailed Description

Lookup external tcam 8 Byte associated data only result.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element::raw\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element::valid](#)

Result valid.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element::match](#)

Match.

Must be set by user. Recommend to set to 1 to match ezdp definition

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element::lookup\\_error](#)

< Result element type.

Must be set by user. Recommend to set to EZDP\_USER\_DEFINED\_ASSOC\_DATA1 or EZDP\_USER\_DEFINED\_ASSOC\_DATA2 or EZDP\_USER\_DEFINED\_ASSOC\_DATA3 to match other response types definition Lookup error. Will be on if any error occurred. Applicable for the first result element only.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element::truncated](#)

Truncated response indication - result returned was truncated due to being larger than result\_len.  
Applicable for the first result element only

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element::pad0](#)

Reserved bit 24.

uint8\_t

[ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element::assoc\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

56 bit Associated data

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)



## ezdp\_lookup\_ext\_tcaml\_index\_16B\_data\_result\_element Struct Reference

Lookup external tcaml index result with 16 Byte associated data.

### Data Fields

- union {
- uint32\_t [raw\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]
- struct {
- unsigned [valid](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.*   unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.*   unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.*   unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.*   unsigned [pad0](#) :  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE
- *Reserved bits 23 to 24.*   unsigned [device\\_id](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE
- *Responding device id.*   unsigned [index](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE
- *compare result index*   uint8\_t [assoc\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- 16B Associated data   }
- };

### Detailed Description

Lookup external tcaml index result with 16 Byte associated data.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::raw\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::valid](#)

Result valid.

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::match](#)

Match.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::lookup\\_error](#)**

< Result element type.

Lookup error. Will be on if any error occurred. For the exact error look in the error flags returned in `ezdp_lookup_ext_tcaml_retval_t`. Applicable for the first result element only.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::truncated](#)**

Truncated response indication - result returned was truncated due to being larger than `result_len`.

Applicable for the first result element only

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::pad0](#)**

Reserved bits 23 to 24.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::device\\_id](#)**

Responding device id.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::index](#)**

compare result index

**uint8\_t**

**[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_16B\\_data\\_result\\_element::assoc\\_data](#)**[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

16B Associated data

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- `dpe/dp/include/ezdp\_search\_defs.h`

## ezdp\_lookup\_ext\_tcaml\_index\_32B\_data\_result\_element Struct Reference

Lookup external tcaml index result with 32 Byte associated data.

### Data Fields

- union {
- uint32\_t [raw\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]
- struct {
- unsigned [valid](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.*   unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.*   unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.*   unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.*   unsigned [pad0](#) :  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE
- *Reserved bits 23 to 24.*   unsigned [device\\_id](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE
- *Responding device id.*   unsigned [index](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE
- *compare result index*   uint8\_t [assoc\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- 32B Associated data   }
- };

### Detailed Description

Lookup external tcaml index result with 32 Byte associated data.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::raw\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::valid](#)

Result valid.

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::match](#)

Match.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::lookup\\_error](#)**

< Result element type.

Lookup error. Will be on if any error occurred. For the exact error look in the error flags returned in `ezdp_lookup_ext_tcaml_retval_t`. Applicable for the first result element only.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::truncated](#)**

Truncated response indication - result returned was truncated due to being larger than `result_len`.

Applicable for the first result element only

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::pad0](#)**

Reserved bits 23 to 24.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::device\\_id](#)**

Responding device id.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::index](#)**

compare result index

**uint8\_t**

**[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_32B\\_data\\_result\\_element::assoc\\_data](#)**[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

32B Associated data

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- `dpe/dp/include/ezdp\_search\_defs.h`

## ezdp\_lookup\_ext\_tcaml\_index\_4B\_data\_result\_element Struct Reference

Lookup external tcaml index result with 4 Byte associated data.

### Data Fields

- union {
- uint32\_t [raw\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]
- struct {
- unsigned [valid](#): EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.*   unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.*   unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.*   unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.*   unsigned [pad0](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE
- *Reserved bits 23 to 24.*   unsigned [device\\_id](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE
- *Responding device id.*   unsigned [index](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE
- *compare result index*   uint8\_t [assoc\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- 4B Associated data   }
- };

### Detailed Description

Lookup external tcaml index result with 4 Byte associated data.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::raw\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::valid](#)

Result valid.

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::match](#)

Match.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::lookup\\_error](#)**

< Result element type.

Lookup error. Will be on if any error occurred. For the exact error look in the error flags returned in `ezdp_lookup_ext_tcaml_retval_t`. Applicable for the first result element only.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::truncated](#)**

Truncated response indication - result returned was truncated due to being larger than `result_len`.

Applicable for the first result element only

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::pad0](#)**

Reserved bits 23 to 24.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::device\\_id](#)**

Responding device id.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::index](#)**

compare result index

**uint8\_t**

**[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_4B\\_data\\_result\\_element::assoc\\_data](#)**[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

4B Associated data

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- `dpe/dp/include/ezdp\_search\_defs.h`

## ezdp\_lookup\_ext\_tcaml\_index\_8B\_data\_result\_element Struct Reference

Lookup external tcaml index result with 8 Byte associated data.

### Data Fields

- union {
- uint32\_t [raw\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]
- struct {
- unsigned [valid](#): EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.*   unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.*   unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- < *Result element type.*   unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.*   unsigned [pad0](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE
- *Reserved bits 23 to 24.*   unsigned [device\\_id](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE
- *Responding device id.*   unsigned [index](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE
- *compare result index*   uint8\_t [assoc\\_data](#)  
[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]
- 8B Associated data   }
- };

### Detailed Description

Lookup external tcaml index result with 8 Byte associated data.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::raw\\_data](#)[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::valid](#)

Result valid.

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::match](#)

Match.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::lookup\\_error](#)**

< Result element type.

Lookup error. Will be on if any error occurred. For the exact error look in the error flags returned in `ezdp_lookup_ext_tcaml_retval_t`. Applicable for the first result element only.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::truncated](#)**

Truncated response indication - result returned was truncated due to being larger than `result_len`.

Applicable for the first result element only

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::pad0](#)**

Reserved bits 23 to 24.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::device\\_id](#)**

Responding device id.

**unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::index](#)**

compare result index

**uint8\_t**

**[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_8B\\_data\\_result\\_element::assoc\\_data](#)**[EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT]

8B Associated data

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- `dpe/dp/include/ezdp\_search\_defs.h`



## ezdp\_lookup\_ext\_tcaml\_index\_result\_element Struct Reference

Lookup external tcaml index result element.

### Data Fields

- union {
- [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_result\\_element\\_t raw\\_data](#)
- struct {
- unsigned [valid](#): EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_VALID\_SIZE
- *Result valid.* unsigned [match](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_MATCH\_SIZE
- *Match.* unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE
- *< Result element type.* unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.* unsigned  
[any\\_match](#): EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_SIZE
- *Any match - there is at least one valid match.* unsigned [pad0](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_RESERVED23\_SIZE
- *Reserved bit 23.* unsigned [device\\_id](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE
- *Responding device id.* unsigned [index](#):  
EZDP\_LOOKUP\_EXT\_TCAML\_INDEX\_RESULT\_ELEMENT\_INDEX\_SIZE
- *compare result index* }
- };

### Detailed Description

Lookup external tcaml index result element.

### Field Documentation

[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_result\\_element\\_t](#)  
[ezdp\\_lookup\\_ext\\_tcaml\\_index\\_result\\_element::raw\\_data](#)

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_result\\_element::valid](#)

Result valid.

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_result\\_element::match](#)

Match.

unsigned [ezdp\\_lookup\\_ext\\_tcaml\\_index\\_result\\_element::lookup\\_error](#)

< Result element type.

Lookup error. Will be on if any error occurred. For the exact error look in the error flags returned in `ezdp_lookup_ext_tcam_retval_t`. Applicable for the first result element only.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element::truncated](#)

Truncated response indication - result returned was truncated due to being larger than `result_len`.  
Applicable for the first result element only

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element::any\\_match](#)

Any match - there is at least one valid match.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element::pad0](#)

Reserved bit 23.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element::device\\_id](#)

Responding device id.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element::index](#)

compare result index

union { ... }

---

The documentation for this struct was generated from the following file:

- `dpe/dp/include/ezdp_search_defs.h`

## ezdp\_lookup\_ext\_tcam\_retval Struct Reference

Lookup external tcam return value.

### Data Fields

- union {
- [ezdp\\_lookup\\_ext\\_tcam\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_RESERVED\_BIT8\_31\_SIZE
- *Reserved bits 8 to 31.* unsigned [lookup\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_ERROR\_SIZE
- *Aggregated lookup error flag.* unsigned [truncated](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNCATED\_SIZE
- *Truncated response indication - result returned was truncated due to being larger than result\_len.* unsigned [multi\\_match](#): EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_MATCH\_SIZE
- *Multi match indication - more than one result element is a match.* unsigned [any\\_match](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MATCH\_SIZE
- *Any match indication - at least one result element is a match.* unsigned [time\\_out\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_OUT\_ERROR\_SIZE
- *Time out error - no response received within configured time frame.* unsigned [device\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_ERROR\_SIZE
- *External TCAM device error - errors returned by the NL12K unit.* unsigned [mac\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ERROR\_SIZE
- *Mac error - error identified by interlaken mac.* unsigned [no\\_context\\_match\\_error](#):  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONTEXT\_MATCH\_ERROR\_SIZE
- *No context match error - returned result's context does not match request context.* }
- };

### Detailed Description

Lookup external tcam return value.

### Field Documentation

[ezdp\\_lookup\\_ext\\_tcam\\_retval\\_t ezdp\\_lookup\\_ext\\_tcam\\_retval::raw\\_data](#)

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::pad0](#)

Reserved bits 8 to 31.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::lookup\\_error](#)

Aggregated lookup error flag.

Will be on if any error occurred out of the following: time\_out\_error, device\_error, mac\_error or no\_context\_match\_error.

unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::truncated](#)

Truncated response indication - result returned was truncated due to being larger than result\_len.

**unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::multi\\_match](#)**

Multi match indication - more than one result element is a match.

Applicable to index mode only.

**unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::any\\_match](#)**

Any match indication - at least one result element is a match.

Applicable to index mode only.

**unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::time\\_out\\_error](#)**

Time out error - no response received within configured time frame.

**unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::device\\_error](#)**

External TCAM device error - errors returned by the NL12K unit.

**unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::mac\\_error](#)**

Mac error - error identified by interlaken mac.

**unsigned [ezdp\\_lookup\\_ext\\_tcam\\_retval::no\\_context\\_match\\_error](#)**

No context match error - returned result's context does not match request context.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_int\_tcam\_12B\_data\_result Struct Reference

Lookup internal tcam 12 byte associated data result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [match](#): EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT\_MATCH\_SIZE
- Match indication.     unsigned [data0](#): EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT\_DATA0\_SIZE
- 31 msb of user defined associated data     uint32\_t [data1](#)
- Bytes 4 to 7 of user defined associated data.     uint32\_t [data2](#)
- Bytes 8 to 11 of user defined associated data.     }
- };

### Detailed Description

Lookup internal tcam 12 byte associated data result.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result::raw\\_data](#)[EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result::match](#)

Match indication.

unsigned [ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result::data0](#)

31 msb of user defined associated data

uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result::data1](#)

Bytes 4 to 7 of user defined associated data.

uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result::data2](#)

Bytes 8 to 11 of user defined associated data.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_int\_tcam\_16B\_data\_result Struct Reference

Lookup internal tcam 16 byte associated data result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [match](#): EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT\_MATCH\_SIZE
- Match indication.    unsigned [data0](#): EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT\_DATA0\_SIZE
- 31 msb of user defined associated data    uint32\_t [data1](#)
- Bytes 4 to 7 of user defined associated data.    uint32\_t [data2](#)
- Bytes 8 to 11 of user defined associated data.    uint32\_t [data3](#)
- Bytes 12 to 15 of user defined associated data.    }
- };

### Detailed Description

Lookup internal tcam 16 byte associated data result.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result::raw\\_data](#)[EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result::match](#)

Match indication.

unsigned [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result::data0](#)

31 msb of user defined associated data

uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result::data1](#)

Bytes 4 to 7 of user defined associated data.

uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result::data2](#)

Bytes 8 to 11 of user defined associated data.

uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result::data3](#)

Bytes 12 to 15 of user defined associated data.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_int\_tcam\_4B\_data\_result Struct Reference

Lookup internal tcam 4 byte associated data result.

### Data Fields

- union {
- [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result\\_t raw\\_data](#)
- struct {
- unsigned [match](#): EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESULT\_MATCH\_SIZE
- Match indication. unsigned [data](#): EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESULT\_DATA\_SIZE
- 31 bits of user defined associated data }
- };

---

### Detailed Description

Lookup internal tcam 4 byte associated data result.

---

### Field Documentation

[ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result\\_t ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result::raw\\_data](#)

unsigned [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result::match](#)

Match indication.

unsigned [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result::data](#)

31 bits of user defined associated data

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_int\_tcam\_8B\_data\_result Struct Reference

Lookup internal tcam 8 byte associated data result.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [match](#): EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_MATCH\_SIZE
- Match indication.     unsigned [data0](#): EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_DATA0\_SIZE
- 31 msb of user defined associated data     uint32\_t [data1](#)
- 32 lsb of user defined associated data     }
- };

### Detailed Description

Lookup internal tcam 8 byte associated data result.

### Field Documentation

uint32\_t

[ezdp\\_lookup\\_int\\_tcam\\_8B\\_data\\_result::raw\\_data](#)[EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_lookup\\_int\\_tcam\\_8B\\_data\\_result::match](#)

Match indication.

unsigned [ezdp\\_lookup\\_int\\_tcam\\_8B\\_data\\_result::data0](#)

31 msb of user defined associated data

uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_8B\\_data\\_result::data1](#)

32 lsb of user defined associated data

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)



## ezdp\_lookup\_int\_tcam\_result Struct Reference

Lookup ITCAM result definition.

### Data Fields

- union {
- struct [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result\\_standard](#)  
[EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_MAX\_NUM]
- *Result for standard internal TCAM lookup.* struct [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result](#) [assoc\\_4B\\_data](#)
- *4 byte associated user data.* struct [ezdp\\_lookup\\_int\\_tcam\\_8B\\_data\\_result](#) [assoc\\_8B\\_data](#)
- *8 byte associated user data.* struct [ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result](#) [assoc\\_12B\\_data](#)
- *12 byte associated user data.* struct [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result](#) [assoc\\_16B\\_data](#)
- *16 byte associated user data.* };

### Detailed Description

Lookup ITCAM result definition.

### Field Documentation

struct [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result](#)  
[ezdp\\_lookup\\_int\\_tcam\\_result::standard](#)[EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_MAX\_NUM] [read]

Result for standard internal TCAM lookup. Up to 4 results.

struct [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result](#) [ezdp\\_lookup\\_int\\_tcam\\_result::assoc\\_4B\\_data](#) [read]

4 byte associated user data. Overrides first TCAM result.

struct [ezdp\\_lookup\\_int\\_tcam\\_8B\\_data\\_result](#) [ezdp\\_lookup\\_int\\_tcam\\_result::assoc\\_8B\\_data](#) [read]

8 byte associated user data. Overrides first 2 TCAM results.

struct [ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result](#) [ezdp\\_lookup\\_int\\_tcam\\_result::assoc\\_12B\\_data](#) [read]

12 byte associated user data.

Overrides first 3 TCAM results.

struct [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result](#) [ezdp\\_lookup\\_int\\_tcam\\_result::assoc\\_16B\\_data](#) [read]

16 byte associated user data. Overrides all TCAM results.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_int\_tcam\_retval Struct Reference

Lookup ITCAM retval definition.

### Data Fields

- union {
- [ezdp\\_lookup\\_int\\_tcam\\_retval\\_t raw\\_data](#)
- *Retval 32 bit raw data.* struct [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result standard](#)
- *Result for standard internal TCAM lookup.* struct [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result assoc\\_data](#)
- *4 byte associated user data.* };

---

### Detailed Description

Lookup ITCAM retval definition.

---

### Field Documentation

[ezdp\\_lookup\\_int\\_tcam\\_retval\\_t ezdp\\_lookup\\_int\\_tcam\\_retval::raw\\_data](#)

Retval 32 bit raw data.

struct [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result ezdp\\_lookup\\_int\\_tcam\\_retval::standard](#) [read]

Result for standard internal TCAM lookup.

struct [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result ezdp\\_lookup\\_int\\_tcam\\_retval::assoc\\_data](#) [read]

4 byte associated user data.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_int\_tcam\_standard\_result Struct Reference

Lookup internal tcam standard result.

### Data Fields

- union {
- [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result\\_t raw\\_data](#)
- struct {
- unsigned [match](#): EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_MATCH\_SIZE
- *Match indication.* unsigned [pad0](#):  
EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_RESERVED0\_15\_SIZE
- *Reserved bits 0 to 15.* unsigned [index](#):  
EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_INDEX\_SIZE
- *Lookup index result.* }
- };

---

### Detailed Description

Lookup internal tcam standard result.

---

### Field Documentation

[ezdp\\_lookup\\_int\\_tcam\\_standard\\_result\\_t ezdp\\_lookup\\_int\\_tcam\\_standard\\_result::raw\\_data](#)

unsigned [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result::match](#)

Match indication.

unsigned [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result::pad0](#)

Reserved bits 0 to 15.

unsigned [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result::index](#)

Lookup index result.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_lookup\_retval Struct Reference

Lookup return value.

### Data Fields

- union {
- [ezdp\\_lookup\\_retval\\_t raw\\_data](#)
- struct {
- unsigned [mem\\_error](#): EZDP\_LOOKUP\_RETVAL\_MEM\_ERROR\_SIZE
- Memory error indication.   unsigned [info](#): EZDP\_LOOKUP\_RETVAL\_INFO\_SIZE
- Operation specific additional information is available.   unsigned [success](#): EZDP\_LOOKUP\_RETVAL\_SUCCESS\_SIZE
- Operation success indication - No memory error and match.   unsigned [match](#): EZDP\_LOOKUP\_RETVAL\_MATCH\_SIZE
- Match indication (in lookup).   unsigned [data](#): EZDP\_LOOKUP\_RETVAL\_DATA\_SIZE
- The first 28 bits of the lookup result.   }
- };

---

### Detailed Description

Lookup return value.

---

### Field Documentation

[ezdp\\_lookup\\_retval\\_t ezdp\\_lookup\\_retval::raw\\_data](#)

unsigned [ezdp\\_lookup\\_retval::mem\\_error](#)

Memory error indication.

unsigned [ezdp\\_lookup\\_retval::info](#)

Operation specific additional information is available.

Currently only used by UIP lookup to indicate availability of extended result.

unsigned [ezdp\\_lookup\\_retval::success](#)

Operation success indication - No memory error and match.

unsigned [ezdp\\_lookup\\_retval::match](#)

Match indication (in lookup).

unsigned [ezdp\\_lookup\\_retval::data](#)

The first 28 bits of the lookup result.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_search\\_defs.h](#)

## ezdp\_mem\_pool\_config Struct Reference

memory pool configuration data structure

### Data Fields

- struct [ezdp\\_sum\\_addr base\\_addr](#)
  - *The start address of the memory.* uint16\_t [index\\_pool\\_id](#)
  - *BMU index pool id to be used by memory pool.* uint16\_t [obj\\_size](#)
- The size of the memory element/object.*
- 

### Detailed Description

memory pool configuration data structure

---

### Field Documentation

struct [ezdp\\_sum\\_addr ezdp\\_mem\\_pool\\_config::base\\_addr](#) [read]

The start address of the memory.

uint16\_t [ezdp\\_mem\\_pool\\_config::index\\_pool\\_id](#)

BMU index pool id to be used by memory pool.

NOTE: Pool id which should be configured/enabled to NPS

uint16\_t [ezdp\\_mem\\_pool\\_config::obj\\_size](#)

The size of the memory element/object.

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pool\\_defs.h](#)

## ezdp\_mem\_section\_info Struct Reference

### Data Fields

- uint32\_t [private\\_cmem\\_size](#)
  - The size of the private cmem section rounded to multiply of 32 bytes. uint32\_t [shared\\_cmem\\_size](#)
  - The size of the shared cmem section rounded to multiply of 32 bytes. uint32\_t [cache\\_size](#)
  - The size of the cmem used for thread data cache. uint32\_t [imem\\_private\\_data\\_size](#)
  - The size of the imem private data section . uint32\_t [imem\\_half\\_cluster\\_data\\_size](#)
  - The size of the imem half cluster data section . uint32\_t [imem\\_1\\_cluster\\_data\\_size](#)
  - The size of the imem 1 cluster data section . uint32\_t [imem\\_2\\_cluster\\_data\\_size](#)
  - The size of the imem 2 cluster data section . uint32\_t [imem\\_4\\_cluster\\_data\\_size](#)
  - The size of the imem 4 cluster data section . uint32\_t [imem\\_16\\_cluster\\_data\\_size](#)
  - The size of the imem 16 cluster data section . uint32\_t [imem\\_all\\_cluster\\_data\\_size](#)
  - The size of the imem all cluster data section . uint32\_t [emem\\_data\\_size](#)
  - The size of the emem shared data section . uint32\_t [imem\\_half\\_cluster\\_code\\_size](#)
  - The size of the imem half cluster code section . uint32\_t [imem\\_1\\_cluster\\_code\\_size](#)
  - The size of the imem half cluster code section . uint32\_t [imem\\_2\\_cluster\\_code\\_size](#)
  - The size of the imem 2 cluster code section . uint32\_t [imem\\_4\\_cluster\\_code\\_size](#)
  - The size of the imem 4 cluster code section . uint32\_t [imem\\_16\\_cluster\\_code\\_size](#)
  - The size of the imem 16 cluster code section . uint32\_t [imem\\_all\\_cluster\\_code\\_size](#)
- The size of the imem all cluster code section .
- 

### Field Documentation

#### uint32\_t [ezdp\\_mem\\_section\\_info::private\\_cmem\\_size](#)

The size of the private cmem section rounded to multiply of 32 bytes.

#### uint32\_t [ezdp\\_mem\\_section\\_info::shared\\_cmem\\_size](#)

The size of the shared cmem section rounded to multiply of 32 bytes.

#### uint32\_t [ezdp\\_mem\\_section\\_info::cache\\_size](#)

The size of the cmem used for thread data cache.

Limitation: The total core data cache (thread\_cache\_size\*number of threads) values is 0K, 1K, 2K, 4K, 8K and 16K The minimal thread data cache size is 256 bytes

#### uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_private\\_data\\_size](#)

The size of the imem private data section .

#### uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_half\\_cluster\\_data\\_size](#)

The size of the imem half cluster data section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_1\\_cluster\\_data\\_size](#)**

The size of the imem 1 cluster data section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_2\\_cluster\\_data\\_size](#)**

The size of the imem 2 cluster data section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_4\\_cluster\\_data\\_size](#)**

The size of the imem 4 cluster data section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_16\\_cluster\\_data\\_size](#)**

The size of the imem 16 cluster data section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_all\\_cluster\\_data\\_size](#)**

The size of the imem all cluster data section .

**uint32\_t [ezdp\\_mem\\_section\\_info::emem\\_data\\_size](#)**

The size of the emem shared data section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_half\\_cluster\\_code\\_size](#)**

The size of the imem half cluster code section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_1\\_cluster\\_code\\_size](#)**

The size of the imem half cluster code section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_2\\_cluster\\_code\\_size](#)**

The size of the imem 2 cluster code section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_4\\_cluster\\_code\\_size](#)**

The size of the imem 4 cluster code section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_16\\_cluster\\_code\\_size](#)**

The size of the imem 16 cluster code section .

**uint32\_t [ezdp\\_mem\\_section\\_info::imem\\_all\\_cluster\\_code\\_size](#)**

The size of the imem all cluster code section .

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp.h](#)



## ezdp\_output\_queue\_status Struct Reference

PMU output queue status definition (based on PMU system info).

### Data Fields

- union {
- [ezdp\\_output\\_queue\\_status\\_t raw\\_data](#)
- struct {
- unsigned [\\_pad0](#); EZDP\_OUTPUT\_QUEUE\_STATUS\_RESERVED18\_31\_SIZE
- Reserved bits 18 to 31.     unsigned [congestion](#); EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_SIZE
- The queue in congestion.     unsigned [ready](#); EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_SIZE
- The queue is ready to accept traffic.     uint16\_t [size](#)
- The total number of jobs in this list.     }
- };

### Detailed Description

PMU output queue status definition (based on PMU system info).

### Field Documentation

#### [ezdp\\_output\\_queue\\_status\\_t ezdp\\_output\\_queue\\_status::raw\\_data](#)

##### unsigned [ezdp\\_output\\_queue\\_status::\\_pad0](#)

Reserved bits 18 to 31.

##### unsigned [ezdp\\_output\\_queue\\_status::congestion](#)

The queue in congestion.

The size of the queue pass the threshold.

##### unsigned [ezdp\\_output\\_queue\\_status::ready](#)

The queue is ready to accept traffic.

##### uint16\_t [ezdp\\_output\\_queue\\_status::size](#)

The total number of jobs in this list.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_job\\_defs.h](#)

## ezdp\_pci\_addr Struct Reference

PCI Address data structure.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_PCI\_ADDR\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_PCI\_ADDR\_RESERVED29\_30\_SIZE
- < Address type     unsigned [virt\\_func\\_en](#): EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_SIZE
- Virtual Function Enable.     unsigned [phy\\_func](#): EZDP\_PCI\_ADDR\_PHY\_FUNC\_SIZE
- Physical function.     uint8\_t [virt\\_func](#)
- Virtual function.     unsigned [\\_\\_pad1](#) : EZDP\_PCI\_ADDR\_RESERVED14\_15\_SIZE
- Reserved bits 14-15.     unsigned [msid](#): EZDP\_PCI\_ADDR\_MSID\_SIZE
- < Type of the MSID     unsigned [\\_\\_pad2](#) : EZDP\_PCI\_ADDR\_RESERVED4\_7\_SIZE
- Reserved bits 4-7.     unsigned [address\\_msb](#): EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_SIZE
- 3 msb of 36 bits PCI address     uint32\_t [address](#)
- 32 lsb of 35 bit extended address     }
- };

### Detailed Description

PCI Address data structure.

### Field Documentation

uint32\_t [ezdp\\_pci\\_addr::raw\\_data](#)[EZDP\_PCI\_ADDR\_WORD\_COUNT]

unsigned [ezdp\\_pci\\_addr::\\_\\_pad0](#)

Address type. Reserved bits 29-30

unsigned [ezdp\\_pci\\_addr::virt\\_func\\_en](#)

Virtual Function Enable.

unsigned [ezdp\\_pci\\_addr::phy\\_func](#)

Physical function.

uint8\_t [ezdp\\_pci\\_addr::virt\\_func](#)

Virtual function.

unsigned [ezdp\\_pci\\_addr::\\_\\_pad1](#)

Reserved bits 14-15.

unsigned [ezdp\\_pci\\_addr::msid](#)

Type of the MSID. Select MSID of the structure

unsigned [ezdp\\_pci\\_addr::\\_\\_pad2](#)

Reserved bits 4-7.

**unsigned [ezdp\\_pci\\_addr::address\\_msb](#)**

3 msb of 36 bits PCI address

**uint32\_t [ezdp\\_pci\\_addr::address](#)**

32 lsb of 35 bit extended address

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_memory\\_defs.h](#)

## ezdp\_pci\_info Struct Reference

PCI info for describing to which endpoint, physical function, virtual function and queue the frame is to be sent.

### Data Fields

- union {
- [ezdp\\_pci\\_info\\_t raw\\_data](#)
- struct {
- unsigned [pad0](#) : EZDP\_PCI\_INFO\_RESERVED16\_32\_SIZE
- *Reserved bits 16 to 32.* unsigned [virt\\_func\\_en](#): EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_SIZE
- *Enable/disable virtual function.* unsigned [queue](#): EZDP\_PCI\_INFO\_QUEUE\_SIZE
- *Queue ID.* unsigned [phys\\_func](#): EZDP\_PCI\_INFO\_PHYS\_FUNC\_SIZE
- *Physical function.* unsigned [endpoint](#): EZDP\_PCI\_INFO\_ENDPOINT\_SIZE
- *Destination end point (PCI device ID).* unsigned [virt\\_func](#): EZDP\_PCI\_INFO\_VIRT\_FUNC\_SIZE
- *Virtual function number.* }
- };

### Detailed Description

PCI info for describing to which endpoint, physical function, virtual function and queue the frame is to be sent.

### Field Documentation

#### [ezdp\\_pci\\_info\\_t ezdp\\_pci\\_info::raw\\_data](#)

##### unsigned [ezdp\\_pci\\_info::pad0](#)

Reserved bits 16 to 32.

##### unsigned [ezdp\\_pci\\_info::virt\\_func\\_en](#)

Enable/disable virtual function.

##### unsigned [ezdp\\_pci\\_info::queue](#)

Queue ID.

##### unsigned [ezdp\\_pci\\_info::phys\\_func](#)

Physical function.

##### unsigned [ezdp\\_pci\\_info::endpoint](#)

Destination end point (PCI device ID).

##### unsigned [ezdp\\_pci\\_info::virt\\_func](#)

Virtual function number.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)

## ezdp\_pci\_msg Struct Reference

Message from PCI queue.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_PCI\_MSG\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0\\_\\_](#) : EZDP\_PCI\_MSG\_ECC\_SIZE
- ECC.     struct [ezdp\\_pci\\_msg\\_ctrl ctrl](#)
- < Message encoding     union {
- struct [ezdp\\_pci\\_msg\\_payload\\_elbi elbi\\_payload](#)
- ELBI message payload.     struct [ezdp\\_pci\\_msg\\_payload\\_atl atl\\_payload](#)
- ATS message payload.     struct [ezdp\\_pci\\_msg\\_payload\\_msix msix\\_payload](#)
- MSIX message payload.     }
- }
- };

### Detailed Description

Message from PCI queue.

### Field Documentation

uint32\_t [ezdp\\_pci\\_msg::raw\\_data](#) [EZDP\_PCI\_MSG\_WORD\_COUNT]

unsigned [ezdp\\_pci\\_msg:: \\_\\_pad0\\_\\_](#)  
ECC.

struct [ezdp\\_pci\\_msg\\_ctrl ezdp\\_pci\\_msg::ctrl](#) [read]  
Message encoding. Message control configuration

struct [ezdp\\_pci\\_msg\\_payload\\_elbi ezdp\\_pci\\_msg::elbi\\_payload](#) [read]  
ELBI message payload.

struct [ezdp\\_pci\\_msg\\_payload\\_atl ezdp\\_pci\\_msg::atl\\_payload](#) [read]  
ATS message payload.

struct [ezdp\\_pci\\_msg\\_payload\\_msix ezdp\\_pci\\_msg::msix\\_payload](#) [read]  
MSIX message payload.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)

## ezdp\_pci\_msg\_ctrl Struct Reference

PCI message control.

### Data Fields

- union {
- [ezdp\\_pci\\_msg\\_ctrl\\_t raw\\_data](#)
- struct {
- unsigned [virt\\_func\\_en](#): EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_SIZE
- *Virtual Function Enable.*   unsigned [bar\\_num](#): EZDP\_PCI\_MSG\_CTRL\_BAR\_NUM\_SIZE
- *BAR number.*           unsigned [pad0](#) : EZDP\_PCI\_MSG\_CTRL\_RESERVED10\_11\_SIZE
- *Reserved bits 10-11.*   unsigned [phy\\_func](#): EZDP\_PCI\_MSG\_CTRL\_PHY\_FUNC\_SIZE
- *Physical function.*     unsigned [pad1](#) : EZDP\_PCI\_MSG\_CTRL\_RESERVED8\_SIZE
- *Reserved bit number 8.*   unsigned [virt\\_func](#): EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_SIZE
- *Virtual function number.*   }
- };

### Detailed Description

PCI message control.

### Field Documentation

[ezdp\\_pci\\_msg\\_ctrl\\_t ezdp\\_pci\\_msg\\_ctrl::raw\\_data](#)

unsigned [ezdp\\_pci\\_msg\\_ctrl::virt\\_func\\_en](#)

Virtual Function Enable.

unsigned [ezdp\\_pci\\_msg\\_ctrl::bar\\_num](#)

BAR number.

unsigned [ezdp\\_pci\\_msg\\_ctrl:: pad0](#)

Reserved bits 10-11.

unsigned [ezdp\\_pci\\_msg\\_ctrl::phy\\_func](#)

Physical function.

unsigned [ezdp\\_pci\\_msg\\_ctrl:: pad1](#)

Reserved bit number 8.

unsigned [ezdp\\_pci\\_msg\\_ctrl::virt\\_func](#)

Virtual function number.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)

## ezdp\_pci\_msg\_payload\_ats Struct Reference

PCI ATS message payload.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_RESERVED\_SIZE
- *Reserved Byte.*     uint32\_t [data\\_msb](#)
- *ATS MSB data.*     uint32\_t [data\\_lsb](#)
- *ATS LSB data.*     }
- };

### Detailed Description

PCI ATS message payload.

### Field Documentation

uint32\_t [ezdp\\_pci\\_msg\\_payload\\_ats::raw\\_data](#)[EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_WORD\_COUNT]

unsigned [ezdp\\_pci\\_msg\\_payload\\_ats::pad0](#)

Reserved Byte.

uint32\_t [ezdp\\_pci\\_msg\\_payload\\_ats::data\\_msb](#)

ATS MSB data.

uint32\_t [ezdp\\_pci\\_msg\\_payload\\_ats::data\\_lsb](#)

ATS LSB data.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)

## ezdp\_pci\_msg\_payload\_elbi Struct Reference

PCI ELBI message payload.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_WORD\_COUNT]
- struct {
- unsigned [\\_pad0](#) : EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_RESERVED\_SIZE
- *Reserved Byte.*     uint32\_t [address](#)
- *ELBI Address.*     uint32\_t [data](#)
- *ELBI data.*     }
- };

---

### Detailed Description

PCI ELBI message payload.

---

### Field Documentation

uint32\_t [ezdp\\_pci\\_msg\\_payload\\_elbi::raw\\_data](#)[EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_WORD\_COUNT]

unsigned [ezdp\\_pci\\_msg\\_payload\\_elbi::\\_pad0](#)

Reserved Byte.

uint32\_t [ezdp\\_pci\\_msg\\_payload\\_elbi::address](#)

ELBI Address.

uint32\_t [ezdp\\_pci\\_msg\\_payload\\_elbi::data](#)

ELBI data.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)



## ezdp\_pci\_msg\_payload\_msix Struct Reference

PCI MSIX message payload.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED0\_31\_SIZE
- *Reserved bits 0 to 31.*   unsigned [\\_\\_pad1](#) :
- EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_32\_63\_SIZE
- *Reserved bits 32 to 63.*   unsigned [\\_\\_pad2](#) :
- EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED66\_95\_SIZE
- *Reserved bits 66 to 95.*   unsigned [vector\\_index](#) :
- EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_INDEX\_SIZE
- *Requested MSIX vector index.*   }
- };

---

### Detailed Description

PCI MSIX message payload.

---

### Field Documentation

uint32\_t

[ezdp\\_pci\\_msg\\_payload\\_msix::raw\\_data](#)[EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_WORD\_COUNT]

unsigned [ezdp\\_pci\\_msg\\_payload\\_msix:: \\_\\_pad0](#)

Reserved bits 0 to 31.

unsigned [ezdp\\_pci\\_msg\\_payload\\_msix:: \\_\\_pad1](#)

Reserved bits 32 to 63.

unsigned [ezdp\\_pci\\_msg\\_payload\\_msix:: \\_\\_pad2](#)

Reserved bits 66 to 95.

unsigned [ezdp\\_pci\\_msg\\_payload\\_msix::vector\\_index](#)

Requested MSIX vector index.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_pci\\_defs.h](#)

## ezdp\_posted\_ctr\_msg Struct Reference

Posted counter message queue definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_POSTED\_CTR\_MSG\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_POSTED\_CTR\_MSG\_ECC\_SIZE
- ECC.    unsigned [pad1](#) : EZDP\_POSTED\_CTR\_MSG\_RESERVED8\_23\_SIZE
- *reserved bits 8-23*    unsigned [overrun\\_error\\_condition](#):  
EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_SIZE
- *Queue was overrun and old messages are lost.*    unsigned [pad2](#) :  
EZDP\_POSTED\_CTR\_MSG\_RESERVED5\_6\_SIZE
- *reserved bits 5-6*    unsigned [clear](#): EZDP\_POSTED\_CTR\_MSG\_CLEAR\_SIZE
- *Counter was cleared in memory.*    unsigned [flush](#): EZDP\_POSTED\_CTR\_MSG\_FLUSH\_SIZE
- *Flush was executed before reporting.*    struct [ezdp\\_sum\\_addr sum\\_addr](#)
- *< Posted counter message type*    uint64\_t [value](#)
- *64 bit message value from a summarize address*    }
- };

### Detailed Description

Posted counter message queue definition.

### Field Documentation

uint32\_t [ezdp\\_posted\\_ctr\\_msg::raw\\_data](#)[EZDP\_POSTED\_CTR\_MSG\_WORD\_COUNT]

unsigned [ezdp\\_posted\\_ctr\\_msg:: pad0](#)

ECC.

unsigned [ezdp\\_posted\\_ctr\\_msg:: pad1](#)

reserved bits 8-23

unsigned [ezdp\\_posted\\_ctr\\_msg::overrun\\_error\\_condition](#)

Queue was overrun and old messages are lost.

unsigned [ezdp\\_posted\\_ctr\\_msg:: pad2](#)

reserved bits 5-6

unsigned [ezdp\\_posted\\_ctr\\_msg::clear](#)

Counter was cleared in memory.

unsigned [ezdp\\_posted\\_ctr\\_msg::flush](#)

Flush was executed before reporting.

struct [ezdp\\_sum\\_addr ezdp\\_posted\\_ctr\\_msg::sum\\_addr](#) [read]

< Posted counter message type  
memory summarize address

uint64\_t [ezdp\\_posted\\_ctr\\_msg::value](#)

64 bit message value from a summarize address

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_ring\_cfg Struct Reference

ring (array queue) configuration data structure

### Data Fields

- struct [ezdp\\_sum\\_addr base\\_addr](#)
  - Base address to start of queue. struct [ezdp\\_sum\\_addr control\\_addr](#)
  - Control address for managing the queue - must be in resolution of 16B. uint32\_t [size](#)  
Maximum number of elements in array.
- 

### Detailed Description

ring (array queue) configuration data structure

---

### Field Documentation

struct [ezdp\\_sum\\_addr ezdp\\_ring\\_cfg::base\\_addr](#) [read]

Base address to start of queue.

struct [ezdp\\_sum\\_addr ezdp\\_ring\\_cfg::control\\_addr](#) [read]

Control address for managing the queue - must be in resolution of 16B.

uint32\_t [ezdp\\_ring\\_cfg::size](#)

Maximum number of elements in array.

Must be a power of 2 (16,32,64,128...)

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_queue\\_defs.h](#)

## ezdp\_rtc Struct Reference

[ezdp\\_rtc](#) struct for ezdp

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_RTC\_WORD\_COUNT]
- struct {
- uint32\_t [sec](#)
- *The real time clock in resolution of seconds.*    uint32\_t [nsec](#)
- *The real time clock in resolution of nano seconds.*   }
- };

---

### Detailed Description

[ezdp\\_rtc](#) struct for ezdp

---

### Field Documentation

uint32\_t [ezdp\\_rtc::raw\\_data](#)[EZDP\_RTC\_WORD\_COUNT]

uint32\_t [ezdp\\_rtc::sec](#)

The real time clock in resolution of seconds.

uint32\_t [ezdp\\_rtc::nsec](#)

The real time clock in resolution of nano seconds.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_time\\_defs.h](#)

## ezdp\_security\_handle Struct Reference

security handle configuration data structure

### Data Fields

- union {
- [ezdp\\_security\\_handle\\_t raw\\_data](#)
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_SECURITY\_HANDLE\_RESERVED24\_31\_SIZE
- < Security algorithm type    unsigned [\\_\\_pad1](#) : EZDP\_SECURITY\_HANDLE\_RESERVED8\_15\_SIZE
- Reserved bits 8 to 15.    uint8\_t [context\\_id](#)
- Security context ID.    }
- };

### Detailed Description

security handle configuration data structure

### Field Documentation

[ezdp\\_security\\_handle\\_t ezdp\\_security\\_handle::raw\\_data](#)

unsigned [ezdp\\_security\\_handle:: \\_\\_pad0](#)

< Security algorithm type  
Reserved bits 24 to 31

unsigned [ezdp\\_security\\_handle:: \\_\\_pad1](#)

Reserved bits 8 to 15.

uint8\_t [ezdp\\_security\\_handle::context\\_id](#)

Security context ID.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_security\\_defs.h](#)

## ezdp\_single\_ctr\_cfg Struct Reference

On-demand single value counter configuration definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_SINGLE\_CTR\_CFG\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0](#) : EZDP\_SINGLE\_CTR\_CFG\_ECC\_SIZE
- ECC.    unsigned [\\_\\_pad1](#) : EZDP\_SINGLE\_CTR\_CFG\_SUB\_TYPE\_SIZE
- Counter sub type (long=0).    unsigned [report\\_exceeded](#);
- EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDED\_SIZE
- Number of bits threshold (1-58) for generating exceed message.    unsigned [\\_\\_pad2](#) :
- EZDP\_SINGLE\_CTR\_CFG\_ZERO\_SIZE
- Zero bit.    unsigned [enable\\_exceed\\_message](#);
- EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_SIZE
- Enable threshold exceed message.    unsigned [\\_\\_pad3](#) :
- EZDP\_SINGLE\_CTR\_CFG\_RESERVED0\_10\_SIZE
- Reserved bits 0 to 10.    unsigned [\\_\\_pad4](#) : EZDP\_SINGLE\_CTR\_CFG\_RESERVED32\_63\_SIZE
- Reserved bits 32 to 63.    uint64\_t [value](#)
- Counter value.    }
- };

---

### Detailed Description

On-demand single value counter configuration definition.

---

### Field Documentation

uint32\_t [ezdp\\_single\\_ctr\\_cfg::raw\\_data](#)[EZDP\_SINGLE\_CTR\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_single\\_ctr\\_cfg:: \\_\\_pad0](#)

ECC.

unsigned [ezdp\\_single\\_ctr\\_cfg:: \\_\\_pad1](#)

Counter sub type (long=0).

unsigned [ezdp\\_single\\_ctr\\_cfg::report\\_exceeded](#)

Number of bits threshold (1-58) for generating exceed message.

unsigned [ezdp\\_single\\_ctr\\_cfg:: \\_\\_pad2](#)

Zero bit.

unsigned [ezdp\\_single\\_ctr\\_cfg::enable\\_exceed\\_message](#)

Enable threshold exceed message.

unsigned [ezdp\\_single\\_ctr\\_cfg::pad3](#)

Reserved bits 0 to 10.

unsigned [ezdp\\_single\\_ctr\\_cfg::pad4](#)

Reserved bits 32 to 63.

uint64\_t [ezdp\\_single\\_ctr\\_cfg::value](#)

Counter value.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)



## ezdp\_small\_linked\_buffers\_desc Struct Reference

Small linked buffers descriptor. May hold up to 3 buffs.

### Data Fields

- struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) [line](#) [EZDP\_SMALL\_LBD]  
*Array of 1 linked buffers line, which contains up to 3 buffers.*
- 

### Detailed Description

Small linked buffers descriptor. May hold up to 3 buffs.

---

### Field Documentation

struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) [ezdp\\_small\\_linked\\_buffers\\_desc::line](#)[EZDP\_SMALL\_LBD]  
[read]

Array of 1 linked buffers line, which contains up to 3 buffers.

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_frame\\_defs.h](#)

## ezdp\_sum\_addr Struct Reference

Summarized Address data structure.

### Data Fields

- union {
- [ezdp\\_sum\\_addr\\_t raw\\_data](#)
- struct {
- unsigned [msid](#): EZDP\_SUM\_ADDR\_MSID\_SIZE
- < Memory space type     unsigned [element\\_index](#): EZDP\_SUM\_ADDR\_ELEMENT\_INDEX\_SIZE
- Index of the element within a specific structure.     }
- };

### Detailed Description

Summarized Address data structure.

### Field Documentation

[ezdp\\_sum\\_addr\\_t ezdp\\_sum\\_addr::raw\\_data](#)

unsigned [ezdp\\_sum\\_addr::msid](#)

< Memory space type

Memory system ID

unsigned [ezdp\\_sum\\_addr::element\\_index](#)

Index of the element within a specific structure.

union { ... }

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_memory\\_defs.h](#)

## ezdp\_sum\_addr\_table\_desc Struct Reference

Structure definition table entry data structure.

### Data Fields

- union {
- [ezdp\\_sum\\_addr\\_table\\_desc t raw\\_data](#)
- struct {
- unsigned [key\\_shuff\\_en](#): EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_SIZE
- *Enable/Disable key shuffling.* unsigned [key\\_shuff\\_bits](#):  
EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_BITS\_SIZE
- *Bitmap for table key shuffling.* unsigned [pad0](#):  
EZDP\_SUM\_ADDR\_TABLE\_DESC\_RESERVED25\_26\_SIZE
- *< Memory space type* unsigned [key\\_size](#): EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SIZE\_SIZE
- *Table key size in bits (up to 32) / Hash1 size in bits (up to 32).* unsigned [msid](#):  
EZDP\_SUM\_ADDR\_TABLE\_DESC\_MSID\_SIZE
- *Memory system ID.* uint16\_t [base\\_index](#)
- *Base index of the structure This is the logical offset within a specific MSID, Index Resolution: IMEM: 1KB  
EMEM: 1MB.* }
- };

### Detailed Description

Structure definition table entry data structure.

### Field Documentation

[ezdp\\_sum\\_addr\\_table\\_desc t ezdp\\_sum\\_addr\\_table\\_desc::raw\\_data](#)

unsigned [ezdp\\_sum\\_addr\\_table\\_desc::key\\_shuff\\_en](#)

Enable/Disable key shuffling.

unsigned [ezdp\\_sum\\_addr\\_table\\_desc::key\\_shuff\\_bits](#)

Bitmap for table key shuffling.

unsigned [ezdp\\_sum\\_addr\\_table\\_desc:: pad0](#)

< Memory space type

Reserved bits 25 to 26

unsigned [ezdp\\_sum\\_addr\\_table\\_desc::key\\_size](#)

Table key size in bits (up to 32) / Hash1 size in bits (up to 32).

Used for masking the input key

unsigned [ezdp\\_sum\\_addr\\_table\\_desc::msid](#)

Memory system ID.

uint16\_t [ezdp\\_sum\\_addr\\_table\\_desc::base\\_index](#)

Base index of the structure This is the logical offset within a specific MSID, Index Resolution: IMEM: 1KB  
EMEM: 1MB.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_memory\\_defs.h](#)

## ezdp\_tb\_ctr\_cfg Struct Reference

Token bucket counter configuration definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_TB\_CTR\_CFG\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_TB\_CTR\_CFG\_RESERVED26\_31\_SIZE
- *Reserved bits 26 to 31.*   unsigned [coupling\\_flag](#): EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_SIZE
- *Enable/disable coupling flag.*   unsigned [color\\_aware](#): EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_SIZE
- *Enable/disable color awareness.*   unsigned [excess\\_profile\\_id](#): EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_SIZE
- *< The marking algorithm used for this profile*   unsigned [commit\\_profile\\_id](#): EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_SIZE
- *Id of commit token bucket profile in the token bucket profile table.*   unsigned [pad1](#) : EZDP\_TB\_CTR\_CFG\_RESERVED32\_63\_SIZE
- *Reserved bits 32 to 63.*   unsigned [pad2](#) : EZDP\_TB\_CTR\_CFG\_RESERVED64\_95\_SIZE
- *Reserved bits 64 to 95.*   unsigned [pad3](#) : EZDP\_TB\_CTR\_CFG\_RESERVED96\_127\_SIZE
- *Reserved bits 96 to 127.*   }
- };

### Detailed Description

Token bucket counter configuration definition.

### Field Documentation

uint32\_t [ezdp\\_tb\\_ctr\\_cfg::raw\\_data](#)[EZDP\_TB\_CTR\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_tb\\_ctr\\_cfg::pad0](#)

Reserved bits 26 to 31.

unsigned [ezdp\\_tb\\_ctr\\_cfg::coupling\\_flag](#)

Enable/disable coupling flag.

Relevant only for trTCM MEF algorithm.

unsigned [ezdp\\_tb\\_ctr\\_cfg::color\\_aware](#)

Enable/disable color awareness.

Relevant only for srTCM, trTCM and trTCM MEF algorithms.

unsigned [ezdp\\_tb\\_ctr\\_cfg::excess\\_profile\\_id](#)

< The marking algorithm used for this profile

Id of excess token bucket profile in the token bucket profile table

**unsigned [ezdp\\_tb\\_ctr\\_cfg::commit\\_profile\\_id](#)**

Id of commit token bucket profile in the token bucket profile table.

**unsigned [ezdp\\_tb\\_ctr\\_cfg::pad1](#)**

Reserved bits 32 to 63.

**unsigned [ezdp\\_tb\\_ctr\\_cfg::pad2](#)**

Reserved bits 64 to 95.

**unsigned [ezdp\\_tb\\_ctr\\_cfg::pad3](#)**

Reserved bits 96 to 127.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_tb\_ctr\_result Struct Reference

Token bucket counter result value definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_TB\_CTR\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_TB\_CTR\_RESULT\_RESERVED0\_31\_SIZE
- Reserved bits 0 to 31.    unsigned [pad1](#) : EZDP\_TB\_CTR\_RESULT\_RESERVED60\_63\_SIZE
- Reserved bits 60 to 63.   unsigned [empty\\_excess\\_bucket Ug](#) : EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_UG\_SIZE
- Empty excess bucket ultra green Close to burst size indication after the force dec command.   unsigned [empty\\_commit\\_bucket Ug](#) : EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_UG\_SIZE
- Empty commit bucket ultra green Close to burst size indication after the force dec command.   unsigned [empty\\_excess\\_bucket](#) : EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_SIZE
- Indicate that excess bucket is empty.   unsigned [empty\\_commit\\_bucket](#) : EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_SIZE
- Indicate that commit bucket is empty.   unsigned [pad2](#) : EZDP\_TB\_CTR\_RESULT\_RESERVED34\_57\_SIZE
- Reserved bits 34 to 57.   }
- };

### Detailed Description

Token bucket counter result value definition.

### Field Documentation

uint32\_t [ezdp\\_tb\\_ctr\\_result::raw\\_data](#) [EZDP\_TB\_CTR\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_tb\\_ctr\\_result::pad0](#)

Reserved bits 0 to 31.

unsigned [ezdp\\_tb\\_ctr\\_result::pad1](#)

Reserved bits 60 to 63.

unsigned [ezdp\\_tb\\_ctr\\_result::empty\\_excess\\_bucket Ug](#)

Empty excess bucket ultra green Close to burst size indication after the force dec command.

unsigned [ezdp\\_tb\\_ctr\\_result::empty\\_commit\\_bucket Ug](#)

Empty commit bucket ultra green Close to burst size indication after the force dec command.

unsigned [ezdp\\_tb\\_ctr\\_result::empty\\_excess\\_bucket](#)

Indicate that excess bucket is empty.

Note1: The value can be negative. Note2: Applicable to increment/decrement command only.

**unsigned [ezdp\\_tb\\_ctr\\_result::empty\\_commit\\_bucket](#)**

Indicate that commit bucket is empty.

Note1: The value can be negative. Note2: Applicable to increment/decrement command only.

**unsigned [ezdp\\_tb\\_ctr\\_result::pad2](#)**

Reserved bits 34 to 57.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)



## ezdp\_version Struct Reference

version info data structure

### Data Fields

- const char \* [project\\_name](#)
  - *Project name.* const char \* [module\\_name](#)
  - *Module name.* uint32\_t [major\\_version](#)
  - *Major version.* uint32\_t [minor\\_version](#)
  - *Minor version.* uint8\_t [version\\_char](#)
  - *Version version.* const char \* [version\\_string](#)
  - *Version string.* uint8\_t [major\\_patch\\_version](#)
  - *Major patch version.* uint8\_t [minor\\_patch\\_version](#)
  - *Minor patch version.* uint8\_t [micro\\_patch\\_version](#)
  - *Micro patch version.* const int8\_t \* [build\\_number](#)
  - *Build number.* const char \* [creation\\_date](#)
  - *Creation date.* const char \* [creation\\_time](#)
- Creation time.*
- 

### Detailed Description

version info data structure

---

### Field Documentation

const char\* [ezdp\\_version::project\\_name](#)

Project name.

const char\* [ezdp\\_version::module\\_name](#)

Module name.

uint32\_t [ezdp\\_version::major\\_version](#)

Major version.

uint32\_t [ezdp\\_version::minor\\_version](#)

Minor version.

uint8\_t [ezdp\\_version::version\\_char](#)

Version version.

const char\* [ezdp\\_version::version\\_string](#)

Version string.

**uint8\_t [ezdp\\_version::major\\_patch\\_version](#)**

Major patch version.

**uint8\_t [ezdp\\_version::minor\\_patch\\_version](#)**

Minor patch version.

**uint8\_t [ezdp\\_version::micro\\_patch\\_version](#)**

Micro patch version.

**const int8\_t\* [ezdp\\_version::build\\_number](#)**

Build number.

**const char\* [ezdp\\_version::creation\\_date](#)**

Creation date.

**const char\* [ezdp\\_version::creation\\_time](#)**

Creation time.

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_version.h](#)

## ezdp\_watchdog\_accumulative\_window\_cfg Struct Reference

Watchdog accumulative window configuration definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_ALERT\_SIZE
- ( Max alert) or (Min alert)   unsigned [pad1](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MIN\_THRESHOLD\_ALERT\_SIZE
- Check resulted in passing of the Min threshold.   unsigned [pad2](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MAX\_THRESHOLD\_ALERT\_SIZE
- Check resulted in passing of the Max threshold.   unsigned [pad3](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED5\_28\_SIZE
- reserved bits 5-28   unsigned [accumulative events](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_ACCUMULATIVE\_EVENTS\_SIZE
- Counts the number of lost events deficit as compared to per profile expected events in a given scan period.   unsigned [pad4](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED63\_SIZE
- reserved bit 63   unsigned [pad5](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PARITY\_SIZE
- The counter parity value.   unsigned [profile\\_id](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_SIZE
- A pointer to one of 16 profiles, where each profile holds one set of minimum and maximum event thresholds.   unsigned [valid](#) : EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_SIZE
- Indicates if the counters value is valid.   unsigned [pad6](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_INIT\_BIT\_SIZE
- Raise after CTOP initialize the counter.   unsigned [curr\\_events](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_CURR\_EVENTS\_SIZE
- The current number of events in the corresponding session.   unsigned [last\\_events](#) :  
EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_LAST\_EVENTS\_SIZE
- The previous current event counter to calculate the new events.   }
- };

### Detailed Description

Watchdog accumulative window configuration definition.

### Field Documentation

uint32\_t  
[ezdp\\_watchdog\\_accumulative\\_window\\_cfg::raw\\_data](#)[EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg:: pad0](#)

( Max alert) or (Min alert)

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg:: pad1](#)

Check resulted in passing of the Min threshold.

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::pad2](#)

Check resulted in passing of the Max threshold.

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::pad3](#)

reserved bits 5-28

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::accumulative\\_events](#)

Counts the number of lost events deficit as compared to per profile expected events in a given scan period.

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::pad4](#)

reserved bit 63

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::pad5](#)

The counter parity value.

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::profile\\_id](#)

A pointer to one of 16 profiles, where each profile holds one set of minimum and maximum event thresholds.

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::valid](#)

Indicates if the counters value is valid.

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::pad6](#)

Raise after CTOP initialize the counter.

Reset after the first check

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::curr\\_events](#)

The current number of events in the corresponding session.

unsigned [ezdp\\_watchdog\\_accumulative\\_window\\_cfg::last\\_events](#)

The previous current event counter to calculate the new events.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_watchdog\_ctr\_cfg Struct Reference

Watchdog counter configuration definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_WATCHDOG\_CTR\_CFG\_WORD\_COUNT]
- struct {
- unsigned [\\_\\_pad0\\_\\_](#): EZDP\_WATCHDOG\_CTR\_CFG\_ECC\_SIZE
- ECC.   unsigned [\\_\\_pad1\\_\\_](#): EZDP\_WATCHDOG\_CTR\_CFG\_SUB\_TYPE\_SIZE
- Counter sub type (bitwise=9).   unsigned [\\_\\_pad2\\_\\_](#):
- EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED0\_18\_SIZE
- Reserved bits 0 to 18.   unsigned [\\_\\_pad3\\_\\_](#): EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED32\_63\_SIZE
- Reserved bits 32 to 63.   union {
- struct [ezdp\\_watchdog\\_accumulative\\_window\\_cfg accumulative\\_window](#)
- Accumulative window mode configuration for WD counter.   struct [ezdp\\_watchdog\\_sliding\\_window\\_cfg sliding\\_window](#)
- Sliding window mode configuration for WD counter.   }
- }
- };

### Detailed Description

Watchdog counter configuration definition.

### Field Documentation

uint32\_t [ezdp\\_watchdog\\_ctr\\_cfg::raw\\_data](#)[EZDP\_WATCHDOG\_CTR\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_watchdog\\_ctr\\_cfg:: \\_\\_pad0\\_\\_](#)  
ECC.

unsigned [ezdp\\_watchdog\\_ctr\\_cfg:: \\_\\_pad1\\_\\_](#)  
Counter sub type (bitwise=9).

unsigned [ezdp\\_watchdog\\_ctr\\_cfg:: \\_\\_pad2\\_\\_](#)  
Reserved bits 0 to 18.

unsigned [ezdp\\_watchdog\\_ctr\\_cfg:: \\_\\_pad3\\_\\_](#)  
Reserved bits 32 to 63.

struct [ezdp\\_watchdog\\_accumulative\\_window\\_cfg ezdp\\_watchdog\\_ctr\\_cfg::accumulative\\_window](#)  
[read]

Accumulative window mode configuration for WD counter.

struct [ezdp\\_watchdog\\_sliding\\_window\\_cfg ezdp\\_watchdog\\_ctr\\_cfg::sliding\\_window](#) [read]  
Sliding window mode configuration for WD counter.

**union { ... }**

---

**The documentation for this struct was generated from the following file:**

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_watchdog\_ctr\_check\_result Struct Reference

Watchdog counter check result definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_WORD\_COUNT]
- struct {
- unsigned [alert](#): EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_SIZE
- ( Max alert) or (Min alert)   unsigned [min\\_threshold\\_alert](#):  
EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MIN\_THRESHOLD\_ALERT\_SIZE
- Check resulted in passing of the Min threshold.   unsigned [max\\_threshold\\_alert](#):  
EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MAX\_THRESHOLD\_ALERT\_SIZE
- Check resulted in passing of the Max threshold.   unsigned [pad0](#) :  
EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_RESERVED4\_28\_SIZE
- reserved bits 4-28   unsigned [pad1](#) :  
EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ACCUMULATIVE\_EVENTS\_SIZE
- Counts the number of lost events deficit as compared to per profile expected events in a given scan period 5  
MSB.   unsigned [pad2](#) : EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_WINDOW\_RELATED\_SIZE
- Counts the number of lost events deficit as compared to per profile expected events in a given scan  
period.   unsigned [pad3](#) : EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_RESERVED62\_SIZE
- reserved bit - 62   unsigned [pad4](#) : EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_PROFILE\_ID\_SIZE
- A pointer to one of 16 profiles, where each profile holds one set of minimum and maximum event  
thresholds.   unsigned [pad5](#) : EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_VALID\_SIZE
- Indicates if the counters value is valid.   unsigned [pad6](#) :  
EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_INIT\_BIT\_SIZE
- Raise after CTOP initialize the counter.   unsigned [pad7](#) :  
EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_CURR\_EVENTS\_SIZE
- The current number of events in the corresponding session.   unsigned [pad8](#) :  
EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_LAST\_EVENTS\_SIZE
- The previous current event counter to calculate the new events.   }
- };

### Detailed Description

Watchdog counter check result definition.

### Field Documentation

uint32\_t  
[ezdp\\_watchdog\\_ctr\\_check\\_result::raw\\_data](#)[EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_WORD\_COUNT]

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result::alert](#)

( Max alert) or (Min alert)

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result::min\\_threshold\\_alert](#)

Check resulted in passing of the Min threshold.

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result::max\\_threshold\\_alert](#)

Check resulted in passing of the Max threshold.

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad0](#)

reserved bits 4-28

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad1](#)

Counts the number of lost events deficit as compared to per profile expected events in a given scan period 5 MSB.

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad2](#)

Counts the number of lost events deficit as compared to per profile expected events in a given scan period.

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad3](#)

reserved bit - 62

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad4](#)

A pointer to one of 16 profiles, where each profile holds one set of minimum and maximum event thresholds.

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad5](#)

Indicates if the counters value is valid.

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad6](#)

Raise after CTOP initialize the counter.

Reset after the first check

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad7](#)

The current number of events in the corresponding session.

unsigned [ezdp\\_watchdog\\_ctr\\_check\\_result:: pad8](#)

The previous current event counter to calculate the new events.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)



## ezdp\_watchdog\_ctr\_start\_result Struct Reference

Watchdog counter check result definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_WATCHDOG\_CTR\_START\_RESULT\_WORD\_COUNT]
- struct {
- uint32\_t [msb](#)
- Counter MSB.   uint32\_t [lsb](#)
- Counter LSB.   }
- };

---

### Detailed Description

Watchdog counter check result definition.

---

### Field Documentation

uint32\_t  
[ezdp\\_watchdog\\_ctr\\_start\\_result::raw\\_data](#)[EZDP\_WATCHDOG\_CTR\_START\_RESULT\_WORD\_COUNT]

uint32\_t [ezdp\\_watchdog\\_ctr\\_start\\_result::msb](#)

Counter MSB.

uint32\_t [ezdp\\_watchdog\\_ctr\\_start\\_result::lsb](#)

Counter LSB.

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

## ezdp\_watchdog\_sliding\_window\_cfg Struct Reference

Watchdog sliding window configuration definition.

### Data Fields

- union {
- uint32\_t [raw\\_data](#) [EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_WORD\_COUNT]
- struct {
- unsigned [pad0](#) : EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_ALERT\_SIZE
- ( Max alert) or (Min alert)   unsigned [pad1](#) :  
EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_MIN\_THRESHOLD\_ALERT\_SIZE
- Check resulted in passing of the Min threshold.   unsigned [pad2](#) :  
EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_MAX\_THRESHOLD\_ALERT\_SIZE
- Check resulted in passing of the Max threshold.   unsigned [pad3](#) :  
EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_RESERVED5\_28\_SIZE
- reserved bits 5-28   unsigned [valid windows](#):  
EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_VALID\_WINDOWS\_SIZE
- Counts the number of valid sliding windows.   unsigned [pad4](#) :  
EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_RESERVED63\_SIZE
- reserved bit 63   unsigned [pad5](#) : EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_PARITY\_SIZE
- The counter parity value.   unsigned [profile\\_id](#):  
EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_PROFILE\_ID\_SIZE
- A pointer to one of 16 profiles, where each profile holds one set of minimum and maximum event thresholds.   unsigned [valid](#): EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_VALID\_SIZE
- Indicates if the counters value is valid.   unsigned [pad6](#) :  
EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_RESERVED56\_SIZE
- reserved bit 56   unsigned [counters](#): EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_COUNTERS\_SIZE
- 3/4/6/8/12/24 windows counters   }
- };

### Detailed Description

Watchdog sliding window configuration definition.

### Field Documentation

uint32\_t

[ezdp\\_watchdog\\_sliding\\_window\\_cfg::raw\\_data](#)[EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_WORD\_COUNT]

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg:: pad0](#)

( Max alert) or (Min alert)

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg:: pad1](#)

Check resulted in passing of the Min threshold.

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg:: pad2](#)

Check resulted in passing of the Max threshold.

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg:: pad3](#)

reserved bits 5-28

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg::valid\\_windows](#)

Counts the number of valid sliding windows.

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg:: pad4](#)

reserved bit 63

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg:: pad5](#)

The counter parity value.

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg::profile\\_id](#)

A pointer to one of 16 profiles, where each profile holds one set of minimum and maximum event thresholds.

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg::valid](#)

Indicates if the counters value is valid.

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg:: pad6](#)

reserved bit 56

unsigned [ezdp\\_watchdog\\_sliding\\_window\\_cfg::counters](#)

3/4/6/8/12/24 windows counters

union { ... }

---

The documentation for this struct was generated from the following file:

- dpe/dp/include/[ezdp\\_counter\\_defs.h](#)

# File Documentation

## dpe/dp/include/ezdp.h File Reference

### Data Structures

- struct [ezdp\\_mem\\_section\\_info](#)

### Defines

- #define [EZDP\\_MEM\\_CFG\\_USE\\_ALTER\\_CMEM](#) 0x1
- *MEM configuration flags.* #define [EZDP\\_MEM\\_CFG\\_USE\\_ALTER\\_SHARED\\_CMEM](#) 0x2
- *Indicate if alternative shared CMEM should be used.* #define [EZDP\\_MEM\\_CFG\\_IMEM\\_PRIVATE\\_DATA\\_CACHABLE](#) 0x4
- *Indicate if IMEM private memory should be cachable.* #define [EZDP\\_MEM\\_CFG\\_IMEM\\_HALF\\_CLUSTER\\_DATA\\_CACHABLE](#) 0x8
- *Indicate if IMEM half cluster memory should be cachable.* #define [EZDP\\_MEM\\_CFG\\_IMEM\\_1\\_CLUSTER\\_DATA\\_CACHABLE](#) 0x10
- *Indicate if IMEM 1 cluster memory should be cachable.* #define [EZDP\\_MEM\\_CFG\\_IMEM\\_2\\_CLUSTER\\_DATA\\_CACHABLE](#) 0x20
- *Indicate if IMEM 2 cluster memory should be cachable.* #define [EZDP\\_MEM\\_CFG\\_IMEM\\_4\\_CLUSTER\\_DATA\\_CACHABLE](#) 0x40
- *Indicate if IMEM 4 cluster memory should be cachable.* #define [EZDP\\_MEM\\_CFG\\_IMEM\\_16\\_CLUSTER\\_DATA\\_CACHABLE](#) 0x80
- *Indicate if IMEM 16 cluster memory should be cachable.* #define [EZDP\\_MEM\\_CFG\\_IMEM\\_ALL\\_CLUSTER\\_DATA\\_CACHABLE](#) 0x100
- *Indicate if IMEM all cluster memory should be cachable.* #define [EZDP\\_MEM\\_CFG\\_EMEM\\_DATA\\_CACHABLE](#) 0x200

### **Indicate if EMEM shared memory should be cachable. Typedefs**

- typedef bool(\* [EZDP\\_MEM\\_CTOR\\_FUNC](#) )(enum [ezdp\\_data\\_mem\\_space](#) data\_ms\_type, uintptr\_t user\_data)
- *Type definition for the user memory constructor handling.* typedef void(\* [EZDP\\_MAIN\\_FUNC](#) )(void)

### **Type definition for the data plane application's main frame handling loop function. Enumerations**

- enum [ezdp\\_data\\_mem\\_space](#) { [EZDP\\_CMEM\\_DATA](#), [EZDP\\_SHARED\\_CMEM\\_DATA](#), [EZDP\\_IMEM\\_PRIVATE\\_DATA](#), [EZDP\\_IMEM\\_HALF\\_CLUSTER\\_DATA](#), [EZDP\\_IMEM\\_1\\_CLUSTER\\_DATA](#), [EZDP\\_IMEM\\_2\\_CLUSTER\\_DATA](#), [EZDP\\_IMEM\\_4\\_CLUSTER\\_DATA](#), [EZDP\\_IMEM\\_16\\_CLUSTER\\_DATA](#), [EZDP\\_IMEM\\_ALL\\_CLUSTER\\_DATA](#), [EZDP\\_EMEM\\_DATA](#) }
- memory space types.*

### Functions

- uint32\_t [ezdp\\_sync\\_cp](#) (void)
- Sync until CP is up. uint32\_t [ezdp\\_init\\_global](#) (uint32\_t app\_id)
- Initialize the data-plane application, Should be called once per data-plane executable. uint32\_t [ezdp\\_init\\_local](#) (uint32\_t app\_id, int32\_t cpu\_id, [EZDP\\_MEM\\_CTOR\\_FUNC](#) mem\_ctor\_func, uintptr\_t mem\_ctor\_data, uint32\_t flags)
- Initialize the data-plane process. void [ezdp\\_run](#) ([EZDP\\_MAIN\\_FUNC](#) func, uint32\_t wait\_count)
- Run a data plane application. struct [ezdp\\_version](#) \* [ezdp\\_get\\_version](#) (void)
- Return the version information of the ezdp library. char \* [ezdp\\_get\\_err\\_msg](#) (void)
- Return string of the last error. void [ezdp\\_get\\_mem\\_section\\_info](#) (struct [ezdp\\_mem\\_section\\_info](#) \*mem\_info, uint32\_t flags)
- Return sizes of the memory sections. const char \* [ezdp\\_mem\\_section\\_info\\_str](#) (struct [ezdp\\_mem\\_section\\_info](#) \*mem\_info)

*Return printable string of the memory sizes.*

---

## Define Documentation

**#define EZDP\_MEM\_CFG\_USE\_ALTER\_CMEM 0x1**

MEM configuration flags.

Indicate if alternative private CMEM should be used. Used in order to determine the size of the CMEM section NOTE: alternative CMEM doesn't contain compile time init values

**#define EZDP\_MEM\_CFG\_USE\_ALTER\_SHARED\_CMEM 0x2**

Indicate if alternative shared CMEM should be used.

Used in order to determine the size of the CMEM section NOTE: alternative CMEM doesn't contain compile time init values

**#define EZDP\_MEM\_CFG\_IMEM\_PRIVATE\_DATA\_CACHABLE 0x4**

Indicate if IMEM private memory should be cachable.

**#define EZDP\_MEM\_CFG\_IMEM\_HALF\_CLUSTER\_DATA\_CACHABLE 0x8**

Indicate if IMEM half cluster memory should be cachable.

**#define EZDP\_MEM\_CFG\_IMEM\_1\_CLUSTER\_DATA\_CACHABLE 0x10**

Indicate if IMEM 1 cluster memory should be cachable.

**#define EZDP\_MEM\_CFG\_IMEM\_2\_CLUSTER\_DATA\_CACHABLE 0x20**

Indicate if IMEM 2 cluster memory should be cachable.

**#define EZDP\_MEM\_CFG\_IMEM\_4\_CLUSTER\_DATA\_CACHABLE 0x40**

Indicate if IMEM 4 cluster memory should be cachable.

**#define EZDP\_MEM\_CFG\_IMEM\_16\_CLUSTER\_DATA\_CACHABLE 0x80**

Indicate if IMEM 16 cluster memory should be cachable.

```
#define EZDP_MEM_CFG_IMEM_ALL_CLUSTER_DATA_CACHABLE 0x100
```

Indicate if IMEM all cluster memory should be cachable.

```
#define EZDP_MEM_CFG_EMEM_DATA_CACHABLE 0x200
```

Indicate if EMEM shared memory should be cachable.

---

## Typedef Documentation

```
typedef bool(* EZDP_MEM_CTOR_FUNC)(enum ezdp_data_mem_space data_ms_type, uintptr_t user_data)
```

Type definition for the user memory constructor handling.

```
typedef void(* EZDP_MAIN_FUNC)(void)
```

Type definition for the data plane application's main frame handling loop function.

---

## Enumeration Type Documentation

```
enum ezdp_data_mem_space
```

memory space types.

### Enumerator:

**EZDP\_CMEM\_DATA** Private CMEM data memory space.

**EZDP\_SHARED\_CMEM\_DATA** Shared CMEM data memory space.

**EZDP\_IMEM\_PRIVATE\_DATA** Thread private data memory space.

**EZDP\_IMEM\_HALF\_CLUSTER\_DATA** Sub cluster data memory space.

**EZDP\_IMEM\_1\_CLUSTER\_DATA** Single cluster data memory space.

**EZDP\_IMEM\_2\_CLUSTER\_DATA** Dual cluster data memory space.

**EZDP\_IMEM\_4\_CLUSTER\_DATA** Quad cluster data memory space.

**EZDP\_IMEM\_16\_CLUSTER\_DATA** 16 cluster data memory space.

**EZDP\_IMEM\_ALL\_CLUSTER\_DATA** All cluster data memory space.

**EZDP\_EMEM\_DATA** External memory shared data.

---

## Function Documentation

### uint32\_t ezdp\_sync\_cp (void)

Sync until CP is up.

Wait until CP is up and till memory is configured and ready to use

#### Returns:

- 0 (operation success), The function may fail for any of the errors specified for routines: open,close,iocctl In such case the errno value of the failure routine is returned Use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

### uint32\_t ezdp\_init\_global (uint32\_t app\_id)

Initialize the data-plane application, Should be called once per data-plane executable.

Used to allocate shared memory, which is later used in ezdp\_init\_local function

#### Parameters:

[in] *app\_id* - Determines the application id

#### Returns:

- 0 (operation success), The function may fail for any of the errors specified for routines: open,mmap,ftruncate,ftok,semget,semop In such case the errno value of the failure routine is returned Use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

### uint32\_t ezdp\_init\_local (uint32\_t app\_id, int32\_t cpu\_id, [EZDP\\_MEM\\_CTOR\\_FUNC](#) mem\_ctor\_func, uintptr\_t mem\_ctor\_data, uint32\_t flags)

Initialize the data-plane process.

Should be called once per data-plane process (e.g. after each fork).

Copy code and data to internal memories and configure cmem configuration and fnt mapping

#### Parameters:

[in] *app\_id* - Determines to which application id this application belong to.

[in] *cpu\_id* - Determines the logical processor to attach the process to.

[in] *mem\_ctor\_func* - Pointer constructor function. Will be called per memory space. Used to initialize the memory. Number of call to the constructor will be according to number of memory replications For example: for 4 cluster memory, construct will be called 4 types to allow initialize memory in in each quad cluster

[in] *mem\_ctor\_data* - Data to provide to construct function

[in] *flags* - Bitwise OR of one or more of the EZDP\_MEM\_CFG\_\* flags defined above

#### Note:

mem\_ctor\_func should be one for all cpus of the applications and initialize all application memories

#### Returns:

- 0 (operation success), EINVAL (illegal/invalid configuration), ENODEV (ezdp\_init\_global was not called) EPROCUNAVAIL (data-plane process already initialize) The function may also fail for any of the errors specified for routines: open, iocctl, close, malloc, mmap, mprotect, sched\_setaffinity, semop. In such case the errno value of the failure routine is returned Use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

**void ezdp\_run ([EZDP\\_MAIN\\_FUNC](#) func, uint32\_t wait\_count)**

Run a data plane application.

The function receives the data plane application's main frame handling loop function as a parameter. It performs the required preparation for the data plane application and then executes the passed function.

**Parameters:**

- [in] *func* - Data plane application function to run.
- [in] *wait\_count* - Number of additional threads to wait for till starting the execution of data plane application function

**Returns:**

none

**struct [ezdp\\_version](#)\* ezdp\_get\_version (void) [read]**

Return the version information of the ezdp library.

Use `ezdp_version_get_string` to get the info string.

**Returns:**

The pointer to the version info of the ezdp library.

**char\* ezdp\_get\_err\_msg (void)**

Return string of the last error.

Returns pointer to string. This string provide more details about the error returns by the functions which return error.

**Returns:**

char\* - The pointer to the error message string

**void ezdp\_get\_mem\_section\_info (struct [ezdp\\_mem\\_section\\_info](#) \* mem\_info, uint32\_t flags)**

Return sizes of the memory sections.

**Parameters:**

- [out] *mem\_info* - Pointer to [ezdp\\_mem\\_section\\_info](#) to write respond
- [in] *flags* - Bitwise OR of one or more of the `EZDP_MEM_CFG_*` flags defined above

**Returns:**

void

**const char\* ezdp\_mem\_section\_info\_str (struct [ezdp\\_mem\\_section\\_info](#) \* mem\_info)**

Return printable string of the memory sizes.

**Parameters:**

- [in] *mem\_info* - Pointer to [ezdp\\_mem\\_section\\_info](#) to convert to string format

**Returns:**

point to string



## dpe/dp/include/ezdp\_atomic.h File Reference

### Functions

- static `__always_inline uint32_t ezdp_atomic_read8_ext_addr` (struct `ezdp_ext_addr` \*addr)
- Atomically read an 8 bit value from an extended address. static `__always_inline uint32_t ezdp_atomic_read16_ext_addr` (struct `ezdp_ext_addr` \*addr)
- Atomically read a 16 bit value from an extended address. static `__always_inline uint32_t ezdp_atomic_read32_ext_addr` (struct `ezdp_ext_addr` \*addr)
- Atomically read a 32 bit value from an extended address. static `__always_inline uint32_t ezdp_atomic_read32_sum_addr` (ezdp\_sum\_addr\_t addr)
- Atomically read a 32 bit value from a summarized address. static `__always_inline void ezdp_atomic_read32_sum_addr_async` (ezdp\_sum\_addr\_t addr, uint32\_t \_\_cmem \*value)
- Non-blocking/posted version of `ezdp_atomic_read32_sum_addr()`. static `__always_inline uint32_t ezdp_atomic_read64_sum_addr` (ezdp\_sum\_addr\_t addr, uint64\_t \_\_cmem \*value)
- Atomically read a 64 bit value from a summarized address. static `__always_inline void ezdp_atomic_read64_sum_addr_async` (ezdp\_sum\_addr\_t addr, uint64\_t \_\_cmem \*value)
- Non-blocking/posted version of `ezdp_atomic_read64_sum_addr()`. static `__always_inline void ezdp_atomic_write8_ext_addr` (struct `ezdp_ext_addr` \*addr, uint8\_t value)
- Atomically write an 8 bit value to an extended address. static `__always_inline void ezdp_atomic_write8_ext_addr_async` (struct `ezdp_ext_addr` \*addr, uint8\_t value)
- Non-blocking/posted version of `ezdp_atomic_write8_ext_addr()`. static `__always_inline void ezdp_atomic_write16_ext_addr` (struct `ezdp_ext_addr` \*addr, uint16\_t value)
- Atomically write a 16 bit value to an extended address. static `__always_inline void ezdp_atomic_write16_ext_addr_async` (struct `ezdp_ext_addr` \*addr, uint16\_t value)
- Non-blocking/posted version of `ezdp_atomic_write16_ext_addr()`. static `__always_inline void ezdp_atomic_write32_ext_addr` (struct `ezdp_ext_addr` \*addr, uint32\_t value)
- Atomically write a 32 bit value to an extended address. static `__always_inline void ezdp_atomic_write32_ext_addr_async` (struct `ezdp_ext_addr` \*addr, uint32\_t value)
- Non-blocking/posted version of `ezdp_atomic_write16_ext_addr()`. static `__always_inline void ezdp_atomic_write32_sum_addr` (ezdp\_sum\_addr\_t addr, uint32\_t value)
- Atomically write a 32 bit value to a summarized address. static `__always_inline void ezdp_atomic_write32_sum_addr_async` (ezdp\_sum\_addr\_t addr, uint32\_t value)
- Non-blocking/posted version of `ezdp_atomic_write16_sum_addr()`. static `__always_inline void ezdp_atomic_write64_sum_addr` (ezdp\_sum\_addr\_t addr, uint64\_t value)
- Atomically write a 64 bit value to a summarized address. static `__always_inline void ezdp_atomic_write64_sum_addr_async` (ezdp\_sum\_addr\_t addr, uint64\_t value)
- Non-blocking/posted version of `ezdp_atomic_write64_sum_addr()`. static `__always_inline uint32_t ezdp_atomic_xchg32_ext_addr` (struct `ezdp_ext_addr` \*addr, uint32\_t value)
- Atomically exchange a 32 bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_xchg32_sum_addr` (ezdp\_sum\_addr\_t addr, uint32\_t value)
- Atomically exchange a 32 bit value in a summarized address. static `__always_inline uint32_t ezdp_atomic_cmpxchg8_ext_addr` (uint8\_t compare\_value, struct `ezdp_ext_addr` \*addr, uint8\_t value)
- Atomically compare and exchange an 8 bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_cmpxchg16_ext_addr` (uint16\_t compare\_value, struct `ezdp_ext_addr` \*addr, uint16\_t value)
- Atomically compare and exchange a 16 bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_cmpxchg32_ext_addr` (uint32\_t compare\_value, struct `ezdp_ext_addr` \*addr, uint32\_t value)
- Atomically compare and exchange a 32 bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_cmpxchg32_sum_addr` (uint32\_t compare\_value, ezdp\_sum\_addr\_t addr, uint32\_t value)
- Atomically compare and exchange a 32 bit value in a summarized address. static `__always_inline uint32_t ezdp_atomic_read_and_tst8_ext_addr` (struct `ezdp_ext_addr` \*addr, bool \*fail\_flag)
- Atomic read, test and set an 8 bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_read_and_tst16_ext_addr` (struct `ezdp_ext_addr` \*addr, bool \*fail\_flag)
- Atomic read, test and set a 16 bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_read_and_tst32_ext_addr` (struct `ezdp_ext_addr` \*addr, bool \*fail\_flag)
- Atomic read, test and set a 32 bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_read_and_tst32_sum_addr` (ezdp\_sum\_addr\_t addr, bool \*fail\_flag)

- Atomic read, test and set a 32 bit value in a summarized address. static `__always_inline uint32_t ezdp_atomic_read_and_clear8_ext_addr` (struct `ezdp_ext_addr` \*addr)
- Atomically read and clear an 8-bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_read_and_clear16_ext_addr` (struct `ezdp_ext_addr` \*addr)
- Atomically read and clear a 16-bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_read_and_clear32_ext_addr` (struct `ezdp_ext_addr` \*addr)
- Atomically read and clear a 32-bit value in an extended address. static `__always_inline uint32_t ezdp_atomic_read_and_clear32_sum_addr` (ezdp\_sum\_addr\_t addr)
- Atomically read and clear a 32-bit value in a summarized address. static `__always_inline uint32_t ezdp_atomic_read_and_clear64_sum_addr` (ezdp\_sum\_addr\_t addr, uint64\_t \_\_cmem \*orig\_value)
- Atomically read and clear a 64-bit value in a summarized address. static `__always_inline void ezdp_atomic_add8_ext_addr` (struct `ezdp_ext_addr` \*addr, int8\_t value)
- Atomically perform an 8 bit logical ADD operation on an extended address. static `__always_inline void ezdp_atomic_add8_ext_addr_async` (struct `ezdp_ext_addr` \*addr, int8\_t value)
- Non-blocking/posted version of `ezdp_atomic_add8_ext_addr()`. static `__always_inline int32_t ezdp_atomic_read_and_add8_ext_addr` (struct `ezdp_ext_addr` \*addr, int8\_t value, bool \*overflow\_flag)
- Atomically read and perform an 8 bit logical ADD operation on an extended address. static `__always_inline void ezdp_atomic_add16_ext_addr` (struct `ezdp_ext_addr` \*addr, int16\_t value)
- Atomically perform a 16 bit logical ADD operation on an extended address. static `__always_inline void ezdp_atomic_add16_ext_addr_async` (struct `ezdp_ext_addr` \*addr, int16\_t value)
- Non-blocking/posted version of `ezdp_atomic_add16_ext_addr()`. static `__always_inline int32_t ezdp_atomic_read_and_add16_ext_addr` (struct `ezdp_ext_addr` \*addr, int16\_t value, bool \*overflow\_flag)
- Atomically read and perform a 16 bit logical ADD operation on an extended address. static `__always_inline void ezdp_atomic_add32_ext_addr` (struct `ezdp_ext_addr` \*addr, int32\_t value)
- Atomically perform a 32 bit logical ADD operation on an extended address. static `__always_inline void ezdp_atomic_add32_ext_addr_async` (struct `ezdp_ext_addr` \*addr, int32\_t value)
- Non-blocking/posted version of `ezdp_atomic_add32_ext_addr()`. static `__always_inline int32_t ezdp_atomic_read_and_add32_ext_addr` (struct `ezdp_ext_addr` \*addr, int32\_t value, bool \*overflow\_flag)
- Atomically read and perform a 32 bit logical ADD operation on an extended address. static `__always_inline void ezdp_atomic_add32_sum_addr` (ezdp\_sum\_addr\_t addr, int32\_t value)
- Atomically perform a 32 bit logical ADD operation on a summarized address. static `__always_inline void ezdp_atomic_add32_sum_addr_async` (ezdp\_sum\_addr\_t addr, int32\_t value)
- Non-blocking/posted version of `ezdp_atomic_add32_sum_addr()`. static `__always_inline int32_t ezdp_atomic_read_and_add32_sum_addr` (ezdp\_sum\_addr\_t addr, int32\_t value, bool \*overflow\_flag)
- Atomically read and perform a 32 bit logical ADD operation on a summarized address. static `__always_inline void ezdp_atomic_add64_sum_addr` (ezdp\_sum\_addr\_t addr, int32\_t value)
- Atomically perform a 64 bit logical ADD operation on a summarized address. static `__always_inline void ezdp_atomic_add64_sum_addr_async` (ezdp\_sum\_addr\_t addr, int32\_t value)
- Non-blocking/posted version of `ezdp_atomic_add64_sum_addr()`. static `__always_inline int32_t ezdp_atomic_read_and_add64_sum_addr` (ezdp\_sum\_addr\_t addr, int32\_t value, int64\_t \_\_cmem \*orig\_value, bool \*overflow\_flag)
- Atomically read and perform a 64 bit logical ADD operation on a summarized address. static `__always_inline void ezdp_atomic_dual_add32_ext_addr` (struct `ezdp_ext_addr` \*addr, int16\_t value1, int16\_t value2)
- Atomically perform a dual ADD operation to the two 32-bit variables pointed to by the extended address. static `__always_inline void ezdp_atomic_dual_add32_ext_addr_async` (struct `ezdp_ext_addr` \*addr, int16\_t value1, int16\_t value2)
- Non-blocking/posted version of `ezdp_atomic_dual_add32_ext_addr()`. static `__always_inline void ezdp_atomic_dual_add32_sum_addr` (ezdp\_sum\_addr\_t addr, int16\_t value1, int16\_t value2)
- Atomically perform a dual ADD operation to the two 32-bit variables pointed to by the summarized address. static `__always_inline void ezdp_atomic_dual_add32_sum_addr_async` (ezdp\_sum\_addr\_t addr, int16\_t value1, int16\_t value2)
- Non-blocking/posted version of `ezdp_atomic_dual_add32_sum_addr()`. static `__always_inline int32_t ezdp_atomic_read_and_dual_add32_sum_addr` (ezdp\_sum\_addr\_t addr, int16\_t value1, int16\_t value2, struct `ezdp_dual_add32_result` \_\_cmem \*orig\_value, bool \*overflow\_flag)
- Atomically read and perform a dual ADD operation to the two 32-bit variables pointed to by the summarized address. static `__always_inline void ezdp_atomic_dual_add64_sum_addr` (ezdp\_sum\_addr\_t addr, int16\_t value1, int16\_t value2)

- Atomically perform dual ADD operation to the two 64-bit variables pointed to by the summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_dual\\_add64\\_sum\\_addr\\_async](#) (ezdp\_sum\_addr\_t addr, int16\_t value1, int16\_t value2)
- Non-blocking/posted version of [ezdp\\_atomic\\_dual\\_add64\\_sum\\_addr\(\)](#). static \_\_always\_inline int32\_t [ezdp\\_atomic\\_read\\_and\\_dual\\_add64\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, int16\_t value1, int16\_t value2, struct [ezdp\\_dual\\_add64\\_result](#) \_\_cmem \*orig\_value, bool \*overflow\_flag)
- Atomically read and perform a dual ADD operation to the two 64-bit variables pointed to by the summarized address. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc8\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, bool \*zero\_flag)
- Atomically read and increment an 8 bit value in an extended address. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc16\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, bool \*zero\_flag)
- Atomically read and increment a 16 bit value in an extended address. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc32\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, bool \*zero\_flag)
- Atomically read and increment a 32 bit value in an extended address. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc32\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, bool \*zero\_flag)
- Atomically read and increment a 32 bit value in a summarized address. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc64\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint64\_t \_\_cmem \*orig\_value, bool \*zero\_flag)
- Atomically read and increment a 64 bit value in a summarized address. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_dec8\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, bool \*unaffected\_flag)
- Atomically read and decrement conditionally by one an 8-bit value in an extended address (zero value does not underflow). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_dec16\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, bool \*unaffected\_flag)
- Atomically read and decrement conditionally by one a 16-bit value in an extended address (zero value does not underflow). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_dec32\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, bool \*unaffected\_flag)
- Atomically read and decrement conditionally by one a 32-bit value in an extended address (zero value does not underflow). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_dec32\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, bool \*unaffected\_flag)
- Atomically read and decrement conditionally by one a 32-bit value in a summarized address (zero value does not underflow). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_dec64\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint64\_t \_\_cmem \*orig\_value, bool \*unaffected\_flag)
- Atomically read and decrement conditionally by one a 64-bit value in a summarized address (zero value does not underflow). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc32\\_cond\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, bool res\_mode, bool \*unaffected\_flag)
- Atomically read and increment conditionally a 32-bit value in an extended address. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc32\\_cond\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, bool res\_mode, bool \*unaffected\_flag)
- Atomically read and increment conditionally a 32-bit value in a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_and8\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint8\_t value)
- Atomically perform an 8 bit logical AND operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_and8\\_ext\\_addr\\_async](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint8\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_and8\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_and8\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint8\_t value, bool \*no\_chng\_flag)
- Atomically read and perform an 8 bit logical AND operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_and16\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint16\_t value)
- Atomically perform a 16 bit logical AND operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_and16\\_ext\\_addr\\_async](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint16\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_and16\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_and16\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint16\_t value, bool \*no\_chng\_flag)
- Atomically read and perform a 16 bit logical AND operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_and32\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint32\_t value)
- Atomically perform a 32 bit logical AND operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_and32\\_ext\\_addr\\_async](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint32\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_and32\\_ext\\_addr\\_async\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_and32\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, uint32\_t value, bool \*no\_chng\_flag)
- Atomically read and perform a 32 bit logical AND operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_and32\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint32\_t value)

- Atomically perform a 32 bit logical AND operation on a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_and32\\_sum\\_addr\\_async](#) (ezdp\_sum\_addr\_t addr, uint32\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_and32\\_sum\\_addr\\_async\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_and32\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint32\_t value, bool \*no\_chng\_flag)
- Atomically read and perform a 32 bit logical AND operation on a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_or8\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint8\_t value)
- Atomically perform an 8 bit logical OR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_or8\\_ext\\_addr\\_async](#) (struct ezdp\_ext\_addr \*addr, uint8\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_or8\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_or8\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint8\_t value, bool \*no\_chng\_flag)
- Atomically read and perform an 8 bit logical OR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_or8\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint8\_t value)
- Atomically perform an 8 bit logical OR operation on a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_or8\\_sum\\_addr\\_async](#) (ezdp\_sum\_addr\_t addr, uint8\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_or8\\_sum\\_addr\(\)](#). static \_\_always\_inline void [ezdp\\_atomic\\_or16\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint16\_t value)
- Atomically perform a 16 bit logical OR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_or16\\_ext\\_addr\\_async](#) (struct ezdp\_ext\_addr \*addr, uint16\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_or16\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_or16\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint16\_t value, bool \*no\_chng\_flag)
- Atomically read and perform a 16 bit logical OR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_or16\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint16\_t value)
- Atomically perform a 16 bit logical OR operation on a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_or16\\_sum\\_addr\\_async](#) (ezdp\_sum\_addr\_t addr, uint16\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_or16\\_sum\\_addr\(\)](#). static \_\_always\_inline void [ezdp\\_atomic\\_or32\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint32\_t value)
- Atomically perform a 32 bit logical OR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_or32\\_ext\\_addr\\_async](#) (struct ezdp\_ext\_addr \*addr, uint32\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_or32\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_or32\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint32\_t value, bool \*no\_chng\_flag)
- Atomically read and perform a 32 bit logical OR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_or32\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint32\_t value)
- Atomically perform a 32 bit logical OR operation on a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_or32\\_sum\\_addr\\_async](#) (ezdp\_sum\_addr\_t addr, uint32\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_or32\\_sum\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_or32\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint32\_t value, bool \*no\_chng\_flag)
- Atomically read and perform a 32 bit logical OR operation on a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_xor8\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint8\_t value)
- Atomically perform an 8 bit logical XOR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_xor8\\_ext\\_addr\\_async](#) (struct ezdp\_ext\_addr \*addr, uint8\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_xor8\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_xor8\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint8\_t value, bool \*no\_id\_flag)
- Atomically read and perform an 8 bit logical XOR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_xor16\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint16\_t value)
- Atomically perform a 16 bit logical XOR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_xor16\\_ext\\_addr\\_async](#) (struct ezdp\_ext\_addr \*addr, uint16\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_xor8\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_xor16\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint16\_t value, bool \*no\_id\_flag)
- Atomically read and perform a 16 bit logical XOR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_xor32\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint32\_t value)
- Atomically perform a 32 bit logical XOR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_xor32\\_ext\\_addr\\_async](#) (struct ezdp\_ext\_addr \*addr, uint32\_t value)
- Non-blocking/posted version of [ezdp\\_atomic\\_xor32\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_xor32\\_ext\\_addr](#) (struct ezdp\_ext\_addr \*addr, uint32\_t value, bool \*no\_id\_flag)
- Atomically read and perform a 32 bit logical XOR operation on an extended address. static \_\_always\_inline void [ezdp\\_atomic\\_xor32\\_sum\\_addr](#) (ezdp\_sum\_addr\_t addr, uint32\_t value)
- Atomically perform a 32 bit logical XOR operation on a summarized address. static \_\_always\_inline void [ezdp\\_atomic\\_xor32\\_sum\\_addr\\_async](#) (ezdp\_sum\_addr\_t addr, uint32\_t value)



- Non-blocking/posted version of [ezdp\\_atomic\\_xor32\\_sum\\_addr\(\)](#). static `__always_inline uint32_t ezdp_atomic_read_and_xor32_sum_addr(ezdp_sum_addr_t addr, uint32_t value, bool *no_id_flag)`  
Atomically read and perform a 32 bit logical XOR operation on a summarized address.
- 

## Function Documentation

**static \_\_always\_inline uint32\_t ezdp\_atomic\_read8\_ext\_addr (struct [ezdp\\_ext\\_addr](#) \* addr)**  
[static]

Atomically read an 8 bit value from an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

**Returns:**

Read value. The value is limited to 8 bits

**static \_\_always\_inline uint32\_t ezdp\_atomic\_read16\_ext\_addr (struct [ezdp\\_ext\\_addr](#) \* addr)**  
[static]

Atomically read a 16 bit value from an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

**Note:**

Address must be 2-byte aligned.

**Returns:**

Read value. The value is limited to 16 bits

**static \_\_always\_inline uint32\_t ezdp\_atomic\_read32\_ext\_addr (struct [ezdp\\_ext\\_addr](#) \* addr)**  
[static]

Atomically read a 32 bit value from an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

**Note:**

Address must be 4-byte aligned.

**Returns:**

Read value

**static \_\_always\_inline uint32\_t ezdp\_atomic\_read32\_sum\_addr ([ezdp\\_sum\\_addr\\_t](#) addr)**  
[static]

Atomically read a 32 bit value from a summarized address.

**Parameters:**

[in] *addr* - summarized address

**Returns:**

Read value

```
static __always_inline void ezdp_atomic_read32_sum_addr_async (ezdp\_sum\_addr\_t addr,
uint32_t __cmem * value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_read32\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address

[out] *value* - pointer in CMEM to write the value in

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read64_sum_addr (ezdp\_sum\_addr\_t addr,
uint64_t __cmem * value) [static]
```

Atomically read a 64 bit value from a summarized address.

**Parameters:**

[in] *addr* - summarized address

[out] *value* - pointer in CMEM to write the value in

**Returns:**

The 32 MSB of the read value

```
static __always_inline void ezdp_atomic_read64_sum_addr_async (ezdp\_sum\_addr\_t addr,
uint64_t __cmem * value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_read64\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address

[out] *value* - pointer in CMEM to write the value in

**Returns:**

void

```
static __always_inline void ezdp_atomic_write8_ext_addr (struct ezdp\_ext\_addr * addr,  uint8_t
value) [static]
```

Atomically write an 8 bit value to an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - write value

**Returns:**

void

```
static __always_inline void ezdp_atomic_write8_ext_addr_async (struct ezdp\_ext\_addr * addr,  
uint8_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_write8\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - write value

**Returns:**

void

```
static __always_inline void ezdp_atomic_write16_ext_addr (struct ezdp\_ext\_addr * addr,  
uint16_t value) [static]
```

Atomically write a 16 bit value to an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - write value

**Note:**

Address must be 2-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_write16_ext_addr_async (struct ezdp\_ext\_addr * addr,  
uint16_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_write16\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - write value

**Note:**

Address must be 2-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_write32_ext_addr (struct ezdp\_ext\_addr * addr,  
uint32_t value) [static]
```

Atomically write a 32 bit value to an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - write value

**Note:**

Address must be 4-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_write32_ext_addr_async (struct ezdp\_ext\_addr * addr,  
uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_write16\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - write value

**Note:**

Address must be 4-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_write32_sum_addr (ezdp\_sum\_addr\_t addr, uint32_t  
value) [static]
```

Atomically write a 32 bit value to a summarized address.

**Parameters:**

[in] *addr* - summarized address

[in] *value* - write value

**Returns:**

void

```
static __always_inline void ezdp_atomic_write32_sum_addr_async (ezdp\_sum\_addr\_t addr,  
uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_write16\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address

[in] *value* - write value

**Returns:**

void

```
static __always_inline void ezdp_atomic_write64_sum_addr (ezdp\_sum\_addr\_t addr, uint64_t  
value) [static]
```

Atomically write a 64 bit value to a summarized address.

**Parameters:**

[in] *addr* - summarized address

[in] *value* - write value

**Returns:**

void



```
static __always_inline void ezdp_atomic_write64_sum_addr_async (ezdp\_sum\_addr\_t addr,
uint64_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_write64\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address  
[in] *value* - write value

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_xchg32_ext_addr (struct ezdp\_ext\_addr * addr,
uint32_t value) [static]
```

Atomically exchange a 32 bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - value to exchange

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original value.

```
static __always_inline uint32_t ezdp_atomic_xchg32_sum_addr (ezdp\_sum\_addr\_t addr,
uint32_t value) [static]
```

Atomically exchange a 32 bit value in a summarized address.

**Parameters:**

[in] *addr* - summarized address  
[in] *value* - value to exchange

**Returns:**

Original value

```
static __always_inline uint32_t ezdp_atomic_cmpxchg8_ext_addr (uint8_t compare_value,
struct ezdp\_ext\_addr * addr, uint8_t value) [static]
```

Atomically compare and exchange an 8 bit value in an extended address.

Compares the passed value with the existing value in memory, and exchanges the values only if they are equal.

**Parameters:**

[in] *compare\_value* - value to compare before the change  
[in] *addr* - pointer to extended address  
[in] *value* - value to exchange

**Returns:**

Original value The return value is limited to 8 bits

```
static __always_inline uint32_t ezdp_atomic_cmpxchg16_ext_addr (uint16_t compare_value,
struct ezdp\_ext\_addr * addr,   uint16_t value) [static]
```

Atomically compare and exchange a 16 bit value in an extended address.

Compares the passed value with the existing value in memory, and exchanges the values only if they are equal.

**Parameters:**

- [in] *compare\_value* - value to compare before the change
- [in] *addr* - pointer to extended address
- [in] *value* - value to exchange

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original value The return value is limited to 16 bits

```
static __always_inline uint32_t ezdp_atomic_cmpxchg32_ext_addr (uint32_t compare_value,
struct ezdp\_ext\_addr * addr,   uint32_t value) [static]
```

Atomically compare and exchange a 32 bit value in an extended address.

Compares the passed value with the existing value in memory, and exchanges the values only if they are equal.

**Parameters:**

- [in] *compare\_value* - value to compare before the change
- [in] *addr* - pointer to extended address
- [in] *value* - value to exchange

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original value

```
static __always_inline uint32_t ezdp_atomic_cmpxchg32_sum_addr (uint32_t compare_value,
ezdp\_sum\_addr t addr,   uint32_t value) [static]
```

Atomically compare and exchange a 32 bit value in a summarized address.

Compares the passed value with the existing value in memory, and exchanges the values only if they are equal.

**Parameters:**

- [in] *compare\_value* - value to compare before the change
- [in] *addr* - summarized address
- [in] *value* - value to exchange

**Returns:**

Original value

```
static __always_inline uint32_t ezdp_atomic_read_and_tst8_ext_addr (struct ezdp\_ext\_addr *
addr,   bool * fail_flag) [static]
```

Atomic read, test and set an 8 bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
 [out] *fail\_flag* - will be true if failed in setting the bits

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_tst16_ext_addr (struct ezdp\_ext\_addr *  

addr, bool * fail_flag) [static]
```

Atomic read, test and set a 16 bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
 [out] *fail\_flag* - will be true if failed in setting the bits

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_tst32_ext_addr (struct ezdp\_ext\_addr *  

addr, bool * fail_flag) [static]
```

Atomic read, test and set a 32 bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
 [out] *fail\_flag* - will be true if failed in setting the bits

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_tst32_sum_addr (ezdp\_sum\_addr\_t addr,  

bool * fail_flag) [static]
```

Atomic read, test and set a 32 bit value in a summarized address.

**Parameters:**

[in] *addr* - summarized address  
 [out] *fail\_flag* - will be true if failed in setting the bits

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_clear8_ext_addr (struct ezdp\_ext\_addr *  

addr) [static]
```

Atomically read and clear an 8-bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_clear16_ext_addr (struct ezdp\_ext\_addr *  
addr) [static]
```

Atomically read and clear a 16-bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_clear32_ext_addr (struct ezdp\_ext\_addr *  
addr) [static]
```

Atomically read and clear a 32-bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_clear32_sum_addr (ezdp\_sum\_addr\_t addr,  
addr) [static]
```

Atomically read and clear a 32-bit value in a summarized address.

**Parameters:**

[in] *addr* - summarized address

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_clear64_sum_addr (ezdp\_sum\_addr\_t addr,  
addr, uint64_t __cmem * orig_value) [static]
```

Atomically read and clear a 64-bit value in a summarized address.

**Parameters:**

[in] *addr* - summarized address

[out] *orig\_value* - the address in CMEM to write the original value in

**Returns:**

The 32 MSB of the original/read value

```
static __always_inline void ezdp_atomic_add8_ext_addr (struct ezdp\_ext\_addr * addr, int8_t value) [static]
```

Atomically perform an 8 bit logical ADD operation on an extended address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - value to add

**Returns:**

void

```
static __always_inline void ezdp_atomic_add8_ext_addr_async (struct ezdp\_ext\_addr * addr, int8_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_add8\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - value to add

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline int32_t ezdp_atomic_read_and_add8_ext_addr (struct ezdp\_ext\_addr * addr, int8_t value, bool * overflow_flag) [static]
```

Atomically read and perform an 8 bit logical ADD operation on an extended address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - value to add

[out] *overflow\_flag* - will be true if an overflow occurred An overflow on signed add is defined as changing originally positive memory contents to be negative when adding a positive number, or changing originally negative memory contents to be positive when adding a negative number

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline void ezdp_atomic_add16_ext_addr (struct ezdp\_ext\_addr * addr, int16_t value) [static]
```

Atomically perform a 16 bit logical ADD operation on an extended address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value

**Returns:**

void

```
static __always_inline void ezdp_atomic_add16_ext_addr_async (struct ezdp\_ext\_addr * addr,
int16_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_add16\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - value to add

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline int32_t ezdp_atomic_read_and_add16_ext_addr (struct ezdp\_ext\_addr *
addr, int16_t value, bool * overflow_flag) [static]
```

Atomically read and perform a 16 bit logical ADD operation on an extended address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value  
 [out] *overflow\_flag* - will be true if an overflow occurred An overflow on signed add is defined as changing originally positive memory contents to be negative when adding a positive number, or changing originally negative memory contents to be positive when adding a negative number

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline void ezdp_atomic_add32_ext_addr (struct ezdp\_ext\_addr * addr, int32_t
value) [static]
```

Atomically perform a 32 bit logical ADD operation on an extended address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - value to add

**Returns:**

void

```
static __always_inline void ezdp_atomic_add32_ext_addr_async (struct ezdp\_ext\_addr * addr,
int32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_add32\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - value to add

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline int32_t ezdp_atomic_read_and_add32_ext_addr (struct ezdp\_ext\_addr *  
addr, int32_t value, bool * overflow_flag) [static]
```

Atomically read and perform a 32 bit logical ADD operation on an extended address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - value to add  
 [out] *overflow\_flag* - will be true if an overflow occurred An overflow on signed add is defined as changing originally positive memory contents to be negative when adding a positive number, or changing originally negative memory contents to be positive when adding a negative number

**Returns:**

Original/Read value.

```
static __always_inline void ezdp_atomic_add32_sum_addr (ezdp\_sum\_addr\_t addr, int32_t  
value) [static]
```

Atomically perform a 32 bit logical ADD operation on a summarized address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address  
 [in] *value* - value to add

**Returns:**

void

```
static __always_inline void ezdp_atomic_add32_sum_addr_async (ezdp\_sum\_addr\_t addr,  
int32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_add32\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address  
 [in] *value* - value to add

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline int32_t ezdp_atomic_read_and_add32_sum_addr (ezdp\_sum\_addr\_t addr,
int32_t value,    bool * overflow_flag) [static]
```

Atomically read and perform a 32 bit logical ADD operation on a summarized address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address

[in] *value* - value to add

[out] *overflow\_flag* - will be true if an overflow occurred An overflow on signed add is defined as changing originally positive memory contents to be negative when adding a positive number, or changing originally negative memory contents to be positive when adding a negative number

**Returns:**

Original/Read value.

```
static __always_inline void ezdp_atomic_add64_sum_addr (ezdp\_sum\_addr\_t addr,    int32_t
value) [static]
```

Atomically perform a 64 bit logical ADD operation on a summarized address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address

[in] *value* - value to add

**Returns:**

void

```
static __always_inline void ezdp_atomic_add64_sum_addr_async (ezdp\_sum\_addr\_t addr,
int32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_add64\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address

[in] *value* - value to add

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline int32_t ezdp_atomic_read_and_add64_sum_addr (ezdp\_sum\_addr\_t addr,
int32_t value,    int64_t __cmem * orig_value,    bool * overflow_flag) [static]
```

Atomically read and perform a 64 bit logical ADD operation on a summarized address.

Performs a logical ADD operation between the existing value in memory and the value passed, and stores the result back into memory.



**Parameters:**

[in] *addr* - summarized address  
 [in] *value* - value to add  
 [out] *orig\_value* - the address in CMEM to write the original value in  
 [out] *overflow\_flag* - will be true if an overflow occurred An overflow on signed add is defined as changing originally positive memory contents to be negative when adding a positive number, or changing originally negative memory contents to be positive when adding a negative number

**Returns:**

The 32 MSB of the original/read value

```
static __always_inline void ezdp_atomic_dual_add32_ext_addr (struct ezdp\_ext\_addr * addr,
int16_t value1,    int16_t value2) [static]
```

Atomically perform a dual ADD operation to the two 32-bit variables pointed to by the extended address.

Performs a dual ADD operation between the existing 2 values, 32 bits each, in memory and the 2 values passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value1* - value1 to add to variable 1  
 [in] *value2* - value2 to add to variable 2

**Returns:**

void

```
static __always_inline void ezdp_atomic_dual_add32_ext_addr_async (struct ezdp\_ext\_addr *
addr,    int16_t value1,    int16_t value2) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_dual\\_add32\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value1* - value1 to add to variable 1  
 [in] *value2* - value2 to add to variable 2

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_atomic_dual_add32_sum_addr (ezdp\_sum\_addr\_t addr,
int16_t value1,    int16_t value2) [static]
```

Atomically perform a dual ADD operation to the two 32-bit variables pointed to by the summarized address.

Performs a dual ADD operation between the existing 2 values, 32 bits each, in memory and the 2 values passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address  
 [in] *value1* - value1 to add to variable 1  
 [in] *value2* - value2 to add to variable 2

**Returns:**

void

```
static __always_inline void ezdp_atomic_dual_add32_sum_addr_async (ezdp\_sum\_addr\_t addr,
int16_t value1, int16_t value2) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_dual\\_add32\\_sum\\_addr\(\)](#).

**Parameters:**

- [in] *addr* - summarized address
- [in] *value1* - value1 to add to variable 1
- [in] *value2* - value2 to add to variable 2

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline int32_t ezdp_atomic_read_and_dual_add32_sum_addr (ezdp\_sum\_addr\_t
addr, int16_t value1, int16_t value2, struct ezdp\_dual\_add32\_result __cmem * orig_value,
bool * overflow_flag) [static]
```

Atomically read and perform a dual ADD operation to the two 32-bit variables pointed to by the summarized address.

Performs a dual ADD operation between the existing 2 values, 32 bits each, in memory and the 2 values passed, and stores the result back into memory.

**Parameters:**

- [in] *addr* - summarized address
- [in] *value1* - value1 to add to variable 1
- [in] *value2* - value2 to add to variable 2
- [out] *orig\_value* - pointer in CMEM to write both original values to
- [out] *overflow\_flag* - will be true if an overflow occurred in at least one counter An overflow on signed add is defined as changing originally positive memory contents to be negative when adding a positive number, or changing originally negative memory contents to be positive when adding a negative number

**Returns:**

Original value of variable 1

```
static __always_inline void ezdp_atomic_dual_add64_sum_addr (ezdp\_sum\_addr\_t addr,
int16_t value1, int16_t value2) [static]
```

Atomically perform dual ADD operation to the two 64-bit variables pointed to by the summarized address.

Performs dual ADD operation between the existing 2 values, 64 bit each, in memory and the 2 values passed, and stores the result back into memory.

**Parameters:**

- [in] *addr* - summarized address
- [in] *value1* - value1 to add to variable 1
- [in] *value2* - value2 to add to variable 1

**Returns:**

void

```
static __always_inline void ezdp_atomic_dual_add64_sum_addr_async (ezdp\_sum\_addr\_t addr,
int16_t value1, int16_t value2) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_dual\\_add64\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address  
 [in] *value1* - value1 to add to variable 1  
 [in] *value2* - value2 to add to variable 2

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline int32_t ezdp_atomic_read_and_dual_add64_sum_addr (ezdp\_sum\_addr\_t
addr, int16\_t value1, int16\_t value2, struct ezdp\_dual\_add64\_result __cmem * orig_value,
bool * overflow_flag) [static]
```

Atomically read and perform a dual ADD operation to the two 64-bit variables pointed to by the summarized address.

Performs a dual ADD operation between the existing 2 values, 64 bits each, in memory and the 2 values passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address  
 [in] *value1* - value1 to add to variable 1  
 [in] *value2* - value2 to add to variable 1  
 [out] *orig\_value* - pointer in CMEM to write both original values to  
 [out] *overflow\_flag* - will be true if an overflow occurred in at least one counter An overflow on signed add is defined as changing originally positive memory contents to be negative when adding a positive number, or changing originally negative memory contents to be positive when adding a negative number

**Returns:**

32 MSB original/read value of variable 2

```
static __always_inline uint32_t ezdp_atomic_read_and_inc8_ext_addr (struct ezdp\_ext\_addr *
addr, bool * zero_flag) [static]
```

Atomically read and increment an 8 bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
 [out] *zero\_flag* - will be true if the memory content was rolled back to zero

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_inc16_ext_addr (struct ezdp\_ext\_addr *
addr, bool * zero_flag) [static]
```

Atomically read and increment a 16 bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
 [out] *zero\_flag* - will be true if the memory content was rolled back to zero

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_inc32_ext_addr (struct ezdp\_ext\_addr *  
addr, bool * zero_flag) [static]
```

Atomically read and increment a 32 bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address

[out] *zero\_flag* - will be true if the memory content was rolled back to zero

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_inc32_sum_addr (ezdp\_sum\_addr\_t addr,  
bool * zero_flag) [static]
```

Atomically read and increment a 32 bit value in a summarized address.

**Parameters:**

[in] *addr* - summarized address

[out] *zero\_flag* - will be true if the memory content was rolled back to zero

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_inc64_sum_addr (ezdp\_sum\_addr\_t addr,  
uint64_t __cmem * orig_value, bool * zero_flag) [static]
```

Atomically read and increment a 64 bit value in a summarized address.

**Parameters:**

[in] *addr* - summarized address

[out] *orig\_value* - the address in CMEM to write the original value in

[out] *zero\_flag* - will be true if the memory content was rolled back to zero

**Returns:**

The 32 MSB of the original/read value

```
static __always_inline uint32_t ezdp_atomic_read_and_dec8_ext_addr (struct ezdp\_ext\_addr *  
addr, bool * unaffected_flag) [static]
```

Atomically read and decrement conditionally by one an 8-bit value in an extended address (zero value does not underflow).

**Parameters:**

[in] *addr* - pointer to extended address

[out] *unaffected\_flag* - will be true if the memory content was not modified (original selected data value was zero)

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_dec16_ext_addr (struct ezdp\_ext\_addr *  
addr, bool * unaffected_flag) [static]
```

Atomically read and decrement conditionally by one a 16-bit value in an extended address (zero value does not underflow).

**Parameters:**

[in] *addr* - pointer to extended address

[out] *unaffected\_flag* - will be true if the memory content was not modified (original selected data value was zero)

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline uint32_t ezdp_atomic_read_and_dec32_ext_addr (struct ezdp\_ext\_addr *  
addr, bool * unaffected_flag) [static]
```

Atomically read and decrement conditionally by one a 32-bit value in an extended address (zero value does not underflow).

**Parameters:**

[in] *addr* - pointer to extended address

[out] *unaffected\_flag* - will be true if the memory content was not modified (original selected data value was zero)

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_dec32_sum_addr (ezdp\_sum\_addr\_t addr,  
bool * unaffected_flag) [static]
```

Atomically read and decrement conditionally by one a 32-bit value in a summarized address (zero value does not underflow).

**Parameters:**

[in] *addr* - summarized address

[out] *unaffected\_flag* - will be true if the memory content was not modified (original selected data value was zero)

**Returns:**

Original/Read value.

```
static __always_inline uint32_t ezdp_atomic_read_and_dec64_sum_addr (ezdp\_sum\_addr\_t addr,
uint64_t __cmem * orig_value,    bool * unaffected_flag) [static]
```

Atomically read and decrement conditionally by one a 64-bit value in a summarized address (zero value does not underflow).

**Parameters:**

[in] *addr* - summarized address  
[out] *orig\_value* - the address in CMEM to write the original value in  
[out] *unaffected\_flag* - will be true if the memory content was not modified (original selected data value was zero)

**Returns:**

The 32 MSB of the original/read value

```
static __always_inline uint32_t ezdp_atomic_read_and_inc32_cond_ext_addr (struct
ezdp\_ext\_addr * addr,    bool res_mode,    bool * unaffected_flag) [static]
```

Atomically read and increment conditionally a 32-bit value in an extended address.

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *res\_mode* - the reservation mode  
[out] *unaffected\_flag* - will be true if the memory content was not modified

**Note:**

Address must be 4-byte aligned.

**Returns:**

The original accessed counter value

```
static __always_inline uint32_t ezdp_atomic_read_and_inc32_cond_sum_addr (ezdp\_sum\_addr\_t
addr,    bool res_mode,    bool * unaffected_flag) [static]
```

Atomically read and increment conditionally a 32-bit value in a summarized address.

**Parameters:**

[in] *addr* - summarized address  
[in] *res\_mode* - the reservation mode  
[out] *unaffected\_flag* - will be true if the memory content was not modified

**Returns:**

The original accessed counter value

```
static __always_inline void ezdp_atomic_and8_ext_addr (struct ezdp\_ext\_addr * addr,    uint8_t
value) [static]
```

Atomically perform an 8 bit logical AND operation on an extended address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - update value

**Returns:**

void

```
static __always_inline void ezdp_atomic_and8_ext_addr_async (struct ezdp\_ext\_addr * addr,
uint8_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_and8\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_and8_ext_addr (struct ezdp\_ext\_addr *
addr, uint8_t value, bool * no_chng_flag) [static]
```

Atomically read and perform an 8 bit logical AND operation on an extended address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

[out] *no\_chng\_flag* - True when AND operation resulted in no change to original value

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline void ezdp_atomic_and16_ext_addr (struct ezdp\_ext\_addr * addr, uint16_t
value) [static]
```

Atomically perform a 16 bit logical AND operation on an extended address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

**Note:**

Address must be 2-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_and16_ext_addr_async (struct ezdp\_ext\_addr * addr,
uint16_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_and16\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - value to and with

**Note:**

Address must be 2-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_and16_ext_addr (struct ezdp\_ext\_addr *  
addr, uint16_t value, bool * no_chng_flag) [static]
```

Atomically read and perform a 16 bit logical AND operation on an extended address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value  
 [out] *no\_chng\_flag* - True when AND operation resulted in no change to original value

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline void ezdp_atomic_and32_ext_addr (struct ezdp\_ext\_addr * addr, uint32_t  
value) [static]
```

Atomically perform a 32 bit logical AND operation on an extended address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value

**Note:**

Address must be 4-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_and32_ext_addr_async (struct ezdp\_ext\_addr * addr,  
uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_and32\\_ext\\_addr\\_async\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value

**Note:**

Address must be 4-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.



**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_and32_ext_addr (struct ezdp\_ext\_addr *  
addr, uint32_t value, bool * no_chng_flag) [static]
```

Atomically read and perform a 32 bit logical AND operation on an extended address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

[out] *no\_chng\_flag* - True when AND operation resulted in no change to original value

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original/Read value

```
static __always_inline void ezdp_atomic_and32_sum_addr (ezdp\_sum\_addr\_t addr, uint32_t  
value) [static]
```

Atomically perform a 32 bit logical AND operation on a summarized address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address

[in] *value* - update value

**Returns:**

void

```
static __always_inline void ezdp_atomic_and32_sum_addr_async (ezdp\_sum\_addr\_t addr,  
uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_and32\\_sum\\_addr\\_async\(\)](#).

**Parameters:**

[in] *addr* - summarized address

[in] *value* - update value

**Note:**

Address must be 4-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_and32_sum_addr (ezdp\_sum\_addr\_t addr,  
uint32_t value, bool * no_chng_flag) [static]
```

Atomically read and perform a 32 bit logical AND operation on a summarized address.

Performs a logical AND operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address  
 [in] *value* - update value  
 [out] *no\_chng\_flag* - True when AND operation resulted in no change to original value

**Returns:**

Original/Read value

```
static __always_inline void ezdp_atomic_or8_ext_addr (struct ezdp\_ext\_addr * addr,  uint8_t
value) [static]
```

Atomically perform an 8 bit logical OR operation on an extended address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value to OR with

**Returns:**

void

```
static __always_inline void ezdp_atomic_or8_ext_addr_async (struct ezdp\_ext\_addr * addr,
uint8_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_or8\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value to OR with

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_or8_ext_addr (struct ezdp\_ext\_addr *
addr,  uint8_t value,  bool * no_chng_flag) [static]
```

Atomically read and perform an 8 bit logical OR operation on an extended address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value to OR with  
 [out] *no\_chng\_flag* - True when OR operation resulted in no change to original value

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline void ezdp_atomic_or8_sum_addr (ezdp\_sum\_addr\_t addr, uint8_t value)
[static]
```

Atomically perform an 8 bit logical OR operation on a summarized address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address  
[in] *value* - update value to OR with

**Returns:**

void

```
static __always_inline void ezdp_atomic_or8_sum_addr_async (ezdp\_sum\_addr\_t addr, uint8_t
value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_or8\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address  
[in] *value* - update value to OR with

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_atomic_or16_ext_addr (struct ezdp\_ext\_addr * addr, uint16_t
value) [static]
```

Atomically perform a 16 bit logical OR operation on an extended address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - update value to OR with

**Note:**

Address must be 2-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_or16_ext_addr_async (struct ezdp\_ext\_addr * addr,
uint16_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_or16\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
[in] *value* - value to OR with

**Note:**

Address must be 2-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_or16_ext_addr (struct ezdp\_ext\_addr *  
addr, uint16_t value, bool * no_chng_flag) [static]
```

Atomically read and perform a 16 bit logical OR operation on an extended address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value to OR with

[out] *no\_chng\_flag* - True when OR operation resulted in no change to original value

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline void ezdp_atomic_or16_sum_addr (ezdp\_sum\_addr\_t addr, uint16_t  
value) [static]
```

Atomically perform a 16 bit logical OR operation on a summarized address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address

[in] *value* - update value to OR with

**Returns:**

void

```
static __always_inline void ezdp_atomic_or16_sum_addr_async (ezdp\_sum\_addr\_t addr,  
uint16_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_or16\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address

[in] *value* - value to OR with

**Note:**

Address must be 2-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_atomic_or32_ext_addr (struct ezdp\_ext\_addr * addr, uint32_t  
value) [static]
```

Atomically perform a 32 bit logical OR operation on an extended address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value to OR with

**Note:**

Address must be 4-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_or32_ext_addr_async (struct ezdp\_ext\_addr * addr,
uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_or32\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - value to OR with

**Note:**

Address must be 4-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_or32_ext_addr (struct ezdp\_ext\_addr *
addr, uint32_t value, bool * no_chng_flag) [static]
```

Atomically read and perform a 32 bit logical OR operation on an extended address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address  
 [in] *value* - update value to OR with  
 [out] *no\_chng\_flag* - True when OR operation resulted in no change to original value

**Note:**

Address must be 4-byte aligned.

**Returns:**

Original/Read value

```
static __always_inline void ezdp_atomic_or32_sum_addr (ezdp\_sum\_addr\_t addr, uint32_t
value) [static]
```

Atomically perform a 32 bit logical OR operation on a summarized address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address  
 [in] *value* - update value to OR with

**Returns:**

void

```
static __always_inline void ezdp_atomic_or32_sum_addr_async (ezdp\_sum\_addr\_t addr,
uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_or32\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *addr* - summarized address

[in] *value* - value to OR with

**Note:**

Address must be 4-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_or32_sum_addr (ezdp\_sum\_addr\_t addr,
uint32_t value,    bool * no_chng_flag) [static]
```

Atomically read and perform a 32 bit logical OR operation on a summarized address.

Performs a logical OR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - summarized address

[in] *value* - update value to OR with

[out] *no\_chng\_flag* - True when OR operation resulted in no change to original value

**Returns:**

Original/Read value

```
static __always_inline void ezdp_atomic_xor8_ext_addr (struct ezdp\_ext\_addr * addr,    uint8_t
value) [static]
```

Atomically perform an 8 bit logical XOR operation on an extended address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

**Returns:**

void

```
static __always_inline void ezdp_atomic_xor8_ext_addr_async (struct ezdp\_ext\_addr * addr,
uint8_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_xor8\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - value to XOR with

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_xor8_ext_addr (struct ezdp\_ext\_addr *  
addr, uint8_t value, bool * no_id_flag) [static]
```

Atomically read and perform an 8 bit logical XOR operation on an extended address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

[out] *no\_id\_flag* - True when the value passed to function and the value in memory are not identical.

**Returns:**

Original/Read value. The value is limited to 8 bits

```
static __always_inline void ezdp_atomic_xor16_ext_addr (struct ezdp\_ext\_addr * addr, uint16_t  
value) [static]
```

Atomically perform a 16 bit logical XOR operation on an extended address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

**Note:**

Address must be 2-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_xor16_ext_addr_async (struct ezdp\_ext\_addr * addr,  
uint16_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_xor8\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

**Note:**

Address must be 2-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_xor16_ext_addr (struct ezdp\_ext\_addr *  
addr, uint16_t value, bool * no_id_flag) [static]
```

Atomically read and perform a 16 bit logical XOR operation on an extended address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

[out] *no\_id\_flag* - True when the value passed to function and the value in memory are not identical.

**Note:**

Address must be 2-byte aligned.

**Returns:**

Original/Read value. The value is limited to 16 bits

```
static __always_inline void ezdp_atomic_xor32_ext_addr (struct ezdp\_ext\_addr * addr, uint32_t  
value) [static]
```

Atomically perform a 32 bit logical XOR operation on an extended address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

**Note:**

Address must be 4-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_atomic_xor32_ext_addr_async (struct ezdp\_ext\_addr * addr,  
uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_xor32\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *addr* - pointer to extended address

[in] *value* - update value

**Note:**

Address must be 4-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_xor32_ext_addr (struct ezdp\_ext\_addr *  
addr, uint32_t value, bool * no_id_flag) [static]
```

Atomically read and perform a 32 bit logical XOR operation on an extended address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.



**Parameters:**[in] *addr* - pointer to extended address[in] *value* - update value[out] *no\_id\_flag* - True when the value passed to function and the value in memory are not identical.**Note:**

Address must be 4-byte aligned.

**Returns:**

Original/Read value.

```
static __always_inline void ezdp_atomic_xor32_sum_addr (ezdp\_sum\_addr\_t addr, uint32_t value) [static]
```

Atomically perform a 32 bit logical XOR operation on a summarized address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**[in] *addr* - summarized address[in] *value* - update value**Returns:**

void

```
static __always_inline void ezdp_atomic_xor32_sum_addr_async (ezdp\_sum\_addr\_t addr, uint32_t value) [static]
```

Non-blocking/posted version of [ezdp\\_atomic\\_xor32\\_sum\\_addr\(\)](#).

**Parameters:**[in] *addr* - summarized address[in] *value* - update value**Note:**

Address must be 4-byte aligned. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_atomic_read_and_xor32_sum_addr (ezdp\_sum\_addr\_t addr, uint32_t value, bool * no_id_flag) [static]
```

Atomically read and perform a 32 bit logical XOR operation on a summarized address.

Performs a logical XOR operation between the existing value in memory and the value passed, and stores the result back into memory.

**Parameters:**[in] *addr* - summarized address[in] *value* - update value[out] *no\_id\_flag* - True when the value passed to function and the value in memory are not identical.**Returns:**

Original/Read value.

## dpe/dp/include/ezdp\_counter.h File Reference

### Functions

- static `__always_inline` void [ezdp\\_write\\_single\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_single\\_ctr\\_cfg](#) \*counter)
- *Configure single counter and its initial value.* static `__always_inline` void [ezdp\\_write\\_single\\_ctr\\_cfg\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_single\\_ctr\\_cfg](#) \*counter)
- *Non blocking version of [ezdp\\_write\\_single\\_ctr\\_cfg](#).* static `__always_inline` void [ezdp\\_read\\_single\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_single\\_ctr\\_cfg](#) \_\_cmem \*counter)
- *Read single counter configuration.* static `__always_inline` void [ezdp\\_read\\_single\\_ctr\\_cfg\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_single\\_ctr\\_cfg](#) \_\_cmem \*counter)
- *Non blocking version of [ezdp\\_read\\_single\\_ctr\\_cfg](#).* static `__always_inline` void [ezdp\\_write\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t value)
- *Initialize single counter with the value specified.* static `__always_inline` void [ezdp\\_write\\_single\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t value)
- *Non blocking version of [ezdp\\_write\\_single\\_ctr](#).* static `__always_inline` uint32\_t [ezdp\\_xchg\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t value, uint64\_t \_\_cmem \*orig\_value)
- *Write single counter with the value specified and read previous counter value.* static `__always_inline` uint32\_t [ezdp\\_read\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t \_\_cmem \*value, bool \*overflow)
- *Read single counter value.* static `__always_inline` void [ezdp\\_inc\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t increment\_value)
- *Increment single counter by the value specified.* static `__always_inline` void [ezdp\\_inc\\_single\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t increment\_value)
- *Non blocking version of [ezdp\\_inc\\_single\\_ctr](#).* static `__always_inline` uint32\_t [ezdp\\_read\\_and\\_inc\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t increment\_value, uint64\_t \_\_cmem \*orig\_value, bool \*overflow)
- *Increment single counter by the value specified and read previous counter value.* static `__always_inline` void [ezdp\\_dec\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value)
- *Decrement single counter by the value specified.* static `__always_inline` void [ezdp\\_dec\\_single\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value)
- *Non blocking version of [ezdp\\_dec\\_single\\_ctr](#).* static `__always_inline` uint32\_t [ezdp\\_read\\_and\\_dec\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value, uint64\_t \_\_cmem \*orig\_value, bool \*overflow)
- *Decrement single counter by the value specified and read previous counter value.* static `__always_inline` void [ezdp\\_reset\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Reset single counter to zero.* static `__always_inline` void [ezdp\\_reset\\_single\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Non blocking version of [ezdp\\_reset\\_single\\_ctr](#).* static `__always_inline` uint32\_t [ezdp\\_read\\_and\\_reset\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t \_\_cmem \*orig\_value)
- *Reset single counter to zero and read previous counter value.* static `__always_inline` void [ezdp\\_cond\\_dec\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value)
- *Conditionally decrement single counter by the value specified.* static `__always_inline` void [ezdp\\_cond\\_dec\\_single\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value)
- *Non blocking version of [ezdp\\_cond\\_dec\\_single\\_ctr](#).* static `__always_inline` uint32\_t [ezdp\\_read\\_and\\_cond\\_dec\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value, uint64\_t \_\_cmem \*orig\_value, bool \*failure\_ind)
- *Conditionally decrement single counter by the value specified and read previous counter value.* static `__always_inline` void [ezdp\\_prefetch\\_single\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Prefetch single counter into the local cache.* static `__always_inline` void [ezdp\\_prefetch\\_single\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Non blocking version of [ezdp\\_prefetch\\_single\\_ctr](#).* static `__always_inline` void [ezdp\\_write\\_dual\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_dual\\_ctr\\_cfg](#) \*counter)
- *Configure dual counter and its initial values (byte and event).* static `__always_inline` void [ezdp\\_write\\_dual\\_ctr\\_cfg\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_dual\\_ctr\\_cfg](#) \*counter)
- *Non blocking version of [ezdp\\_write\\_dual\\_ctr\\_cfg](#).* static `__always_inline` void [ezdp\\_read\\_dual\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_dual\\_ctr\\_cfg](#) \_\_cmem \*counter)
- *Read dual counter configuration.* static `__always_inline` uint32\_t [ezdp\\_read\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_dual\\_ctr\\_result](#) \_\_cmem \*value, bool \*overflow)
- *Read dual counter values (byte and event).* static `__always_inline` void [ezdp\\_inc\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t inc\_byte\_value, uint16\_t inc\_event\_value)

- Increment dual counter with the values specified (byte and event). static \_\_always\_inline void [ezdp\\_inc\\_dual\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t inc\_byte\_value, uint16\_t inc\_event\_value)
- Non blocking version of [ezdp\\_inc\\_dual\\_ctr](#). static \_\_always\_inline uint32\_t [ezdp\\_read\\_and\\_inc\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t inc\_byte\_value, uint16\_t inc\_event\_value, struct [ezdp\\_dual\\_ctr\\_result](#) \_\_cmem \*orig\_value, bool \*overflow)
- Increment dual counter with the values specified (byte and event) and read previous counter values. static \_\_always\_inline void [ezdp\\_dec\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t dec\_byte\_value)
- Decrement dual counter's byte value by the value specified and event value by 1. static \_\_always\_inline void [ezdp\\_dec\\_dual\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t dec\_byte\_value)
- Non blocking version of [ezdp\\_dec\\_dual\\_ctr](#). static \_\_always\_inline uint32\_t [ezdp\\_read\\_and\\_dec\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t dec\_byte\_value, struct [ezdp\\_dual\\_ctr\\_result](#) \_\_cmem \*orig\_value, bool \*overflow)
- Decrement dual counter's byte value by the value specified and event value by 1, and read previous counter values. static \_\_always\_inline void [ezdp\\_reset\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Reset dual counter values (byte and event) to zero. static \_\_always\_inline void [ezdp\\_reset\\_dual\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Non blocking version of [ezdp\\_reset\\_dual\\_ctr](#). static \_\_always\_inline uint32\_t [ezdp\\_read\\_and\\_reset\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_dual\\_ctr\\_result](#) \_\_cmem \*orig\_value)
- Reset dual counter values (byte and event) to zero and read previous counter values. static \_\_always\_inline void [ezdp\\_prefetch\\_dual\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Prefetch dual counter into the local cache. static \_\_always\_inline void [ezdp\\_prefetch\\_dual\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Non blocking version of [ezdp\\_prefetch\\_dual\\_ctr](#). static \_\_always\_inline void [ezdp\\_write\\_bitwise\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t value)
- Configure bitwise counter and its initial value. static \_\_always\_inline void [ezdp\\_write\\_bitwise\\_ctr\\_cfg\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t value)
- Non blocking version of [ezdp\\_write\\_bitwise\\_ctr\\_cfg](#). static \_\_always\_inline void [ezdp\\_read\\_bitwise\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_bitwise\\_ctr\\_cfg](#) \_\_cmem \*counter)
- Read bitwise counter configuration. static \_\_always\_inline void [ezdp\\_read\\_bitwise\\_ctr\\_cfg\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_bitwise\\_ctr\\_cfg](#) \_\_cmem \*counter)
- Non blocking version of [ezdp\\_read\\_bitwise\\_ctr\\_cfg](#). static \_\_always\_inline void [ezdp\\_write\\_bits\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset)
- Write the value to the selected bits in the bitwise counter. static \_\_always\_inline void [ezdp\\_write\\_bits\\_bitwise\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset)
- Non blocking version of [ezdp\\_write\\_bits\\_bitwise\\_ctr](#). static \_\_always\_inline void [ezdp\\_xchg\\_bits\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset, uint64\_t \_\_cmem \*orig\_value)
- Write the value to the selected bits in the bitwise counter and read previous counter value. static \_\_always\_inline uint32\_t [ezdp\\_read\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t \_\_cmem \*value)
- Read bitwise counter value. static \_\_always\_inline void [ezdp\\_read\\_bits\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset, uint64\_t \_\_cmem \*value)
- Read the selected bits from the bitwise counter. static \_\_always\_inline void [ezdp\\_inc\\_bits\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t increment\_value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset)
- Increment the selected bits in the bitwise counter by the value specified. static \_\_always\_inline void [ezdp\\_inc\\_bits\\_bitwise\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t increment\_value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset)
- Non blocking version of [ezdp\\_inc\\_bits\\_bitwise\\_ctr](#). static \_\_always\_inline void [ezdp\\_read\\_and\\_inc\\_bits\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t increment\_value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset, uint64\_t \_\_cmem \*orig\_value)
- Increment the selected bits in the bitwise counter by the value specified and read previous counter value. static \_\_always\_inline void [ezdp\\_dec\\_bits\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset)
- Decrement the selected bits in the bitwise counter by the value specified. static \_\_always\_inline void [ezdp\\_dec\\_bits\\_bitwise\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset)
- Non blocking version of [ezdp\\_dec\\_bits\\_bitwise\\_ctr](#). static \_\_always\_inline void [ezdp\\_read\\_and\\_dec\\_bits\\_bitwise\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t decrement\_value, enum [ezdp\\_bitwise\\_size](#) size, uint8\_t offset, uint64\_t \_\_cmem \*orig\_value)

- Decrement the selected bits in the bitwise counter by the value specified and read previous counter value. static `__always_inline void ezdp\_reset\_bitwise\_ctr (ezdp\_sum\_addr\_t addr)`
- Reset bitwise counter value to zero. static `__always_inline void ezdp\_reset\_bitwise\_ctr\_async (ezdp\_sum\_addr\_t addr)`
- Non blocking version of `ezdp_reset_bitwise_ctr`. static `__always_inline void ezdp\_read\_and\_reset\_bitwise\_ctr (ezdp\_sum\_addr\_t addr, uint64_t __cmem *orig_value)`
- Reset bitwise counter value to zero and read previous counter value. static `__always_inline void ezdp\_set\_bits\_bitwise\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset)`
- Set the selected bits in the bitwise counter according to the value specified. static `__always_inline void ezdp\_set\_bits\_bitwise\_ctr\_async (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset)`
- Non blocking version of `ezdp_set_bits_bitwise_ctr`. static `__always_inline void ezdp\_read\_and\_set\_bits\_bitwise\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset, uint64_t __cmem *orig_value)`
- Set the selected bits in the bitwise counter according to the value specified and read previous counter value. static `__always_inline void ezdp\_clear\_bits\_bitwise\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset)`
- Clear the selected bits in the bitwise counter according to the value specified. static `__always_inline void ezdp\_clear\_bits\_bitwise\_ctr\_async (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset)`
- Non blocking version of `ezdp_clear_bits_bitwise_ctr`. static `__always_inline void ezdp\_read\_and\_clear\_bits\_bitwise\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset, uint64_t __cmem *orig_value)`
- Clear the selected bits in the bitwise counter according to the value specified and read previous counter value. static `__always_inline void ezdp\_read\_and\_cond\_write\_bits\_bitwise\_ctr (ezdp\_sum\_addr\_t addr, uint8_t value, enum ezdp\_bitwise\_size size, uint8_t offset, uint8_t cmp_value, uint64_t __cmem *orig_value, bool *success_ind)`
- Read, compare and conditionally set the specified bits in the bitwise counter with the value specified. static `__always_inline void ezdp\_prefetch\_bitwise\_ctr (ezdp\_sum\_addr\_t addr)`
- Prefetch bitwise counter into the local cache. static `__always_inline void ezdp\_prefetch\_bitwise\_ctr\_async (ezdp\_sum\_addr\_t addr)`
- Non blocking version of `ezdp_prefetch_bitwise_ctr`. static `__always_inline void ezdp\_write\_tb\_ctr\_cfg (ezdp\_sum\_addr\_t addr, struct ezdp\_tb\_ctr\_cfg *counter)`
- Configure token bucket counter. static `__always_inline void ezdp\_write\_tb\_ctr\_cfg\_async (ezdp\_sum\_addr\_t addr, struct ezdp\_tb\_ctr\_cfg *counter)`
- Non blocking version of `ezdp_write_tb_ctr_cfg`. static `__always_inline void ezdp\_read\_tb\_ctr\_cfg (ezdp\_sum\_addr\_t addr, struct ezdp\_tb\_ctr\_cfg __cmem *counter)`
- Read token bucket counter configuration. static `__always_inline void ezdp\_update\_tb\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color)`
- Update a token bucket counter with the specified value (e.g. static `__always_inline void ezdp\_update\_tb\_ctr\_async (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color)`
- Non blocking version of `ezdp_update_tb_ctr`. static `__always_inline void ezdp\_read\_tb\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem *result_color)`
- Get the resulting color after updating a token bucket counter with the specified value (e.g. static `__always_inline void ezdp\_read\_tb\_ctr\_async (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem *result_color)`
- Non blocking version of `ezdp_read_tb_color`. static `__always_inline void ezdp\_check\_tb\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem *result_value)`
- Get the resulting color after updating a token bucket with the specified value (e.g. static `__always_inline void ezdp\_check\_tb\_ctr\_async (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem *result_value)`
- Non blocking version of `ezdp_check_tb_ctr`. static `__always_inline void ezdp\_inc\_tb\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color, bool inc_commit_bucket, bool inc_excess_bucket)`
- Force increment token buckets with the specified value (e.g. static `__always_inline void ezdp\_inc\_tb\_ctr\_async (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color, bool inc_commit_bucket, bool inc_excess_bucket)`
- Non blocking version of `ezdp_inc_tb_ctr`. static `__always_inline void ezdp\_read\_and\_inc\_tb\_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum ezdp\_tb\_color pre_color, bool inc_commit_bucket, bool inc_excess_bucket, struct ezdp\_tb\_ctr\_result __cmem *result_value)`

- Force increment token buckets with the specified value (e.g. static \_\_always\_inline void [ezdp\\_dec\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, enum [ezdp\\_tb\\_color](#) pre\_color, bool dec\_commit\_bucket, bool dec\_excess\_bucket)
- Force decrement token buckets with the specified value (e.g. static \_\_always\_inline void [ezdp\\_dec\\_tb\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, enum [ezdp\\_tb\\_color](#) pre\_color, bool dec\_commit\_bucket, bool dec\_excess\_bucket)
- Non blocking version of [ezdp\\_dec\\_tb\\_ctr](#). static \_\_always\_inline void [ezdp\\_read\\_and\\_dec\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, enum [ezdp\\_tb\\_color](#) pre\_color, bool dec\_commit\_bucket, bool dec\_excess\_bucket, struct [ezdp\\_tb\\_ctr\\_result](#) \_\_cmem \*result\_value)
- Force decrement token buckets with the specified value (e.g. static \_\_always\_inline void [ezdp\\_prefetch\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Prefetch token bucket counter into the local cache. static \_\_always\_inline void [ezdp\\_prefetch\\_tb\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Non blocking version of [ezdp\\_prefetch\\_tb\\_ctr](#). static \_\_always\_inline void [ezdp\\_write\\_hier\\_tb\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_hier\\_tb\\_ctr\\_cfg](#) \*counter)
- Configure hierarchical token bucket counter. static \_\_always\_inline void [ezdp\\_write\\_hier\\_tb\\_ctr\\_cfg\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_hier\\_tb\\_ctr\\_cfg](#) \*counter)
- Non blocking version of [ezdp\\_write\\_hier\\_tb\\_ctr\\_cfg](#). static \_\_always\_inline void [ezdp\\_read\\_hier\\_tb\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_hier\\_tb\\_ctr\\_cfg](#) \_\_cmem \*counter)
- Read hierarchical token bucket counter configuration. static \_\_always\_inline void [ezdp\\_inc\\_hier\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, bool update\_ctr0, bool update\_acc1)
- Increment hierarchical token bucket counter accumulator(s) by the value specified. static \_\_always\_inline void [ezdp\\_inc\\_hier\\_tb\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, bool update\_ctr0, bool update\_acc1)
- Non blocking version of [ezdp\\_inc\\_hier\\_tb\\_ctr](#). static \_\_always\_inline void [ezdp\\_read\\_and\\_inc\\_hier\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint16\_t value, bool update\_ctr0, bool update\_acc1, struct [ezdp\\_hier\\_tb\\_result](#) \_\_cmem \*result)
- Increment hierarchical token bucket counter accumulator(s) by the value specified and read previous counter value. static \_\_always\_inline void [ezdp\\_update\\_hier\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_hier\\_tb\\_update](#) \*ctr\_update)
- Update hierarchical token bucket counter state, app bits or clear accumulators. static \_\_always\_inline void [ezdp\\_update\\_hier\\_tb\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_hier\\_tb\\_update](#) \*ctr\_update)
- Non blocking version of [ezdp\\_inc\\_hier\\_tb\\_ctr](#). static \_\_always\_inline void [ezdp\\_read\\_and\\_update\\_hier\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_hier\\_tb\\_update](#) \*ctr\_update, struct [ezdp\\_hier\\_tb\\_result](#) \_\_cmem \*result)
- Update hierarchical token bucket counter state, app bits or clear accumulators and read previous counter value. static \_\_always\_inline void [ezdp\\_change\\_state\\_hier\\_tb\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_hier\\_tb\\_result](#) \_\_cmem \*result)
- Change hierarchical token bucket state to Ph1. static \_\_always\_inline void [ezdp\\_write\\_watchdog\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_watchdog\\_ctr\\_cfg](#) \*counter)
- Configure watchdog counter. static \_\_always\_inline void [ezdp\\_write\\_watchdog\\_ctr\\_cfg\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_watchdog\\_ctr\\_cfg](#) \*counter)
- Non blocking version of [ezdp\\_write\\_watchdog\\_ctr\\_cfg](#). static \_\_always\_inline void [ezdp\\_read\\_watchdog\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_watchdog\\_ctr\\_cfg](#) \_\_cmem \*counter)
- Read watchdog counter configuration. static \_\_always\_inline void [ezdp\\_start\\_watchdog\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) counter\_address)
- Start the watchdog counter. static \_\_always\_inline void [ezdp\\_start\\_watchdog\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) counter\_address)
- Non blocking version of [ezdp\\_start\\_watchdog\\_ctr](#). static \_\_always\_inline void [ezdp\\_inc\\_watchdog\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Increment the watchdog counter events by one. static \_\_always\_inline void [ezdp\\_inc\\_watchdog\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Non blocking version of [ezdp\\_inc\\_watchdog\\_ctr](#). static \_\_always\_inline void [ezdp\\_check\\_watchdog\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) counter\_address, struct [ezdp\\_watchdog\\_ctr\\_check\\_result](#) \_\_cmem \*check\_result)
- Check the number of events in the watchdog counter. static \_\_always\_inline void [ezdp\\_check\\_watchdog\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) counter\_address, struct [ezdp\\_watchdog\\_ctr\\_check\\_result](#) \_\_cmem \*check\_result)
- Non blocking version of [ezdp\\_check\\_watchdog\\_ctr](#). static \_\_always\_inline void [ezdp\\_prefetch\\_watchdog\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- Prefetch watchdog\_counter into the local cache. static \_\_always\_inline void [ezdp\\_prefetch\\_watchdog\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)



- *Non blocking version of ezdp\_prefetch\_watchdog\_ctr.* static \_\_always\_inline bool [ezdp\\_init\\_ctr\\_msg\\_queue\\_desc](#) (uint32\_t partition\_id, uint32\_t queue\_mask, [ezdp\\_ctr\\_msg\\_queue\\_desc\\_t](#) \*ctr\_queue\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- *Initialize counter message queue descriptor.* static \_\_always\_inline bool [ezdp\\_read\\_ctr\\_msg](#) ([ezdp\\_ctr\\_msg\\_queue\\_desc\\_t](#) \*ctr\_queue\_desc, struct [ezdp\\_ctr\\_msg](#) \*msg, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- *Read counter message from message queue.* static \_\_always\_inline void [ezdp\\_write\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t value)
- *Initialize posted counter with the value specified.* static \_\_always\_inline void [ezdp\\_write\\_posted\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t value)
- *Non blocking version of ezdp\_write\_posted\_ctr.* static \_\_always\_inline void [ezdp\\_dual\\_write\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t counter1, uint64\_t counter2)
- *Initialize two successive posted counter with the value specified.* static \_\_always\_inline void [ezdp\\_dual\\_write\\_posted\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, uint64\_t counter1, uint64\_t counter2)
- *Non blocking version of ezdp\_dual\_write\_posted\_ctr.* static \_\_always\_inline void [ezdp\\_add\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, int32\_t value)
- *Add signed value to posted counter.* static \_\_always\_inline void [ezdp\\_add\\_posted\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, int32\_t value)
- *Non blocking version of ezdp\_add\_posted\_ctr.* static \_\_always\_inline void [ezdp\\_dual\\_add\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, int16\_t counter1, int16\_t counter2)
- *Add signed values to two successive posted counters.* static \_\_always\_inline void [ezdp\\_dual\\_add\\_posted\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr, int16\_t counter1, int16\_t counter2)
- *Non blocking version of ezdp\_add\_posted\_ctr\_dual.* static \_\_always\_inline void [ezdp\\_report\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) sum\_addr, bool flush)
- *Generate posted counter value report.* static \_\_always\_inline void [ezdp\\_report\\_and\\_clear\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) sum\_addr, bool flush)
- *Generate posted counter value report and reset the counter to zero.* static \_\_always\_inline void [ezdp\\_dual\\_report\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, bool flush)
- *Generate posted counter value report for two successive counters.* static \_\_always\_inline void [ezdp\\_dual\\_report\\_and\\_clear\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, bool flush)
- *Generate posted counter value report for two successive counters and reset both counter values to zero.* static \_\_always\_inline void [ezdp\\_reset\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Reset posted counter.* static \_\_always\_inline void [ezdp\\_reset\\_posted\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Non blocking version of ezdp\_reset\_posted\_ctr.* static \_\_always\_inline void [ezdp\\_dual\\_reset\\_posted\\_ctr](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Reset two successive posted counter.* static \_\_always\_inline void [ezdp\\_dual\\_reset\\_posted\\_ctr\\_async](#) ([ezdp\\_sum\\_addr\\_t](#) addr)
- *Non blocking version of ezdp\_dual\_reset\_posted\_ctr.* static \_\_always\_inline bool [ezdp\\_init\\_posted\\_ctr\\_msg\\_queue\\_desc](#) (uint32\_t partition\_id, uint32\_t queue\_mask, [ezdp\\_posted\\_ctr\\_msg\\_queue\\_desc\\_t](#) \*posted\_ctr\_queue\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- *Initialize posted counter message queue descriptor.* static \_\_always\_inline bool [ezdp\\_read\\_posted\\_ctr\\_msg](#) ([ezdp\\_posted\\_ctr\\_msg\\_queue\\_desc\\_t](#) \*ctr\_queue\_desc, struct [ezdp\\_posted\\_ctr\\_msg](#) \*msg, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)

*Read posted counter message from message queue.*

## Function Documentation

static \_\_always\_inline void [ezdp\\_write\\_single\\_ctr\\_cfg](#) ([ezdp\\_sum\\_addr\\_t](#) addr, struct [ezdp\\_single\\_ctr\\_cfg](#) \*counter) [static]

Configure single counter and its initial value.

### Parameters:

- [in] *addr* - Address of counter
- [in] *counter* - counter configuration + value

**Returns:**

void

```
static __always_inline void ezdp_write_single_ctr_cfg (ezdp_sum_addr_t addr, struct
ezdp_single_ctr_cfg * counter) [static]
```

Non blocking version of ezdp\_write\_single\_ctr\_cfg.

**Parameters:**

[in] *addr* - Address of counter

[in] *counter* - counter configuration + value

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_single_ctr_cfg (ezdp_sum_addr_t addr, struct
ezdp_single_ctr_cfg __cmem * counter) [static]
```

Read single counter configuration.

**Parameters:**

[in] *addr* - Address of counter

[out] *counter* - Address at CMEM to write counter config

**Returns:**

void

```
static __always_inline void ezdp_read_single_ctr_cfg_async (ezdp_sum_addr_t addr, struct
ezdp_single_ctr_cfg __cmem * counter) [static]
```

Non blocking version of ezdp\_read\_single\_ctr\_cfg.

**Parameters:**

[in] *addr* - Address of counter

[out] *counter* - Address at CMEM to write counter config

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_write_single_ctr (ezdp_sum_addr_t addr, uint64_t value)
[static]
```

Initialize single counter with the value specified.

**Parameters:**

[in] *addr* - Address of counter

[in] *value* - Counter value

**Returns:**

void

```
static __always_inline void ezdp_write_single_ctr_async (ezdp\_sum\_addr\_t addr,  uint64_t
value) [static]
```

Non blocking version of ezdp\_write\_single\_ctr.

**Parameters:**

[in] *addr* - Address of counter

[in] *value* - Counter value

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_xchg_single_ctr (ezdp\_sum\_addr\_t addr,  uint64_t value,
uint64_t __cmem * orig_value) [static]
```

Write single counter with the value specified and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter

[in] *value* - Counter value

[out] *orig\_value* - Address at CMEM to write old counter value

**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline uint32_t ezdp_read_single_ctr (ezdp\_sum\_addr\_t addr,  uint64_t __cmem
* value,  bool * overflow) [static]
```

Read single counter value.

**Parameters:**

[in] *addr* - Address of counter

[out] *value* - Address at CMEM to write counter value

[out] *overflow* - over flow flag of previous operation

**Returns:**

uint32\_t - 32 MSB value

```
static __always_inline void ezdp_inc_single_ctr (ezdp\_sum\_addr\_t addr,  uint16_t
increment_value) [static]
```

Increment single counter by the value specified.

**Parameters:**

[in] *addr* - Address of counter



[in] *increment\_value* - value to enlarge

**Returns:**

void

```
static __always_inline void ezdp_inc_single_ctr_async (ezdp\_sum\_addr\_t addr,  uint16_t
increment_value) [static]
```

Non blocking version of ezdp\_inc\_single\_ctr.

**Parameters:**

[in] *addr* - Address of counter

[in] *increment\_value* - value to enlarge

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_and_inc_single_ctr (ezdp\_sum\_addr\_t addr,  uint16_t
increment_value,  uint64_t __cmem * orig_value,  bool * overflow) [static]
```

Increment single counter by the value specified and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter

[in] *increment\_value* - value to enlarge

[out] *orig\_value* - Address at CMEM to write old counter value

[out] *overflow* - over flow flag of previous operation

**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline void ezdp_dec_single_ctr (ezdp\_sum\_addr\_t addr,  uint16_t
decrement_value) [static]
```

Decrement single counter by the value specified.

**Parameters:**

[in] *addr* - Address of counter

[in] *decrement\_value* - value to reduce

**Returns:**

void

```
static __always_inline void ezdp_dec_single_ctr_async (ezdp\_sum\_addr\_t addr,  uint16_t
decrement_value) [static]
```

Non blocking version of ezdp\_dec\_single\_ctr.

**Parameters:**

[in] *addr* - Address of counter

[in] *decrement\_value* - value to reduce

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_and_dec_single_ctr (ezdp\_sum\_addr\_t addr,
uint16_t decrement_value,  uint64_t __cmem * orig_value,  bool * overflow) [static]
```

Decrement single counter by the value specified and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter

[in] *decrement\_value* - value to reduce

[out] *orig\_value* - Address at CMEM to write old counter value

[out] *overflow* - over flow flag of previous operation

**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline void ezdp_reset_single_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Reset single counter to zero.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_reset_single_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of ezdp\_reset\_single\_ctr.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_and_reset_single_ctr (ezdp\_sum\_addr\_t addr,
uint64_t __cmem * orig_value) [static]
```

Reset single counter to zero and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter

[out] *orig\_value* - Address at CMEM to write old counter value

**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline void ezdp_cond_dec_single_ctr (ezdp\_sum\_addr\_t addr,  uint16_t
decrement_value) [static]
```

Conditionally decrement single counter by the value specified.

If the decremented value is greater than zero, performs decrement operation, otherwise does not perform the operation.

**Parameters:**[in] *addr* - Address of counter[in] *decrement\_value* - value to reduce**Returns:**

void

```
static __always_inline void ezdp_cond_dec_single_ctr_async (ezdp\_sum\_addr\_t addr,  uint16_t
decrement_value) [static]
```

Non blocking version of `ezdp_cond_dec_single_ctr`.

**Parameters:**[in] *addr* - Address of counter[in] *decrement\_value* - value to reduce**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_and_cond_dec_single_ctr (ezdp\_sum\_addr\_t addr,
uint16_t decrement_value,  uint64_t __cmem * orig_value,  bool * failure_ind) [static]
```

Conditionally decrement single counter by the value specified and read previous counter value.

If the decremented value is greater than zero, performs decrement operation, otherwise does not perform the operation.

**Parameters:**[in] *addr* - Address of counter[in] *decrement\_value* - value to reduce[out] *orig\_value* - Address at CMEM to write old counter value[out] *failure\_ind* - failure indication**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline void ezdp_prefetch_single_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Prefetch single counter into the local cache.

Load counter into the cache to hide latency of the accessing this counter.

**Parameters:**[in] *addr* - Address of counter

**Returns:**

none

```
static __always_inline void ezdp_prefetch_single_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of ezdp\_prefetch\_single\_ctr.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_write_dual_ctr_cfg (ezdp\_sum\_addr\_t addr, struct
ezdp\_dual\_ctr\_cfg * counter) [static]
```

Configure dual counter and its initial values (byte and event).

**Parameters:**

[in] *addr* - Address of counter

[in] *counter* - counter configuration + value

**Returns:**

void

```
static __always_inline void ezdp_write_dual_ctr_cfg_async (ezdp\_sum\_addr\_t addr, struct
ezdp\_dual\_ctr\_cfg * counter) [static]
```

Non blocking version of ezdp\_write\_dual\_ctr\_cfg.

**Parameters:**

[in] *addr* - Address of counter

[in] *counter* - counter configuration + value

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_dual_ctr_cfg (ezdp\_sum\_addr\_t addr, struct
ezdp\_dual\_ctr\_cfg __cmem * counter) [static]
```

Read dual counter configuration.

**Parameters:**

[in] *addr* - Address of counter

[out] *counter* - Address at CMEM to write counter config

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_dual_ctr(ezdp\_sum\_addr\_t addr, struct
ezdp\_dual\_ctr\_result __cmem * value, bool * overflow) [static]
```

Read dual counter values (byte and event).

**Parameters:**

[in] *addr* - Address of counter  
 [out] *value* - Address at CMEM to write counter value  
 [out] *overflow* - over flow flag of previous operation

**Returns:**

uint32\_t - 32 MSB value

```
static __always_inline void ezdp_inc_dual_ctr(ezdp\_sum\_addr\_t addr, uint16_t inc_byte_value,
uint16_t inc_event_value) [static]
```

Increment dual counter with the values specified (byte and event).

**Parameters:**

[in] *addr* - Address of counter  
 [in] *inc\_byte\_value* - byte value to increase  
 [in] *inc\_event\_value* - event value to increase

**Returns:**

void

```
static __always_inline void ezdp_inc_dual_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t
inc_byte_value, uint16_t inc_event_value) [static]
```

Non blocking version of ezdp\_inc\_dual\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *inc\_byte\_value* - byte value to increase  
 [in] *inc\_event\_value* - event value to increase

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_and_inc_dual_ctr(ezdp\_sum\_addr\_t addr, uint16_t
inc_byte_value, uint16_t inc_event_value, struct ezdp\_dual\_ctr\_result __cmem * orig_value,
bool * overflow) [static]
```

Increment dual counter with the values specified (byte and event) and read previous counter values.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *inc\_byte\_value* - byte value to increase  
 [in] *inc\_event\_value* - event value to increase  
 [out] *orig\_value* - Address at CMEM to write old counter value  
 [out] *overflow* - over flow flag of previous operation

**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline void ezdp_dec_dual_ctr (ezdp_sum_addr_t addr,  uint16_t
dec_byte_value) [static]
```

Decrement dual counter's byte value by the value specified and event value by 1.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *dec\_byte\_value* - value to reduce byte counter

**Returns:**

void

```
static __always_inline void ezdp_dec_dual_ctr_async (ezdp_sum_addr_t addr,  uint16_t
dec_byte_value) [static]
```

Non blocking version of ezdp\_dec\_dual\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *dec\_byte\_value* - value to reduce byte counter

**Note:**

1. Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination. 2. event will reduce by 1

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_and_dec_dual_ctr (ezdp_sum_addr_t addr,  uint16_t
dec_byte_value,  struct ezdp_dual_ctr_result __cmem * orig_value,  bool * overflow) [static]
```

Decrement dual counter's byte value by the value specified and event value by 1, and read previous counter values.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *dec\_byte\_value* - value to reduce byte counter  
 [out] *orig\_value* - Address at CMEM to write old counter value  
 [out] *overflow* - over flow flag of previous operation

**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline void ezdp_reset_dual_ctr (ezdp_sum_addr_t addr) [static]
```

Reset dual counter values (byte and event) to zero.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_reset_dual_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of `ezdp_reset_dual_ctr`.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_and_reset_dual_ctr (ezdp\_sum\_addr\_t addr, struct ezdp\_dual\_ctr\_result __cmem * orig_value) [static]
```

Reset dual counter values (byte and event) to zero and read previous counter values.

**Parameters:**

[in] *addr* - Address of counter

[out] *orig\_value* - Address at CMEM to write old counter value

**Returns:**

uint32\_t - old 32 MSB value

```
static __always_inline void ezdp_prefetch_dual_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Prefetch dual counter into the local cache.

Load counter into the cache to hide latency of the accessing this counter.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

none

```
static __always_inline void ezdp_prefetch_dual_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of `ezdp_prefetch_dual_ctr`.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_write_bitwise_ctr_cfg (ezdp\_sum\_addr\_t addr,   uint64_t value)
[static]
```

Configure bitwise counter and its initial value.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - counter value

**Returns:**

void

```
static __always_inline void ezdp_write_bitwise_ctr_cfg_async (ezdp\_sum\_addr\_t addr,   uint64_t
value) [static]
```

Non blocking version of ezdp\_write\_bitwise\_ctr\_cfg.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - counter value

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_bitwise_ctr_cfg (ezdp\_sum\_addr\_t addr,   struct
ezdp\_bitwise\_ctr\_cfg __cmem * counter) [static]
```

Read bitwise counter configuration.

**Parameters:**

[in] *addr* - Address of counter  
 [out] *counter* - Address at CMEM to write counter config

**Returns:**

void

```
static __always_inline void ezdp_read_bitwise_ctr_cfg_async (ezdp\_sum\_addr\_t addr,   struct
ezdp\_bitwise\_ctr\_cfg __cmem * counter) [static]
```

Non blocking version of ezdp\_read\_bitwise\_ctr\_cfg.

**Parameters:**

[in] *addr* - Address of counter



[out] *counter* - Address at CMEM to write counter config

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_write_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Write the value to the selected bits in the bitwise counter.

**Parameters:**

[in] *addr* - Address of counter  
[in] *value* - bit value  
[in] *size* - The size of the operation  
[in] *offset* - Target bit offset. Must be multiply of the size

**Note:**

This command with the Read option may be used to implement a swap

**Returns:**

void

```
static __always_inline void ezdp_write_bits_bitwise_ctr_async(ezdp\_sum\_addr\_t addr,
uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Non blocking version of ezdp\_write\_bits\_bitwise\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
[in] *value* - bit value  
[in] *size* - The size of the operation  
[in] *offset* - Target bit offset. Must be multiply of the size

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_xchg_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_bitwise\_size size, uint8_t offset, uint64_t __cmem * orig_value) [static]
```

Write the value to the selected bits in the bitwise counter and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter  
[in] *value* - bits value  
[in] *size* - The size of the operation  
[in] *offset* - Target bit offset. Must be multiply of the size  
[out] *orig\_value* - Address at CMEM to write old counter value

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_bitwise_ctr(ezdp\_sum\_addr\_t addr, uint64_t
__cmem * value) [static]
```

Read bitwise counter value.

**Parameters:**

[in] *addr* - Address of counter

[out] *value* - Address at CMEM to write counter value

**Returns:**

uint32\_t - 32 MSB value

```
static __always_inline void ezdp_read_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr, enum
ezdp\_bitwise\_size size, uint8_t offset, uint64_t __cmem * value) [static]
```

Read the selected bits from the bitwise counter.

**Parameters:**

[in] *addr* - Address of counter

[in] *size* - The size of the operation

[in] *offset* - Target bit offset. Must be multiply of the size

[out] *value* - Address at CMEM to write counter value

**Returns:**

void

```
static __always_inline void ezdp_inc_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr, uint16_t
increment_value, enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Increment the selected bits in the bitwise counter by the value specified.

**Parameters:**

[in] *addr* - Address of counter

[in] *increment\_value* - value to enlarge

[in] *size* - The size of the operation

[in] *offset* - Target bit offset. Must be multiply of the size

**Returns:**

void

```
static __always_inline void ezdp_inc_bits_bitwise_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t
increment_value, enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Non blocking version of ezdp\_inc\_bits\_bitwise\_ctr.

**Parameters:**

[in] *addr* - Address of counter

[in] *increment\_value* - value to enlarge

[in] *size* - The size of the operation

[in] *offset* - Target bit offset. Must be multiply of the size

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_and_inc_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr,
uint16_t increment_value, enum ezdp\_bitwise\_size size, uint8_t offset, uint64_t __cmem *
orig_value) [static]
```

Increment the selected bits in the bitwise counter by the value specified and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *increment\_value* - value to enlarge  
 [in] *size* - The size of the operation  
 [in] *offset* - Target bit offset. Must be multiply of the size  
 [out] *orig\_value* - Address at CMEM to write old counter value (16 bits)

**Returns:**

void

```
static __always_inline void ezdp_dec_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr, uint16_t
decrement_value, enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Decrement the selected bits in the bitwise counter by the value specified.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *decrement\_value* - value to reduce  
 [in] *size* - The size of the operation  
 [in] *offset* - Target bit offset. Must be multiply of the size

**Returns:**

void

```
static __always_inline void ezdp_dec_bits_bitwise_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t
decrement_value, enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Non blocking version of `ezdp_dec_bits_bitwise_ctr`.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *decrement\_value* - value to reduce  
 [in] *size* - The size of the operation  
 [in] *offset* - Target bit offset. Must be multiply of the size

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_and_dec_bits_bitwise_ctr (ezdp\_sum\_addr\_t addr,
uint16_t decrement_value, enum ezdp\_bitwise\_size size, uint8_t offset, uint64_t __cmem *
orig_value) [static]
```

Decrement the selected bits in the bitwise counter by the value specified and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter  
[in] *decrement\_value* - value to reduce  
[in] *size* - The size of the operation  
[in] *offset* - Target bit offset. Must be multiply of the size  
[out] *orig\_value* - Address at CMEM to write old counter value (16 bits)

**Returns:**

void

```
static __always_inline void ezdp_reset_bitwise_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Reset bitwise counter value to zero.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_reset_bitwise_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of `ezdp_reset_bitwise_ctr`.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_and_reset_bitwise_ctr (ezdp\_sum\_addr\_t addr, uint64_t
__cmem * orig_value) [static]
```

Reset bitwise counter value to zero and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter  
[out] *orig\_value* - Address at CMEM to write old counter value

**Returns:**

void

```
static __always_inline void ezdp_set_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Set the selected bits in the bitwise counter according to the value specified.

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - event value to increase
- [in] *size* - The size of the operation
- [in] *offset* - Target bit offset. Must be multiply of the size

**Returns:**

void

```
static __always_inline void ezdp_set_bits_bitwise_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t
value, enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Non blocking version of `ezdp_set_bits_bitwise_ctr`.

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - event value to increase
- [in] *size* - The size of the operation
- [in] *offset* - Target bit offset. Must be multiply of the size

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_and_set_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr,
uint16_t value, enum ezdp\_bitwise\_size size, uint8_t offset, uint64_t __cmem * orig_value)
[static]
```

Set the selected bits in the bitwise counter according to the value specified and read previous counter value.

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - event value to increase
- [in] *size* - The size of the operation
- [in] *offset* - Target bit offset. Must be multiply of the size
- [out] *orig\_value* - Address at CMEM to write old counter value (16 bits)

**Returns:**

void

```
static __always_inline void ezdp_clear_bits_bitwise_ctr(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_bitwise\_size size, uint8_t offset) [static]
```

Clear the selected bits in the bitwise counter according to the value specified.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - bits value  
 [in] *size* - The size of the operation  
 [in] *offset* - Target bit offset. Must be multiply of the size

**Returns:**

void

```
static __always_inline void ezdp_clear_bits_bitwise_ctr (ezdp_sum_addr_t addr,
uint16_t value,  enum ezdp_bitwise_size size,  uint8_t offset) [static]
```

Non blocking version of ezdp\_clear\_bits\_bitwise\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - bits value  
 [in] *size* - The size of the operation  
 [in] *offset* - Target bit offset. Must be multiply of the size

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_and_clear_bits_bitwise_ctr (ezdp_sum_addr_t addr,
uint16_t value,  enum ezdp_bitwise_size size,  uint8_t offset,  uint64_t __cmem * orig_value)
[static]
```

Clear the selected bits in the bitwise counter according to the value specified and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - bits value  
 [in] *size* - The size of the operation  
 [in] *offset* - Target bit offset. Must be multiply of the size  
 [out] *orig\_value* - Address at CMEM to write old counter value (16 bits)

**Returns:**

void

```
static __always_inline void ezdp_read_and_cond_write_bits_bitwise_ctr (ezdp_sum_addr_t addr,
uint8_t value,  enum ezdp_bitwise_size size,  uint8_t offset,  uint8_t cmp_value,  uint64_t
__cmem * orig_value,  bool * success_ind) [static]
```

Read, compare and conditionally set the specified bits in the bitwise counter with the value specified.

If value of the selected bits, based on size (1, 2 or 4 bits), is match cmp\_value, it replaces the bits with the new value Old value and success/failure indication is returned.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - bits value  
 [in] *size* - The size of the operation Applicable values are 1,2,4 Encoded as log  
 [in] *offset* - Target bit offset. Must be multiply of the size  
 [in] *cmp\_value* - compare value

[out] *orig\_value* - Address at CMEM to write old counter value (16 bits)  
 [out] *success\_ind* - success indication

**Note:**

This command may be used for a state machine implementation

**Returns:**

void

```
static __always_inline void ezdp_prefetch_bitwise_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Prefetch bitwise counter into the local cache.

Load counter into the cache to hide latency of the accessing this counter.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

none

```
static __always_inline void ezdp_prefetch_bitwise_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of `ezdp_prefetch_bitwise_ctr`.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_write_tb_ctr_cfg (ezdp\_sum\_addr\_t addr, struct  

ezdp\_tb\_ctr\_cfg * counter) [static]
```

Configure token bucket counter.

**Parameters:**

[in] *addr* - Address of counter

[in] *counter* - counter configuration

**Returns:**

void

```
static __always_inline void ezdp_write_tb_ctr_cfg_async (ezdp\_sum\_addr\_t addr, struct  

ezdp\_tb\_ctr\_cfg * counter) [static]
```

Non blocking version of `ezdp_write_tb_ctr_cfg`.

**Parameters:**

[in] *addr* - Address of counter

[in] *counter* - counter configuration

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_tb_ctr_cfg (ezdp\_sum\_addr\_t addr, struct
ezdp\_tb\_ctr\_cfg __cmem * counter) [static]
```

Read token bucket counter configuration.

**Parameters:**

[in] *addr* - Address of counter  
 [out] *counter* - Address at CMEM to write counter config

**Returns:**

void

```
static __always_inline void ezdp_update_tb_ctr (ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color) [static]
```

Update a token bucket counter with the specified value (e.g. packet length).

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - Value to add to TB counter, e.g. packet length in bytes  
 [in] *pre\_color* - Pre-Color (e.g. packet original color)

**Returns:**

void

```
static __always_inline void ezdp_update_tb_ctr_async (ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color) [static]
```

Non blocking version of `ezdp_update_tb_ctr`.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - Value to add to TB counter, e.g. packet length in bytes  
 [in] *pre\_color* - Pre-Color (e.g. packet original color)

**Returns:**

void

```
static __always_inline void ezdp_read_tb_ctr (ezdp\_sum\_addr\_t addr, uint16_t value, enum
ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem * result_color) [static]
```

Get the resulting color after updating a token bucket counter with the specified value (e.g. packet length).

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - Value to add to TB counter, e.g. packet length in bytes



[in] *pre\_color* - Pre-Color (e.g. packet original color)  
 [out] *result\_color* - Address at CMEM to write result color

**Returns:**

void

```
static __always_inline void ezdp_read_tb_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem * result_color) [static]
```

Non blocking version of ezdp\_read\_tb\_color.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - Value to add to TB counter, e.g. packet length in bytes  
 [in] *pre\_color* - Pre-Color (e.g. packet original color)  
 [out] *result\_color* - Address at CMEM to write result color

**Returns:**

void

```
static __always_inline void ezdp_check_tb_ctr(ezdp\_sum\_addr\_t addr, uint16_t value, enum
ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem * result_value) [static]
```

Get the resulting color after updating a token bucket with the specified value (e.g. packet length), without updating it.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - Value to add to TB counter, e.g. packet length in bytes  
 [in] *pre\_color* - Pre-Color (e.g. packet original color)  
 [out] *result\_value* - Address at CMEM to write result color

**Returns:**

*ezdp\_tb\_color* - resulting color

```
static __always_inline void ezdp_check_tb_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color, struct ezdp\_tb\_ctr\_result __cmem * result_value) [static]
```

Non blocking version of ezdp\_check\_tb\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - Value to add to TB counter, e.g. packet length in bytes  
 [in] *pre\_color* - Pre-Color (e.g. packet original color)  
 [out] *result\_value* - Address at CMEM to write result color

**Returns:**

void

```
static __always_inline void ezdp_inc_tb_ctr(ezdp\_sum\_addr\_t addr, uint16_t value, enum
ezdp\_tb\_color pre_color, bool inc_commit_bucket, bool inc_excess_bucket) [static]
```

Force increment token buckets with the specified value (e.g. packet length).

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - Value to add to TB counter, e.g. packet length in bytes
- [in] *pre\_color* - Pre-Color (e.g. packet original color)
- [in] *inc\_commit\_bucket* - true if you want to increment commit bucket
- [in] *inc\_excess\_bucket* - true if you want to increment excess bucket

**Note:**

Can be used to implement hierarchical token bucket algorithms

**Returns:**

void

```
static __always_inline void ezdp_inc_tb_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color, bool inc_commit_bucket, bool inc_excess_bucket) [static]
```

Non blocking version of `ezdp_inc_tb_ctr`.

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - Value to add to TB counter, e.g. packet length in bytes
- [in] *pre\_color* - Pre-Color (e.g. packet original color)
- [in] *inc\_commit\_bucket* - true if you want to increment commit bucket
- [in] *inc\_excess\_bucket* - true if you want to increment excess bucket

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete.

**Returns:**

void

```
static __always_inline void ezdp_read_and_inc_tb_ctr(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color, bool inc_commit_bucket, bool inc_excess_bucket, struct
ezdp\_tb\_ctr\_result __cmem * result_value) [static]
```

Force increment token buckets with the specified value (e.g. packet length) and get resulting color and bucket states.

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - Value to add to TB counter, e.g. packet length in bytes
- [in] *pre\_color* - Pre-Color (e.g. packet original color)
- [in] *inc\_commit\_bucket* - true if you want to increment commit bucket
- [in] *inc\_excess\_bucket* - true if you want to increment excess bucket
- [out] *result\_value* - Address at CMEM to write result color

**Note:**

Can be used to implement hierarchical token bucket algorithms

**Returns:**

none

```
static __always_inline void ezdp_dec_tb_ctr(ezdp\_sum\_addr\_t addr, uint16_t value, enum
ezdp\_tb\_color pre_color, bool dec_commit_bucket, bool dec_excess_bucket) [static]
```

Force decrement token buckets with the specified value (e.g. packet length).

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - Value to decrement to TB counter, e.g. packet length in bytes
- [in] *pre\_color* - Pre-Color (e.g. packet original color)
- [in] *dec\_commit\_bucket* - true if you want to increment commit bucket
- [in] *dec\_excess\_bucket* - true if you want to increment excess bucket

**Note:**

Can be used to implement hierarchical token bucket algorithms

**Returns:**

void

```
static __always_inline void ezdp_dec_tb_ctr_async(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color, bool dec_commit_bucket, bool dec_excess_bucket)
[static]
```

Non blocking version of `ezdp_dec_tb_ctr`.

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - Value to decrement to TB counter, e.g. packet length in bytes
- [in] *pre\_color* - Pre-Color (e.g. packet original color)
- [in] *dec\_commit\_bucket* - true if you want to increment commit bucket
- [in] *dec\_excess\_bucket* - true if you want to increment excess bucket

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_and_dec_tb_ctr(ezdp\_sum\_addr\_t addr, uint16_t value,
enum ezdp\_tb\_color pre_color, bool dec_commit_bucket, bool dec_excess_bucket, struct
ezdp\_tb\_ctr\_result __cmem * result_value) [static]
```

Force decrement token buckets with the specified value (e.g. packet length) and get resulting color and bucket states.

**Parameters:**

- [in] *addr* - Address of counter
- [in] *value* - Value to add to TB counter, e.g. packet length in bytes
- [in] *pre\_color* - Pre-Color (e.g. packet original color)
- [in] *dec\_commit\_bucket* - true if you want to increment commit bucket
- [in] *dec\_excess\_bucket* - true if you want to increment excess bucket
- [out] *result\_value* - Address at CMEM to write result color

**Note:**

Can be used to implement hierarchical token bucket algorithms

**Returns:**

none

```
static __always_inline void ezdp_prefetch_tb_ctr(ezdp\_sum\_addr\_t addr) [static]
```

Prefetch token bucket counter into the local cache.

Load counter into the cache to hide latency of the accessing this counter.

**Parameters:**[in] *addr* - Address of counter**Returns:**

none

```
static __always_inline void ezdp_prefetch_tb_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of `ezdp_prefetch_tb_ctr`.

**Parameters:**[in] *addr* - Address of counter**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_write_hier_tb_ctr_cfg (ezdp\_sum\_addr\_t addr, struct ezdp\_hier\_tb\_ctr\_cfg * counter) [static]
```

Configure hierarchical token bucket counter.

**Parameters:**[in] *addr* - Address of counter[in] *counter* - counter configuration**Returns:**

void

```
static __always_inline void ezdp_write_hier_tb_ctr_cfg_async (ezdp\_sum\_addr\_t addr, struct ezdp\_hier\_tb\_ctr\_cfg * counter) [static]
```

Non blocking version of `ezdp_write_hier_tb_ctr_cfg`.

**Parameters:**[in] *addr* - Address of counter[out] *counter* - Address at CMEM to write counter config**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_hier_tb_ctr_cfg (ezdp\_sum\_addr\_t addr, struct ezdp\_hier\_tb\_ctr\_cfg __cmem * counter) [static]
```

Read hierarchical token bucket counter configuration.

**Parameters:**

[in] *addr* - Address of counter  
 [out] *counter* - Address at CMEM to write counter config

**Returns:**

void

```
static __always_inline void ezdp_inc_hier_tb_ctr (ezdp\_sum\_addr\_t addr,  uint16_t value,
bool update_ctr0,  bool update_acc1) [static]
```

Increment hierarchical token bucket counter accumulator(s) by the value specified.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - 14 bits value to add to TB counter, e.g. packet length in bytes  
 [in] *update\_ctr0* - true if you want to increment accumulator 0  
 [in] *update\_acc1* - true if you want to increment accumulator 1

**Returns:**

void

```
static __always_inline void ezdp_inc_hier_tb_ctr_async (ezdp\_sum\_addr\_t addr,  uint16_t value,
bool update_ctr0,  bool update_acc1) [static]
```

Non blocking version of `ezdp_inc_hier_tb_ctr`.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - 14 bits value to add to TB counter, e.g. packet length in bytes  
 [in] *update\_ctr0* - true if you want to increment accumulator 0  
 [in] *update\_acc1* - true if you want to increment accumulator 1

**Returns:**

void

```
static __always_inline void ezdp_read_and_inc_hier_tb_ctr (ezdp\_sum\_addr\_t addr,  uint16_t
value,  bool update_ctr0,  bool update_acc1,  struct ezdp\_hier\_tb\_result __cmem * result)
[static]
```

Increment hierarchical token bucket counter accumulator(s) by the value specified and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - 14 bits value to add to TB counter, e.g. packet length in bytes  
 [in] *update\_ctr0* - true if you want to increment accumulator 0  
 [in] *update\_acc1* - true if you want to increment accumulator 1  
 [in] *result* - Address at CMEM to write result

**Returns:**

void

```
static __always_inline void ezdp_update_hier_tb_ctr (ezdp\_sum\_addr\_t addr,  struct
ezdp\_hier\_tb\_update * ctr_update) [static]
```

Update hierarchical token bucket counter state, app bits or clear accumulators.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *ctr\_update* - counter update request structure

**Returns:**

void

```
static __always_inline void ezdp_update_hier_tb_ctr_async (ezdp_sum_addr_t addr, struct
ezdp_hier_tb_update * ctr_update) [static]
```

Non blocking version of ezdp\_inc\_hier\_tb\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *ctr\_update* - counter update request structure

**Returns:**

void

```
static __always_inline void ezdp_read_and_update_hier_tb_ctr (ezdp_sum_addr_t addr, struct
ezdp_hier_tb_update * ctr_update, struct ezdp_hier_tb_result __cmem * result) [static]
```

Update hierarchical token bucket counter state, app bits or clear accumulators and read previous counter value.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *ctr\_update* - counter update request structure  
 [in] *result* - Address at CMEM to write result

**Returns:**

void

```
static __always_inline void ezdp_change_state_hier_tb_ctr (ezdp_sum_addr_t addr, struct
ezdp_hier_tb_result __cmem * result) [static]
```

Change hierarchical token bucket state to Ph1.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *result* - Address at CMEM to write result

**Note:**

1. In case of success, must update BE TBs with the accumulators at the result

**Returns:**

void

```
static __always_inline void ezdp_write_watchdog_ctr_cfg (ezdp_sum_addr_t addr, struct
ezdp_watchdog_ctr_cfg * counter) [static]
```

Configure watchdog counter.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *counter* - counter configuration + value

**Returns:**

void

```
static __always_inline void ezdp_write_watchdog_ctr_cfg_async (ezdp_sum_addr_t addr,
struct ezdp_watchdog_ctr_cfg * counter) [static]
```

Non blocking version of ezdp\_write\_watchdog\_ctr\_cfg.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *counter* - counter configuration + value

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_read_watchdog_ctr_cfg (ezdp_sum_addr_t addr, struct
ezdp_watchdog_ctr_cfg __cmem * counter) [static]
```

Read watchdog counter configuration.

**Parameters:**

[in] *addr* - Address of counter  
 [out] *counter* - Address at CMEM to write counter config

**Returns:**

void

```
static __always_inline void ezdp_start_watchdog_ctr (ezdp_sum_addr_t counter_address)
[static]
```

Start the watchdog counter.

Sets the valid bit of the counter to one

**Parameters:**

[in] *counter\_address* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_start_watchdog_ctr_async (ezdp_sum_addr_t counter_address)
[static]
```

Non blocking version of ezdp\_start\_watchdog\_ctr.

**Parameters:**

[in] *counter\_address* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_inc_watchdog_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Increment the watchdog counter events by one.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_inc_watchdog_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of `ezdp_inc_watchdog_ctr`.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_check_watchdog_ctr (ezdp\_sum\_addr\_t counter_address,  
struct ezdp\_watchdog\_ctr\_check\_result __cmem * check_result) [static]
```

Check the number of events in the watchdog counter.

If the number of events is within the required range (min/max) the counter window is shifted. Otherwise, an alert is indicated and the counter operation is stopped.

**Parameters:**

[in] *counter\_address* - Address of counter

[out] *check\_result* - Address at CMEM to write result

**Returns:**

void

```
static __always_inline void ezdp_check_watchdog_ctr_async (ezdp\_sum\_addr\_t  
counter_address, struct ezdp\_watchdog\_ctr\_check\_result __cmem * check_result) [static]
```

Non blocking version of `ezdp_check_watchdog_ctr`.

**Parameters:**

[in] *counter\_address* - Address of counter

[out] *check\_result* - Address at CMEM to write result

**Returns:**

void



```
static __always_inline void ezdp_prefetch_watchdog_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Prefetch watchdog\_counter into the local cache.

Load counter into the cache to hide latency of the accessing this counter.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

none

```
static __always_inline void ezdp_prefetch_watchdog_ctr_async (ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of ezdp\_prefetch\_watchdog\_ctr.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline bool ezdp_init_ctr_msg_queue_desc (uint32_t partition_id, uint32_t queue_mask, ezdp\_ctr\_msg\_queue\_desc\_t * ctr_queue_desc, char __cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Initialize counter message queue descriptor.

**Parameters:**

[in] *partition\_id* - partition id of queue

[in] *queue\_mask* - relevant queue bit mask for stat\_desc 0 mean all queues

[out] *ctr\_queue\_desc* - counter message queue description

[in] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_CTR\_MSG\_QUEUE\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Returns:**

bool - true/false - success/fatal error

```
static __always_inline bool ezdp_read_ctr_msg (ezdp\_ctr\_msg\_queue\_desc\_t * ctr_queue_desc, struct ezdp\_ctr\_msg * msg, char __cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Read counter message from message queue.

**Parameters:**

[in] *ctr\_queue\_desc* - counter message queue description

[out] *msg* - message from queue

[in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_CTR\_MSG\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Returns:**

bool - true/false - success/fatal error

```
static __always_inline void ezdp_write_posted_ctr (ezdp\_sum\_addr\_t addr,   uint64_t value)
[static]
```

Initialize posted counter with the value specified.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - value to write

**Returns:**

void

```
static __always_inline void ezdp_write_posted_ctr_async (ezdp\_sum\_addr\_t addr,   uint64_t
value) [static]
```

Non blocking version of ezdp\_write\_posted\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *value* - value to set

**Returns:**

void

```
static __always_inline void ezdp_dual_write_posted_ctr (ezdp\_sum\_addr\_t addr,   uint64_t
counter1,   uint64_t counter2) [static]
```

Initialize two successive posted counter with the value specified.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *counter1* - value to set counter1  
 [in] *counter2* - value to set counter2

**Note:**

The operation is not atomic and implemented as two write to two successive counters

**Returns:**

void

```
static __always_inline void ezdp_dual_write_posted_ctr_async (ezdp\_sum\_addr\_t addr,
uint64_t counter1,   uint64_t counter2) [static]
```

Non blocking version of ezdp\_dual\_write\_posted\_ctr.

**Parameters:**

[in] *addr* - Address of counter  
 [in] *counter1* - value to set counter1  
 [in] *counter2* - value to set counter2

**Note:**

The operation is not atomic and implemented as two write to two successive counters

**Returns:**

void

```
static __always_inline void ezdp_add_posted_ctr (ezdp\_sum\_addr\_t addr, int32_t value)
[static]
```

Add signed value to posted counter.

**Parameters:**

[in] *addr* - Address of counter

[in] *value* - signed value to add

**Returns:**

void

```
static __always_inline void ezdp_add_posted_ctr_async (ezdp\_sum\_addr\_t addr, int32_t value)
[static]
```

Non blocking version of ezdp\_add\_posted\_ctr.

**Parameters:**

[in] *addr* - Address of counter

[in] *value* - signed value to add

**Returns:**

void

```
static __always_inline void ezdp_dual_add_posted_ctr (ezdp\_sum\_addr\_t addr, int16_t
counter1, int16_t counter2) [static]
```

Add signed values to two successive posted counters.

**Parameters:**

[in] *addr* - Address of counter

[in] *counter1* - signed value to add counter1

[in] *counter2* - signed value to add counter2

**Returns:**

void

```
static __always_inline void ezdp_dual_add_posted_ctr_async (ezdp\_sum\_addr\_t addr, int16_t
counter1, int16_t counter2) [static]
```

Non blocking version of ezdp\_add\_posted\_ctr\_dual.

**Parameters:**

[in] *addr* - Address of counter

[in] *counter1* - signed value to add counter1

[in] *counter2* - signed value to add counter2

**Returns:**

void

```
static __always_inline void ezdp_report_posted_ctr (ezdp\_sum\_addr\_t sum_addr, bool flush)
[static]
```

Generate posted counter value report.

**Parameters:**

[in] *sum\_addr* - summarize address

[in] *flush* - flush before report

**Returns:**

void

```
static __always_inline void ezdp_report_and_clear_posted_ctr (ezdp\_sum\_addr\_t sum_addr,
bool flush) [static]
```

Generate posted counter value report and reset the counter to zero.

**Parameters:**

[in] *sum\_addr* - summarize address

[in] *flush* - flush before report

**Returns:**

void

```
static __always_inline void ezdp_dual_report_posted_ctr (ezdp\_sum\_addr\_t addr, bool flush)
[static]
```

Generate posted counter value report for two successive counters.

**Parameters:**

[in] *addr* - Address of counter

[in] *flush* - flush before report

**Returns:**

void

```
static __always_inline void ezdp_dual_report_and_clear_posted_ctr (ezdp\_sum\_addr\_t addr,
bool flush) [static]
```

Generate posted counter value report for two successive counters and reset both counter values to zero.

**Parameters:**

[in] *addr* - Address of counter

[in] *flush* - flush before report

**Returns:**

void

```
static __always_inline void ezdp_reset_posted_ctr (ezdp\_sum\_addr\_t addr) [static]
```

Reset posted counter.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_reset_posted_ctr_async(ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of ezdp\_reset\_posted\_ctr.

**Parameters:**

[in] *addr* - Address of counter

**Returns:**

void

```
static __always_inline void ezdp_dual_reset_posted_ctr(ezdp\_sum\_addr\_t addr) [static]
```

Reset two successive posted counter.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

The operation is not atomic and implemented as two write to two successive counters

**Returns:**

void

```
static __always_inline void ezdp_dual_reset_posted_ctr_async(ezdp\_sum\_addr\_t addr) [static]
```

Non blocking version of ezdp\_dual\_reset\_posted\_ctr.

**Parameters:**

[in] *addr* - Address of counter

**Note:**

The operation is not atomic and implemented as two write to two successive counters

**Returns:**

void

```
static __always_inline bool ezdp_init_posted_ctr_msg_queue_desc(uint32_t partition_id,
uint32_t queue_mask, ezdp\_posted\_ctr\_msg\_queue\_desc\_t * posted_ctr_queue_desc, char
__cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Initialize posted counter message queue descriptor.

**Parameters:**

[in] *partition\_id* - partition id of queue

[in] *queue\_mask* - relevant queue bit mask for posted\_desc 0 mean all queues  
 [out] *posted\_ctr\_queue\_desc* - counter message queue description  
 [in] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_CTR\_MSG\_QUEUE\_WORK\_AREA\_SIZE  
 [in] *work\_area\_size* - size of work area pointer

**Returns:**

bool - true/false - success/fatal error

```
static __always_inline bool ezdp_read_posted_ctr_msg (ezdp\_posted\_ctr\_msg\_queue\_desc\_t *  
ctr_queue_desc, struct ezdp\_posted\_ctr\_msg * msg, char __cmem * work_area_ptr,  
uint32_t work_area_size) [static]
```

Read posted counter message from message queue.

**Parameters:**

[in] *ctr\_queue\_desc* - counter message queue description  
 [out] *msg* - message from queue  
 [in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_POSTED\_CTR\_MSG\_WORK\_AREA\_SIZE  
 [in] *work\_area\_size* - size of work area pointer

**Returns:**

- true/false - success/fatal error

## dpe/dp/include/ezdp\_counter\_defs.h File Reference

### Data Structures

- struct [ezdp\\_single\\_ctr\\_cfg](#)
- On-demand single value counter configuration definition. struct [ezdp\\_dual\\_ctr\\_result](#)
- On-demand dual value counter result value. struct [ezdp\\_dual\\_ctr](#)
- On-demand dual counter value. struct [ezdp\\_dual\\_ctr\\_cfg](#)
- On-demand dual counter configuration definition. struct [ezdp\\_tb\\_ctr\\_result](#)
- Token bucket counter result value definition. struct [ezdp\\_tb\\_ctr\\_cfg](#)
- Token bucket counter configuration definition. struct [ezdp\\_hier\\_tb\\_ug\\_app\\_bits](#)
- Application bits of Hierarchical token bucket for ultra green feature. struct [ezdp\\_hier\\_tb\\_ctr\\_cfg](#)
- Statistic hierarchical token bucket counter config structure (write cfg usage). struct [ezdp\\_hier\\_tb\\_result](#)
- Hierarchical token bucket counter result value definition. struct [ezdp\\_hier\\_tb\\_update](#)
- Hierarchical token bucket update counter definition. struct [ezdp\\_bitwise\\_ctr\\_cfg](#)
- On-demand bitwise counter configuration definition. struct [ezdp\\_watchdog\\_accumulative\\_window\\_cfg](#)
- Watchdog accumulative window configuration definition. struct [ezdp\\_watchdog\\_sliding\\_window\\_cfg](#)
- Watchdog sliding window configuration definition. struct [ezdp\\_watchdog\\_ctr\\_cfg](#)
- Watchdog counter configuration definition. struct [ezdp\\_watchdog\\_ctr\\_check\\_result](#)
- Watchdog counter check result definition. struct [ezdp\\_watchdog\\_ctr\\_start\\_result](#)
- Watchdog counter check result definition. struct [ezdp\\_ctr\\_msg](#)
- Counter message queue definition. struct [ezdp\\_posted\\_ctr\\_msg](#)

### Posted counter message queue definition. Defines

- #define [EZDP\\_CTR\\_MSG\\_QUEUE\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_init\_msgq\_desc\_working\_area)
- Work area minimal required size definitions. #define [EZDP\\_COUNTER\\_VERSION\\_MAJOR](#) 2
- #define [EZDP\\_COUNTER\\_VERSION\\_MINOR](#) 1
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_RESERVED0\\_10\\_SIZE](#) 11
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_RESERVED0\\_10\\_OFFSET](#) 0
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_SIZE](#) 1
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_OFFSET](#) 11
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_WORD\\_OFFSET](#) 11
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_MASK](#) (1 << EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_WORD\_OFFSET)
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ZERO\\_OFFSET](#) 12
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_REPORT\\_EXCEEDED\\_SIZE](#) 6
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_REPORT\\_EXCEEDED\\_OFFSET](#) 13
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_REPORT\\_EXCEEDED\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_REPORT\\_EXCEEDED\\_WORD\\_OFFSET](#) 13
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_SUB\\_TYPE\\_SIZE](#) 5
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_SUB\\_TYPE\\_OFFSET](#) 19
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_RESERVED32\\_63\\_SIZE](#) 32
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_RESERVED32\\_63\\_OFFSET](#) 32
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_VALUE\\_SIZE](#) 64
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_VALUE\\_OFFSET](#) 64
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_VALUE\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_VALUE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_SINGLE\\_CTR\\_CFG\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_MSB\\_SIZE](#) 31
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_MSB\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_MSB\\_WORD\\_SELECT](#) 0

- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_MSB\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_RESERVED31\\_SIZE](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_RESERVED31\\_OFFSET](#) 31
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_EVENT\\_VALUE\\_SIZE](#) 28
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_EVENT\\_VALUE\\_OFFSET](#) 32
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_EVENT\\_VALUE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_EVENT\\_VALUE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_LSB\\_SIZE](#) 4
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_LSB\\_OFFSET](#) 60
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_LSB\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_BYTE\\_VALUE\\_LSB\\_WORD\\_OFFSET](#) 28
- #define [EZDP\\_DUAL\\_CTR\\_RESULT\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_DUAL\\_CTR\\_BYTE\\_SIZE](#) 64
- #define [EZDP\\_DUAL\\_CTR\\_BYTE\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_BYTE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_BYTE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_EVENT\\_SIZE](#) 64
- #define [EZDP\\_DUAL\\_CTR\\_EVENT\\_OFFSET](#) 64
- #define [EZDP\\_DUAL\\_CTR\\_EVENT\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_DUAL\\_CTR\\_EVENT\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_RESERVED0\\_SIZE](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_RESERVED0\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_VALUE\\_SIZE\\_SIZE](#) 4
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_VALUE\\_SIZE\\_OFFSET](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_VALUE\\_SIZE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_VALUE\\_SIZE\\_WORD\\_OFFSET](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_EVENT\\_REPORT\\_EXCEEDED\\_SIZE](#) 6
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_EVENT\\_REPORT\\_EXCEEDED\\_OFFSET](#) 5
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_EVENT\\_REPORT\\_EXCEEDED\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_EVENT\\_REPORT\\_EXCEEDED\\_WORD\\_OFFSET](#) 5
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_SIZE](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_OFFSET](#) 11
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_WORD\\_OFFSET](#) 11
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_ENABLE\\_EXCEED\\_MESSAGE\\_MASK](#) (1 << EZDP\_DUAL\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_WORD\_OFFSET)
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_CLR\\_ON\\_GC\\_SIZE](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_CLR\\_ON\\_GC\\_OFFSET](#) 12
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_REPORT\\_EXCEEDED\\_SIZE](#) 6
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_REPORT\\_EXCEEDED\\_OFFSET](#) 13
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_REPORT\\_EXCEEDED\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_BYTE\\_REPORT\\_EXCEEDED\\_WORD\\_OFFSET](#) 13
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_RESERVED19\\_23\\_SIZE](#) 5
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_RESERVED19\\_23\\_OFFSET](#) 19
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_VALUE\\_SIZE](#) 128
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_VALUE\\_OFFSET](#) 32
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_VALUE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_VALUE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_CTR\\_CFG\\_WORD\\_COUNT](#) 5
- #define [EZDP\\_TB\\_CTR\\_RESULT\\_RESERVED0\\_31\\_SIZE](#) 32
- #define [EZDP\\_TB\\_CTR\\_RESULT\\_RESERVED0\\_31\\_OFFSET](#) 0
- #define [EZDP\\_TB\\_CTR\\_RESULT\\_COLOR\\_SIZE](#) 2
- #define [EZDP\\_TB\\_CTR\\_RESULT\\_COLOR\\_OFFSET](#) 32
- #define [EZDP\\_TB\\_CTR\\_RESULT\\_COLOR\\_WORD\\_SELECT](#) 1



```

• #define EZDP_TB_CTR_RESULT_COLOR_WORD_OFFSET 0
• #define EZDP_TB_CTR_RESULT_RESERVED34_57_SIZE 22
• #define EZDP_TB_CTR_RESULT_RESERVED34_57_OFFSET 34
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_SIZE 1
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_OFFSET 56
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_WORD_SELECT 1
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_WORD_OFFSET 24
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_WORD_OFFSET)
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_SIZE 1
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_OFFSET 57
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_WORD_SELECT 1
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_WORD_OFFSET 25
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_WORD_OFFSET)
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_SIZE 1
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_OFFSET 58
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_WORD_SELECT 1
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_WORD_OFFSET 26
• #define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_WORD_OFFSET)
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_SIZE 1
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_OFFSET 59
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_WORD_SELECT 1
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_WORD_OFFSET 27
• #define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_WORD_OFFSET)
• #define EZDP_TB_CTR_RESULT_RESERVED60_63_SIZE 4
• #define EZDP_TB_CTR_RESULT_RESERVED60_63_OFFSET 60
• #define EZDP_TB_CTR_RESULT_WORD_COUNT 2
• #define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_SIZE 10
• #define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_OFFSET 0
• #define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_WORD_SELECT 0
• #define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_WORD_OFFSET 0
• #define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_SIZE 10
• #define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_OFFSET 10
• #define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_WORD_SELECT 0
• #define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_WORD_OFFSET 10
• #define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_SIZE 4
• #define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_OFFSET 20
• #define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_WORD_SELECT 0
• #define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_WORD_OFFSET 20
• #define EZDP_TB_CTR_CFG_COLOR_AWARE_SIZE 1
• #define EZDP_TB_CTR_CFG_COLOR_AWARE_OFFSET 24
• #define EZDP_TB_CTR_CFG_COLOR_AWARE_WORD_SELECT 0
• #define EZDP_TB_CTR_CFG_COLOR_AWARE_WORD_OFFSET 24
• #define EZDP_TB_CTR_CFG_COLOR_AWARE_MASK (1 <<
EZDP_TB_CTR_CFG_COLOR_AWARE_WORD_OFFSET)
• #define EZDP_TB_CTR_CFG_COUPLING_FLAG_SIZE 1
• #define EZDP_TB_CTR_CFG_COUPLING_FLAG_OFFSET 25
• #define EZDP_TB_CTR_CFG_COUPLING_FLAG_WORD_SELECT 0
• #define EZDP_TB_CTR_CFG_COUPLING_FLAG_WORD_OFFSET 25
• #define EZDP_TB_CTR_CFG_COUPLING_FLAG_MASK (1 <<
EZDP_TB_CTR_CFG_COUPLING_FLAG_WORD_OFFSET)
• #define EZDP_TB_CTR_CFG_RESERVED26_31_SIZE 6
• #define EZDP_TB_CTR_CFG_RESERVED26_31_OFFSET 26
• #define EZDP_TB_CTR_CFG_RESERVED32_63_SIZE 32

```

- #define [EZDP\\_TB\\_CTR\\_CFG\\_RESERVED32\\_63\\_OFFSET](#) 32
- #define [EZDP\\_TB\\_CTR\\_CFG\\_RESERVED64\\_95\\_SIZE](#) 32
- #define [EZDP\\_TB\\_CTR\\_CFG\\_RESERVED64\\_95\\_OFFSET](#) 64
- #define [EZDP\\_TB\\_CTR\\_CFG\\_RESERVED96\\_127\\_SIZE](#) 32
- #define [EZDP\\_TB\\_CTR\\_CFG\\_RESERVED96\\_127\\_OFFSET](#) 96
- #define [EZDP\\_TB\\_CTR\\_CFG\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_COLOR\\_STATE\\_G\\_SIZE](#) 2
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_COLOR\\_STATE\\_G\\_OFFSET](#) 0
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_COLOR\\_STATE\\_Y\\_SIZE](#) 2
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_COLOR\\_STATE\\_Y\\_OFFSET](#) 2
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_EIGHTH\\_MODE\\_RET\\_BITS\\_SIZE](#) 4
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_EIGHTH\\_MODE\\_RET\\_BITS\\_OFFSET](#) 4
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_APP\\_BITS\\_SIZE](#) 16
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_APP\\_BITS\\_OFFSET](#) 8
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_RESERVED24\\_31\\_SIZE](#) 8
- #define [EZDP\\_HIER\\_TB\\_UG\\_APP\\_BITS\\_RESERVED24\\_31\\_OFFSET](#) 24
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_RESERVED0\\_1\\_SIZE](#) 2
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_RESERVED0\\_1\\_OFFSET](#) 0
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_UPDT\\_THRESHOLD\\_SIZE](#) 5
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_UPDT\\_THRESHOLD\\_OFFSET](#) 2
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_UPDT\\_THRESHOLD\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_UPDT\\_THRESHOLD\\_WORD\\_OFFSET](#) 2
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_FAIL\\_THRESHOLD\\_SIZE](#) 5
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_FAIL\\_THRESHOLD\\_OFFSET](#) 7
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_FAIL\\_THRESHOLD\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR1\\_FAIL\\_THRESHOLD\\_WORD\\_OFFSET](#) 7
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_UPDT\\_THRESHOLD\\_SIZE](#) 5
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_UPDT\\_THRESHOLD\\_OFFSET](#) 12
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_UPDT\\_THRESHOLD\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_UPDT\\_THRESHOLD\\_WORD\\_OFFSET](#) 12
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_FAIL\\_THRESHOLD\\_SIZE](#) 5
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_FAIL\\_THRESHOLD\\_OFFSET](#) 17
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_FAIL\\_THRESHOLD\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR\\_SUM\\_FAIL\\_THRESHOLD\\_WORD\\_OFFSET](#) 17
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_RESERVED22\\_26\\_SIZE](#) 5
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_RESERVED22\\_26\\_OFFSET](#) 22
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_FAIL\\_THRESHOLD\\_SIZE](#) 5
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_FAIL\\_THRESHOLD\\_OFFSET](#) 27
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_FAIL\\_THRESHOLD\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_FAIL\\_THRESHOLD\\_WORD\\_OFFSET](#) 27
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_APP\\_BITS\\_SIZE](#) 24
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_APP\\_BITS\\_OFFSET](#) 32
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_APP\\_BITS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_APP\\_BITS\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_UPDT\\_THRESHOLD\\_SIZE](#) 5
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_UPDT\\_THRESHOLD\\_OFFSET](#) 56
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_UPDT\\_THRESHOLD\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_CTR0\\_UPDT\\_THRESHOLD\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_TIMESTAMP\\_THRESHOLD\\_SIZE](#) 2
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_TIMESTAMP\\_THRESHOLD\\_OFFSET](#) 61
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_TIMESTAMP\\_THRESHOLD\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_TIMESTAMP\\_THRESHOLD\\_WORD\\_OFFSET](#) 29
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_RESERVED63\\_SIZE](#) 1
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_RESERVED63\\_OFFSET](#) 63
- #define [EZDP\\_HIER\\_TB\\_CTR\\_CFG\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_HIER\\_TB\\_RESULT\\_RESERVED0\\_9\\_SIZE](#) 10
- #define [EZDP\\_HIER\\_TB\\_RESULT\\_RESERVED0\\_9\\_OFFSET](#) 0

```

• #define EZDP_HIER_TB_RESULT_CTR1_SIZE 18
• #define EZDP_HIER_TB_RESULT_CTR1_OFFSET 10
• #define EZDP_HIER_TB_RESULT_CTR1_WORD_SELECT 0
• #define EZDP_HIER_TB_RESULT_CTR1_WORD_OFFSET 10
• #define EZDP_HIER_TB_RESULT_STATE_SIZE 2
• #define EZDP_HIER_TB_RESULT_STATE_OFFSET 28
• #define EZDP_HIER_TB_RESULT_STATE_WORD_SELECT 0
• #define EZDP_HIER_TB_RESULT_STATE_WORD_OFFSET 28
• #define EZDP_HIER_TB_RESULT_UPDATE_TASK_SIZE 1
• #define EZDP_HIER_TB_RESULT_UPDATE_TASK_OFFSET 30
• #define EZDP_HIER_TB_RESULT_UPDATE_TASK_WORD_SELECT 0
• #define EZDP_HIER_TB_RESULT_UPDATE_TASK_WORD_OFFSET 30
• #define EZDP_HIER_TB_RESULT_UPDATE_TASK_MASK (1 <<
EZDP_HIER_TB_RESULT_UPDATE_TASK_WORD_OFFSET)
• #define EZDP_HIER_TB_RESULT_FAIL_SIZE 1
• #define EZDP_HIER_TB_RESULT_FAIL_OFFSET 31
• #define EZDP_HIER_TB_RESULT_FAIL_WORD_SELECT 0
• #define EZDP_HIER_TB_RESULT_FAIL_WORD_OFFSET 31
• #define EZDP_HIER_TB_RESULT_FAIL_MASK (1 <<
EZDP_HIER_TB_RESULT_FAIL_WORD_OFFSET)
• #define EZDP_HIER_TB_RESULT_APP_BITS_SIZE 24
• #define EZDP_HIER_TB_RESULT_APP_BITS_OFFSET 32
• #define EZDP_HIER_TB_RESULT_APP_BITS_WORD_SELECT 1
• #define EZDP_HIER_TB_RESULT_APP_BITS_WORD_OFFSET 0
• #define EZDP_HIER_TB_RESULT_RESERVED56_63_SIZE 8
• #define EZDP_HIER_TB_RESULT_RESERVED56_63_OFFSET 56
• #define EZDP_HIER_TB_RESULT_CTR0_SIZE 18
• #define EZDP_HIER_TB_RESULT_CTR0_OFFSET 64
• #define EZDP_HIER_TB_RESULT_CTR0_WORD_SELECT 2
• #define EZDP_HIER_TB_RESULT_CTR0_WORD_OFFSET 0
• #define EZDP_HIER_TB_RESULT_RESERVED82_95_SIZE 14
• #define EZDP_HIER_TB_RESULT_RESERVED82_95_OFFSET 82
• #define EZDP_HIER_TB_RESULT_WORD_COUNT 3
• #define EZDP_HIER_TB_UPDATE_APP_BITS_SIZE 24
• #define EZDP_HIER_TB_UPDATE_APP_BITS_OFFSET 0
• #define EZDP_HIER_TB_UPDATE_RESERVED24_27_SIZE 4
• #define EZDP_HIER_TB_UPDATE_RESERVED24_27_OFFSET 24
• #define EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_SIZE 1
• #define EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_OFFSET 28
• #define EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_MASK (1 <<
EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_OFFSET)
• #define EZDP_HIER_TB_UPDATE_SET_APP_BITS_SIZE 1
• #define EZDP_HIER_TB_UPDATE_SET_APP_BITS_OFFSET 29
• #define EZDP_HIER_TB_UPDATE_SET_APP_BITS_MASK (1 <<
EZDP_HIER_TB_UPDATE_SET_APP_BITS_OFFSET)
• #define EZDP_HIER_TB_UPDATE_CLR_CTR_SIZE 1
• #define EZDP_HIER_TB_UPDATE_CLR_CTR_OFFSET 30
• #define EZDP_HIER_TB_UPDATE_CLR_CTR_MASK (1 <<
EZDP_HIER_TB_UPDATE_CLR_CTR_OFFSET)
• #define EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_SIZE 1
• #define EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_OFFSET 31
• #define EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_MASK (1 <<
EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_OFFSET)
• #define EZDP_BITWISE_CTR_CFG_RESERVED0_18_SIZE 19
• #define EZDP_BITWISE_CTR_CFG_RESERVED0_18_OFFSET 0
• #define EZDP_BITWISE_CTR_CFG_SUB_TYPE_SIZE 5
• #define EZDP_BITWISE_CTR_CFG_SUB_TYPE_OFFSET 19

```

- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_RESERVED32\\_63\\_SIZE](#) 32
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_RESERVED32\\_63\\_OFFSET](#) 32
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_DATA\\_SIZE](#) 64
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_DATA\\_OFFSET](#) 64
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_DATA\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_BITWISE\\_CTR\\_CFG\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_ACCUMULATIVE\\_EVENTS\\_SIZE](#) 5
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_ACCUMULATIVE\\_EVENTS\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_ACCUMULATIVE\\_EVENTS\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_ACCUMULATIVE\\_EVENTS\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_RESERVED5\\_28\\_SIZE](#) 24
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_RESERVED5\\_28\\_OFFSET](#) 5
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_MAX\\_THRESHOLD\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_MAX\\_THRESHOLD\\_ALERT\\_OFFSET](#) 29
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_MIN\\_THRESHOLD\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_MIN\\_THRESHOLD\\_ALERT\\_OFFSET](#) 30
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_ALERT\\_OFFSET](#) 31
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_LAST\\_EVENTS\\_SIZE](#) 12
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_LAST\\_EVENTS\\_OFFSET](#) 32
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_LAST\\_EVENTS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_LAST\\_EVENTS\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_CURR\\_EVENTS\\_SIZE](#) 12
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_CURR\\_EVENTS\\_OFFSET](#) 44
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_CURR\\_EVENTS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_CURR\\_EVENTS\\_WORD\\_OFFSET](#) 12
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_INIT\\_BIT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_INIT\\_BIT\\_OFFSET](#) 56
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_VALID\\_OFFSET](#) 57
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_VALID\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_VALID\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_VALID\\_MASK](#) (1 << EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_SIZE](#) 4
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_OFFSET](#) 58
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_PARITY\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_PARITY\\_OFFSET](#) 62
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_RESERVED63\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_RESERVED63\\_OFFSET](#) 63
- #define [EZDP\\_WATCHDOG\\_ACCUMULATIVE\\_WINDOW\\_CFG\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_WINDOWS\\_SIZE](#) 5
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_WINDOWS\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_WINDOWS\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_WINDOWS\\_WORD\\_OFFSET](#) 0

- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_RESERVED5\\_28\\_SIZE](#) 24
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_RESERVED5\\_28\\_OFFSET](#) 5
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_MAX\\_THRESHOLD\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_MAX\\_THRESHOLD\\_ALERT\\_OFFSET](#) 29
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_MIN\\_THRESHOLD\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_MIN\\_THRESHOLD\\_ALERT\\_OFFSET](#) 30
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_ALERT\\_OFFSET](#) 31
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_COUNTERS\\_SIZE](#) 24
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_COUNTERS\\_OFFSET](#) 32
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_COUNTERS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_COUNTERS\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_RESERVED56\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_RESERVED56\\_OFFSET](#) 56
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_OFFSET](#) 57
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_VALID\\_MASK](#) (1 << EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_SIZE](#) 4
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_OFFSET](#) 58
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_PROFILE\\_ID\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_PARITY\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_PARITY\\_OFFSET](#) 62
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_RESERVED63\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_RESERVED63\\_OFFSET](#) 63
- #define [EZDP\\_WATCHDOG\\_SLIDING\\_WINDOW\\_CFG\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_RESERVED0\\_18\\_SIZE](#) 19
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_RESERVED0\\_18\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_SUB\\_TYPE\\_SIZE](#) 5
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_SUB\\_TYPE\\_OFFSET](#) 19
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_RESERVED32\\_63\\_SIZE](#) 32
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_RESERVED32\\_63\\_OFFSET](#) 32
- #define [EZDP\\_WATCHDOG\\_CTR\\_CFG\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_ACCUMULATIVE\\_EVENTS\\_SIZE](#) 4
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_ACCUMULATIVE\\_EVENTS\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_RESERVED4\\_28\\_SIZE](#) 25
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_RESERVED4\\_28\\_OFFSET](#) 4
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MAX\\_THRESHOLD\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MAX\\_THRESHOLD\\_ALERT\\_OFFSET](#) 29
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MAX\\_THRESHOLD\\_ALERT\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MAX\\_THRESHOLD\\_ALERT\\_WORD\\_OFFSET](#) 29
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MAX\\_THRESHOLD\\_ALERT\\_MASK](#) (1 << EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MAX\_THRESHOLD\_ALERT\_WORD\_OFFSET)
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MIN\\_THRESHOLD\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MIN\\_THRESHOLD\\_ALERT\\_OFFSET](#) 30
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MIN\\_THRESHOLD\\_ALERT\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MIN\\_THRESHOLD\\_ALERT\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_MIN\\_THRESHOLD\\_ALERT\\_MASK](#) (1 << EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MIN\_THRESHOLD\_ALERT\_WORD\_OFFSET)
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_ALERT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_ALERT\\_OFFSET](#) 31



- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_ALERT\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_ALERT\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_ALERT\\_MASK](#) (1 << EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_WORD\_OFFSET)
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_LAST\\_EVENTS\\_SIZE](#) 12
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_LAST\\_EVENTS\\_OFFSET](#) 32
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_CURR\\_EVENTS\\_SIZE](#) 12
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_CURR\\_EVENTS\\_OFFSET](#) 44
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_INIT\\_BIT\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_INIT\\_BIT\\_OFFSET](#) 56
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_VALID\\_OFFSET](#) 57
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_PROFILE\\_ID\\_SIZE](#) 4
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_PROFILE\\_ID\\_OFFSET](#) 58
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_RESERVED62\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_RESERVED62\\_OFFSET](#) 62
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_WINDOW\\_RELATED\\_SIZE](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_WINDOW\\_RELATED\\_OFFSET](#) 63
- #define [EZDP\\_WATCHDOG\\_CTR\\_CHECK\\_RESULT\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_MSB\\_SIZE](#) 32
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_MSB\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_MSB\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_MSB\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_LSB\\_SIZE](#) 32
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_LSB\\_OFFSET](#) 32
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_LSB\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_LSB\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_WATCHDOG\\_CTR\\_START\\_RESULT\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_CTR\\_MSG\\_COUNTER\\_TYPE\\_SIZE](#) 2
- #define [EZDP\\_CTR\\_MSG\\_COUNTER\\_TYPE\\_OFFSET](#) 0
- #define [EZDP\\_CTR\\_MSG\\_COUNTER\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_CTR\\_MSG\\_COUNTER\\_TYPE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_CTR\\_MSG\\_MSG\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_CTR\\_MSG\\_MSG\\_TYPE\\_OFFSET](#) 2
- #define [EZDP\\_CTR\\_MSG\\_MSG\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_CTR\\_MSG\\_MSG\\_TYPE\\_WORD\\_OFFSET](#) 2
- #define [EZDP\\_CTR\\_MSG\\_OVERFLOW\\_SIZE](#) 1
- #define [EZDP\\_CTR\\_MSG\\_OVERFLOW\\_OFFSET](#) 5
- #define [EZDP\\_CTR\\_MSG\\_OVERFLOW\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_CTR\\_MSG\\_OVERFLOW\\_WORD\\_OFFSET](#) 5
- #define [EZDP\\_CTR\\_MSG\\_OVERFLOW\\_MASK](#) (1 << EZDP\_CTR\_MSG\_OVERFLOW\_WORD\_OFFSET)
- #define [EZDP\\_CTR\\_MSG\\_RESERVED6\\_SIZE](#) 1
- #define [EZDP\\_CTR\\_MSG\\_RESERVED6\\_OFFSET](#) 6
- #define [EZDP\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_SIZE](#) 1
- #define [EZDP\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_OFFSET](#) 7
- #define [EZDP\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_WORD\\_OFFSET](#) 7
- #define [EZDP\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_MASK](#) (1 << EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_WORD\_OFFSET)
- #define [EZDP\\_CTR\\_MSG\\_RESERVED8\\_23\\_SIZE](#) 16
- #define [EZDP\\_CTR\\_MSG\\_RESERVED8\\_23\\_OFFSET](#) 8
- #define [EZDP\\_CTR\\_MSG\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_CTR\\_MSG\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_CTR\\_MSG\\_SUM\\_ADDR\\_SIZE](#) 32
- #define [EZDP\\_CTR\\_MSG\\_SUM\\_ADDR\\_OFFSET](#) 32
- #define [EZDP\\_CTR\\_MSG\\_SUM\\_ADDR\\_WORD\\_SELECT](#) 1

- #define [EZDP\\_CTR\\_MSG\\_SUM\\_ADDR\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_CTR\\_MSG\\_WORD\\_COUNT](#) 6
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_MSG\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_MSG\\_TYPE\\_OFFSET](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_MSG\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_MSG\\_TYPE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_FLUSH\\_SIZE](#) 1
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_FLUSH\\_OFFSET](#) 3
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_FLUSH\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_FLUSH\\_WORD\\_OFFSET](#) 3
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_FLUSH\\_MASK](#) (1 << EZDP\_POSTED\_CTR\_MSG\_FLUSH\_WORD\_OFFSET)
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_CLEAR\\_SIZE](#) 1
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_CLEAR\\_OFFSET](#) 4
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_CLEAR\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_CLEAR\\_WORD\\_OFFSET](#) 4
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_CLEAR\\_MASK](#) (1 << EZDP\_POSTED\_CTR\_MSG\_CLEAR\_WORD\_OFFSET)
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_RESERVED5\\_6\\_SIZE](#) 2
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_RESERVED5\\_6\\_OFFSET](#) 5
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_SIZE](#) 1
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_OFFSET](#) 7
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_WORD\\_OFFSET](#) 7
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_OVERRUN\\_ERROR\\_CONDITION\\_MASK](#) (1 << EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_WORD\_OFFSET)
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_RESERVED8\\_23\\_SIZE](#) 16
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_RESERVED8\\_23\\_OFFSET](#) 8
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_SUM\\_ADDR\\_SIZE](#) 32
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_SUM\\_ADDR\\_OFFSET](#) 32
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_SUM\\_ADDR\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_SUM\\_ADDR\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_VALUE\\_SIZE](#) 64
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_VALUE\\_OFFSET](#) 64
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_VALUE\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_VALUE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_CTR\\_MSG\\_WORK\\_AREA\\_SIZE](#) ((sizeof(struct ezdp\_stat\_msg\_int) > sizeof(struct ezdp\_double\_ctr\_int)/2) ? sizeof(struct ezdp\_stat\_msg\_int) : sizeof(struct ezdp\_double\_ctr\_int)/2)
- #define [EZDP\\_POSTED\\_CTR\\_MSG\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_stat\_msg\_int)

## Typedefs

- typedef uint32\_t [ezdp\\_hier\\_tb\\_ug\\_app\\_bits\\_t](#)
- typedef uint32\_t [ezdp\\_hier\\_tb\\_update\\_t](#)
- typedef struct ezdp\_get\_color\_hier\_tb\_ctr\_working\_area [ezdp\\_get\\_color\\_hier\\_tb\\_ctr\\_working\\_area\\_t](#)
- typedef uint32\_t(\* [EZDP\\_HIER\\_TB\\_UPDATE\\_BES](#))(uint32\_t app\_bits, uint32\_t ctr\_g, uint32\_t ctr\_y, [ezdp\\_get\\_color\\_hier\\_tb\\_ctr\\_working\\_area\\_t](#) \_\_cmem \*wa\_ptr, void \*user\_data)
- Type definition for the data plane application's updating BE TBs according to algorithm and user information.
- typedef struct ezdp\_msgq\_desc [ezdp\\_ctr\\_msg\\_queue\\_desc\\_t](#)
- Statistic message queue descriptor. typedef struct ezdp\_msgq\_desc [ezdp\\_posted\\_ctr\\_msg\\_queue\\_desc\\_t](#)

## Posted message queue descriptor. Enumerations

- enum [ezdp\\_tb\\_color](#) { [EZDP\\_RED\\_TRAFFIC](#) = 0x0, [EZDP\\_YELLOW\\_TRAFFIC](#) = 0x1, [EZDP\\_GREEN\\_TRAFFIC](#) = 0x2 }

*tb color possible values.*

- enum [ezdp\\_tb\\_algo](#) { [EZDP\\_INACTIVE](#) = 0x0, [EZDP\\_SINGLE\\_BUCKET](#) = 0x1, [EZDP\\_SR\\_TCM](#) = 0x4, [EZDP\\_TR\\_TCM](#) = 5, [EZDP\\_TR\\_TCM\\_MEF](#) = 0x8 }

*tb algo possible values.*

- enum [ezdp\\_hier\\_tb\\_state](#) { [EZDP\\_UG\\_FAST\\_PATH](#) = 0x0, [EZDP\\_UG\\_SLOW\\_PATH](#) = 0x3 }

*hier tb state possible values.*

- enum [ezdp\\_ctr\\_type](#) { [EZDP\\_SINGLE\\_CTR](#) = 0x0, [EZDP\\_DUAL\\_CTR](#) = 0x1 }

*ctr type possible values.*

- enum [ezdp\\_ctr\\_msg\\_type](#) { [EZDP\\_PERIODIC\\_CTR\\_MSG](#) = 0x0, [EZDP\\_THRESHOLD\\_CTR\\_MSG](#) = 0x1, [EZDP\\_NULL\\_CTR\\_MSG](#) = 0x3 }

*ctr msg type possible values.*

- enum [ezdp\\_msg\\_posted\\_ctr\\_type](#) { [EZDP\\_PERIODIC\\_POSTED\\_CTR\\_MSG](#) = 0x0, [EZDP\\_THRESHOLD\\_POSTED\\_CTR\\_MSG](#) = 0x1, [EZDP\\_REPORT\\_POSTED\\_CTR\\_MSG](#) = 0x2, [EZDP\\_NULL\\_POSTED\\_CTR\\_MSG](#) = 0x3 }

*msg posted ctr type possible values.*

- enum [ezdp\\_bitwise\\_size](#) { [EZDP\\_1\\_BITS](#) = EZASM\_BITWISE\_SIZE\_1\_BIT, [EZDP\\_2\\_BITS](#) = EZASM\_BITWISE\_SIZE\_2\_BITS, [EZDP\\_4\\_BITS](#) = EZASM\_BITWISE\_SIZE\_4\_BITS, [EZDP\\_8\\_BITS](#) = EZASM\_BITWISE\_SIZE\_8\_BITS, [EZDP\\_16\\_BITS](#) = EZASM\_BITWISE\_SIZE\_16\_BITS }

*bitwise size possible values.*

## Variables

- enum [ezdp\\_tb\\_color](#)(\* [EZDP\\_HIER\\_TB\\_CALC\\_COLOR](#))(uint32\_t app\_bits, enum [ezdp\\_tb\\_color](#) pre\_color, uint32\_t \*user\_data)

*User Call Back function calc color according to algorithm and user information.*

## Define Documentation

```
#define EZDP_CTR_MSG_QUEUE_WORK_AREA_SIZE  sizeof(struct
ezdp_init_msgq_desc_working_area )
```

Work area minimal required size definitions.



```
#define EZDP_COUNTER_VERSION_MAJOR 2

#define EZDP_COUNTER_VERSION_MINOR 1

#define EZDP_SINGLE_CTR_CFG_RESERVED0_10_SIZE 11

#define EZDP_SINGLE_CTR_CFG_RESERVED0_10_OFFSET 0

#define EZDP_SINGLE_CTR_CFG_ENABLE_EXCEED_MESSAGE_SIZE 1

#define EZDP_SINGLE_CTR_CFG_ENABLE_EXCEED_MESSAGE_OFFSET 11

#define EZDP_SINGLE_CTR_CFG_ENABLE_EXCEED_MESSAGE_WORD_SELECT 0

#define EZDP_SINGLE_CTR_CFG_ENABLE_EXCEED_MESSAGE_WORD_OFFSET 11

#define EZDP_SINGLE_CTR_CFG_ENABLE_EXCEED_MESSAGE_MASK (1 <<
EZDP_SINGLE_CTR_CFG_ENABLE_EXCEED_MESSAGE_WORD_OFFSET)

#define EZDP_SINGLE_CTR_CFG_ZERO_SIZE 1

#define EZDP_SINGLE_CTR_CFG_ZERO_OFFSET 12

#define EZDP_SINGLE_CTR_CFG_REPORT_EXCEEDED_SIZE 6

#define EZDP_SINGLE_CTR_CFG_REPORT_EXCEEDED_OFFSET 13

#define EZDP_SINGLE_CTR_CFG_REPORT_EXCEEDED_WORD_SELECT 0

#define EZDP_SINGLE_CTR_CFG_REPORT_EXCEEDED_WORD_OFFSET 13

#define EZDP_SINGLE_CTR_CFG_SUB_TYPE_SIZE 5

#define EZDP_SINGLE_CTR_CFG_SUB_TYPE_OFFSET 19

#define EZDP_SINGLE_CTR_CFG_ECC_SIZE 8

#define EZDP_SINGLE_CTR_CFG_ECC_OFFSET 24

#define EZDP_SINGLE_CTR_CFG_RESERVED32_63_SIZE 32

#define EZDP_SINGLE_CTR_CFG_RESERVED32_63_OFFSET 32

#define EZDP_SINGLE_CTR_CFG_VALUE_SIZE 64

#define EZDP_SINGLE_CTR_CFG_VALUE_OFFSET 64

#define EZDP_SINGLE_CTR_CFG_VALUE_WORD_SELECT 2

#define EZDP_SINGLE_CTR_CFG_VALUE_WORD_OFFSET 0

#define EZDP_SINGLE_CTR_CFG_WORD_COUNT 4
```

```
#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_SIZE 31

#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_OFFSET 0

#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_WORD_SELECT 0

#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_WORD_OFFSET 0

#define EZDP_DUAL_CTR_RESULT_RESERVED31_SIZE 1

#define EZDP_DUAL_CTR_RESULT_RESERVED31_OFFSET 31

#define EZDP_DUAL_CTR_RESULT_EVENT_VALUE_SIZE 28

#define EZDP_DUAL_CTR_RESULT_EVENT_VALUE_OFFSET 32

#define EZDP_DUAL_CTR_RESULT_EVENT_VALUE_WORD_SELECT 1

#define EZDP_DUAL_CTR_RESULT_EVENT_VALUE_WORD_OFFSET 0

#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_SIZE 4

#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_OFFSET 60

#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_WORD_SELECT 1

#define EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_WORD_OFFSET 28

#define EZDP_DUAL_CTR_RESULT_WORD_COUNT 2

#define EZDP_DUAL_CTR_BYTE_SIZE 64

#define EZDP_DUAL_CTR_BYTE_OFFSET 0

#define EZDP_DUAL_CTR_BYTE_WORD_SELECT 0

#define EZDP_DUAL_CTR_BYTE_WORD_OFFSET 0

#define EZDP_DUAL_CTR_EVENT_SIZE 64

#define EZDP_DUAL_CTR_EVENT_OFFSET 64

#define EZDP_DUAL_CTR_EVENT_WORD_SELECT 2

#define EZDP_DUAL_CTR_EVENT_WORD_OFFSET 0

#define EZDP_DUAL_CTR_WORD_COUNT 4

#define EZDP_DUAL_CTR_CFG_RESERVED0_SIZE 1

#define EZDP_DUAL_CTR_CFG_RESERVED0_OFFSET 0
```

```
#define EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_SIZE 4

#define EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_OFFSET 1

#define EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_WORD_SELECT 0

#define EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_WORD_OFFSET 1

#define EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEEDED_SIZE 6

#define EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEEDED_OFFSET 5

#define EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEEDED_WORD_SELECT 0

#define EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEEDED_WORD_OFFSET 5

#define EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MESSAGE_SIZE 1

#define EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MESSAGE_OFFSET 11

#define EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MESSAGE_WORD_SELECT 0

#define EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MESSAGE_WORD_OFFSET 11

#define EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MESSAGE_MASK (1 <<
EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MESSAGE_WORD_OFFSET)

#define EZDP_DUAL_CTR_CFG_CLR_ON_GC_SIZE 1

#define EZDP_DUAL_CTR_CFG_CLR_ON_GC_OFFSET 12

#define EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEEDED_SIZE 6

#define EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEEDED_OFFSET 13

#define EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEEDED_WORD_SELECT 0

#define EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEEDED_WORD_OFFSET 13

#define EZDP_DUAL_CTR_CFG_RESERVED19_23_SIZE 5

#define EZDP_DUAL_CTR_CFG_RESERVED19_23_OFFSET 19

#define EZDP_DUAL_CTR_CFG_ECC_SIZE 8

#define EZDP_DUAL_CTR_CFG_ECC_OFFSET 24

#define EZDP_DUAL_CTR_CFG_VALUE_SIZE 128

#define EZDP_DUAL_CTR_CFG_VALUE_OFFSET 32

#define EZDP_DUAL_CTR_CFG_VALUE_WORD_SELECT 1
```

```
#define EZDP_DUAL_CTR_CFG_VALUE_WORD_OFFSET 0

#define EZDP_DUAL_CTR_CFG_WORD_COUNT 5

#define EZDP_TB_CTR_RESULT_RESERVED0_31_SIZE 32

#define EZDP_TB_CTR_RESULT_RESERVED0_31_OFFSET 0

#define EZDP_TB_CTR_RESULT_COLOR_SIZE 2

#define EZDP_TB_CTR_RESULT_COLOR_OFFSET 32

#define EZDP_TB_CTR_RESULT_COLOR_WORD_SELECT 1

#define EZDP_TB_CTR_RESULT_COLOR_WORD_OFFSET 0

#define EZDP_TB_CTR_RESULT_RESERVED34_57_SIZE 22

#define EZDP_TB_CTR_RESULT_RESERVED34_57_OFFSET 34

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_SIZE 1

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_OFFSET 56

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_WORD_SELECT 1

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_WORD_OFFSET 24

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_WORD_OFFSET)

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_SIZE 1

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_OFFSET 57

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_WORD_SELECT 1

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_WORD_OFFSET 25

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_WORD_OFFSET)

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_SIZE 1

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_OFFSET 58

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_WORD_SELECT 1

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_WORD_OFFSET 26

#define EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUCKET_UG_WORD_OFFSET)
```

```
#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_SIZE 1

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_OFFSET 59

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_WORD_SELECT 1

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_WORD_OFFSET 27

#define EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_MASK (1 <<
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUCKET_UG_WORD_OFFSET)

#define EZDP_TB_CTR_RESULT_RESERVED60_63_SIZE 4

#define EZDP_TB_CTR_RESULT_RESERVED60_63_OFFSET 60

#define EZDP_TB_CTR_RESULT_WORD_COUNT 2

#define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_SIZE 10

#define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_OFFSET 0

#define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_WORD_SELECT 0

#define EZDP_TB_CTR_CFG_COMMIT_PROFILE_ID_WORD_OFFSET 0

#define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_SIZE 10

#define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_OFFSET 10

#define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_WORD_SELECT 0

#define EZDP_TB_CTR_CFG_EXCESS_PROFILE_ID_WORD_OFFSET 10

#define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_SIZE 4

#define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_OFFSET 20

#define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_WORD_SELECT 0

#define EZDP_TB_CTR_CFG_ALGORITHM_TYPE_WORD_OFFSET 20

#define EZDP_TB_CTR_CFG_COLOR_AWARE_SIZE 1

#define EZDP_TB_CTR_CFG_COLOR_AWARE_OFFSET 24

#define EZDP_TB_CTR_CFG_COLOR_AWARE_WORD_SELECT 0

#define EZDP_TB_CTR_CFG_COLOR_AWARE_WORD_OFFSET 24

#define EZDP_TB_CTR_CFG_COLOR_AWARE_MASK (1 <<
EZDP_TB_CTR_CFG_COLOR_AWARE_WORD_OFFSET)

#define EZDP_TB_CTR_CFG_COUPLING_FLAG_SIZE 1
```

```
#define EZDP_TB_CTR_CFG_COUPLING_FLAG_OFFSET 25

#define EZDP_TB_CTR_CFG_COUPLING_FLAG_WORD_SELECT 0

#define EZDP_TB_CTR_CFG_COUPLING_FLAG_WORD_OFFSET 25

#define EZDP_TB_CTR_CFG_COUPLING_FLAG_MASK (1 <<
EZDP_TB_CTR_CFG_COUPLING_FLAG_WORD_OFFSET)

#define EZDP_TB_CTR_CFG_RESERVED26_31_SIZE 6

#define EZDP_TB_CTR_CFG_RESERVED26_31_OFFSET 26

#define EZDP_TB_CTR_CFG_RESERVED32_63_SIZE 32

#define EZDP_TB_CTR_CFG_RESERVED32_63_OFFSET 32

#define EZDP_TB_CTR_CFG_RESERVED64_95_SIZE 32

#define EZDP_TB_CTR_CFG_RESERVED64_95_OFFSET 64

#define EZDP_TB_CTR_CFG_RESERVED96_127_SIZE 32

#define EZDP_TB_CTR_CFG_RESERVED96_127_OFFSET 96

#define EZDP_TB_CTR_CFG_WORD_COUNT 4

#define EZDP_HIER_TB_UG_APP_BITS_COLOR_STATE_G_SIZE 2

#define EZDP_HIER_TB_UG_APP_BITS_COLOR_STATE_G_OFFSET 0

#define EZDP_HIER_TB_UG_APP_BITS_COLOR_STATE_Y_SIZE 2

#define EZDP_HIER_TB_UG_APP_BITS_COLOR_STATE_Y_OFFSET 2

#define EZDP_HIER_TB_UG_APP_BITS_EIGHTH_MODE_RET_BITS_SIZE 4

#define EZDP_HIER_TB_UG_APP_BITS_EIGHTH_MODE_RET_BITS_OFFSET 4

#define EZDP_HIER_TB_UG_APP_BITS_APP_BITS_SIZE 16

#define EZDP_HIER_TB_UG_APP_BITS_APP_BITS_OFFSET 8

#define EZDP_HIER_TB_UG_APP_BITS_RESERVED24_31_SIZE 8

#define EZDP_HIER_TB_UG_APP_BITS_RESERVED24_31_OFFSET 24

#define EZDP_HIER_TB_CTR_CFG_RESERVED0_1_SIZE 2

#define EZDP_HIER_TB_CTR_CFG_RESERVED0_1_OFFSET 0

#define EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESHOLD_SIZE 5
```

```
#define EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESHOLD_OFFSET 2

#define EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESHOLD_WORD_SELECT 0

#define EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESHOLD_WORD_OFFSET 2

#define EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESHOLD_SIZE 5

#define EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESHOLD_OFFSET 7

#define EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESHOLD_WORD_SELECT 0

#define EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESHOLD_WORD_OFFSET 7

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_THRESHOLD_SIZE 5

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_THRESHOLD_OFFSET 12

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_THRESHOLD_WORD_SELECT 0

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_THRESHOLD_WORD_OFFSET 12

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_FAIL_THRESHOLD_SIZE 5

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_FAIL_THRESHOLD_OFFSET 17

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_FAIL_THRESHOLD_WORD_SELECT 0

#define EZDP_HIER_TB_CTR_CFG_CTR_SUM_FAIL_THRESHOLD_WORD_OFFSET 17

#define EZDP_HIER_TB_CTR_CFG_RESERVED22_26_SIZE 5

#define EZDP_HIER_TB_CTR_CFG_RESERVED22_26_OFFSET 22

#define EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESHOLD_SIZE 5

#define EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESHOLD_OFFSET 27

#define EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESHOLD_WORD_SELECT 0

#define EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESHOLD_WORD_OFFSET 27

#define EZDP_HIER_TB_CTR_CFG_APP_BITS_SIZE 24

#define EZDP_HIER_TB_CTR_CFG_APP_BITS_OFFSET 32

#define EZDP_HIER_TB_CTR_CFG_APP_BITS_WORD_SELECT 1

#define EZDP_HIER_TB_CTR_CFG_APP_BITS_WORD_OFFSET 0

#define EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESHOLD_SIZE 5
```

```
#define EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESHOLD_OFFSET 56

#define EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESHOLD_WORD_SELECT 1

#define EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESHOLD_WORD_OFFSET 24

#define EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRESHOLD_SIZE 2

#define EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRESHOLD_OFFSET 61

#define EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRESHOLD_WORD_SELECT 1

#define EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRESHOLD_WORD_OFFSET 29

#define EZDP_HIER_TB_CTR_CFG_RESERVED63_SIZE 1

#define EZDP_HIER_TB_CTR_CFG_RESERVED63_OFFSET 63

#define EZDP_HIER_TB_CTR_CFG_WORD_COUNT 2

#define EZDP_HIER_TB_RESULT_RESERVED0_9_SIZE 10

#define EZDP_HIER_TB_RESULT_RESERVED0_9_OFFSET 0

#define EZDP_HIER_TB_RESULT_CTR1_SIZE 18

#define EZDP_HIER_TB_RESULT_CTR1_OFFSET 10

#define EZDP_HIER_TB_RESULT_CTR1_WORD_SELECT 0

#define EZDP_HIER_TB_RESULT_CTR1_WORD_OFFSET 10

#define EZDP_HIER_TB_RESULT_STATE_SIZE 2

#define EZDP_HIER_TB_RESULT_STATE_OFFSET 28

#define EZDP_HIER_TB_RESULT_STATE_WORD_SELECT 0

#define EZDP_HIER_TB_RESULT_STATE_WORD_OFFSET 28

#define EZDP_HIER_TB_RESULT_UPDATE_TASK_SIZE 1

#define EZDP_HIER_TB_RESULT_UPDATE_TASK_OFFSET 30

#define EZDP_HIER_TB_RESULT_UPDATE_TASK_WORD_SELECT 0

#define EZDP_HIER_TB_RESULT_UPDATE_TASK_WORD_OFFSET 30

#define EZDP_HIER_TB_RESULT_UPDATE_TASK_MASK (1 << EZDP_HIER_TB_RESULT_UPDATE_TASK_WORD_OFFSET)

#define EZDP_HIER_TB_RESULT_FAIL_SIZE 1
```



```
#define EZDP_HIER_TB_RESULT_FAIL_OFFSET 31

#define EZDP_HIER_TB_RESULT_FAIL_WORD_SELECT 0

#define EZDP_HIER_TB_RESULT_FAIL_WORD_OFFSET 31

#define EZDP_HIER_TB_RESULT_FAIL_MASK (1 <<
EZDP_HIER_TB_RESULT_FAIL_WORD_OFFSET)

#define EZDP_HIER_TB_RESULT_APP_BITS_SIZE 24

#define EZDP_HIER_TB_RESULT_APP_BITS_OFFSET 32

#define EZDP_HIER_TB_RESULT_APP_BITS_WORD_SELECT 1

#define EZDP_HIER_TB_RESULT_APP_BITS_WORD_OFFSET 0

#define EZDP_HIER_TB_RESULT_RESERVED56_63_SIZE 8

#define EZDP_HIER_TB_RESULT_RESERVED56_63_OFFSET 56

#define EZDP_HIER_TB_RESULT_CTR0_SIZE 18

#define EZDP_HIER_TB_RESULT_CTR0_OFFSET 64

#define EZDP_HIER_TB_RESULT_CTR0_WORD_SELECT 2

#define EZDP_HIER_TB_RESULT_CTR0_WORD_OFFSET 0

#define EZDP_HIER_TB_RESULT_RESERVED82_95_SIZE 14

#define EZDP_HIER_TB_RESULT_RESERVED82_95_OFFSET 82

#define EZDP_HIER_TB_RESULT_WORD_COUNT 3

#define EZDP_HIER_TB_UPDATE_APP_BITS_SIZE 24

#define EZDP_HIER_TB_UPDATE_APP_BITS_OFFSET 0

#define EZDP_HIER_TB_UPDATE_RESERVED24_27_SIZE 4

#define EZDP_HIER_TB_UPDATE_RESERVED24_27_OFFSET 24

#define EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_SIZE 1

#define EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_OFFSET 28

#define EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_MASK (1 <<
EZDP_HIER_TB_UPDATE_COND_SET_ACTIVE_STATE_OFFSET)

#define EZDP_HIER_TB_UPDATE_SET_APP_BITS_SIZE 1

#define EZDP_HIER_TB_UPDATE_SET_APP_BITS_OFFSET 29
```

```
#define EZDP_HIER_TB_UPDATE_SET_APP_BITS_MASK (1 <<
EZDP_HIER_TB_UPDATE_SET_APP_BITS_OFFSET)

#define EZDP_HIER_TB_UPDATE_CLR_CTR_SIZE 1

#define EZDP_HIER_TB_UPDATE_CLR_CTR_OFFSET 30

#define EZDP_HIER_TB_UPDATE_CLR_CTR_MASK (1 <<
EZDP_HIER_TB_UPDATE_CLR_CTR_OFFSET)

#define EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_SIZE 1

#define EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_OFFSET 31

#define EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_MASK (1 <<
EZDP_HIER_TB_UPDATE_SET_ACTIVE_STATE_OFFSET)

#define EZDP_BITWISE_CTR_CFG_RESERVED0_18_SIZE 19

#define EZDP_BITWISE_CTR_CFG_RESERVED0_18_OFFSET 0

#define EZDP_BITWISE_CTR_CFG_SUB_TYPE_SIZE 5

#define EZDP_BITWISE_CTR_CFG_SUB_TYPE_OFFSET 19

#define EZDP_BITWISE_CTR_CFG_ECC_SIZE 8

#define EZDP_BITWISE_CTR_CFG_ECC_OFFSET 24

#define EZDP_BITWISE_CTR_CFG_RESERVED32_63_SIZE 32

#define EZDP_BITWISE_CTR_CFG_RESERVED32_63_OFFSET 32

#define EZDP_BITWISE_CTR_CFG_DATA_SIZE 64

#define EZDP_BITWISE_CTR_CFG_DATA_OFFSET 64

#define EZDP_BITWISE_CTR_CFG_DATA_WORD_SELECT 2

#define EZDP_BITWISE_CTR_CFG_DATA_WORD_OFFSET 0

#define EZDP_BITWISE_CTR_CFG_WORD_COUNT 4

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_SIZE 5

#define
EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_OFFSET 0

#define
EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_WORD_SELECT
0
```

```
#define
EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_WORD_OFFSET
0

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_RESERVED5_28_SIZE 24

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_RESERVED5_28_OFFSET 5

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_MAX_THRESHOLD_ALERT_SIZE 1

#define
EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_MAX_THRESHOLD_ALERT_OFFSET 29

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_MIN_THRESHOLD_ALERT_SIZE 1

#define
EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_MIN_THRESHOLD_ALERT_OFFSET 30

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ALERT_SIZE 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ALERT_OFFSET 31

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_SIZE 12

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_OFFSET 32

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_WORD_SELECT 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_WORD_OFFSET 0

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_SIZE 12

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_OFFSET 44

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_WORD_SELECT 1

#define
EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_WORD_OFFSET 12

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_INIT_BIT_SIZE 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_INIT_BIT_OFFSET 56

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_VALID_SIZE 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_VALID_OFFSET 57

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_VALID_WORD_SELECT 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_VALID_WORD_OFFSET 25

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_VALID_MASK (1 <<
EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_VALID_WORD_OFFSET)
```

```
#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_PROFILE_ID_SIZE 4

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_PROFILE_ID_OFFSET 58

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_PROFILE_ID_WORD_SELECT 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_PROFILE_ID_WORD_OFFSET 26

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_PARITY_SIZE 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_PARITY_OFFSET 62

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_RESERVED63_SIZE 1

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_RESERVED63_OFFSET 63

#define EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_WORD_COUNT 2

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_SIZE 5

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_OFFSET 0

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_WORD_SELECT 0

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_WORD_OFFSET 0

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED5_28_SIZE 24

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED5_28_OFFSET 5

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MAX_THRESHOLD_ALERT_SIZE 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MAX_THRESHOLD_ALERT_OFFSET 29

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MIN_THRESHOLD_ALERT_SIZE 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MIN_THRESHOLD_ALERT_OFFSET 30

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_ALERT_SIZE 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_ALERT_OFFSET 31

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_SIZE 24

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_OFFSET 32

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_WORD_SELECT 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_WORD_OFFSET 0

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED56_SIZE 1
```

```
#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED56_OFFSET 56

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_SIZE 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_OFFSET 57

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WORD_SELECT 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WORD_OFFSET 25

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_MASK (1 <<
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WORD_OFFSET)

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PROFILE_ID_SIZE 4

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PROFILE_ID_OFFSET 58

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PROFILE_ID_WORD_SELECT 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PROFILE_ID_WORD_OFFSET 26

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PARITY_SIZE 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PARITY_OFFSET 62

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED63_SIZE 1

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED63_OFFSET 63

#define EZDP_WATCHDOG_SLIDING_WINDOW_CFG_WORD_COUNT 2

#define EZDP_WATCHDOG_CTR_CFG_RESERVED0_18_SIZE 19

#define EZDP_WATCHDOG_CTR_CFG_RESERVED0_18_OFFSET 0

#define EZDP_WATCHDOG_CTR_CFG_SUB_TYPE_SIZE 5

#define EZDP_WATCHDOG_CTR_CFG_SUB_TYPE_OFFSET 19

#define EZDP_WATCHDOG_CTR_CFG_ECC_SIZE 8

#define EZDP_WATCHDOG_CTR_CFG_ECC_OFFSET 24

#define EZDP_WATCHDOG_CTR_CFG_RESERVED32_63_SIZE 32

#define EZDP_WATCHDOG_CTR_CFG_RESERVED32_63_OFFSET 32

#define EZDP_WATCHDOG_CTR_CFG_WORD_COUNT 4

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_ACCUMULATIVE_EVENTS_SIZE 4

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_ACCUMULATIVE_EVENTS_OFFSET 0
```

```
#define EZDP_WATCHDOG_CTR_CHECK_RESULT_RESERVED4_28_SIZE 25

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_RESERVED4_28_OFFSET 4

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_SIZE 1

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_OFFSET 29

#define
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_WORD_SELECT 0

#define
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_WORD_OFFSET 29

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_MASK (1 <<
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_WORD_OFFSET)

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_SIZE 1

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_OFFSET 30

#define
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_WORD_SELECT 0

#define
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_WORD_OFFSET 30

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_MASK (1 <<
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_WORD_OFFSET)

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_ALERT_SIZE 1

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_ALERT_OFFSET 31

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_ALERT_WORD_SELECT 0

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_ALERT_WORD_OFFSET 31

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_ALERT_MASK (1 <<
EZDP_WATCHDOG_CTR_CHECK_RESULT_ALERT_WORD_OFFSET)

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_LAST_EVENTS_SIZE 12

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_LAST_EVENTS_OFFSET 32

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_CURR_EVENTS_SIZE 12

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_CURR_EVENTS_OFFSET 44

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_INIT_BIT_SIZE 1

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_INIT_BIT_OFFSET 56
```

```
#define EZDP_WATCHDOG_CTR_CHECK_RESULT_VALID_SIZE 1

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_VALID_OFFSET 57

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_PROFILE_ID_SIZE 4

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_PROFILE_ID_OFFSET 58

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_RESERVED62_SIZE 1

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_RESERVED62_OFFSET 62

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_WINDOW_RELATED_SIZE 1

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_WINDOW_RELATED_OFFSET 63

#define EZDP_WATCHDOG_CTR_CHECK_RESULT_WORD_COUNT 2

#define EZDP_WATCHDOG_CTR_START_RESULT_MSB_SIZE 32

#define EZDP_WATCHDOG_CTR_START_RESULT_MSB_OFFSET 0

#define EZDP_WATCHDOG_CTR_START_RESULT_MSB_WORD_SELECT 0

#define EZDP_WATCHDOG_CTR_START_RESULT_MSB_WORD_OFFSET 0

#define EZDP_WATCHDOG_CTR_START_RESULT_LSB_SIZE 32

#define EZDP_WATCHDOG_CTR_START_RESULT_LSB_OFFSET 32

#define EZDP_WATCHDOG_CTR_START_RESULT_LSB_WORD_SELECT 1

#define EZDP_WATCHDOG_CTR_START_RESULT_LSB_WORD_OFFSET 0

#define EZDP_WATCHDOG_CTR_START_RESULT_WORD_COUNT 2

#define EZDP_CTR_MSG_COUNTER_TYPE_SIZE 2

#define EZDP_CTR_MSG_COUNTER_TYPE_OFFSET 0

#define EZDP_CTR_MSG_COUNTER_TYPE_WORD_SELECT 0

#define EZDP_CTR_MSG_COUNTER_TYPE_WORD_OFFSET 0

#define EZDP_CTR_MSG_MSG_TYPE_SIZE 3

#define EZDP_CTR_MSG_MSG_TYPE_OFFSET 2

#define EZDP_CTR_MSG_MSG_TYPE_WORD_SELECT 0

#define EZDP_CTR_MSG_MSG_TYPE_WORD_OFFSET 2
```

```
#define EZDP_CTR_MSG_OVERFLOW_SIZE 1

#define EZDP_CTR_MSG_OVERFLOW_OFFSET 5

#define EZDP_CTR_MSG_OVERFLOW_WORD_SELECT 0

#define EZDP_CTR_MSG_OVERFLOW_WORD_OFFSET 5

#define EZDP_CTR_MSG_OVERFLOW_MASK (1 <<
EZDP_CTR_MSG_OVERFLOW_WORD_OFFSET)

#define EZDP_CTR_MSG_RESERVED6_SIZE 1

#define EZDP_CTR_MSG_RESERVED6_OFFSET 6

#define EZDP_CTR_MSG_OVERRUN_ERROR_CONDITION_SIZE 1

#define EZDP_CTR_MSG_OVERRUN_ERROR_CONDITION_OFFSET 7

#define EZDP_CTR_MSG_OVERRUN_ERROR_CONDITION_WORD_SELECT 0

#define EZDP_CTR_MSG_OVERRUN_ERROR_CONDITION_WORD_OFFSET 7

#define EZDP_CTR_MSG_OVERRUN_ERROR_CONDITION_MASK (1 <<
EZDP_CTR_MSG_OVERRUN_ERROR_CONDITION_WORD_OFFSET)

#define EZDP_CTR_MSG_RESERVED8_23_SIZE 16

#define EZDP_CTR_MSG_RESERVED8_23_OFFSET 8

#define EZDP_CTR_MSG_ECC_SIZE 8

#define EZDP_CTR_MSG_ECC_OFFSET 24

#define EZDP_CTR_MSG_SUM_ADDR_SIZE 32

#define EZDP_CTR_MSG_SUM_ADDR_OFFSET 32

#define EZDP_CTR_MSG_SUM_ADDR_WORD_SELECT 1

#define EZDP_CTR_MSG_SUM_ADDR_WORD_OFFSET 0

#define EZDP_CTR_MSG_WORD_COUNT 6

#define EZDP_POSTED_CTR_MSG_MSG_TYPE_SIZE 3

#define EZDP_POSTED_CTR_MSG_MSG_TYPE_OFFSET 0

#define EZDP_POSTED_CTR_MSG_MSG_TYPE_WORD_SELECT 0

#define EZDP_POSTED_CTR_MSG_MSG_TYPE_WORD_OFFSET 0

#define EZDP_POSTED_CTR_MSG_FLUSH_SIZE 1
```



```
#define EZDP_POSTED_CTR_MSG_FLUSH_OFFSET 3

#define EZDP_POSTED_CTR_MSG_FLUSH_WORD_SELECT 0

#define EZDP_POSTED_CTR_MSG_FLUSH_WORD_OFFSET 3

#define EZDP_POSTED_CTR_MSG_FLUSH_MASK (1 <<
EZDP_POSTED_CTR_MSG_FLUSH_WORD_OFFSET)

#define EZDP_POSTED_CTR_MSG_CLEAR_SIZE 1

#define EZDP_POSTED_CTR_MSG_CLEAR_OFFSET 4

#define EZDP_POSTED_CTR_MSG_CLEAR_WORD_SELECT 0

#define EZDP_POSTED_CTR_MSG_CLEAR_WORD_OFFSET 4

#define EZDP_POSTED_CTR_MSG_CLEAR_MASK (1 <<
EZDP_POSTED_CTR_MSG_CLEAR_WORD_OFFSET)

#define EZDP_POSTED_CTR_MSG_RESERVED5_6_SIZE 2

#define EZDP_POSTED_CTR_MSG_RESERVED5_6_OFFSET 5

#define EZDP_POSTED_CTR_MSG_OVERRUN_ERROR_CONDITION_SIZE 1

#define EZDP_POSTED_CTR_MSG_OVERRUN_ERROR_CONDITION_OFFSET 7

#define EZDP_POSTED_CTR_MSG_OVERRUN_ERROR_CONDITION_WORD_SELECT 0

#define EZDP_POSTED_CTR_MSG_OVERRUN_ERROR_CONDITION_WORD_OFFSET 7

#define EZDP_POSTED_CTR_MSG_OVERRUN_ERROR_CONDITION_MASK (1 <<
EZDP_POSTED_CTR_MSG_OVERRUN_ERROR_CONDITION_WORD_OFFSET)

#define EZDP_POSTED_CTR_MSG_RESERVED8_23_SIZE 16

#define EZDP_POSTED_CTR_MSG_RESERVED8_23_OFFSET 8

#define EZDP_POSTED_CTR_MSG_ECC_SIZE 8

#define EZDP_POSTED_CTR_MSG_ECC_OFFSET 24

#define EZDP_POSTED_CTR_MSG_SUM_ADDR_SIZE 32

#define EZDP_POSTED_CTR_MSG_SUM_ADDR_OFFSET 32

#define EZDP_POSTED_CTR_MSG_SUM_ADDR_WORD_SELECT 1

#define EZDP_POSTED_CTR_MSG_SUM_ADDR_WORD_OFFSET 0

#define EZDP_POSTED_CTR_MSG_VALUE_SIZE 64
```

```

#define EZDP_POSTED_CTR_MSG_VALUE_OFFSET 64

#define EZDP_POSTED_CTR_MSG_VALUE_WORD_SELECT 2

#define EZDP_POSTED_CTR_MSG_VALUE_WORD_OFFSET 0

#define EZDP_POSTED_CTR_MSG_WORD_COUNT 4

#define EZDP_CTR_MSG_WORK_AREA_SIZE ( (sizeof(struct ezdp_stat_msg_int) > sizeof(struct ezdp_double_ctr_int)/2) ? sizeof(struct ezdp_stat_msg_int) : sizeof(struct ezdp_double_ctr_int)/2 )

#define EZDP_POSTED_CTR_MSG_WORK_AREA_SIZE sizeof(struct ezdp_stat_msg_int)

```

---

## Typedef Documentation

typedef uint32\_t [ezdp\\_hier\\_tb Ug App Bits t](#)

typedef uint32\_t [ezdp\\_hier\\_tb Update t](#)

typedef struct ezdp\_get\_color\_hier\_tb\_ctr\_working\_area  
[ezdp\\_get\\_color\\_hier\\_tb\\_ctr\\_working\\_area t](#)

typedef uint32\_t (\* [EZDP\\_HIER\\_TB\\_UPDATE\\_BES](#))(uint32\_t app\_bits, uint32\_t ctr\_g, uint32\_t ctr\_y, [ezdp\\_get\\_color\\_hier\\_tb\\_ctr\\_working\\_area t](#) \_\_cmem \*wa\_ptr, void \*user\_data)

Type definition for the data plane application's updating BE TBs according to algorithm and user information.

### Parameters:

[in] *app\_bits* - 24b user data at HIER\_TB [in] *ctr\_g* - counter/accumulator of pre-color green packets [in] *ctr\_y* - counter/accumulator of pre-color yellow packets [in] *wa\_ptr* - working area pointer (temporary memory on CMEM to be used by the function) [in] *user\_data* - data for passing to user call back function

### Returns:

updated application bits

typedef struct ezdp\_msgq\_desc [ezdp\\_ctr\\_msg\\_queue\\_desc t](#)

Statistic message queue descriptor.

typedef struct ezdp\_msgq\_desc [ezdp\\_posted\\_ctr\\_msg\\_queue\\_desc t](#)

Posted message queue descriptor.

## Enumeration Type Documentation

enum [ezdp\\_tb\\_color](#)

tb color possible values.

**Enumerator:**

*EZDP\_RED\_TRAFFIC* Red color marking traffic.

*EZDP\_YELLOW\_TRAFFIC* Yellow color marking traffic.

*EZDP\_GREEN\_TRAFFIC* Green color marking traffic.

**enum [ezdp\\_tb\\_algo](#)**

tb algo possible values.

**Enumerator:**

*EZDP\_INACTIVE* Inactive.

*EZDP\_SINGLE\_BUCKET* Single bucket.

*EZDP\_SR\_TCM* Single Rate Three Color Marker (srTCM).

*EZDP\_TR\_TCM* Two Rate Three Color Marker (trTCM).

*EZDP\_TR\_TCM\_MEF* Two Rate Three Color Marker - MEF standard (trTCM MEF).

**enum [ezdp\\_hier\\_tb\\_state](#)**

hier tb state possible values.

**Enumerator:**

*EZDP\_UG\_FAST\_PATH* Fast path result in case of ultra green feature.

*EZDP\_UG\_SLOW\_PATH* Slow path result in case of ultra green feature.

**enum [ezdp\\_ctr\\_type](#)**

ctr type possible values.

**Enumerator:**

*EZDP\_SINGLE\_CTR* On-demand counter.

*EZDP\_DUAL\_CTR* On-demand dual counter.

**enum [ezdp\\_ctr\\_msg\\_type](#)**

ctr msg type possible values.

**Enumerator:**

*EZDP\_PERIODIC\_CTR\_MSG* Periodical update message.

*EZDP\_THRESHOLD\_CTR\_MSG* Counter exceeded threshold message.

*EZDP\_NULL\_CTR\_MSG* Null message - there is no message.

**enum [ezdp\\_msg\\_posted\\_ctr\\_type](#)**

msg posted ctr type possible values.

**Enumerator:**

***EZDP\_PERIODIC\_POSTED\_CTR\_MSG*** Periodical update message.

***EZDP\_THRESHOLD\_POSTED\_CTR\_MSG*** Counter exceeded threshold message.

***EZDP\_REPORT\_POSTED\_CTR\_MSG*** Report message.

Applicable only for posted counters.

***EZDP\_NULL\_POSTED\_CTR\_MSG*** Null message - there is no message.

**enum [ezdp\\_bitwise\\_size](#)**

bitwise size possible values.

**Enumerator:**

***EZDP\_1\_BITS*** Size of operation is 1 bit.

***EZDP\_2\_BITS*** Size of operation is 2 bits.

***EZDP\_4\_BITS*** Size of operation is 4 bits.

***EZDP\_8\_BITS*** Size of operation is 8 bits.

***EZDP\_16\_BITS*** Size of operation is 16 bits.

---

**Variable Documentation**

**enum [ezdp\\_tb\\_color](#)(\* [EZDP\\_HIER\\_TB\\_CALC\\_COLOR](#))(uint32\_t app\_bits, enum [ezdp\\_tb\\_color](#) pre\_color, uint32\_t \*user\_data)**

User Call Back function calc color according to algorithm and user information.

**Parameters:**

[in] *app\_bits* - 24b user data at HIER\_TB [in] *pre\_color* - Packet Original Color [in] *user\_data* - data for passing to user call back function

**Returns:**

color

## dpe/dp/include/ezdp\_decode.h File Reference

### Functions

- static `__always_inline` [ezdp\\_decode\\_mac\\_retval\\_t ezdp\\_decode\\_mac](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mac\\_result](#) \_\_cmem \*decode\_result)
- *Parse and decode an Ethernet header.* static `__always_inline` void [ezdp\\_decode\\_mac\\_async](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mac\\_result](#) \_\_cmem \*decode\_result)
- *Non blocking version of [ezdp\\_decode\\_mac\(\)](#).* static `__always_inline` [ezdp\\_decode\\_ipv4\\_retval\\_t ezdp\\_decode\\_ipv4](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, uint32\_t frame\_size, struct [ezdp\\_decode\\_ipv4\\_result](#) \_\_cmem \*decode\_result)
- *Parse and decode an IPv4 header.* static `__always_inline` void [ezdp\\_decode\\_ipv4\\_async](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, uint32\_t frame\_size, struct [ezdp\\_decode\\_ipv4\\_result](#) \_\_cmem \*decode\_result)
- *Non blocking version of [ezdp\\_decode\\_ipv4\(\)](#).* static `__always_inline` [ezdp\\_decode\\_ipv6\\_retval\\_t ezdp\\_decode\\_ipv6](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, uint32\_t frame\_size, struct [ezdp\\_decode\\_ipv6\\_result](#) \_\_cmem \*decode\_result)
- *Parse and decode an IPv6 header.* static `__always_inline` void [ezdp\\_decode\\_ipv6\\_async](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, uint32\_t frame\_size, struct [ezdp\\_decode\\_ipv6\\_result](#) \_\_cmem \*decode\_result)
- *Non blocking version of [ezdp\\_decode\\_ipv6\(\)](#).* static `__always_inline` [ezdp\\_decode\\_mpls\\_result\\_t ezdp\\_decode\\_mpls](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mpls\\_result](#) \_\_cmem \*decode\_result)
- *Parse and decode an MPLS header.* static `__always_inline` void [ezdp\\_decode\\_mpls\\_async](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mpls\\_result](#) \_\_cmem \*decode\_result)
- *Non blocking version of [ezdp\\_decode\\_mpls\(\)](#).* static `__always_inline` [ezdp\\_decode\\_mpls\\_label\\_retval\\_t ezdp\\_decode\\_mpls\\_label](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mpls\\_label\\_result](#) \_\_cmem \*decode\_result)
- *Parse and decode an MPLS label.* static `__always_inline` void [ezdp\\_decode\\_mpls\\_label\\_async](#) (uint8\_t \_\_cmem \*frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mpls\\_label\\_result](#) \_\_cmem \*decode\_result)
- *Non blocking version of [ezdp\\_decode\\_mpls\\_label\(\)](#).* static `__always_inline` [ezdp\\_decode\\_tcp\\_retval\\_t ezdp\\_decode\\_tcp](#) (uint32\_t \_\_cmem \*frame\_ptr)
- *Parse and decode a TCP header.* static `__always_inline` [ezdp\\_decode\\_ip\\_protocol\\_retval\\_t ezdp\\_decode\\_ip\\_protocol](#) (uint32\_t ip\_type, uint32\_t def\_ip\_prot\_0, uint32\_t def\_ip\_prot\_1, uint32\_t def\_ip\_prot\_2, uint32\_t def\_ip\_prot\_3)
- *Decode an IP protocol value.* static `__always_inline` [ezdp\\_decode\\_eth\\_type\\_retval\\_t ezdp\\_decode\\_eth\\_type](#) (uint32\_t eth\_type, uint32\_t def\_eth\_type\_0, uint32\_t def\_eth\_type\_1)

*Decode an Ethernet type value.*

---

### Function Documentation

static `__always_inline` [ezdp\\_decode\\_mac\\_retval\\_t ezdp\\_decode\\_mac](#) (uint8\_t \_\_cmem \* frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mac\\_result](#) \_\_cmem \* decode\_result) [static]

Parse and decode an Ethernet header.

The decode result is written to CMEM. In addition, the first 4 bytes of the decode result are returned.

#### Parameters:

- [in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded
- [in] *size* - number of bytes to decode
- [out] *decode\_result* - decode result

#### Returns:

uint32\_t - according to [ezdp\\_decode\\_dcmac\\_retval](#)

static `__always_inline` void [ezdp\\_decode\\_mac\\_async](#) (uint8\_t \_\_cmem \* frame\_ptr, uint32\_t size, struct [ezdp\\_decode\\_mac\\_result](#) \_\_cmem \* decode\_result) [static]

Non blocking version of [ezdp\\_decode\\_mac\(\)](#).

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
 [in] *size* - number of bytes to decode  
 [out] *decode\_result* - decode result

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the decode result is ready in CMEM.

**Returns:**

none

```
static __always_inline ezdp\_decode\_ipv4\_retval\_t ezdp_decode_ipv4 (uint8_t __cmem *
frame_ptr, uint32_t size, uint32_t frame_size, struct ezdp\_decode\_ipv4\_result __cmem *
decode_result) [static]
```

Parse and decode an IPv4 header.

The decode result is written to CMEM. In addition, the first 4 bytes of the decode result are returned.

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
 [in] *size* - number of bytes to decode  
 [in] *frame\_size* - frame data size  
 [out] *decode\_result* - decode result

**Returns:**

uint32\_t - according to [ezdp\\_decode\\_dcipv4\\_retval](#)

```
static __always_inline void ezdp_decode_ipv4_async (uint8_t __cmem * frame_ptr, uint32_t
size, uint32_t frame_size, struct ezdp\_decode\_ipv4\_result __cmem * decode_result)
[static]
```

Non blocking version of [ezdp\\_decode\\_ipv4\(\)](#).

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
 [in] *size* - number of bytes to decode  
 [in] *frame\_size* - frame data size  
 [out] *decode\_result* - decode result

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the decode result is ready in CMEM.

**Returns:**

none

```
static __always_inline ezdp\_decode\_ipv6\_retval\_t ezdp_decode_ipv6 (uint8_t __cmem *
frame_ptr, uint32_t size, uint32_t frame_size, struct ezdp\_decode\_ipv6\_result __cmem *
decode_result) [static]
```

Parse and decode an IPv6 header.

The decode result is written to CMEM. In addition, the first 4 bytes of the decode result are returned.

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
 [in] *size* - number of bytes to decode  
 [in] *frame\_size* - frame data size  
 [out] *decode\_result* - decode result

**Returns:**

uint32\_t - according to [ezdp\\_decode\\_dcipv6\\_retval](#)

```
static __always_inline void ezdp_decode_ipv6_async (uint8_t __cmem * frame_ptr,  uint32_t
size,  uint32_t frame_size,  struct ezdp\_decode\_ipv6\_result __cmem * decode_result)
[static]
```

Non blocking version of [ezdp\\_decode\\_ipv6\(\)](#).

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
 [in] *size* - number of bytes to decode  
 [in] *frame\_size* - frame data size  
 [out] *decode\_result* - decode result

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the decode result is ready in CMEM.

**Returns:**

none

```
static __always_inline ezdp\_decode\_mpls\_result\_t ezdp_decode_mpls (uint8_t __cmem *
frame_ptr,  uint32_t size,  struct ezdp\_decode\_mpls\_result __cmem * decode_result)
[static]
```

Parse and decode an MPLS header.

The decode result is written to CMEM. In addition, the first 4 bytes of the decode result are returned.

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
 [in] *size* - number of bytes to decode  
 [out] *decode\_result* - decode result

**Returns:**

uint32\_t - according to [ezdp\\_decode\\_mpls\\_result](#)

```
static __always_inline void ezdp_decode_mpls_async (uint8_t __cmem * frame_ptr,  uint32_t
size,  struct ezdp\_decode\_mpls\_result __cmem * decode_result) [static]
```

Non blocking version of [ezdp\\_decode\\_mpls\(\)](#).

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
 [in] *size* - number of bytes to decode  
 [out] *decode\_result* - decode result

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the decode result is ready in CMEM.

**Returns:**

none

```
static __always_inline ezdp\_decode\_mpls\_label\_retval\_t ezdp_decode_mpls_label (uint8_t
__cmem * frame_ptr,  uint32_t size,  struct ezdp\_decode\_mpls\_label\_result __cmem *
decode_result) [static]
```

Parse and decode an MPLS label.

The decode result is written to CMEM. In addition, the first 4 bytes of the decode result are returned.

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
[in] *size* - number of bytes to decode  
[out] *decode\_result* - decode result

**Returns:**

uint32\_t - according to [ezdp\\_decode\\_mpls\\_label\\_retval\\_t](#)

```
static __always_inline void ezdp_decode_mpls_label_async (uint8_t __cmem * frame_ptr,
uint32_t size,  struct ezdp\_decode\_mpls\_label\_result __cmem * decode_result) [static]
```

Non blocking version of [ezdp\\_decode\\_mpls\\_label\(\)](#).

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded  
[in] *size* - number of bytes to decode  
[out] *decode\_result* - decode result

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the decode result is ready in CMEM.

**Returns:**

none

```
static __always_inline ezdp\_decode\_tcp\_retval\_t ezdp_decode_tcp (uint32_t __cmem * frame_ptr)
[static]
```

Parse and decode a TCP header.

**Parameters:**

[in] *frame\_ptr* - pointer to frame header data (in CMEM) to be decoded

**Returns:**

uint32\_t - according to [ezdp\\_decode\\_tcp\\_retval](#)

```
static __always_inline ezdp\_decode\_ip\_protocol\_retval\_t ezdp_decode_ip_protocol (uint32_t
ip_type,  uint32_t def_ip_prot_0,  uint32_t def_ip_prot_1,  uint32_t def_ip_prot_2,  uint32_t
def_ip_prot_3) [static]
```

Decode an IP protocol value.

**Parameters:**

[in] *ip\_type* - IP protocol type



[in] *def\_ip\_prot\_0* - Default/config ip protocol type 0 immediate value to be used by `decode_ip_protocol` instruction to turn on `def_ip_prot_0` flag.  
 [in] *def\_ip\_prot\_1* - Default/config ip protocol type 1 immediate value to be used by `decode_ip_protocol` instruction to turn on `def_ip_prot_1` flag.  
 [in] *def\_ip\_prot\_2* - Default/config ip protocol type 2 immediate value to be used by `decode_ip_protocol` instruction to turn on `def_ip_prot_2` flag.  
 [in] *def\_ip\_prot\_3* - Default/config ip protocol type 3 immediate value to be used by `decode_ip_protocol` instruction to turn on `def_ip_prot_3` flag.

**Returns:**

`uint32_t` - according to [ezdp\\_decode\\_ip\\_protocol\\_retval](#)

```
static __always_inline ezdp\_decode\_eth\_type\_retval\_t ezdp_decode_eth_type(uint32_t eth_type,  

uint32_t def_eth_type_0, uint32_t def_eth_type_1) [static]
```

Decode an Ethernet type value.

**Parameters:**

[in] *eth\_type* - Ethernet type  
 [in] *def\_eth\_type\_0* - Default/config Ethernet type 0 immediate value to be used by `decode_eth_type` instruction to turn on `user_def0` flag.  
 [in] *def\_eth\_type\_1* - Default/config Ethernet type 1 immediate value to be used by `decode_eth_type` instruction to turn on `user_def1` flag.

**Note:**

When both `type0` and `type1` are 0, default/conf values are ignored and a smaller/optimized instruction is used.

**Returns:**

`uint32_t` - according to `decode_eth_type_retval`

## dpe/dp/include/ezdp\_decode\_defs.h File Reference

### Data Structures

- struct [ezdp\\_decode\\_ipv4\\_control](#)
- IPv4 addresses decoding result. struct [ezdp\\_decode\\_ipv4\\_errors](#)
- IPv4 header decode error flags. struct [ezdp\\_decode\\_ip\\_next\\_protocol](#)
- IP protocol type flags. struct [ezdp\\_decode\\_ipv6\\_control](#)
- IPv6 addresses decoding result. struct [ezdp\\_decode\\_ipv6\\_errors](#)
- IPv6 header decode error flags. struct [ezdp\\_decode\\_mac\\_control](#)
- MAC addresses decoding result. struct [ezdp\\_decode\\_mac\\_errors](#)
- MAC header decode error flags. struct [ezdp\\_decode\\_mac\\_protocol\\_type](#)
- Ethernet type definition. struct [ezdp\\_decode\\_tcp\\_errors](#)
- TCP header decode error flags. struct [ezdp\\_decode\\_tcp\\_retval](#)
- Decode TCP return value struct definition. struct [ezdp\\_decode\\_ip\\_protocol\\_retval](#)
- Decode ip protocol return value struct definition. struct [ezdp\\_decode\\_eth\\_type\\_retval](#)
- Decode ip protocol return value struct definition. struct [ezdp\\_decode\\_ipv4\\_retval](#)
- Decode IPv4 return value struct definition. struct [ezdp\\_decode\\_ipv4\\_result](#)
- Decode IPv4 result. struct [ezdp\\_decode\\_ipv6\\_retval](#)
- Decode IPv4 return value struct definition. struct [ezdp\\_decode\\_ipv6\\_result](#)
- Decode IPv6 result. struct [ezdp\\_decode\\_mac\\_retval](#)
- Decode MAC return value struct definition. struct [ezdp\\_decode\\_mac\\_result](#)
- Decode MAC result. struct [ezdp\\_decode\\_mpls\\_retval](#)
- Decode MPLS return value struct definition. struct [ezdp\\_decode\\_mpls\\_result](#)
- [ezdp\\_decode\\_mpls\\_result](#) struct for ezdp struct [ezdp\\_decode\\_mpls\\_label\\_retval](#)
- Decode MPLS label return value struct definition. struct [ezdp\\_decode\\_mpls\\_label\\_result](#)

### [ezdp\\_decode\\_mpls\\_label\\_result](#) struct for ezdp Defines

- #define [EZDP\\_DECODE\\_VERSION\\_MAJOR](#) 2
- #define [EZDP\\_DECODE\\_VERSION\\_MINOR](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_LINK\\_LOCAL\\_MULTICAST\\_RANGE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_LINK\\_LOCAL\\_MULTICAST\\_RANGE\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_LINK\\_LOCAL\\_MULTICAST\\_RANGE\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_INTERNETWORK\\_MULTICAST\\_RANGE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_INTERNETWORK\\_MULTICAST\\_RANGE\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_INTERNETWORK\\_MULTICAST\\_RANGE\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_RESERVED\\_2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_RESERVED\\_2\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_ICMP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_ICMP\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_ICMP\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_CONTROL\_ICMP\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_IGMP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_IGMP\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_IGMP\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_CONTROL\_IGMP\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG0\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG0\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG0\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG1\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG1\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_OFFSET)

- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG2\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IPV4\\_CONTROL\\_USER\\_CONFIG2\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_IS\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_IS\\_MULTICAST\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_IS\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_IS\\_ZERO\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_HEADER\\_LENGTH\\_LT\\_5\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_HEADER\\_LENGTH\\_LT\\_5\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_HEADER\\_LENGTH\\_LT\\_5\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_TOTAL\\_LENGTH\\_GT\\_FRAME\\_LENGTH\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_TOTAL\\_LENGTH\\_GT\\_FRAME\\_LENGTH\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_TOTAL\\_LENGTH\\_GT\\_FRAME\\_LENGTH\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_HEADER\\_LENGTH\\_GT\\_FRAME\\_LENGTH\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_HEADER\\_LENGTH\\_GT\\_FRAME\\_LENGTH\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_HEADER\\_LENGTH\\_GT\\_FRAME\\_LENGTH\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_NOT\\_IPV4\\_VERSION\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_NOT\\_IPV4\\_VERSION\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_NOT\\_IPV4\\_VERSION\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_CHECKSUM\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_CHECKSUM\\_ERROR\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_CHECKSUM\\_ERROR\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_EQUAL\\_DIP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_EQUAL\\_DIP\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_SIP\\_EQUAL\\_DIP\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_DECODE\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_DECODE\\_ERROR\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_DECODE\\_ERROR\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_RESERVED9\\_15\\_SIZE](#) 7
- #define [EZDP\\_DECODE\\_IPV4\\_ERRORS\\_RESERVED9\\_15\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_TCP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_TCP\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_TCP\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_UDP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_UDP\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_UDP\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_MPLS\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_MPLS\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_MPLS\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_GRE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_GRE\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_GRE\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_OFFSET)

- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_IPV4\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_IPV4\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_IPV4\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_IPV6\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_IPV6\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_IPV6\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_ICMP\\_IGMP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_ICMP\\_IGMP\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_ICMP\\_IGMP\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_IGMP\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_OTHER\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_OTHER\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IP\\_NEXT\\_PROTOCOL\\_OTHER\\_MASK](#) (1 << EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_LINK\\_LOCAL\\_MULTICAST\\_RANGE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_LINK\\_LOCAL\\_MULTICAST\\_RANGE\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_LINK\\_LOCAL\\_MULTICAST\\_RANGE\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_INTERNETWORK\\_MULTICAST\\_RANGE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_INTERNETWORK\\_MULTICAST\\_RANGE\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_INTERNETWORK\\_MULTICAST\\_RANGE\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_SOLICITED\\_NODE\\_MULTICAST\\_RANGE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_SOLICITED\\_NODE\\_MULTICAST\\_RANGE\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_SOLICITED\\_NODE\\_MULTICAST\\_RANGE\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_CONTROL\_SOLICITED\_NODE\_MULTICAST\_RANGE\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_RESERVED\\_3\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_RESERVED\\_3\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_DIP\\_IS\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_DIP\\_IS\\_MULTICAST\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_DIP\\_IS\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_DIP\\_IS\\_WELLKNOWN\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_DIP\\_IS\\_WELLKNOWN\\_MULTICAST\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_DIP\\_IS\\_WELLKNOWN\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_WELLKNOWN\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_RESERVED7\\_8\\_SIZE](#) 2
- #define [EZDP\\_DECODE\\_IPV6\\_CONTROL\\_RESERVED7\\_8\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_PAYLOAD\\_GT\\_FRAME\\_LENGTH\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_PAYLOAD\\_GT\\_FRAME\\_LENGTH\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_PAYLOAD\\_GT\\_FRAME\\_LENGTH\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_GT\_FRAME\_LENGTH\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_NOT\\_IPV6\\_VERSION\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_NOT\\_IPV6\\_VERSION\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_NOT\\_IPV6\\_VERSION\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_NOT\_IPV6\_VERSION\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_ZERO\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_ONE\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DIP\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DIP\\_IS\\_ZERO\\_OFFSET](#) 4

- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DIP\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_DIP\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DIP\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DIP\\_IS\\_ONE\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DIP\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_DIP\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_EQUAL\\_DIP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_EQUAL\\_DIP\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_EQUAL\\_DIP\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_EQUAL\_DIP\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DECODE\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DECODE\\_ERROR\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_DECODE\\_ERROR\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_DECODE\_ERROR\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_PAYLOAD\\_MISSING\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_PAYLOAD\\_MISSING\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_PAYLOAD\\_MISSING\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_MISSING\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_MULTICAST\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_SIP\\_IS\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_RESERVED10\\_15\\_SIZE](#) 6
- #define [EZDP\\_DECODE\\_IPV6\\_ERRORS\\_RESERVED10\\_15\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MY\\_MAC\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MY\\_MAC\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MY\\_MAC\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_MY\_MAC\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_0X\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_0X\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_0X\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_LSB\_0X\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_1X\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_1X\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_1X\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_LSB\_1X\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_2X\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_2X\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_LSB\\_2X\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_LSB\_2X\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_OTHER\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_OTHER\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_MAC\\_CONTROL\\_OTHER\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONTROL\_OTHER\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_VRRP\\_MAC\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_VRRP\\_MAC\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_VRRP\\_MAC\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_VRRP\_MAC\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_IPV4\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_IPV4\\_MULTICAST\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_IPV4\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_IPV4\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_IPV6\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_IPV6\\_MULTICAST\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_IPV6\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_IPV6\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG0\\_OFFSET](#) 8



- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG0\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG0\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG1\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG1\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG1\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG2\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG2\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG2\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG3\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG3\\_OFFSET](#) 11
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_USER\\_CONFIG3\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONFIG3\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_SMAC\\_EQUALS\\_DMACE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_SMAC\\_EQUALS\\_DMACE\\_OFFSET](#) 12
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_SMAC\\_EQUALS\\_DMACE\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_CONTROL\_SMAC\_EQUALS\_DMACE\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_RESERVED13\\_15\\_SIZE](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_CONTROL\\_RESERVED13\\_15\\_OFFSET](#) 13
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_SMACE\\_IS\\_NOT\\_UNICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_SMACE\\_IS\\_NOT\\_UNICAST\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_SMACE\\_IS\\_NOT\\_UNICAST\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_ERRORS\_SMACE\_IS\_NOT\_UNICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_SMACE\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_SMACE\\_IS\\_ZERO\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_SMACE\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_ERRORS\_SMACE\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_DMACE\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_DMACE\\_IS\\_ZERO\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_DMACE\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_ERRORS\_DMACE\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_IP\\_VERSION\\_MISMATCH\\_IN\\_PPPOE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_IP\\_VERSION\\_MISMATCH\\_IN\\_PPPOE\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_IP\\_VERSION\\_MISMATCH\\_IN\\_PPPOE\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_ERRORS\_IP\_VERSION\_MISMATCH\_IN\_PPPOE\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_DECODE\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_DECODE\\_ERROR\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_DECODE\\_ERROR\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_ERRORS\_DECODE\_ERROR\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_RESERVED5\\_7\\_SIZE](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_ERRORS\\_RESERVED5\\_7\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_IPV4\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_IPV4\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_IPV4\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV4\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN0\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN0\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG\_VLAN0\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN1\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN1\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG\_VLAN1\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_ARP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_ARP\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_ARP\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_ARP\_OFFSET)

- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_MPLS\\_UNICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_MPLS\\_UNICAST\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_MPLS\\_UNICAST\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_UNICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_MPLS\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_MPLS\\_MULTICAST\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_MPLS\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_IPV6\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_IPV6\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_IPV6\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV6\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_LENGTH\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_LENGTH\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_LENGTH\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_LENGTH\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG0\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG0\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG0\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG1\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG1\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG1\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG2\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG2\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG2\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG3\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG3\\_OFFSET](#) 11
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG3\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG3\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_PPPOE\\_SESSION\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_PPPOE\\_SESSION\\_OFFSET](#) 12
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_PPPOE\\_SESSION\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE\_SESSION\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_PPPOE\\_DISCOVERY\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_PPPOE\\_DISCOVERY\\_OFFSET](#) 13
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_PPPOE\\_DISCOVERY\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE\_DISCOVERY\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN2\\_OFFSET](#) 14
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_USER\\_CONFIG\\_VLAN2\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_CONFIG\_VLAN2\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_RESERVED\\_15\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_PROTOCOL\\_TYPE\\_RESERVED\\_15\\_OFFSET](#) 15
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_DATA\\_OFFSET\\_LT\\_5\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_DATA\\_OFFSET\\_LT\\_5\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_DATA\\_OFFSET\\_LT\\_5\\_MASK](#) (1 << EZDP\_DECODE\_TCP\_ERRORS\_DATA\_OFFSET\_LT\_5\_OFFSET)
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_SYN\\_AND\\_FIN\\_EQ\\_1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_SYN\\_AND\\_FIN\\_EQ\\_1\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_SYN\\_AND\\_FIN\\_EQ\\_1\\_MASK](#) (1 << EZDP\_DECODE\_TCP\_ERRORS\_SYN\_AND\_FIN\_EQ\_1\_OFFSET)
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_DECODE\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_DECODE\\_ERROR\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_DECODE\\_ERROR\\_MASK](#) (1 << EZDP\_DECODE\_TCP\_ERRORS\_DECODE\_ERROR\_OFFSET)

- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_RESERVED3\\_7\\_SIZE](#) 5
- #define [EZDP\\_DECODE\\_TCP\\_ERRORS\\_RESERVED3\\_7\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_ERROR\\_CODES\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_ERROR\\_CODES\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_OPTIONS\\_EXIST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_OPTIONS\\_EXIST\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_OPTIONS\\_EXIST\\_MASK](#) (1 << EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIST\_OFFSET)
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_RESERVED9\\_15\\_SIZE](#) 7
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_RESERVED9\\_15\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_DATA\\_OFFSET\\_SIZE](#) 6
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_DATA\\_OFFSET\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_RESERVED22\\_23\\_SIZE](#) 2
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_RESERVED22\\_23\\_OFFSET](#) 22
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_RESERVED24\\_31\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_TCP\\_RETVAL\\_RESERVED24\\_31\\_OFFSET](#) 24
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_TCP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_TCP\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_TCP\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_TCP\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_UDP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_UDP\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_UDP\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_UDP\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_MPLS\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_MPLS\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_MPLS\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_MPLS\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_GRE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_GRE\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_GRE\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_GRE\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_IPV4\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_IPV4\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_IPV4\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV4\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_IPV6\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_IPV6\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_IPV6\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV6\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_ICMP\\_IGMP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_ICMP\\_IGMP\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_ICMP\\_IGMP\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_ICMP\_IGMP\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_0\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_0\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_0\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_1\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_1\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_1\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_2\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_2\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_2\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_3\\_SIZE](#) 1



- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_3\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_DEF\\_IP\\_PROT\\_3\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROT\_3\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_ESP\\_PROT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_ESP\\_PROT\\_OFFSET](#) 11
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_ESP\\_PROT\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_ESP\_PROT\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_AH\\_PROT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_AH\\_PROT\\_OFFSET](#) 12
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_AH\\_PROT\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_AH\_PROT\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_OTHER\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_OTHER\\_OFFSET](#) 13
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_OTHER\\_MASK](#) (1 << EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_OTHER\_OFFSET)
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_RESERVED14\\_31\\_SIZE](#) 18
- #define [EZDP\\_DECODE\\_IP\\_PROTOCOL\\_RETVAL\\_RESERVED14\\_31\\_OFFSET](#) 14
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_IPV4\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_IPV4\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_IPV4\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ETH\\_8100\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ETH\\_8100\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ETH\\_8100\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ETH\\_88A8\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ETH\\_88A8\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ETH\\_88A8\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ARP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ARP\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_ARP\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_MPLS\\_UNICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_MPLS\\_UNICAST\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_MPLS\\_UNICAST\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_MPLS\\_MULTICAST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_MPLS\\_MULTICAST\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_MPLS\\_MULTICAST\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MULTICAST\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_IPV6\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_IPV6\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_IPV6\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_LENGTH\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_LENGTH\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_LENGTH\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_USER\\_DEF0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_USER\\_DEF0\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_USER\\_DEF0\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_USER\\_DEF1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_USER\\_DEF1\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_USER\\_DEF1\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_PPPOE\\_SESSION\\_SIZE](#) 1

- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_PPPOE\\_SESSION\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_PPPOE\\_SESSION\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_PPPOE\\_DISCOVERY\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_PPPOE\\_DISCOVERY\\_OFFSET](#) 11
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_PPPOE\\_DISCOVERY\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DISCOVERY\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_OTHER\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_OTHER\\_OFFSET](#) 12
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_OTHER\\_MASK](#) (1 << EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_OFFSET)
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_RESERVED13\\_31\\_SIZE](#) 19
- #define [EZDP\\_DECODE\\_ETH\\_TYPE\\_RETVAL\\_RESERVED13\\_31\\_OFFSET](#) 13
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_OPTION\\_EXIST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_OPTION\\_EXIST\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_OPTION\\_EXIST\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_RETVAL\_OPTION\_EXIST\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_USER\\_CONFIG\\_SIP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_USER\\_CONFIG\\_SIP\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_USER\\_CONFIG\\_SIP\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_RETVAL\_USER\_CONFIG\_SIP\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_RESERVED\\_2\\_6\\_SIZE](#) 5
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_RESERVED\\_2\\_6\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_FIRST\\_FRAGMENT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_FIRST\\_FRAGMENT\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_FIRST\\_FRAGMENT\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_RETVAL\_FIRST\_FRAGMENT\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_ERROR\\_CODES\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_ERROR\\_CODES\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_CONTROL\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_RETVAL\\_CONTROL\\_OFFSET](#) 24
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_OPTION\\_EXIST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_OPTION\\_EXIST\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_OPTION\\_EXIST\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_OPTION\\_EXIST\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_OPTION\\_EXIST\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_RESULT\_OPTION\_EXIST\_WORD\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_USER\\_CONFIG\\_SIP\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_USER\\_CONFIG\\_SIP\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_USER\\_CONFIG\\_SIP\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_USER\\_CONFIG\\_SIP\\_WORD\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_USER\\_CONFIG\\_SIP\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_RESULT\_USER\_CONFIG\_SIP\_WORD\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_RESERVED\\_2\\_6\\_SIZE](#) 5
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_RESERVED\\_2\\_6\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_FIRST\\_FRAGMENT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_FIRST\\_FRAGMENT\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_FIRST\\_FRAGMENT\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_FIRST\\_FRAGMENT\\_WORD\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_FIRST\\_FRAGMENT\\_MASK](#) (1 << EZDP\_DECODE\_IPV4\_RESULT\_FIRST\_FRAGMENT\_WORD\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_ERROR\\_CODES\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_ERROR\\_CODES\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_ERROR\\_CODES\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_ERROR\\_CODES\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_CONTROL\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_CONTROL\\_OFFSET](#) 24

- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_CONTROL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_CONTROL\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_SIP\\_DIP\\_HASH\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_SIP\\_DIP\\_HASH\\_OFFSET](#) 32
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_SIP\\_DIP\\_HASH\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_SIP\\_DIP\\_HASH\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_NEXT\\_PROTOCOL\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_NEXT\\_PROTOCOL\\_OFFSET](#) 48
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_NEXT\\_PROTOCOL\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_NEXT\\_PROTOCOL\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_RESERVED\\_56\\_63\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_RESERVED\\_56\\_63\\_OFFSET](#) 56
- #define [EZDP\\_DECODE\\_IPV4\\_RESULT\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_CONTROL\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_CONTROL\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_OPTIONS\\_EXIST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_OPTIONS\\_EXIST\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_OPTIONS\\_EXIST\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_RETVAL\_OPTIONS\_EXIST\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_RESERVED9\\_11\\_SIZE](#) 3
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_RESERVED9\\_11\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_LINK\\_LOCAL\\_ADDRESS\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_LINK\\_LOCAL\\_ADDRESS\\_OFFSET](#) 12
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_LINK\\_LOCAL\\_ADDRESS\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_RETVAL\_LINK\_LOCAL\_ADDRESS\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_SITE\\_LOCAL\\_ADDRESS\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_SITE\\_LOCAL\\_ADDRESS\\_OFFSET](#) 13
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_SITE\\_LOCAL\\_ADDRESS\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_RETVAL\_SITE\_LOCAL\_ADDRESS\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_GLOBAL\\_ADDRESSES\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_GLOBAL\\_ADDRESSES\\_OFFSET](#) 14
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_GLOBAL\\_ADDRESSES\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_RETVAL\_GLOBAL\_ADDRESSES\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_RESERVED\\_15\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_RESERVED\\_15\\_OFFSET](#) 15
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_ERROR\\_CODES\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_IPV6\\_RETVAL\\_ERROR\\_CODES\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_CONTROL\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_CONTROL\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_CONTROL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_CONTROL\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_OPTIONS\\_EXIST\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_OPTIONS\\_EXIST\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_OPTIONS\\_EXIST\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_OPTIONS\\_EXIST\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_OPTIONS\\_EXIST\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_RESULT\_OPTIONS\_EXIST\_WORD\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_RESERVED9\\_11\\_SIZE](#) 3
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_RESERVED9\\_11\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_LINK\\_LOCAL\\_ADDRESS\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_LINK\\_LOCAL\\_ADDRESS\\_OFFSET](#) 12
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_LINK\\_LOCAL\\_ADDRESS\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_LINK\\_LOCAL\\_ADDRESS\\_WORD\\_OFFSET](#) 12
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_LINK\\_LOCAL\\_ADDRESS\\_MASK](#) (1 << EZDP\_DECODE\_IPV6\_RESULT\_LINK\_LOCAL\_ADDRESS\_WORD\_OFFSET)
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_SITE\\_LOCAL\\_ADDRESS\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_IPV6\\_RESULT\\_SITE\\_LOCAL\\_ADDRESS\\_OFFSET](#) 13

```

• #define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_SELECT 0
• #define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_OFFSET 13
• #define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_MASK (1 <<
EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_OFFSET)
• #define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_SIZE 1
• #define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_OFFSET 14
• #define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_SELECT 0
• #define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_OFFSET 14
• #define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_MASK (1 <<
EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_OFFSET)
• #define EZDP_DECODE_IPV6_RESULT_RESERVED_15_SIZE 1
• #define EZDP_DECODE_IPV6_RESULT_RESERVED_15_OFFSET 15
• #define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_SIZE 16
• #define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_OFFSET 16
• #define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_WORD_SELECT 0
• #define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_WORD_OFFSET 16
• #define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_SIZE 16
• #define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_OFFSET 32
• #define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_WORD_SELECT 1
• #define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_WORD_OFFSET 0
• #define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_SIZE 8
• #define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_OFFSET 48
• #define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_WORD_SELECT 1
• #define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_WORD_OFFSET 16
• #define EZDP_DECODE_IPV6_RESULT_RESERVED_56_63_SIZE 8
• #define EZDP_DECODE_IPV6_RESULT_RESERVED_56_63_OFFSET 56
• #define EZDP_DECODE_IPV6_RESULT_WORD_COUNT 2
• #define EZDP_DECODE_MAC_RETVAL_CONTROL_SIZE 16
• #define EZDP_DECODE_MAC_RETVAL_CONTROL_OFFSET 0
• #define EZDP_DECODE_MAC_RETVAL_ERROR_CODES_SIZE 8
• #define EZDP_DECODE_MAC_RETVAL_ERROR_CODES_OFFSET 16
• #define EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_SIZE 1
• #define EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_OFFSET 24
• #define EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_MASK (1 <<
EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_OFFSET)
• #define EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_SIZE 1
• #define EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_OFFSET 25
• #define EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_MASK (1 <<
EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_OFFSET)
• #define EZDP_DECODE_MAC_RETVAL_RESERVED26_27_SIZE 2
• #define EZDP_DECODE_MAC_RETVAL_RESERVED26_27_OFFSET 26
• #define EZDP_DECODE_MAC_RETVAL_NUMBER_OF_TAGS_SIZE 3
• #define EZDP_DECODE_MAC_RETVAL_NUMBER_OF_TAGS_OFFSET 28
• #define EZDP_DECODE_MAC_RETVAL_RESERVED_31_SIZE 1
• #define EZDP_DECODE_MAC_RETVAL_RESERVED_31_OFFSET 31
• #define EZDP_DECODE_MAC_RESULT_CONTROL_SIZE 16
• #define EZDP_DECODE_MAC_RESULT_CONTROL_OFFSET 0
• #define EZDP_DECODE_MAC_RESULT_CONTROL_WORD_SELECT 0
• #define EZDP_DECODE_MAC_RESULT_CONTROL_WORD_OFFSET 0
• #define EZDP_DECODE_MAC_RESULT_ERROR_CODES_SIZE 8
• #define EZDP_DECODE_MAC_RESULT_ERROR_CODES_OFFSET 16
• #define EZDP_DECODE_MAC_RESULT_ERROR_CODES_WORD_SELECT 0
• #define EZDP_DECODE_MAC_RESULT_ERROR_CODES_WORD_OFFSET 16
• #define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_SIZE 1
• #define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_OFFSET 24
• #define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_WORD_SELECT 0
• #define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_WORD_OFFSET 24

```

- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_IPV4\\_IN\\_PPPOE\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_RESULT\_IPV4\_IN\_PPPOE\_WORD\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_IPV6\\_IN\\_PPPOE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_IPV6\\_IN\\_PPPOE\\_OFFSET](#) 25
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_IPV6\\_IN\\_PPPOE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_IPV6\\_IN\\_PPPOE\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_IPV6\\_IN\\_PPPOE\\_MASK](#) (1 << EZDP\_DECODE\_MAC\_RESULT\_IPV6\_IN\_PPPOE\_WORD\_OFFSET)
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_RESERVED26\\_27\\_SIZE](#) 2
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_RESERVED26\\_27\\_OFFSET](#) 26
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_NUMBER\\_OF\\_TAGS\\_SIZE](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_NUMBER\\_OF\\_TAGS\\_OFFSET](#) 28
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_NUMBER\\_OF\\_TAGS\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_NUMBER\\_OF\\_TAGS\\_WORD\\_OFFSET](#) 28
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_RESERVED\\_31\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_RESERVED\\_31\\_OFFSET](#) 31
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG1\\_PROTOCOL\\_TYPE\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG1\\_PROTOCOL\\_TYPE\\_OFFSET](#) 32
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG1\\_PROTOCOL\\_TYPE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG1\\_PROTOCOL\\_TYPE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG2\\_PROTOCOL\\_TYPE\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG2\\_PROTOCOL\\_TYPE\\_OFFSET](#) 48
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG2\\_PROTOCOL\\_TYPE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG2\\_PROTOCOL\\_TYPE\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG3\\_PROTOCOL\\_TYPE\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG3\\_PROTOCOL\\_TYPE\\_OFFSET](#) 64
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG3\\_PROTOCOL\\_TYPE\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_TAG3\\_PROTOCOL\\_TYPE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAST\\_TAG\\_PROTOCOL\\_TYPE\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAST\\_TAG\\_PROTOCOL\\_TYPE\\_OFFSET](#) 80
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAST\\_TAG\\_PROTOCOL\\_TYPE\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAST\\_TAG\\_PROTOCOL\\_TYPE\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_DA\\_SA\\_HASH\\_SIZE](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_DA\\_SA\\_HASH\\_OFFSET](#) 96
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_DA\\_SA\\_HASH\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_DA\\_SA\\_HASH\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAYER2\\_SIZE\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAYER2\\_SIZE\\_OFFSET](#) 112
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAYER2\\_SIZE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_LAYER2\\_SIZE\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_RESERVED120\\_127\\_SIZE](#) 8
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_RESERVED120\\_127\\_OFFSET](#) 120
- #define [EZDP\\_DECODE\\_MAC\\_RESULT\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_DECODE\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_DECODE\\_ERROR\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_DECODE\\_ERROR\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ERROR\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED1\\_7\\_SIZE](#) 7
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED1\\_7\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LAST\\_ENTRY\\_IN\\_STACK\\_SIZE](#) 2
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LAST\\_ENTRY\\_IN\\_STACK\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED10\\_15\\_SIZE](#) 6
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED10\\_15\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL1\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL1\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL1\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_IS\_ZERO\_OFFSET)



- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL2\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL2\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 17
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL2\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL3\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL3\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 18
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL3\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL4\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL4\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 19
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL4\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED20\\_23\\_SIZE](#) 4
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED20\\_23\\_OFFSET](#) 20
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL1\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL1\\_TTL\\_IS\\_ONE\\_OFFSET](#) 24
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL1\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL2\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL2\\_TTL\\_IS\\_ONE\\_OFFSET](#) 25
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL2\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL3\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL3\\_TTL\\_IS\\_ONE\\_OFFSET](#) 26
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL3\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL4\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL4\\_TTL\\_IS\\_ONE\\_OFFSET](#) 27
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_LABEL4\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED28\\_31\\_SIZE](#) 4
- #define [EZDP\\_DECODE\\_MPLS\\_RETVAL\\_RESERVED28\\_31\\_OFFSET](#) 28
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_DECODE\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_DECODE\\_ERROR\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_DECODE\\_ERROR\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERROR\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED1\\_7\\_SIZE](#) 7
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED1\\_7\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LAST\\_ENTRY\\_IN\\_STACK\\_SIZE](#) 2
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LAST\\_ENTRY\\_IN\\_STACK\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED10\\_15\\_SIZE](#) 6
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED10\\_15\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL1\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL1\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 16
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL1\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL2\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL2\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 17
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL2\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL3\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL3\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 18
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL3\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL4\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL4\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 19
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL4\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ZERO\_OFFSET)

- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED20\\_23\\_SIZE](#) 4
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED20\\_23\\_OFFSET](#) 20
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL1\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL1\\_TTL\\_IS\\_ONE\\_OFFSET](#) 24
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL1\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL2\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL2\\_TTL\\_IS\\_ONE\\_OFFSET](#) 25
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL2\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL3\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL3\\_TTL\\_IS\\_ONE\\_OFFSET](#) 26
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL3\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL4\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL4\\_TTL\\_IS\\_ONE\\_OFFSET](#) 27
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_LABEL4\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED28\\_31\\_SIZE](#) 4
- #define [EZDP\\_DECODE\\_MPLS\\_RESULT\\_RESERVED28\\_31\\_OFFSET](#) 28
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_END\\_OF\\_STACK\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_END\\_OF\\_STACK\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_END\\_OF\\_STACK\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_END\_OF\_STACK\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_RESERVED\\_LABEL\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_RESERVED\\_LABEL\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_RESERVED\\_LABEL\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED\_LABEL\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_TTL\\_IS\\_ONE\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG0\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG0\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG0\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG1\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG1\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG1\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG2\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG2\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG2\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG3\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG3\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_USER\\_CONFIG3\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG3\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_EXCEPTION\\_BIT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_EXCEPTION\\_BIT\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_EXCEPTION\\_BIT\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_EXCEPTION\_BIT\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_STOP\\_BIT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_STOP\\_BIT\\_OFFSET](#) 9

- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_STOP\\_BIT\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_STOP\_BIT\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_RESERVED10\\_31\\_SIZE](#) 22
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RETVAL\\_RESERVED10\\_31\\_OFFSET](#) 10
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_END\\_OF\\_STACK\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_END\\_OF\\_STACK\\_OFFSET](#) 0
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_END\\_OF\\_STACK\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_END\_OF\_STACK\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_RESERVED\\_LABEL\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_RESERVED\\_LABEL\\_OFFSET](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_RESERVED\\_LABEL\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_RESERVED\_LABEL\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_TTL\\_IS\\_ZERO\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_TTL\\_IS\\_ZERO\\_OFFSET](#) 2
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_TTL\\_IS\\_ZERO\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_TTL\_IS\_ZERO\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_TTL\\_IS\\_ONE\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_TTL\\_IS\\_ONE\\_OFFSET](#) 3
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_TTL\\_IS\\_ONE\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_TTL\_IS\_ONE\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG0\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG0\\_OFFSET](#) 4
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG0\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG0\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG1\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG1\\_OFFSET](#) 5
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG1\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG1\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG2\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG2\\_OFFSET](#) 6
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG2\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG2\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG3\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG3\\_OFFSET](#) 7
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_USER\\_CONFIG3\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_USER\_CONFIG3\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_EXCEPTION\\_BIT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_EXCEPTION\\_BIT\\_OFFSET](#) 8
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_EXCEPTION\\_BIT\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_EXCEPTION\_BIT\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_STOP\\_BIT\\_SIZE](#) 1
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_STOP\\_BIT\\_OFFSET](#) 9
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_STOP\\_BIT\\_MASK](#) (1 << EZDP\_DECODE\_MPLS\_LABEL\_RESULT\_STOP\_BIT\_OFFSET)
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_RESERVED10\\_31\\_SIZE](#) 22
- #define [EZDP\\_DECODE\\_MPLS\\_LABEL\\_RESULT\\_RESERVED10\\_31\\_OFFSET](#) 10

## Typedefs

- typedef uint8\_t [ezdp\\_decode\\_ipv4\\_control\\_t](#)
- typedef uint16\_t [ezdp\\_decode\\_ipv4\\_errors\\_t](#)
- typedef uint8\_t [ezdp\\_decode\\_ip\\_next\\_protocol\\_t](#)
- typedef uint8\_t [ezdp\\_decode\\_ipv6\\_control\\_t](#)
- typedef uint16\_t [ezdp\\_decode\\_ipv6\\_errors\\_t](#)
- typedef uint16\_t [ezdp\\_decode\\_mac\\_control\\_t](#)
- typedef uint8\_t [ezdp\\_decode\\_mac\\_errors\\_t](#)
- typedef uint16\_t [ezdp\\_decode\\_mac\\_protocol\\_type\\_t](#)
- typedef uint8\_t [ezdp\\_decode\\_tcp\\_errors\\_t](#)



- typedef uint32\_t [ezdp\\_decode\\_tcp\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_ip\\_protocol\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_eth\\_type\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_ipv4\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_ipv6\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_mac\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_mpls\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_mpls\\_result\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_mpls\\_label\\_retval\\_t](#)
  - typedef uint32\_t [ezdp\\_decode\\_mpls\\_label\\_result\\_t](#)
-

## Define Documentation

```
#define EZDP_DECODE_VERSION_MAJOR 2

#define EZDP_DECODE_VERSION_MINOR 1

#define EZDP_DECODE_IPV4_CONTROL_LINK_LOCAL_MULTICAST_RANGE_SIZE 1

#define EZDP_DECODE_IPV4_CONTROL_LINK_LOCAL_MULTICAST_RANGE_OFFSET 0

#define EZDP_DECODE_IPV4_CONTROL_LINK_LOCAL_MULTICAST_RANGE_MASK (1 <<
EZDP_DECODE_IPV4_CONTROL_LINK_LOCAL_MULTICAST_RANGE_OFFSET)

#define EZDP_DECODE_IPV4_CONTROL_INTERNETWORK_MULTICAST_RANGE_SIZE 1

#define EZDP_DECODE_IPV4_CONTROL_INTERNETWORK_MULTICAST_RANGE_OFFSET 1

#define EZDP_DECODE_IPV4_CONTROL_INTERNETWORK_MULTICAST_RANGE_MASK (1 <<
EZDP_DECODE_IPV4_CONTROL_INTERNETWORK_MULTICAST_RANGE_OFFSET)

#define EZDP_DECODE_IPV4_CONTROL_RESERVED_2_SIZE 1

#define EZDP_DECODE_IPV4_CONTROL_RESERVED_2_OFFSET 2

#define EZDP_DECODE_IPV4_CONTROL_ICMP_SIZE 1

#define EZDP_DECODE_IPV4_CONTROL_ICMP_OFFSET 3

#define EZDP_DECODE_IPV4_CONTROL_ICMP_MASK (1 <<
EZDP_DECODE_IPV4_CONTROL_ICMP_OFFSET)

#define EZDP_DECODE_IPV4_CONTROL_IGMP_SIZE 1

#define EZDP_DECODE_IPV4_CONTROL_IGMP_OFFSET 4

#define EZDP_DECODE_IPV4_CONTROL_IGMP_MASK (1 <<
EZDP_DECODE_IPV4_CONTROL_IGMP_OFFSET)

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG0_SIZE 1

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG0_OFFSET 5

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG0_MASK (1 <<
EZDP_DECODE_IPV4_CONTROL_USER_CONFIG0_OFFSET)

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG1_SIZE 1

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG1_OFFSET 6

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG1_MASK (1 <<
EZDP_DECODE_IPV4_CONTROL_USER_CONFIG1_OFFSET)

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG2_SIZE 1
```

```
#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG2_OFFSET 7

#define EZDP_DECODE_IPV4_CONTROL_USER_CONFIG2_MASK (1 <<
EZDP_DECODE_IPV4_CONTROL_USER_CONFIG2_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_SIP_IS_MULTICAST_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_SIP_IS_MULTICAST_OFFSET 0

#define EZDP_DECODE_IPV4_ERRORS_SIP_IS_MULTICAST_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_SIP_IS_MULTICAST_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_SIP_IS_ZERO_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_SIP_IS_ZERO_OFFSET 1

#define EZDP_DECODE_IPV4_ERRORS_SIP_IS_ZERO_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_SIP_IS_ZERO_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_LT_5_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_LT_5_OFFSET 2

#define EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_LT_5_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_LT_5_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_TOTAL_LENGTH_GT_FRAME_LENGTH_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_TOTAL_LENGTH_GT_FRAME_LENGTH_OFFSET 3

#define EZDP_DECODE_IPV4_ERRORS_TOTAL_LENGTH_GT_FRAME_LENGTH_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_TOTAL_LENGTH_GT_FRAME_LENGTH_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_GT_FRAME_LENGTH_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_GT_FRAME_LENGTH_OFFSET 4

#define EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_GT_FRAME_LENGTH_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_HEADER_LENGTH_GT_FRAME_LENGTH_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_NOT_IPV4_VERSION_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_NOT_IPV4_VERSION_OFFSET 5

#define EZDP_DECODE_IPV4_ERRORS_NOT_IPV4_VERSION_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_NOT_IPV4_VERSION_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_CHECKSUM_ERROR_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_CHECKSUM_ERROR_OFFSET 6

#define EZDP_DECODE_IPV4_ERRORS_CHECKSUM_ERROR_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_CHECKSUM_ERROR_OFFSET)
```

```
#define EZDP_DECODE_IPV4_ERRORS_SIP_EQUAL_DIP_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_SIP_EQUAL_DIP_OFFSET 7

#define EZDP_DECODE_IPV4_ERRORS_SIP_EQUAL_DIP_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_SIP_EQUAL_DIP_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_DECODE_ERROR_SIZE 1

#define EZDP_DECODE_IPV4_ERRORS_DECODE_ERROR_OFFSET 8

#define EZDP_DECODE_IPV4_ERRORS_DECODE_ERROR_MASK (1 <<
EZDP_DECODE_IPV4_ERRORS_DECODE_ERROR_OFFSET)

#define EZDP_DECODE_IPV4_ERRORS_RESERVED9_15_SIZE 7

#define EZDP_DECODE_IPV4_ERRORS_RESERVED9_15_OFFSET 9

#define EZDP_DECODE_IP_NEXT_PROTOCOL_TCP_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_TCP_OFFSET 0

#define EZDP_DECODE_IP_NEXT_PROTOCOL_TCP_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_TCP_OFFSET)

#define EZDP_DECODE_IP_NEXT_PROTOCOL_UDP_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_UDP_OFFSET 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_UDP_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_UDP_OFFSET)

#define EZDP_DECODE_IP_NEXT_PROTOCOL_MPLS_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_MPLS_OFFSET 2

#define EZDP_DECODE_IP_NEXT_PROTOCOL_MPLS_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_MPLS_OFFSET)

#define EZDP_DECODE_IP_NEXT_PROTOCOL_GRE_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_GRE_OFFSET 3

#define EZDP_DECODE_IP_NEXT_PROTOCOL_GRE_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_GRE_OFFSET)

#define EZDP_DECODE_IP_NEXT_PROTOCOL_IPV4_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_IPV4_OFFSET 4

#define EZDP_DECODE_IP_NEXT_PROTOCOL_IPV4_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_IPV4_OFFSET)
```

```
#define EZDP_DECODE_IP_NEXT_PROTOCOL_IPV6_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_IPV6_OFFSET 5

#define EZDP_DECODE_IP_NEXT_PROTOCOL_IPV6_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_IPV6_OFFSET)

#define EZDP_DECODE_IP_NEXT_PROTOCOL_ICMP_IGMP_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_ICMP_IGMP_OFFSET 6

#define EZDP_DECODE_IP_NEXT_PROTOCOL_ICMP_IGMP_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_ICMP_IGMP_OFFSET)

#define EZDP_DECODE_IP_NEXT_PROTOCOL_OTHER_SIZE 1

#define EZDP_DECODE_IP_NEXT_PROTOCOL_OTHER_OFFSET 7

#define EZDP_DECODE_IP_NEXT_PROTOCOL_OTHER_MASK (1 <<
EZDP_DECODE_IP_NEXT_PROTOCOL_OTHER_OFFSET)

#define EZDP_DECODE_IPV6_CONTROL_LINK_LOCAL_MULTICAST_RANGE_SIZE 1

#define EZDP_DECODE_IPV6_CONTROL_LINK_LOCAL_MULTICAST_RANGE_OFFSET 0

#define EZDP_DECODE_IPV6_CONTROL_LINK_LOCAL_MULTICAST_RANGE_MASK (1 <<
EZDP_DECODE_IPV6_CONTROL_LINK_LOCAL_MULTICAST_RANGE_OFFSET)

#define EZDP_DECODE_IPV6_CONTROL_INTERNETWORK_MULTICAST_RANGE_SIZE 1

#define EZDP_DECODE_IPV6_CONTROL_INTERNETWORK_MULTICAST_RANGE_OFFSET 1

#define EZDP_DECODE_IPV6_CONTROL_INTERNETWORK_MULTICAST_RANGE_MASK (1 <<
EZDP_DECODE_IPV6_CONTROL_INTERNETWORK_MULTICAST_RANGE_OFFSET)

#define EZDP_DECODE_IPV6_CONTROL_SOLICITED_NODE_MULTICAST_RANGE_SIZE 1

#define EZDP_DECODE_IPV6_CONTROL_SOLICITED_NODE_MULTICAST_RANGE_OFFSET 2

#define EZDP_DECODE_IPV6_CONTROL_SOLICITED_NODE_MULTICAST_RANGE_MASK (1 <<
EZDP_DECODE_IPV6_CONTROL_SOLICITED_NODE_MULTICAST_RANGE_OFFSET)

#define EZDP_DECODE_IPV6_CONTROL_RESERVED_3_SIZE 1

#define EZDP_DECODE_IPV6_CONTROL_RESERVED_3_OFFSET 3

#define EZDP_DECODE_IPV6_CONTROL_DIP_IS_MULTICAST_SIZE 1

#define EZDP_DECODE_IPV6_CONTROL_DIP_IS_MULTICAST_OFFSET 4

#define EZDP_DECODE_IPV6_CONTROL_DIP_IS_MULTICAST_MASK (1 <<
EZDP_DECODE_IPV6_CONTROL_DIP_IS_MULTICAST_OFFSET)
```

```
#define EZDP_DECODE_IPV6_CONTROL_DIP_IS_WELLKNOWN_MULTICAST_SIZE 1

#define EZDP_DECODE_IPV6_CONTROL_DIP_IS_WELLKNOWN_MULTICAST_OFFSET 5

#define EZDP_DECODE_IPV6_CONTROL_DIP_IS_WELLKNOWN_MULTICAST_MASK (1 <<
EZDP_DECODE_IPV6_CONTROL_DIP_IS_WELLKNOWN_MULTICAST_OFFSET)

#define EZDP_DECODE_IPV6_CONTROL_RESERVED7_8_SIZE 2

#define EZDP_DECODE_IPV6_CONTROL_RESERVED7_8_OFFSET 6

#define EZDP_DECODE_IPV6_ERRORS_PAYLOAD_GT_FRAME_LENGTH_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_PAYLOAD_GT_FRAME_LENGTH_OFFSET 0

#define EZDP_DECODE_IPV6_ERRORS_PAYLOAD_GT_FRAME_LENGTH_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_GT_FRAME_LENGTH_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_OFFSET 1

#define EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_ZERO_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_ZERO_OFFSET 2

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_ZERO_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_SIP_IS_ZERO_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_ONE_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_ONE_OFFSET 3

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_ONE_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_SIP_IS_ONE_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_OFFSET 4

#define EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_OFFSET 5

#define EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_OFFSET)
```

```
#define EZDP_DECODE_IPV6_ERRORS_SIP_EQUAL_DIP_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_SIP_EQUAL_DIP_OFFSET 6

#define EZDP_DECODE_IPV6_ERRORS_SIP_EQUAL_DIP_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_SIP_EQUAL_DIP_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_DECODE_ERROR_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_DECODE_ERROR_OFFSET 7

#define EZDP_DECODE_IPV6_ERRORS_DECODE_ERROR_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_DECODE_ERROR_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_PAYLOAD_MISSING_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_PAYLOAD_MISSING_OFFSET 8

#define EZDP_DECODE_IPV6_ERRORS_PAYLOAD_MISSING_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_MISSING_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_MULTICAST_SIZE 1

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_MULTICAST_OFFSET 9

#define EZDP_DECODE_IPV6_ERRORS_SIP_IS_MULTICAST_MASK (1 <<
EZDP_DECODE_IPV6_ERRORS_SIP_IS_MULTICAST_OFFSET)

#define EZDP_DECODE_IPV6_ERRORS_RESERVED10_15_SIZE 6

#define EZDP_DECODE_IPV6_ERRORS_RESERVED10_15_OFFSET 10

#define EZDP_DECODE_MAC_CONTROL_MY_MAC_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_MY_MAC_OFFSET 0

#define EZDP_DECODE_MAC_CONTROL_MY_MAC_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_MY_MAC_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_0X_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_0X_OFFSET 1

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_0X_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_0X_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_1X_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_1X_OFFSET 2

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_1X_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_1X_OFFSET)
```

```
#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_2X_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_2X_OFFSET 3

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_2X_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_2X_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_OTHER_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_OTHER_OFFSET 4

#define EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_OTHER_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_OTHER_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_VRRP_MAC_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_VRRP_MAC_OFFSET 5

#define EZDP_DECODE_MAC_CONTROL_VRRP_MAC_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_VRRP_MAC_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_IPV4_MULTICAST_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_IPV4_MULTICAST_OFFSET 6

#define EZDP_DECODE_MAC_CONTROL_IPV4_MULTICAST_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_IPV4_MULTICAST_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_IPV6_MULTICAST_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_IPV6_MULTICAST_OFFSET 7

#define EZDP_DECODE_MAC_CONTROL_IPV6_MULTICAST_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_IPV6_MULTICAST_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG0_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG0_OFFSET 8

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG0_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_USER_CONFIG0_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG1_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG1_OFFSET 9

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG1_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_USER_CONFIG1_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG2_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG2_OFFSET 10
```



```
#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG2_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_USER_CONFIG2_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG3_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG3_OFFSET 11

#define EZDP_DECODE_MAC_CONTROL_USER_CONFIG3_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_USER_CONFIG3_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_SMAC_EQUALS_DMAC_SIZE 1

#define EZDP_DECODE_MAC_CONTROL_SMAC_EQUALS_DMAC_OFFSET 12

#define EZDP_DECODE_MAC_CONTROL_SMAC_EQUALS_DMAC_MASK (1 <<
EZDP_DECODE_MAC_CONTROL_SMAC_EQUALS_DMAC_OFFSET)

#define EZDP_DECODE_MAC_CONTROL_RESERVED13_15_SIZE 3

#define EZDP_DECODE_MAC_CONTROL_RESERVED13_15_OFFSET 13

#define EZDP_DECODE_MAC_ERRORS_SMAC_IS_NOT_UNICAST_SIZE 1

#define EZDP_DECODE_MAC_ERRORS_SMAC_IS_NOT_UNICAST_OFFSET 0

#define EZDP_DECODE_MAC_ERRORS_SMAC_IS_NOT_UNICAST_MASK (1 <<
EZDP_DECODE_MAC_ERRORS_SMAC_IS_NOT_UNICAST_OFFSET)

#define EZDP_DECODE_MAC_ERRORS_SMAC_IS_ZERO_SIZE 1

#define EZDP_DECODE_MAC_ERRORS_SMAC_IS_ZERO_OFFSET 1

#define EZDP_DECODE_MAC_ERRORS_SMAC_IS_ZERO_MASK (1 <<
EZDP_DECODE_MAC_ERRORS_SMAC_IS_ZERO_OFFSET)

#define EZDP_DECODE_MAC_ERRORS_DMAC_IS_ZERO_SIZE 1

#define EZDP_DECODE_MAC_ERRORS_DMAC_IS_ZERO_OFFSET 2

#define EZDP_DECODE_MAC_ERRORS_DMAC_IS_ZERO_MASK (1 <<
EZDP_DECODE_MAC_ERRORS_DMAC_IS_ZERO_OFFSET)

#define EZDP_DECODE_MAC_ERRORS_IP_VERSION_MISMATCH_IN_PPPOE_SIZE 1

#define EZDP_DECODE_MAC_ERRORS_IP_VERSION_MISMATCH_IN_PPPOE_OFFSET 3

#define EZDP_DECODE_MAC_ERRORS_IP_VERSION_MISMATCH_IN_PPPOE_MASK (1 <<
EZDP_DECODE_MAC_ERRORS_IP_VERSION_MISMATCH_IN_PPPOE_OFFSET)

#define EZDP_DECODE_MAC_ERRORS_DECODE_ERROR_SIZE 1

#define EZDP_DECODE_MAC_ERRORS_DECODE_ERROR_OFFSET 4
```

```
#define EZDP_DECODE_MAC_ERRORS_DECODE_ERROR_MASK (1 <<
EZDP_DECODE_MAC_ERRORS_DECODE_ERROR_OFFSET)

#define EZDP_DECODE_MAC_ERRORS_RESERVED5_7_SIZE 3

#define EZDP_DECODE_MAC_ERRORS_RESERVED5_7_OFFSET 5

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV4_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV4_OFFSET 0

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV4_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV4_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN0_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN0_OFFSET 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN0_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN0_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN1_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN1_OFFSET 2

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN1_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN1_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_ARP_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_ARP_OFFSET 3

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_ARP_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_ARP_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_UNICAST_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_UNICAST_OFFSET 4

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_UNICAST_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_UNICAST_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_MULTICAST_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_MULTICAST_OFFSET 5

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_MULTICAST_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_MPLS_MULTICAST_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV6_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV6_OFFSET 6
```

```
#define EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV6_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV6_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_LENGTH_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_LENGTH_OFFSET 7

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_LENGTH_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_LENGTH_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG0_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG0_OFFSET 8

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG0_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG0_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG1_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG1_OFFSET 9

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG1_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG1_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG2_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG2_OFFSET 10

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG2_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG2_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG3_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG3_OFFSET 11

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG3_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG3_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_SESSION_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_SESSION_OFFSET 12

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_SESSION_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_SESSION_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_DISCOVERY_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_DISCOVERY_OFFSET 13

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_DISCOVERY_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPOE_DISCOVERY_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN2_SIZE 1
```

```
#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN2_OFFSET 14

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN2_MASK (1 <<
EZDP_DECODE_MAC_PROTOCOL_TYPE_USER_CONFIG_VLAN2_OFFSET)

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_RESERVED_15_SIZE 1

#define EZDP_DECODE_MAC_PROTOCOL_TYPE_RESERVED_15_OFFSET 15

#define EZDP_DECODE_TCP_ERRORS_DATA_OFFSET_LT_5_SIZE 1

#define EZDP_DECODE_TCP_ERRORS_DATA_OFFSET_LT_5_OFFSET 0

#define EZDP_DECODE_TCP_ERRORS_DATA_OFFSET_LT_5_MASK (1 <<
EZDP_DECODE_TCP_ERRORS_DATA_OFFSET_LT_5_OFFSET)

#define EZDP_DECODE_TCP_ERRORS_SYN_AND_FIN_EQ_1_SIZE 1

#define EZDP_DECODE_TCP_ERRORS_SYN_AND_FIN_EQ_1_OFFSET 1

#define EZDP_DECODE_TCP_ERRORS_SYN_AND_FIN_EQ_1_MASK (1 <<
EZDP_DECODE_TCP_ERRORS_SYN_AND_FIN_EQ_1_OFFSET)

#define EZDP_DECODE_TCP_ERRORS_DECODE_ERROR_SIZE 1

#define EZDP_DECODE_TCP_ERRORS_DECODE_ERROR_OFFSET 2

#define EZDP_DECODE_TCP_ERRORS_DECODE_ERROR_MASK (1 <<
EZDP_DECODE_TCP_ERRORS_DECODE_ERROR_OFFSET)

#define EZDP_DECODE_TCP_ERRORS_RESERVED3_7_SIZE 5

#define EZDP_DECODE_TCP_ERRORS_RESERVED3_7_OFFSET 3

#define EZDP_DECODE_TCP_RETVAL_ERROR_CODES_SIZE 8

#define EZDP_DECODE_TCP_RETVAL_ERROR_CODES_OFFSET 0

#define EZDP_DECODE_TCP_RETVAL_OPTIONS_EXIST_SIZE 1

#define EZDP_DECODE_TCP_RETVAL_OPTIONS_EXIST_OFFSET 8

#define EZDP_DECODE_TCP_RETVAL_OPTIONS_EXIST_MASK (1 <<
EZDP_DECODE_TCP_RETVAL_OPTIONS_EXIST_OFFSET)

#define EZDP_DECODE_TCP_RETVAL_RESERVED9_15_SIZE 7

#define EZDP_DECODE_TCP_RETVAL_RESERVED9_15_OFFSET 9

#define EZDP_DECODE_TCP_RETVAL_DATA_OFFSET_SIZE 6

#define EZDP_DECODE_TCP_RETVAL_DATA_OFFSET_OFFSET 16
```

```
#define EZDP_DECODE_TCP_RETVAL_RESERVED22_23_SIZE 2

#define EZDP_DECODE_TCP_RETVAL_RESERVED22_23_OFFSET 22

#define EZDP_DECODE_TCP_RETVAL_RESERVED24_31_SIZE 8

#define EZDP_DECODE_TCP_RETVAL_RESERVED24_31_OFFSET 24

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_TCP_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_TCP_OFFSET 0

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_TCP_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_TCP_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_UDP_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_UDP_OFFSET 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_UDP_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_UDP_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_MPLS_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_MPLS_OFFSET 2

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_MPLS_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_MPLS_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_GRE_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_GRE_OFFSET 3

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_GRE_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_GRE_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV4_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV4_OFFSET 4

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV4_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV4_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV6_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV6_OFFSET 5

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV6_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV6_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_ICMP_IGMP_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_ICMP_IGMP_OFFSET 6
```

```
#define EZDP_DECODE_IP_PROTOCOL_RETVAL_ICMP_IGMP_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_ICMP_IGMP_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_0_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_0_OFFSET 7

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_0_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_0_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_1_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_1_OFFSET 8

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_1_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_1_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_2_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_2_OFFSET 9

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_2_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_2_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_3_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_3_OFFSET 10

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_3_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_3_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_ESP_PROT_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_ESP_PROT_OFFSET 11

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_ESP_PROT_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_ESP_PROT_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_AH_PROT_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_AH_PROT_OFFSET 12

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_AH_PROT_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_AH_PROT_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_OTHER_SIZE 1

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_OTHER_OFFSET 13

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_OTHER_MASK (1 <<
EZDP_DECODE_IP_PROTOCOL_RETVAL_OTHER_OFFSET)

#define EZDP_DECODE_IP_PROTOCOL_RETVAL_RESERVED14_31_SIZE 18
```

```
#define EZDP_DECODE_IP_PROTOCOL_RETVAL_RESERVED14_31_OFFSET 14

#define EZDP_DECODE_ETH_TYPE_RETVAL_IPV4_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_IPV4_OFFSET 0

#define EZDP_DECODE_ETH_TYPE_RETVAL_IPV4_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_IPV4_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_ETH_8100_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_ETH_8100_OFFSET 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_ETH_8100_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_ETH_8100_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_ETH_88A8_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_ETH_88A8_OFFSET 2

#define EZDP_DECODE_ETH_TYPE_RETVAL_ETH_88A8_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_ETH_88A8_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_ARP_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_ARP_OFFSET 3

#define EZDP_DECODE_ETH_TYPE_RETVAL_ARP_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_ARP_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_UNICAST_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_UNICAST_OFFSET 4

#define EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_UNICAST_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_UNICAST_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_MULTICAST_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_MULTICAST_OFFSET 5

#define EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_MULTICAST_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_MPLS_MULTICAST_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_IPV6_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_IPV6_OFFSET 6

#define EZDP_DECODE_ETH_TYPE_RETVAL_IPV6_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_IPV6_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_LENGTH_SIZE 1
```

```
#define EZDP_DECODE_ETH_TYPE_RETVAL_LENGTH_OFFSET 7

#define EZDP_DECODE_ETH_TYPE_RETVAL_LENGTH_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_LENGTH_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF0_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF0_OFFSET 8

#define EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF0_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF0_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF1_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF1_OFFSET 9

#define EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF1_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_USER_DEF1_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_SESSION_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_SESSION_OFFSET 10

#define EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_SESSION_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_SESSION_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_DISCOVERY_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_DISCOVERY_OFFSET 11

#define EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_DISCOVERY_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_PPPOE_DISCOVERY_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_OTHER_SIZE 1

#define EZDP_DECODE_ETH_TYPE_RETVAL_OTHER_OFFSET 12

#define EZDP_DECODE_ETH_TYPE_RETVAL_OTHER_MASK (1 <<
EZDP_DECODE_ETH_TYPE_RETVAL_OTHER_OFFSET)

#define EZDP_DECODE_ETH_TYPE_RETVAL_RESERVED13_31_SIZE 19

#define EZDP_DECODE_ETH_TYPE_RETVAL_RESERVED13_31_OFFSET 13

#define EZDP_DECODE_IPV4_RETVAL_OPTION_EXIST_SIZE 1

#define EZDP_DECODE_IPV4_RETVAL_OPTION_EXIST_OFFSET 0

#define EZDP_DECODE_IPV4_RETVAL_OPTION_EXIST_MASK (1 <<
EZDP_DECODE_IPV4_RETVAL_OPTION_EXIST_OFFSET)

#define EZDP_DECODE_IPV4_RETVAL_USER_CONFIG_SIP_SIZE 1
```



```
#define EZDP_DECODE_IPV4_RETVAL_USER_CONFIG_SIP_OFFSET 1

#define EZDP_DECODE_IPV4_RETVAL_USER_CONFIG_SIP_MASK (1 <<
EZDP_DECODE_IPV4_RETVAL_USER_CONFIG_SIP_OFFSET)

#define EZDP_DECODE_IPV4_RETVAL_RESERVED_2_6_SIZE 5

#define EZDP_DECODE_IPV4_RETVAL_RESERVED_2_6_OFFSET 2

#define EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_SIZE 1

#define EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_OFFSET 7

#define EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_MASK (1 <<
EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_OFFSET)

#define EZDP_DECODE_IPV4_RETVAL_ERROR_CODES_SIZE 16

#define EZDP_DECODE_IPV4_RETVAL_ERROR_CODES_OFFSET 8

#define EZDP_DECODE_IPV4_RETVAL_CONTROL_SIZE 8

#define EZDP_DECODE_IPV4_RETVAL_CONTROL_OFFSET 24

#define EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_SIZE 1

#define EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_OFFSET 0

#define EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_WORD_SELECT 0

#define EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_WORD_OFFSET 0

#define EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_MASK (1 <<
EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_WORD_OFFSET)

#define EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_SIZE 1

#define EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_OFFSET 1

#define EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_WORD_SELECT 0

#define EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_WORD_OFFSET 1

#define EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_MASK (1 <<
EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_WORD_OFFSET)

#define EZDP_DECODE_IPV4_RESULT_RESERVED_2_6_SIZE 5

#define EZDP_DECODE_IPV4_RESULT_RESERVED_2_6_OFFSET 2

#define EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_SIZE 1

#define EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_OFFSET 7
```

```
#define EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_WORD_SELECT 0

#define EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_WORD_OFFSET 7

#define EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_MASK (1 <<
EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_WORD_OFFSET)

#define EZDP_DECODE_IPV4_RESULT_ERROR_CODES_SIZE 16

#define EZDP_DECODE_IPV4_RESULT_ERROR_CODES_OFFSET 8

#define EZDP_DECODE_IPV4_RESULT_ERROR_CODES_WORD_SELECT 0

#define EZDP_DECODE_IPV4_RESULT_ERROR_CODES_WORD_OFFSET 8

#define EZDP_DECODE_IPV4_RESULT_CONTROL_SIZE 8

#define EZDP_DECODE_IPV4_RESULT_CONTROL_OFFSET 24

#define EZDP_DECODE_IPV4_RESULT_CONTROL_WORD_SELECT 0

#define EZDP_DECODE_IPV4_RESULT_CONTROL_WORD_OFFSET 24

#define EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_SIZE 16

#define EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_OFFSET 32

#define EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_WORD_SELECT 1

#define EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_WORD_OFFSET 0

#define EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_SIZE 8

#define EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_OFFSET 48

#define EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_WORD_SELECT 1

#define EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_WORD_OFFSET 16

#define EZDP_DECODE_IPV4_RESULT_RESERVED_56_63_SIZE 8

#define EZDP_DECODE_IPV4_RESULT_RESERVED_56_63_OFFSET 56

#define EZDP_DECODE_IPV4_RESULT_WORD_COUNT 2

#define EZDP_DECODE_IPV6_RETVAL_CONTROL_SIZE 8

#define EZDP_DECODE_IPV6_RETVAL_CONTROL_OFFSET 0

#define EZDP_DECODE_IPV6_RETVAL_OPTIONS_EXIST_SIZE 1

#define EZDP_DECODE_IPV6_RETVAL_OPTIONS_EXIST_OFFSET 8
```

```
#define EZDP_DECODE_IPV6_RETVAL_OPTIONS_EXIST_MASK (1 <<
EZDP_DECODE_IPV6_RETVAL_OPTIONS_EXIST_OFFSET)

#define EZDP_DECODE_IPV6_RETVAL_RESERVED9_11_SIZE 3

#define EZDP_DECODE_IPV6_RETVAL_RESERVED9_11_OFFSET 9

#define EZDP_DECODE_IPV6_RETVAL_LINK_LOCAL_ADDRESS_SIZE 1

#define EZDP_DECODE_IPV6_RETVAL_LINK_LOCAL_ADDRESS_OFFSET 12

#define EZDP_DECODE_IPV6_RETVAL_LINK_LOCAL_ADDRESS_MASK (1 <<
EZDP_DECODE_IPV6_RETVAL_LINK_LOCAL_ADDRESS_OFFSET)

#define EZDP_DECODE_IPV6_RETVAL_SITE_LOCAL_ADDRESS_SIZE 1

#define EZDP_DECODE_IPV6_RETVAL_SITE_LOCAL_ADDRESS_OFFSET 13

#define EZDP_DECODE_IPV6_RETVAL_SITE_LOCAL_ADDRESS_MASK (1 <<
EZDP_DECODE_IPV6_RETVAL_SITE_LOCAL_ADDRESS_OFFSET)

#define EZDP_DECODE_IPV6_RETVAL_GLOBAL_ADDRESSES_SIZE 1

#define EZDP_DECODE_IPV6_RETVAL_GLOBAL_ADDRESSES_OFFSET 14

#define EZDP_DECODE_IPV6_RETVAL_GLOBAL_ADDRESSES_MASK (1 <<
EZDP_DECODE_IPV6_RETVAL_GLOBAL_ADDRESSES_OFFSET)

#define EZDP_DECODE_IPV6_RETVAL_RESERVED_15_SIZE 1

#define EZDP_DECODE_IPV6_RETVAL_RESERVED_15_OFFSET 15

#define EZDP_DECODE_IPV6_RETVAL_ERROR_CODES_SIZE 16

#define EZDP_DECODE_IPV6_RETVAL_ERROR_CODES_OFFSET 16

#define EZDP_DECODE_IPV6_RESULT_CONTROL_SIZE 8

#define EZDP_DECODE_IPV6_RESULT_CONTROL_OFFSET 0

#define EZDP_DECODE_IPV6_RESULT_CONTROL_WORD_SELECT 0

#define EZDP_DECODE_IPV6_RESULT_CONTROL_WORD_OFFSET 0

#define EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_SIZE 1

#define EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_OFFSET 8

#define EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_WORD_SELECT 0

#define EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_WORD_OFFSET 8
```

```
#define EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_MASK (1 <<
EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_WORD_OFFSET)

#define EZDP_DECODE_IPV6_RESULT_RESERVED9_11_SIZE 3

#define EZDP_DECODE_IPV6_RESULT_RESERVED9_11_OFFSET 9

#define EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_SIZE 1

#define EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_OFFSET 12

#define EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_WORD_SELECT 0

#define EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_WORD_OFFSET 12

#define EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_MASK (1 <<
EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_WORD_OFFSET)

#define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_SIZE 1

#define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_OFFSET 13

#define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_SELECT 0

#define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_OFFSET 13

#define EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_MASK (1 <<
EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_OFFSET)

#define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_SIZE 1

#define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_OFFSET 14

#define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_SELECT 0

#define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_OFFSET 14

#define EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_MASK (1 <<
EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_OFFSET)

#define EZDP_DECODE_IPV6_RESULT_RESERVED_15_SIZE 1

#define EZDP_DECODE_IPV6_RESULT_RESERVED_15_OFFSET 15

#define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_SIZE 16

#define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_OFFSET 16

#define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_WORD_SELECT 0

#define EZDP_DECODE_IPV6_RESULT_ERROR_CODES_WORD_OFFSET 16

#define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_SIZE 16
```

```
#define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_OFFSET 32

#define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_WORD_SELECT 1

#define EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_WORD_OFFSET 0

#define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_SIZE 8

#define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_OFFSET 48

#define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_WORD_SELECT 1

#define EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_WORD_OFFSET 16

#define EZDP_DECODE_IPV6_RESULT_RESERVED_56_63_SIZE 8

#define EZDP_DECODE_IPV6_RESULT_RESERVED_56_63_OFFSET 56

#define EZDP_DECODE_IPV6_RESULT_WORD_COUNT 2

#define EZDP_DECODE_MAC_RETVAL_CONTROL_SIZE 16

#define EZDP_DECODE_MAC_RETVAL_CONTROL_OFFSET 0

#define EZDP_DECODE_MAC_RETVAL_ERROR_CODES_SIZE 8

#define EZDP_DECODE_MAC_RETVAL_ERROR_CODES_OFFSET 16

#define EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_SIZE 1

#define EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_OFFSET 24

#define EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_MASK (1 << EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_OFFSET)

#define EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_SIZE 1

#define EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_OFFSET 25

#define EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_MASK (1 << EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_OFFSET)

#define EZDP_DECODE_MAC_RETVAL_RESERVED26_27_SIZE 2

#define EZDP_DECODE_MAC_RETVAL_RESERVED26_27_OFFSET 26

#define EZDP_DECODE_MAC_RETVAL_NUMBER_OF_TAGS_SIZE 3

#define EZDP_DECODE_MAC_RETVAL_NUMBER_OF_TAGS_OFFSET 28

#define EZDP_DECODE_MAC_RETVAL_RESERVED_31_SIZE 1

#define EZDP_DECODE_MAC_RETVAL_RESERVED_31_OFFSET 31
```

```
#define EZDP_DECODE_MAC_RESULT_CONTROL_SIZE 16

#define EZDP_DECODE_MAC_RESULT_CONTROL_OFFSET 0

#define EZDP_DECODE_MAC_RESULT_CONTROL_WORD_SELECT 0

#define EZDP_DECODE_MAC_RESULT_CONTROL_WORD_OFFSET 0

#define EZDP_DECODE_MAC_RESULT_ERROR_CODES_SIZE 8

#define EZDP_DECODE_MAC_RESULT_ERROR_CODES_OFFSET 16

#define EZDP_DECODE_MAC_RESULT_ERROR_CODES_WORD_SELECT 0

#define EZDP_DECODE_MAC_RESULT_ERROR_CODES_WORD_OFFSET 16

#define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_SIZE 1

#define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_OFFSET 24

#define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_WORD_SELECT 0

#define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_WORD_OFFSET 24

#define EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_MASK (1 <<
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_WORD_OFFSET)

#define EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_SIZE 1

#define EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_OFFSET 25

#define EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_WORD_SELECT 0

#define EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_WORD_OFFSET 25

#define EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_MASK (1 <<
EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_WORD_OFFSET)

#define EZDP_DECODE_MAC_RESULT_RESERVED26_27_SIZE 2

#define EZDP_DECODE_MAC_RESULT_RESERVED26_27_OFFSET 26

#define EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_SIZE 3

#define EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_OFFSET 28

#define EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_WORD_SELECT 0

#define EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_WORD_OFFSET 28

#define EZDP_DECODE_MAC_RESULT_RESERVED_31_SIZE 1

#define EZDP_DECODE_MAC_RESULT_RESERVED_31_OFFSET 31
```

```
#define EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_SIZE 16

#define EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_OFFSET 32

#define EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_WORD_SELECT 1

#define EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_WORD_OFFSET 0

#define EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_SIZE 16

#define EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_OFFSET 48

#define EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_WORD_SELECT 1

#define EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_WORD_OFFSET 16

#define EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_SIZE 16

#define EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_OFFSET 64

#define EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_WORD_SELECT 2

#define EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_WORD_OFFSET 0

#define EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_SIZE 16

#define EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_OFFSET 80

#define EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_WORD_SELECT 2

#define EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_WORD_OFFSET 16

#define EZDP_DECODE_MAC_RESULT_DA_SA_HASH_SIZE 16

#define EZDP_DECODE_MAC_RESULT_DA_SA_HASH_OFFSET 96

#define EZDP_DECODE_MAC_RESULT_DA_SA_HASH_WORD_SELECT 3

#define EZDP_DECODE_MAC_RESULT_DA_SA_HASH_WORD_OFFSET 0

#define EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_SIZE 8

#define EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_OFFSET 112

#define EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_WORD_SELECT 3

#define EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_WORD_OFFSET 16

#define EZDP_DECODE_MAC_RESULT_RESERVED120_127_SIZE 8

#define EZDP_DECODE_MAC_RESULT_RESERVED120_127_OFFSET 120
```

```
#define EZDP_DECODE_MAC_RESULT_WORD_COUNT 4

#define EZDP_DECODE_MPLS_RETVAL_DECODE_ERROR_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_DECODE_ERROR_OFFSET 0

#define EZDP_DECODE_MPLS_RETVAL_DECODE_ERROR_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_DECODE_ERROR_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_RESERVED1_7_SIZE 7

#define EZDP_DECODE_MPLS_RETVAL_RESERVED1_7_OFFSET 1

#define EZDP_DECODE_MPLS_RETVAL_LAST_ENTRY_IN_STACK_SIZE 2

#define EZDP_DECODE_MPLS_RETVAL_LAST_ENTRY_IN_STACK_OFFSET 8

#define EZDP_DECODE_MPLS_RETVAL_RESERVED10_15_SIZE 6

#define EZDP_DECODE_MPLS_RETVAL_RESERVED10_15_OFFSET 10

#define EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ZERO_OFFSET 16

#define EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ZERO_OFFSET 17

#define EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ZERO_OFFSET 18

#define EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ZERO_OFFSET 19

#define EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_RESERVED20_23_SIZE 4

#define EZDP_DECODE_MPLS_RETVAL_RESERVED20_23_OFFSET 20
```



```
#define EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ONE_OFFSET 24

#define EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL1_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ONE_OFFSET 25

#define EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ONE_OFFSET 26

#define EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ONE_OFFSET 27

#define EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RETVAL_RESERVED28_31_SIZE 4

#define EZDP_DECODE_MPLS_RETVAL_RESERVED28_31_OFFSET 28

#define EZDP_DECODE_MPLS_RESULT_DECODE_ERROR_SIZE 1

#define EZDP_DECODE_MPLS_RESULT_DECODE_ERROR_OFFSET 0

#define EZDP_DECODE_MPLS_RESULT_DECODE_ERROR_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_DECODE_ERROR_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_RESERVED1_7_SIZE 7

#define EZDP_DECODE_MPLS_RESULT_RESERVED1_7_OFFSET 1

#define EZDP_DECODE_MPLS_RESULT_LAST_ENTRY_IN_STACK_SIZE 2

#define EZDP_DECODE_MPLS_RESULT_LAST_ENTRY_IN_STACK_OFFSET 8

#define EZDP_DECODE_MPLS_RESULT_RESERVED10_15_SIZE 6

#define EZDP_DECODE_MPLS_RESULT_RESERVED10_15_OFFSET 10

#define EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ZERO_SIZE 1
```

```
#define EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ZERO_OFFSET 16

#define EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ZERO_OFFSET 17

#define EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ZERO_OFFSET 18

#define EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ZERO_OFFSET 19

#define EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_RESERVED20_23_SIZE 4

#define EZDP_DECODE_MPLS_RESULT_RESERVED20_23_OFFSET 20

#define EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ONE_OFFSET 24

#define EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL1_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ONE_OFFSET 25

#define EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL2_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ONE_OFFSET 26

#define EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL3_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ONE_SIZE 1
```

```
#define EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ONE_OFFSET 27

#define EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_RESULT_LABEL4_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_RESULT_RESERVED28_31_SIZE 4

#define EZDP_DECODE_MPLS_RESULT_RESERVED28_31_OFFSET 28

#define EZDP_DECODE_MPLS_LABEL_RETVAL_END_OF_STACK_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_END_OF_STACK_OFFSET 0

#define EZDP_DECODE_MPLS_LABEL_RETVAL_END_OF_STACK_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_END_OF_STACK_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED_LABEL_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED_LABEL_OFFSET 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED_LABEL_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED_LABEL_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ZERO_OFFSET 2

#define EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ONE_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ONE_OFFSET 3

#define EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG0_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG0_OFFSET 4

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG0_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG0_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG1_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG1_OFFSET 5

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG1_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG1_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG2_SIZE 1
```

```
#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG2_OFFSET 6

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG2_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG2_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG3_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG3_OFFSET 7

#define EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG3_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_USER_CONFIG3_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_EXCEPTION_BIT_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_EXCEPTION_BIT_OFFSET 8

#define EZDP_DECODE_MPLS_LABEL_RETVAL_EXCEPTION_BIT_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_EXCEPTION_BIT_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_STOP_BIT_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RETVAL_STOP_BIT_OFFSET 9

#define EZDP_DECODE_MPLS_LABEL_RETVAL_STOP_BIT_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RETVAL_STOP_BIT_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED10_31_SIZE 22

#define EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED10_31_OFFSET 10

#define EZDP_DECODE_MPLS_LABEL_RESULT_END_OF_STACK_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_END_OF_STACK_OFFSET 0

#define EZDP_DECODE_MPLS_LABEL_RESULT_END_OF_STACK_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_END_OF_STACK_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED_LABEL_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED_LABEL_OFFSET 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED_LABEL_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED_LABEL_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ZERO_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ZERO_OFFSET 2

#define EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ZERO_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ZERO_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ONE_SIZE 1
```

```
#define EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ONE_OFFSET 3

#define EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ONE_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ONE_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG0_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG0_OFFSET 4

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG0_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG0_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG1_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG1_OFFSET 5

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG1_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG1_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG2_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG2_OFFSET 6

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG2_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG2_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG3_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG3_OFFSET 7

#define EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG3_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG3_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_EXCEPTION_BIT_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_EXCEPTION_BIT_OFFSET 8

#define EZDP_DECODE_MPLS_LABEL_RESULT_EXCEPTION_BIT_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_EXCEPTION_BIT_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_STOP_BIT_SIZE 1

#define EZDP_DECODE_MPLS_LABEL_RESULT_STOP_BIT_OFFSET 9

#define EZDP_DECODE_MPLS_LABEL_RESULT_STOP_BIT_MASK (1 <<
EZDP_DECODE_MPLS_LABEL_RESULT_STOP_BIT_OFFSET)

#define EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED10_31_SIZE 22

#define EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED10_31_OFFSET 10
```

## Typedef Documentation

typedef uint8\_t [ezdp\\_decode\\_ipv4\\_control\\_t](#)

typedef uint16\_t [ezdp\\_decode\\_ipv4\\_errors\\_t](#)

typedef uint8\_t [ezdp\\_decode\\_ip\\_next\\_protocol\\_t](#)

typedef uint8\_t [ezdp\\_decode\\_ipv6\\_control\\_t](#)

typedef uint16\_t [ezdp\\_decode\\_ipv6\\_errors\\_t](#)

typedef uint16\_t [ezdp\\_decode\\_mac\\_control\\_t](#)

typedef uint8\_t [ezdp\\_decode\\_mac\\_errors\\_t](#)

typedef uint16\_t [ezdp\\_decode\\_mac\\_protocol\\_type\\_t](#)

typedef uint8\_t [ezdp\\_decode\\_tcp\\_errors\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_tcp\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_ip\\_protocol\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_eth\\_type\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_ipv4\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_ipv6\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_mac\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_mpls\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_mpls\\_result\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_mpls\\_label\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_decode\\_mpls\\_label\\_result\\_t](#)

## dpe/dp/include/ezdp\_defs.h File Reference

### Defines

- #define [likely](#)(x) \_\_builtin\_expect(!!(x),1)
  - #define [unlikely](#)(x) \_\_builtin\_expect(!!(x),0)
  - #define [no inline](#) \_\_attribute\_\_((noinline))
  - #define [unused](#) \_\_attribute\_\_((unused))
  - #define [packed](#) \_\_attribute\_\_((packed))
  - #define [packed\\_struct](#)
  - #define [aligned\\_cmem\\_ext\\_addr](#) \_\_attribute\_\_((aligned(8)))
  - #define [cmem](#)
  - #define [cmem\\_var](#) \_\_attribute\_\_((section(".cmem")))
  - #define [cmem\\_shared\\_var](#) \_\_attribute\_\_((section(".cmem\_shared")))
  - #define [alter\\_cmem\\_var](#) \_\_attribute\_\_((section(".cmem\_alter")))
  - #define [alter\\_cmem\\_shared\\_var](#) \_\_attribute\_\_((section(".cmem\_shared\_alter")))
  - #define [imem\\_private\\_var](#) \_\_attribute\_\_((section(".fmt\_slot0")))
  - #define [imem\\_half\\_cluster\\_var](#) \_\_attribute\_\_((section(".fmt\_slot2")))
  - #define [imem\\_1\\_cluster\\_var](#) \_\_attribute\_\_((section(".fmt\_slot4")))
  - #define [imem\\_2\\_cluster\\_var](#) \_\_attribute\_\_((section(".fmt\_slot6")))
  - #define [imem\\_4\\_cluster\\_var](#) \_\_attribute\_\_((section(".fmt\_slot8")))
  - #define [imem\\_16\\_cluster\\_var](#) \_\_attribute\_\_((section(".fmt\_slot10")))
  - #define [imem\\_all\\_cluster\\_var](#) \_\_attribute\_\_((section(".fmt\_slot12")))
  - #define [emem\\_var](#) \_\_attribute\_\_((section(".fmt\_slot14")))
  - #define [imem\\_half\\_cluster\\_func](#) \_\_attribute\_\_((section(".fmt\_slot3")))
  - #define [imem\\_1\\_cluster\\_func](#) \_\_attribute\_\_((section(".fmt\_slot5")))
  - #define [imem\\_2\\_cluster\\_func](#) \_\_attribute\_\_((section(".fmt\_slot7")))
  - #define [imem\\_4\\_cluster\\_func](#) \_\_attribute\_\_((section(".fmt\_slot9")))
  - #define [imem\\_16\\_cluster\\_func](#) \_\_attribute\_\_((section(".fmt\_slot11")))
  - #define [imem\\_all\\_cluster\\_func](#) \_\_attribute\_\_((section(".fmt\_slot13")))
-

## Define Documentation

```
#define likely(x) __builtin_expect(!!(x),1)

#define unlikely(x) __builtin_expect(!!(x),0)

#define __no_inline __attribute__((noinline))

#define __unused __attribute__((unused))

#define __packed __attribute__((packed))

#define __packed_struct

#define __aligned_cmem_ext_addr __attribute__((aligned (8)))

#define __cmem

#define __cmem_var __attribute__((section(".cmem")))

#define __cmem_shared_var __attribute__((section(".cmem_shared")))

#define __alter_cmem_var __attribute__((section(".cmem_alter")))

#define __alter_cmem_shared_var __attribute__((section(".cmem_shared_alter")))

#define __imem_private_var __attribute__((section(".fmt_slot0")))

#define __imem_half_cluster_var __attribute__((section(".fmt_slot2")))

#define __imem_1_cluster_var __attribute__((section(".fmt_slot4")))

#define __imem_2_cluster_var __attribute__((section(".fmt_slot6")))

#define __imem_4_cluster_var __attribute__((section(".fmt_slot8")))

#define __imem_16_cluster_var __attribute__((section(".fmt_slot10")))

#define __imem_all_cluster_var __attribute__((section(".fmt_slot12")))

#define __emem_var __attribute__((section(".fmt_slot14")))

#define __imem_half_cluster_func __attribute__((section(".fmt_slot3")))

#define __imem_1_cluster_func __attribute__((section(".fmt_slot5")))

#define __imem_2_cluster_func __attribute__((section(".fmt_slot7")))

#define __imem_4_cluster_func __attribute__((section(".fmt_slot9")))

#define __imem_16_cluster_func __attribute__((section(".fmt_slot11")))
```



```
#define __imem_all_cluster_func __attribute__((section(".fmt_slot13")))
```

## dpe/dp/include/ezdp\_dma.h File Reference

### Functions

- static `__always_inline uint32_t ezdp_copy_data_by_ext_addr` (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Copy data between two extended addresses. static `__always_inline void ezdp_copy_data_by_ext_addr_async` (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_data\\_by\\_ext\\_addr\(\)](#). static `__always_inline void ezdp_load_data_from_ext_addr` (void \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Copy data from an extended address to CMEM. static `__always_inline void ezdp_load_data_from_ext_addr_async` (void \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_load\\_data\\_from\\_ext\\_addr\(\)](#). static `__always_inline uint32_t ezdp_load_16_byte_data_from_ext_addr` (void \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \*src\_ptr, uint32\_t flags)
- Copy 16 bytes from an extended address to CMEM. static `__always_inline void ezdp_load_16_byte_data_from_ext_addr_async` (void \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \*src\_ptr, uint32\_t flags)
- Non blocking version of [ezdp\\_load\\_16\\_byte\\_data\\_from\\_ext\\_addr\(\)](#). static `__always_inline uint32_t ezdp_load_32_byte_data_from_ext_addr` (void \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \*src\_ptr, uint32\_t flags)
- Copy 32 bytes from an extended address to CMEM. static `__always_inline void ezdp_load_32_byte_data_from_ext_addr_async` (void \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \*src\_ptr, uint32\_t flags)
- Non blocking version of [ezdp\\_load\\_32\\_byte\\_data\\_from\\_ext\\_addr\(\)](#). static `__always_inline uint32_t ezdp_store_data_to_ext_addr` (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Copy data from CMEM to an extended address. static `__always_inline void ezdp_store_data_to_ext_addr_async` (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_store\\_data\\_to\\_ext\\_addr\(\)](#). static `__always_inline void ezdp_store_16_byte_data_to_ext_addr` (struct [ezdp\\_ext\\_addr](#) \*dst\_ptr, void \_\_cmem \*src\_ptr, uint32\_t flags)
- Copy 16 bytes from CMEM to an extended address. static `__always_inline void ezdp_store_16_byte_data_to_ext_addr_async` (struct [ezdp\\_ext\\_addr](#) \*dst\_ptr, void \_\_cmem \*src\_ptr, uint32\_t flags)
- Non blocking version of [ezdp\\_store\\_16\\_byte\\_data\\_to\\_ext\\_addr\(\)](#). static `__always_inline void ezdp_store_32_byte_data_to_ext_addr` (struct [ezdp\\_ext\\_addr](#) \*dst\_ptr, void \_\_cmem \*src\_ptr, uint32\_t flags)
- Copy 32 bytes from CMEM to an extended address. static `__always_inline void ezdp_store_32_byte_data_to_ext_addr_async` (struct [ezdp\\_ext\\_addr](#) \*dst\_ptr, void \_\_cmem \*src\_ptr, uint32\_t flags)
- Non blocking version of [ezdp\\_store\\_32\\_byte\\_data\\_to\\_ext\\_addr\(\)](#). static `__always_inline uint32_t ezdp_load_data_from_sum_addr` (struct [ezdp\\_sum\\_addr](#) \_\_cmem \*src\_sum\_addr\_ptr, uint32\_t sum\_addr\_entry\_size, uint32\_t sum\_addr\_offset, uint8\_t \_\_cmem \*dst\_ptr, uint32\_t size, uint32\_t flags)
- Load data from a summarized address to CMEM. static `__always_inline void ezdp_load_data_from_sum_addr_async` (struct [ezdp\\_sum\\_addr](#) \_\_cmem \*src\_sum\_addr\_ptr, uint32\_t sum\_addr\_entry\_size, uint32\_t sum\_addr\_offset, uint8\_t \_\_cmem \*dst\_ptr, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_load\\_data\\_from\\_sum\\_addr\(\)](#). static `__always_inline uint32_t ezdp_load_16_byte_data_from_sum_addr` ([ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint8\_t \_\_cmem \*ptr, uint32\_t flags)
- Load 16 bytes from a summarized address to CMEM. static `__always_inline void ezdp_load_16_byte_data_from_sum_addr_async` ([ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint8\_t \_\_cmem \*ptr, uint32\_t flags)
- Load 16 bytes from a summarized address to CMEM. static `__always_inline uint32_t ezdp_load_32_byte_data_from_sum_addr` ([ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint8\_t \_\_cmem \*ptr, uint32\_t flags)
- Load 32 bytes from a summarized address to CMEM. static `__always_inline void ezdp_load_32_byte_data_from_sum_addr_async` ([ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint8\_t \_\_cmem \*ptr, uint32\_t flags)

- *Load 32 bytes from a summarized address to CMEM.* static \_\_always\_inline uint32\_t [ezdp\\_store\\_data\\_to\\_sum\\_addr](#) (uint8\_t \_\_cmem \*src\_ptr, struct [ezdp\\_sum\\_addr](#) \_\_cmem \*dst\_sum\_addr\_ptr, uint32\_t sum\_addr\_entry\_size, uint32\_t sum\_addr\_offset, uint32\_t size, uint32\_t flags)
- *Store data from CMEM to summarized address.* static \_\_always\_inline void [ezdp\\_store\\_data\\_to\\_sum\\_addr\\_async](#) (uint8\_t \_\_cmem \*src\_ptr, struct [ezdp\\_sum\\_addr](#) \_\_cmem \*dst\_sum\_addr\_ptr, uint32\_t sum\_addr\_entry\_size, uint32\_t sum\_addr\_offset, uint32\_t size, uint32\_t flags)
- *Non blocking version of [ezdp\\_store\\_data\\_to\\_sum\\_addr\(\)](#).* static \_\_always\_inline void [ezdp\\_store\\_16\\_byte\\_data\\_to\\_sum\\_addr](#) (uint8\_t \_\_cmem \*ptr, [ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint32\_t flags)
- *Store 16 bytes from CMEM to a summarized address.* static \_\_always\_inline void [ezdp\\_store\\_16\\_byte\\_data\\_to\\_sum\\_addr\\_async](#) (uint8\_t \_\_cmem \*ptr, [ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint32\_t flags)
- *Non blocking version of [ezdp\\_store\\_16\\_byte\\_data\\_to\\_sum\\_addr\(\)](#).* static \_\_always\_inline void [ezdp\\_store\\_32\\_byte\\_data\\_to\\_sum\\_addr](#) (uint8\_t \_\_cmem \*ptr, [ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint32\_t flags)
- *Store 32 bytes from CMEM to a summarized address.* static \_\_always\_inline void [ezdp\\_store\\_32\\_byte\\_data\\_to\\_sum\\_addr\\_async](#) (uint8\_t \_\_cmem \*ptr, [ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size, uint32\_t offset, uint32\_t flags)

Non blocking version of [ezdp\\_store\\_32\\_byte\\_data\\_to\\_sum\\_addr\(\)](#).

---

## Function Documentation

static \_\_always\_inline uint32\_t ezdp\_copy\_data\_by\_ext\_addr (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags) [static]

Copy data between two extended addresses.

### Parameters:

- [in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)
- [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

### Note:

In EMEM atomic operation is limited to 128 bytes. In IMEM atomic operation is limited to 32 bytes.

### Returns:

uint32\_t - 16 bit checksum value

static \_\_always\_inline void ezdp\_copy\_data\_by\_ext\_addr\_async (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags) [static]

Non blocking version of [ezdp\\_copy\\_data\\_by\\_ext\\_addr\(\)](#).

### Parameters:

- [in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)
- [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_load_data_from_ext_addr (void __cmem * dst_ptr, struct
ezdp_ext_addr __cmem * src_ptr, uint32_t size, uint32_t flags) [static]
```

Copy data from an extended address to CMEM.

**Parameters:**

[out] *dst\_ptr* - pointer to the destination array in CMEM where the content is to be copied  
 [in] *src\_ptr* - pointer to the source extended address (in CMEM in 8 byte alignment) to be copied  
 [in] *size* - number of bytes to copy  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

In EMEM atomic operation is limited to 128 bytes. In IMEM atomic operation is limited to 32 bytes.

**Returns:**

none

```
static __always_inline void ezdp_load_data_from_ext_addr_async (void __cmem * dst_ptr,
struct ezdp_ext_addr __cmem * src_ptr, uint32_t size, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_load\\_data\\_from\\_ext\\_addr\(\)](#).

**Parameters:**

[out] *dst\_ptr* - pointer in CMEM to copy data to  
 [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)  
 [in] *size* - number of bytes to copy  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_load_16_byte_data_from_ext_addr (void __cmem * dst_ptr,
struct ezdp_ext_addr * src_ptr, uint32_t flags) [static]
```

Copy 16 bytes from an extended address to CMEM.

**Parameters:**

[out] *dst\_ptr* - pointer to the destination array in CMEM where the content is to be copied  
 [in] *src\_ptr* - pointer to the source extended address  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

The extended address must be 16-byte aligned.

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_load_16_byte_data_from_ext_addr_async(void __cmem *
dst_ptr, struct ezdp\_ext\_addr * src_ptr, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_load\\_16\\_byte\\_data\\_from\\_ext\\_addr\(\)](#).

**Parameters:**

[out] *dst\_ptr* - pointer in CMEM to copy data to

[in] *src\_ptr* - pointer to source extended address

[in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_load_32_byte_data_from_ext_addr(void __cmem * dst_ptr,
struct ezdp\_ext\_addr * src_ptr, uint32_t flags) [static]
```

Copy 32 bytes from an extended address to CMEM.

**Parameters:**

[out] *dst\_ptr* - pointer to the destination array in CMEM where the content is to be copied

[in] *src\_ptr* - pointer to the source extended address

[in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

The extended address must be 16-byte aligned.

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_load_32_byte_data_from_ext_addr_async(void __cmem *
dst_ptr, struct ezdp\_ext\_addr * src_ptr, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_load\\_32\\_byte\\_data\\_from\\_ext\\_addr\(\)](#).

**Parameters:**

[out] *dst\_ptr* - pointer in CMEM to copy data to

[in] *src\_ptr* - pointer to source extended address

[in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_store_data_to_ext_addr(struct ezdp\_ext\_addr __cmem *
dst_ptr, void __cmem * src_ptr, uint32_t size, uint32_t flags) [static]
```

Copy data from CMEM to an extended address.

**Parameters:**

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
 [in] *src\_ptr* - pointer in CMEM to copy data from  
 [in] *size* - number of bytes to copy  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

returned 16 bit checksum is applicable only when memory type is external or memory type is internal and *msid* > 5 In EMEM atomic operation is limited to 128 bytes. In IMEM atomic operation is limited to 32 bytes.

**Returns:**

uint32\_t - 16 bit checksum value

```
static __always_inline void ezdp_store_data_to_ext_addr_async (struct ezdp\_ext\_addr __cmem *  
dst_ptr, void __cmem * src_ptr, uint32_t size, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_store\\_data\\_to\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
 [in] *src\_ptr* - pointer in CMEM to copy data from  
 [in] *size* - number of bytes to copy  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

non

```
static __always_inline void ezdp_store_16_byte_data_to_ext_addr (struct ezdp\_ext\_addr * dst_ptr,  
void __cmem * src_ptr, uint32_t flags) [static]
```

Copy 16 bytes from CMEM to an extended address.

**Parameters:**

[in] *dst\_ptr* - pointer to destination extended address  
 [in] *src\_ptr* - pointer in CMEM to copy data from  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

The extended address must be 16-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_store_16_byte_data_to_ext_addr_async (struct ezdp\_ext\_addr *  
dst_ptr, void __cmem * src_ptr, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_store\\_16\\_byte\\_data\\_to\\_ext\\_addr\(\)](#).

**Parameters:**

- [in] *dst\_ptr* - pointer in CMEM to copy data to
- [in] *src\_ptr* - pointer to source extended address
- [in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_store_32_byte_data_to_ext_addr (struct ezdp\_ext\_addr * dst_ptr,
void __cmem * src_ptr,  uint32_t flags) [static]
```

Copy 32 bytes from CMEM to an extended address.

**Parameters:**

- [in] *dst\_ptr* - pointer to destination extended address
- [in] *src\_ptr* - pointer in CMEM to copy data from
- [in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

The extended address must be 16-byte aligned.

**Returns:**

void

```
static __always_inline void ezdp_store_32_byte_data_to_ext_addr_async (struct ezdp\_ext\_addr *
dst_ptr,  void __cmem * src_ptr,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_store\\_32\\_byte\\_data\\_to\\_ext\\_addr\(\)](#).

**Parameters:**

- [in] *dst\_ptr* - pointer in CMEM to copy data to
- [in] *src\_ptr* - pointer to source extended address
- [in] *flags* - execution flags. Bitwise OR of zero or more flags. No flags defined for this function yet.

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_load_data_from_sum_addr (struct ezdp\_sum\_addr __cmem *
src_sum_addr_ptr,  uint32_t sum_addr_entry_size,  uint32_t sum_addr_offset,  uint8_t
__cmem * dst_ptr,  uint32_t size,  uint32_t flags) [static]
```

Load data from a summarized address to CMEM.

**Parameters:**

- [in] *src\_sum\_addr\_ptr* - pointer to summarized address in CMEM

[in] *sum\_addr\_entry\_size* - the entry size in bytes. (Applicable values are 16,32,64,128,256)  
 [in] *sum\_addr\_offset* - the offset in bytes to add to base address (Applicable values are 0,16,32,48,64,...,240)  
 [out] *dst\_ptr* - pointer to the destination in CMEM  
 [in] *size* - the amount of data to load in byte (Applicable values are 0,16,32,48,64,...,240,256)  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

In EMEM atomic operation is limited to 128 bytes. In IMEM atomic operation is limited to 32 bytes.

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_load_data_from_sum_addr_async(struct ezdp_sum_addr
__cmem * src_sum_addr_ptr,  uint32_t sum_addr_entry_size,  uint32_t sum_addr_offset,
uint8_t __cmem * dst_ptr,  uint32_t size,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_load\\_data\\_from\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *src\_sum\_addr\_ptr* - pointer to summarized address in CMEM  
 [in] *sum\_addr\_entry\_size* - the entry size in bytes. (Applicable values are 16,32,64,128,256)  
 [in] *sum\_addr\_offset* - the offset in bytes to add to base address (Applicable values are 0,16,32,48,64,...,240)  
 [out] *dst\_ptr* - pointer to the destination in CMEM  
 [in] *size* - the amount of data to load in byte (Applicable values are 0,16,32,48,64,...,240,256)  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_load_16_byte_data_from_sum_addr (ezdp_sum_addr_t
sum_addr,  uint32_t entry_size,  uint32_t offset,  uint8_t __cmem * ptr,  uint32_t flags)
[static]
```

Load 16 bytes from a summarized address to CMEM.

**Parameters:**

[in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory  
 [in] *ptr* - pointer to address in CMEM  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_load_16_byte_data_from_sum_addr_async (ezdp_sum_addr_t
sum_addr,  uint32_t entry_size,  uint32_t offset,  uint8_t __cmem * ptr,  uint32_t flags)
[static]
```



Load 16 bytes from a summarized address to CMEM.

**Parameters:**

[in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory  
 [in] *ptr* - pointer to address in CMEM  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_load_32_byte_data_from_sum_addr (ezdp_sum_addr_t
sum_addr, uint32_t entry_size, uint32_t offset, uint8_t __cmem * ptr, uint32_t flags)
[static]
```

Load 32 bytes from a summarized address to CMEM.

**Parameters:**

[in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory  
 [in] *ptr* - pointer to address in CMEM  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_load_32_byte_data_from_sum_addr_async (ezdp_sum_addr_t
sum_addr, uint32_t entry_size, uint32_t offset, uint8_t __cmem * ptr, uint32_t flags)
[static]
```

Load 32 bytes from a summarized address to CMEM.

**Parameters:**

[in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory  
 [in] *ptr* - pointer to address in CMEM  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline uint32_t ezdp_store_data_to_sum_addr (uint8_t __cmem * src_ptr, struct
ezdp_sum_addr __cmem * dst_sum_addr_ptr, uint32_t sum_addr_entry_size, uint32_t
sum_addr_offset, uint32_t size, uint32_t flags) [static]
```

Store data from CMEM to summarized address.

#### Parameters:

[in] *src\_ptr* - pointer to the source in CMEM  
[out] *dst\_sum\_addr\_ptr* - pointer to summarized address of the destination in CMEM  
[in] *sum\_addr\_entry\_size* - the entry size in bytes. (Applicable values are 16,32,64,128,256)  
[in] *sum\_addr\_offset* - the offset in bytes to add to base address (Applicable values are 0,16,32,48,64,...,240)  
[in] *size* - the amount of data to load in byte (Applicable values are 0,16,32,48,64,...,240,256)  
[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

#### Note:

returned 16 bit checksum is applicable only when memory type is external or memory type is internal and *msid* > 5. In EMEM atomic operation is limited to 128 bytes. In IMEM atomic operation is limited to 32 bytes.

#### Returns:

uint32\_t - 16 bit checksum value

```
static __always_inline void ezdp_store_data_to_sum_addr_async (uint8_t __cmem * src_ptr,
struct ezdp_sum_addr __cmem * dst_sum_addr_ptr, uint32_t sum_addr_entry_size, uint32_t
sum_addr_offset, uint32_t size, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_store\\_data\\_to\\_sum\\_addr\(\)](#).

#### Parameters:

[in] *src\_ptr* - pointer to the source in CMEM  
[out] *dst\_sum\_addr\_ptr* - pointer to summarized address of the destination in CMEM  
[in] *sum\_addr\_entry\_size* - the entry size in bytes. (Applicable values are 16,32,64,128,256)  
[in] *sum\_addr\_offset* - the offset in bytes to add to base address (Applicable values are 0,16,32,48,64,...,240)  
[in] *size* - the amount of data to load in byte (Applicable values are 0,16,32,48,64,...,240,256)  
[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

#### Note:

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

#### Returns:

void

```
static __always_inline void ezdp_store_16_byte_data_to_sum_addr (uint8_t __cmem * ptr,
ezdp_sum_addr t sum_addr, uint32_t entry_size, uint32_t offset, uint32_t flags) [static]
```

Store 16 bytes from CMEM to a summarized address.

#### Parameters:

[in] *ptr* - pointer to address in CMEM

[in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Returns:**

void

```
static __always_inline void ezdp_store_16_byte_data_to_sum_addr_async(uint8_t __cmem * ptr,
ezdp_sum_addr_t sum_addr,  uint32_t entry_size,  uint32_t offset,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_store\\_16\\_byte\\_data\\_to\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *ptr* - pointer to address in CMEM  
 [in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

```
static __always_inline void ezdp_store_32_byte_data_to_sum_addr(uint8_t __cmem * ptr,
ezdp_sum_addr_t sum_addr,  uint32_t entry_size,  uint32_t offset,  uint32_t flags) [static]
```

Store 32 bytes from CMEM to a summarized address.

**Parameters:**

[in] *ptr* - pointer to address in CMEM  
 [in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Returns:**

void

```
static __always_inline void ezdp_store_32_byte_data_to_sum_addr_async(uint8_t __cmem * ptr,
ezdp_sum_addr_t sum_addr,  uint32_t entry_size,  uint32_t offset,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_store\\_32\\_byte\\_data\\_to\\_sum\\_addr\(\)](#).

**Parameters:**

[in] *ptr* - pointer to address in CMEM  
 [in] *sum\_addr* - summarized address  
 [in] *entry\_size* - size of entry  
 [in] *offset* - offset into memory

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

void

## dpe/dp/include/ezdp\_frame.h File Reference

### Functions

- static `__always_inline` [ezdp\\_buffer\\_desc\\_t ezdp\\_alloc\\_buf](#) (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type, uint32\_t buf\_budget\_id)
- *Allocate a single frame buffer.* static `__always_inline` void [ezdp\\_free\\_buf](#) (uint32\_t buf\_budget\_id, [ezdp\\_buffer\\_desc\\_t](#) bd\_raw\_data)
- *Free a single frame buffer.* static `__always_inline` void [ezdp\\_free\\_buf\\_async](#) (uint32\_t buf\_budget\_id, [ezdp\\_buffer\\_desc\\_t](#) bd\_raw\_data)
- *Non blocking version of [ezdp\\_free\\_buf\(\)](#).* static `__always_inline` [ezdp\\_buffer\\_desc\\_t ezdp\\_alloc\\_multi\\_buf](#) (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type, uint32\_t buf\_budget\_id, uint32\_t num\_of\_bufs, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*bds\_ptr)
- *Allocate multiple frame buffers.* static `__always_inline` void [ezdp\\_alloc\\_multi\\_buf\\_async](#) (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type, uint32\_t buf\_budget\_id, uint32\_t num\_of\_bufs, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*bds\_ptr)
- *Non blocking version of [ezdp\\_alloc\\_multi\\_buf\(\)](#).* static `__always_inline` bool [ezdp\\_buf\\_alloc\\_failed](#) ([ezdp\\_buffer\\_desc\\_t](#) ret)
- *Check if allocation of the buffer failed.* static `__always_inline` void [ezdp\\_free\\_multi\\_buf](#) (uint32\_t buf\_budget\_id, uint32\_t num\_of\_bufs, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*bds\_ptr)
- *Free multiple frame buffers.* static `__always_inline` void [ezdp\\_free\\_multi\\_buf\\_async](#) (uint32\_t buf\_budget\_id, uint32\_t num\_of\_bufs, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*bds\_ptr)
- *Non blocking version of [ezdp\\_free\\_multi\\_buf\(\)](#).* static `__always_inline` uint32\_t [ezdp\\_read\\_free\\_buf](#) (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type)
- *The number of buffers available to be obtained.* static `__always_inline` void [ezdp\\_rebudget\\_buf](#) (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type, uint32\_t free\_buf\_budget\_id, uint32\_t free\_num\_of\_bufs, uint32\_t alloc\_buf\_budget\_id, uint32\_t alloc\_num\_of\_bufs)
- *Update the budget to which buffers are credited.* static `__always_inline` void [ezdp\\_rebudget\\_buf\\_async](#) (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type, uint32\_t free\_buf\_budget\_id, uint32\_t free\_num\_of\_bufs, uint32\_t alloc\_buf\_budget\_id, uint32\_t alloc\_num\_of\_bufs)
- *Non blocking version of [ezdp\\_rebudget\\_buf\(\)](#).* static `__always_inline` uint32\_t [ezdp\\_copy\\_frame\\_data](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- *Copy data between two frame buffers.* static `__always_inline` void [ezdp\\_copy\\_frame\\_data\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- *Non blocking version of [ezdp\\_copy\\_frame\\_data\(\)](#).* static `__always_inline` uint32\_t [ezdp\\_clone\\_frame\\_data](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t bd\_offset, uint32\_t size, uint32\_t flags)
- *Copy data between two frame buffers, with the same source and destination offset (optimized).* static `__always_inline` void [ezdp\\_clone\\_frame\\_data\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t bd\_offset, uint32\_t size, uint32\_t flags)
- *Non blocking version of [ezdp\\_clone\\_frame\\_data\(\)](#).* static `__always_inline` uint32\_t [ezdp\\_copy\\_frame\\_data\\_to\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- *Copy data from a frame buffer to an extended address.* static `__always_inline` void [ezdp\\_copy\\_frame\\_data\\_to\\_ext\\_addr\\_async](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- *Non blocking version of [ezdp\\_copy\\_frame\\_data\\_to\\_ext\\_addr\(\)](#).* static `__always_inline` uint32\_t [ezdp\\_copy\\_frame\\_data\\_from\\_ext\\_addr](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- *Copy data from an extended address to a frame buffer.* static `__always_inline` void [ezdp\\_copy\\_frame\\_data\\_from\\_ext\\_addr\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- *Non blocking version of [ezdp\\_copy\\_frame\\_data\\_from\\_ext\\_addr\\_async\(\)](#).* static `__always_inline` uint32\_t [ezdp\\_load\\_frame\\_data](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)

- Copy data from a frame buffer to CMEM. static \_\_always\_inline void [ezdp\\_load\\_frame\\_data\\_async](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_load\\_frame\\_data\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_store\\_frame\\_data](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Copy data from CMEM to a frame buffer. static \_\_always\_inline void [ezdp\\_store\\_frame\\_data\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_store\\_frame\\_data\(\)](#). static \_\_always\_inline void [ezdp\\_copy\\_frame\\_lbd](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- Copy LBD data between two frame buffers. static \_\_always\_inline void [ezdp\\_copy\\_frame\\_lbd\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_frame\\_lbd\(\)](#). static \_\_always\_inline void [ezdp\\_clone\\_frame\\_lbd](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t bd\_offset, uint32\_t size, uint32\_t flags)
- Copy LBD data between two frame buffers, with the same source and destination offset (optimized). static \_\_always\_inline void [ezdp\\_clone\\_frame\\_lbd\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t bd\_offset, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_clone\\_frame\\_lbd\(\)](#). static \_\_always\_inline void [ezdp\\_copy\\_frame\\_lbd\\_to\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- Copy LBD data from a frame buffer to an extended address. static \_\_always\_inline void [ezdp\\_copy\\_frame\\_lbd\\_to\\_ext\\_addr\\_async](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_frame\\_lbd\\_to\\_ext\\_addr\(\)](#). static \_\_always\_inline void [ezdp\\_copy\\_frame\\_lbd\\_from\\_ext\\_addr](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Copy LBD data from an extended address to a frame buffer. static \_\_always\_inline void [ezdp\\_copy\\_frame\\_lbd\\_from\\_ext\\_addr\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_frame\\_lbd\\_from\\_ext\\_addr\(\)](#). static \_\_always\_inline void [ezdp\\_load\\_frame\\_lbd](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- Copy LBD data from a frame buffer to CMEM. static \_\_always\_inline void [ezdp\\_load\\_frame\\_lbd\\_async](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_load\\_frame\\_lbd\(\)](#). static \_\_always\_inline void [ezdp\\_store\\_frame\\_lbd](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Copy LBD data from CMEM to a frame buffer. static \_\_always\_inline void [ezdp\\_store\\_frame\\_lbd\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Non blocking version of [ezdp\\_store\\_frame\\_lbd](#). static \_\_always\_inline [ezdp\\_buffer\\_desc\\_t ezdp\\_alloc\\_mc\\_buf](#) (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type, uint32\_t buf\_budget\_id, uint16\_t ref\_counter)
- Allocate a single frame buffer and set its multicast reference counter. static \_\_always\_inline void [ezdp\\_free\\_mc\\_buf](#) (uint32\_t buf\_budget\_id, [ezdp\\_buffer\\_desc\\_t](#) bd)
- Free a multicast frame buffer. static \_\_always\_inline void [ezdp\\_write\\_mc\\_buf\\_counter](#) ([ezdp\\_buffer\\_desc\\_t](#) bd, uint16\_t value)
- Set a frame buffer's multicast reference counter. static \_\_always\_inline void [ezdp\\_write\\_mc\\_buf\\_counter\\_async](#) ([ezdp\\_buffer\\_desc\\_t](#) bd, uint16\_t value)
- Non blocking version of [ezdp\\_write\\_mc\\_buf\\_counter\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_read\\_mc\\_buf\\_counter](#) ([ezdp\\_buffer\\_desc\\_t](#) bd)
- Get a frame buffer's multicast reference counter. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_inc\\_mc\\_buf\\_counter](#) ([ezdp\\_buffer\\_desc\\_t](#) bd)
- Atomically read and increment a frame buffer's multicast reference counter. static \_\_always\_inline uint32\_t [ezdp\\_atomic\\_read\\_and\\_dec\\_mc\\_buf\\_counter](#) ([ezdp\\_buffer\\_desc\\_t](#) bd)

- *Atomically read and conditionally decrement a frame buffer's multicast reference counter.* static `__always_inline uint32_t ezdp\_calc\_frame\_data\_checksum` (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*bd\_ptr, uint32\_t bd\_offset, uint32\_t size, uint32\_t flags)
- *Calculate checksum of frame data buffer.* static `__always_inline uint32_t ezdp\_buf\_data\_len` (uint32\_t header\_offset, uint32\_t free\_bytes, uint32\_t max\_length)
- *Calculate the length of the buffer based on header offset and free bytes.* static `__always_inline uint32_t ezdp\_lbd\_len` (uint32\_t data\_buf\_count)
- *Calculate LBD buffer length according to BD count.* static `__always_inline uint32_t ezdp\_calc\_header\_offset` (uint32\_t min\_header\_offset, bool prefer\_128B\_offset, uint32\_t buffer\_length)
- *Calculate optimized frame header offset.* static `__always_inline void ezdp\_inc\_tm\_imem\_buf\_ctr` (uint32\_t value)
- *Increment TM IMEM buffer counter.* static `__always_inline void ezdp\_inc\_tm\_imem\_buf\_ctr\_async` (uint32\_t value)
- *Non blocking version of [\\_ezdp\\_inc\\_tm\\_imem\\_buf\\_ctr](#).* static `__always_inline void ezdp\_dec\_tm\_imem\_buf\_ctr` (uint32\_t value)
- *Decrement TM IMEM buffer counter.* static `__always_inline void ezdp\_dec\_tm\_imem\_buf\_ctr\_async` (uint32\_t value)
- *Non blocking version of [\\_ezdp\\_dec\\_tm\\_imem\\_buf\\_ctr](#).* static `__always_inline uint32_t ezdp\_read\_tm\_imem\_buf\_ctr` (void)
- *Number of IMEM buffers used by frames currently being processed by TM.* static `__always_inline int32_t ezdp\_get\_first\_buf` (struct [ezdp\\_frame\\_desc](#) \*fd, struct [ezdp\\_buffer\\_desc](#) \*buf\_desc, struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) \_\_cmem \*lbd\_line, [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#) \*iter\_st)
- *Gets first buffer from frame.* static `__always_inline int32_t ezdp\_get\_next\_buf` (struct [ezdp\\_frame\\_desc](#) \*fd, struct [ezdp\\_buffer\\_desc](#) \*buf\_desc, struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) \_\_cmem \*lbd\_line, [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#) \*iter\_st)
- *Get next buffer from frame.* static `__always_inline bool ezdp\_init\_frame` (struct [ezdp\\_frame\\_desc](#) \_\_cmem \*frame\_desc, enum [ezdp\\_frame\\_type](#) frame\_type, [ezdp\\_buffer\\_desc\\_t](#) buf\_desc, uint32\_t buf\_len, struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) \_\_cmem \*lbd\_line, uint32\_t header\_offset, uint32\_t budget\_id, [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#) \*iter\_st)
- *Create a new frame by setting its frame descriptor params and init frame iterator.* static `__always_inline void ezdp\_append\_buf` (struct [ezdp\\_frame\\_desc](#) \_\_cmem \*frame\_desc, [ezdp\\_buffer\\_desc\\_t](#) buf\_desc, uint32\_t buf\_len, struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) \_\_cmem \*lbd\_line, [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#) \*iter\_st)
- *Add newly allocated (by user) buffer to frame pointed by iterator.* static `__always_inline void ezdp\_sync\_frame` (struct [ezdp\\_frame\\_desc](#) \_\_cmem \*frame\_desc, struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) \_\_cmem \*lbd\_line, [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#) \*iter\_st, bool force)

Store last LBD line to memory, if required.

## Function Documentation

static `__always_inline ezdp\_buffer\_desc\_t ezdp\_alloc\_buf` (enum [ezdp\\_buffer\\_mem\\_type](#) pool\_type, uint32\_t buf\_budget\_id) [static]

Allocate a single frame buffer.

The returned buffer descriptor's type is initially set to be a data buffer. This can later be changed by the caller to match the actual intended use.

### Parameters:

- [in] *pool\_type* - pool type to allocate buffer from
- [in] *buf\_budget\_id* - budget id for accounting

### Returns:

[ezdp\\_buffer\\_desc\\_t](#) - use [ezdp\\_buf\\_alloc\\_failed](#) API to check if allocation success or fail

static `__always_inline void ezdp\_free\_buf` (uint32\_t buf\_budget\_id, [ezdp\\_buffer\\_desc\\_t](#) bd\_raw\_data) [static]

Free a single frame buffer.



**Parameters:**

[in] *buf\_budget\_id* - budget id to charge  
 [in] *bd\_raw\_data* - BD raw data

**Returns:**

none

```
static __always_inline void ezdp_free_buf_async (uint32_t buf_budget_id, ezdp\_buffer\_desc\_t
bd_raw_data) [static]
```

Non blocking version of [ezdp\\_free\\_buf\(\)](#).

**Parameters:**

[in] *buf\_budget\_id* - budget id to charge  
 [in] *bd\_raw\_data* - BD raw data

**Returns:**

none

```
static __always_inline ezdp\_buffer\_desc\_t ezdp_alloc_multi_buf (enum ezdp\_buffer\_mem\_type
pool_type, uint32_t buf_budget_id, uint32_t num_of_bufs, struct ezdp\_buffer\_desc
__cmem * bds_ptr) [static]
```

Allocate multiple frame buffers.

Allocate up to 8 frame buffers from the same pool. The allocated BDs are written to the CMEM. In addition, the first allocated BD is returned. The operation either succeeds to allocate all requested resources or fails without allocating any resources.

**Parameters:**

[in] *pool\_type* - pool type to allocate buffer from (IMEM or EMEM)  
 [in] *num\_of\_bufs* - number of buffers to allocate (1-8)  
 [in] *buf\_budget\_id* - budget id for accounting  
 [out] *bds\_ptr* - pointer to CMEM to write response to

**Returns:**

First allocated [ezdp\\_buffer\\_desc](#) - use `ezdp_buf_alloc_failed` API to check if allocation success or fail

```
static __always_inline void ezdp_alloc_multi_buf_async (enum ezdp\_buffer\_mem\_type pool_type,
uint32_t buf_budget_id, uint32_t num_of_bufs, struct ezdp\_buffer\_desc __cmem * bds_ptr)
[static]
```

Non blocking version of [ezdp\\_alloc\\_multi\\_buf\(\)](#).

**Parameters:**

[in] *pool\_type* - pool type to allocate buffer from (IMEM or EMEM pool)  
 [in] *num\_of\_bufs* - number of buffer to allocate (1-8)  
 [in] *buf\_budget\_id* - budget id for accounting  
 [out] *bds\_ptr* - pointer to CMEM to write response to

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the allocated BDs were written to CMEM.

**Returns:**

none use `ezdp_buf_alloc_failed` API on first allocated buffer to check if allocation success or fail



```
static __always_inline bool ezdp_buf_alloc_failed (ezdp\_buffer\_desc\_t ret) [static]
```

Check if allocation of the buffer failed.

**Parameters:**

[in] *ret* - return value from buffer allocation API

**Returns:**

bool - true if allocation failed

```
static __always_inline void ezdp_free_multi_buf (uint32_t buf_budget_id,   uint32_t num_of_bufs,
struct ezdp\_buffer\_desc __cmem * bds_ptr) [static]
```

Free multiple frame buffers.

Free up to 8 frame buffers, not necessarily from the same pool. The BDs to free are passed in CMEM. NULL BDs are ignored.

**Parameters:**

[in] *num\_of\_bufs* - number of buffers to free (1-8)

[in] *buf\_budget\_id* - budget id to charge

[in] *bds\_ptr* - pointer to BDs in CMEM to recycle

**Returns:**

none

```
static __always_inline void ezdp_free_multi_buf_async (uint32_t buf_budget_id,   uint32_t
num_of_bufs,   struct ezdp\_buffer\_desc __cmem * bds_ptr) [static]
```

Non blocking version of [ezdp\\_free\\_multi\\_buf\(\)](#).

**Parameters:**

[in] *num\_of\_bufs* - number of buffers to recycle (1-8)

[in] *buf\_budget\_id* - budget id to charge

[in] *bds\_ptr* - pointer to BDs in CMEM to recycle

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the BDs were read from CMEM and the request was sent to the BMU.

**Returns:**

none

```
static __always_inline uint32_t ezdp_read_free_buf (enum ezdp\_buffer\_mem\_type pool_type)
[static]
```

The number of buffers available to be obtained.

**Parameters:**

[in] *pool\_type* - buffer pool type

**Returns:**

uint32\_t - number of available/free frame buffers

```
static __always_inline void ezdp_rebudget_buf (enum ezdp\_buffer\_mem\_type pool_type,
uint32_t free_buf_budget_id,  uint32_t free_num_of_bufs,  uint32_t alloc_buf_budget_id,
uint32_t alloc_num_of_bufs) [static]
```

Update the budget to which buffers are credited.

Adds free\_num\_of\_bufs credits back into the budget selected by free\_buf\_budget\_id and subtracts alloc\_num\_of\_bufs credits from the budget selected by alloc\_buf\_budget\_id. The budget is maintained separately for IMEM and EMEM buffers.

**Parameters:**

- [in] *pool\_type* - pool type (IMEM or EMEM pool)
- [in] *free\_buf\_budget\_id* - budget ID to reimburse
- [in] *free\_num\_of\_bufs* - number of buffers to reimburse. Decrement operation (can be zero)
- [in] *alloc\_buf\_budget\_id* - budget ID to charge
- [in] *alloc\_num\_of\_bufs* - number of buffers to charge. Increment operation (can be zero)

**Returns:**

none

```
static __always_inline void ezdp_rebudget_buf_async (enum ezdp\_buffer\_mem\_type pool_type,
uint32_t free_buf_budget_id,  uint32_t free_num_of_bufs,  uint32_t alloc_buf_budget_id,
uint32_t alloc_num_of_bufs) [static]
```

Non blocking version of [ezdp\\_rebudget\\_buf\(\)](#).

**Parameters:**

- [in] *pool\_type* - pool type (IMEM or EMEM pool)
- [in] *free\_buf\_budget\_id* - budget ID to reimburse
- [in] *free\_num\_of\_bufs* - number of buffers to reimburse. Decrement operation (can be zero)
- [in] *alloc\_buf\_budget\_id* - budget ID to charge
- [in] *alloc\_num\_of\_bufs* - number of buffers to charge. Increment operation (can be zero)

**Returns:**

none

```
static __always_inline uint32_t ezdp_copy_frame_data (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  uint32_t dst_bd_offset,  struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t
src_bd_offset,  uint32_t size,  uint32_t flags) [static]
```

Copy data between two frame buffers.

**Parameters:**

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the destination buffer
- [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the source buffer
- [in] *src\_bd\_offset* - offset in the source buffer to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

uint32\_t - 16 bit checksum value

```
static __always_inline void ezdp_copy_frame_data_async (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  uint32_t dst_bd_offset,  struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t
src_bd_offset,  uint32_t size,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_frame\\_data\(\)](#).

#### Parameters:

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the destination buffer
- [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the source buffer
- [in] *src\_bd\_offset* - offset in the source buffer to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

#### Note:

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

#### Returns:

none

```
static __always_inline uint32_t ezdp_clone_frame_data (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t bd_offset,  uint32_t
size,  uint32_t flags) [static]
```

Copy data between two frame buffers, with the same source and destination offset (optimized).

#### Parameters:

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the destination buffer
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the source buffer
- [in] *bd\_offset* - offset in both the source and destination buffers to copy from/to
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

#### Returns:

uin32\_t - 16 bit checksum value

```
static __always_inline void ezdp_clone_frame_data_async (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t bd_offset,  uint32_t
size,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_clone\\_frame\\_data\(\)](#).

#### Parameters:

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the destination buffer
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM with 4 byte alignment) describing the source buffer
- [in] *bd\_offset* - offset in both the source and destination buffers to copy from/to
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

#### Note:

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_copy_frame_data_to_ext_addr (struct ezdp\_ext\_addr
__cmem * dst_ptr, struct ezdp\_buffer\_desc __cmem * src_bd_ptr, uint32_t src_bd_offset,
uint32_t size, uint32_t flags) [static]
```

Copy data from a frame buffer to an extended address.

**Parameters:**

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
[in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer  
[in] *src\_bd\_offset* - offset in the source buffer to copy data from  
[in] *size* - number of bytes to copy  
[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Note:**

returned 16 bit checksum is applicable only when memory type is external or memory type is internal and msid > 5

**Returns:**

uint32\_t - 16 bit checksum value

```
static __always_inline void ezdp_copy_frame_data_to_ext_addr_async (struct ezdp\_ext\_addr
__cmem * dst_ptr, struct ezdp\_buffer\_desc __cmem * src_bd_ptr, uint32_t src_bd_offset,
uint32_t size, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_frame\\_data\\_to\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
[in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer  
[in] *src\_bd\_offset* - offset in the source buffer to copy data from  
[in] *size* - number of bytes to copy  
[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_copy_frame_data_from_ext_addr (struct ezdp\_buffer\_desc
__cmem * dst_bd_ptr, uint32_t dst_bd_offset, struct ezdp\_ext\_addr __cmem * src_ptr,
uint32_t size, uint32_t flags) [static]
```

Copy data from an extended address to a frame buffer.

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer  
[in] *dst\_bd\_offset* - offset in the destination buffer to copy data to  
[in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)  
[in] *size* - number of bytes to copy

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

uint32\_t - 16 bit checksum value

```
static __always_inline void ezdp_copy_frame_data_from_ext_addr_async (struct
ezdp_buffer_desc __cmem * dst_bd_ptr,  uint32_t dst_bd_offset,  struct ezdp_ext_addr
__cmem * src_ptr,  uint32_t size,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_frame\\_data\\_from\\_ext\\_addr\\_async\(\)](#).

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer  
 [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to  
 [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)  
 [in] *size* - number of bytes to copy  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_load_frame_data (void __cmem * dst_ptr,  struct
ezdp_buffer_desc __cmem * src_bd_ptr,  uint32_t src_bd_offset,  uint32_t size,  uint32_t
flags) [static]
```

Copy data from a frame buffer to CMEM.

**Parameters:**

[out] *dst\_ptr* - pointer in CMEM to copy data to  
 [in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer  
 [in] *src\_bd\_offset* - offset in the source buffer to copy data from  
 [in] *size* - number of bytes to copy  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Returns:**

uint32\_t - first 4 bytes of the copied data, which are zero padded if the requested data length is smaller than 4

```
static __always_inline void ezdp_load_frame_data_async (void __cmem * dst_ptr,  struct
ezdp_buffer_desc __cmem * src_bd_ptr,  uint32_t src_bd_offset,  uint32_t size,  uint32_t
flags) [static]
```

Non blocking version of [ezdp\\_load\\_frame\\_data\(\)](#).

**Parameters:**

[out] *dst\_ptr* - pointer in CMEM to copy data to  
 [in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer  
 [in] *src\_bd\_offset* - offset in the source buffer to copy data from  
 [in] *size* - number of bytes to copy  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_store_frame_data (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  uint32_t dst_bd_offset,  void __cmem * src_ptr,  uint32_t size,  uint32_t flags)
[static]
```

Copy data from CMEM to a frame buffer.

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer  
[in] *dst\_bd\_offset* - offset in the destination buffer to copy data to  
[in] *src\_ptr* - pointer in CMEM to copy data from  
[in] *size* - number of bytes to copy  
[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

uin32\_t - 16 bit checksum value

```
static __always_inline void ezdp_store_frame_data_async (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  uint32_t dst_bd_offset,  void __cmem * src_ptr,  uint32_t size,  uint32_t flags)
[static]
```

Non blocking version of [ezdp\\_store\\_frame\\_data\(\)](#).

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer  
[in] *dst\_bd\_offset* - offset in the destination buffer to copy data to  
[in] *src\_ptr* - pointer in CMEM to copy data from  
[in] *size* - number of bytes to copy  
[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_copy_frame_lbd (struct ezdp\_buffer\_desc __cmem * dst_bd_ptr,
uint32_t dst_bd_offset,  struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t
src_bd_offset,  uint32_t size,  uint32_t flags) [static]
```

Copy LBD data between two frame buffers.

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer  
[in] *dst\_bd\_offset* - offset in the destination buffer to copy data to  
[in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer  
[in] *src\_bd\_offset* - offset in the source buffer to copy data from

[in] *size* - number of bytes to copy

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Returns:**

none

```
static __always_inline void ezdp_copy_frame_lbd_async (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  uint32_t dst_bd_offset,  struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t
src_bd_offset,  uint32_t size,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_frame\\_lbd\(\)](#).

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer

[in] *dst\_bd\_offset* - offset in the destination buffer to copy data to

[in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer

[in] *src\_bd\_offset* - offset in the source buffer to copy data from

[in] *size* - number of bytes to copy

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_clone_frame_lbd (struct ezdp\_buffer\_desc __cmem * dst_bd_ptr,
struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t bd_offset,  uint32_t size,  uint32_t
flags) [static]
```

Copy LBD data between two frame buffers, with the same source and destination offset (optimized).

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer

[in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer

[in] *bd\_offset* - offset in both the source and destination buffers to copy from/to

[in] *size* - number of bytes to copy

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Returns:**

none

```
static __always_inline void ezdp_clone_frame_lbd_async (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr,  struct ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t bd_offset,  uint32_t
size,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_clone\\_frame\\_lbd\(\)](#).

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer

[in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer

[in] *bd\_offset* - offset in both the source and destination buffers to copy from/to

[in] *size* - number of bytes to copy

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_copy_frame_lbd_to_ext_addr (struct ezdp\_ext\_addr __cmem *
dst_ptr, struct ezdp\_buffer\_desc __cmem * src_bd_ptr, uint32_t src_bd_offset, uint32_t
size, uint32_t flags) [static]
```

Copy LBD data from a frame buffer to an extended address.

**Parameters:**

- [in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer
- [in] *src\_bd\_offset* - offset in the source buffer to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Returns:**

none

```
static __always_inline void ezdp_copy_frame_lbd_to_ext_addr_async (struct ezdp\_ext\_addr
__cmem * dst_ptr, struct ezdp\_buffer\_desc __cmem * src_bd_ptr, uint32_t src_bd_offset,
uint32_t size, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_frame\\_lbd\\_to\\_ext\\_addr\(\)](#).

**Parameters:**

- [in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer
- [in] *src\_bd\_offset* - offset in the source buffer to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_copy_frame_lbd_from_ext_addr (struct ezdp\_buffer\_desc
__cmem * dst_bd_ptr, uint32_t dst_bd_offset, struct ezdp\_ext\_addr __cmem * src_ptr,
uint32_t size, uint32_t flags) [static]
```

Copy LBD data from an extended address to a frame buffer.

**Parameters:**

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer
- [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to
- [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO



**Returns:**

none

```
static __always_inline void ezdp_copy_frame_lbd_from_ext_addr_async (struct ezdp\_buffer\_desc
__cmem * dst_bd_ptr,  uint32_t dst_bd_offset,  struct ezdp\_ext\_addr __cmem * src_ptr,
uint32_t size,  uint32_t flags) [static]
```

Non blocking version of [\\_ezdp\\_copy\\_frame\\_lbd\\_from\\_ext\\_addr\(\)](#).

**Parameters:**

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer
- [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to
- [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_load_frame_lbd (void __cmem * dst_ptr,  struct
ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t src_bd_offset,  uint32_t size,  uint32_t
flags) [static]
```

Copy LBD data from a frame buffer to CMEM.

**Parameters:**

- [out] *dst\_ptr* - pointer in CMEM to copy data to
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer
- [in] *src\_bd\_offset* - offset in the source buffer to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Returns:**

none

```
static __always_inline void ezdp_load_frame_lbd_async (void __cmem * dst_ptr,  struct
ezdp\_buffer\_desc __cmem * src_bd_ptr,  uint32_t src_bd_offset,  uint32_t size,  uint32_t
flags) [static]
```

Non blocking version of [ezdp\\_load\\_frame\\_lbd\(\)](#).

**Parameters:**

- [out] *dst\_ptr* - pointer in CMEM to copy data to
- [in] *src\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the source buffer
- [in] *src\_bd\_offset* - offset in the source buffer to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_store_frame_lbd (struct ezdp\_buffer\_desc __cmem * dst_bd_ptr,
uint32_t dst_bd_offset, void __cmem * src_ptr, uint32_t size, uint32_t flags) [static]
```

Copy LBD data from CMEM to a frame buffer.

**Parameters:**

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer
- [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to
- [in] *src\_ptr* - pointer in CMEM to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Returns:**

none

```
static __always_inline void ezdp_store_frame_lbd_async (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr, uint32_t dst_bd_offset, void __cmem * src_ptr, uint32_t size, uint32_t flags)
[static]
```

Non blocking version of ezdp\_store\_frame\_lbd.

**Parameters:**

- [in] *dst\_bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the destination buffer
- [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to
- [in] *src\_ptr* - pointer in CMEM to copy data from
- [in] *size* - number of bytes to copy
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline ezdp\_buffer\_desc\_t ezdp_alloc_mc_buf (enum ezdp\_buffer\_mem\_type
pool_type, uint32_t buf_budget_id, uint16_t ref_counter) [static]
```

Allocate a single frame buffer and set its multicast reference counter.

Used for allocation of frame buffers for multicast frames. The buffer is allocated and its multicast reference counter is set to required ref\_counter value

**Parameters:**

- [in] *pool\_type* - pool type to allocate buffer from
- [in] *buf\_budget\_id* - budget id for accounting
- [in] *ref\_counter* - reference counter to set for this buffer

**Returns:**

[ezdp\\_buffer\\_desc\\_t](#) - use [ezdp\\_buf\\_alloc\\_failed](#) API to check if allocation success or fail

```
static __always_inline void ezdp_free_mc_buf (uint32_t buf_budget_id, ezdp\_buffer\_desc\_t bd)
[static]
```

Free a multicast frame buffer.

Used for recycling of frame buffers belonging to multicast frames. The buffer's multicast reference counter is decremented and the buffer is recycled if this was the last reference to the buffer.

**Parameters:**

[in] *buf\_budget\_id* - budget id to charge  
 [in] *bd* - BD raw data

**Returns:**

none

```
static __always_inline void ezdp_write_mc_buf_counter (ezdp\_buffer\_desc\_t bd,  uint16_t
value) [static]
```

Set a frame buffer's multicast reference counter.

**Parameters:**

[in] *bd* - BD describing the buffer  
 [in] *value* - reference count value to set

**Returns:**

void

```
static __always_inline void ezdp_write_mc_buf_counter_async (ezdp\_buffer\_desc\_t bd,
uint16_t value) [static]
```

Non blocking version of [ezdp\\_write\\_mc\\_buf\\_counter\(\)](#).

**Parameters:**

[in] *bd* - BD describing the buffer  
 [in] *value* - reference count value to set

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the value was set.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_mc_buf_counter (ezdp\_buffer\_desc\_t bd) [static]
```

Get a frame buffer's multicast reference counter.

**Parameters:**

[in] *bd* - BD describing the buffer

**Returns:**

uint32\_t - the 16 LSB are the current counter value

```
static __always_inline uint32_t ezdp_atomic_read_and_inc_mc_buf_counter (ezdp\_buffer\_desc\_t
bd) [static]
```

Atomically read and increment a frame buffer's multicast reference counter.

**Parameters:**

[in] *bd* - BD describing the buffer

**Returns:**

uint32\_t - the 16 LSB are the value of the counter before the increment

**static \_\_always\_inline uint32\_t ezdp\_atomic\_read\_and\_dec\_mc\_buf\_counter ([ezdp\\_buffer\\_desc\\_t bd](#)) [static]**

Atomically read and conditionally decrement a frame buffer's multicast reference counter.

**Parameters:**

[in] *bd* - BD describing the buffer

**Returns:**

uint32\_t - the 16 LSB are the value of the counter before the increment A value of 1 indicates the last reference has been released A value of 0 indicates an error (reference counter was already at 0)

**static \_\_always\_inline uint32\_t ezdp\_calc\_frame\_data\_checksum (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \* *bd\_ptr*, uint32\_t *bd\_offset*, uint32\_t *size*, uint32\_t *flags*) [static]**

Calculate checksum of frame data buffer.

**Parameters:**

[in] *bd\_ptr* - pointer to BD (in CMEM in 4 byte alignment) describing the buffer to calculate checksum on

[in] *bd\_offset* - offset in the source buffer to calculate checksum from

[in] *size* - number of bytes to sum

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following: TODO

**Note:**

1. The checksum calculation assumes an even (2 byte aligned) pointer. If the pointer is odd, the checksum result should be swapped.
2. Applicable only for data buffer.

**Returns:**

uint32\_t - 16 bit checksum value

**static \_\_always\_inline uint32\_t ezdp\_buf\_data\_len (uint32\_t *header\_offset*, uint32\_t *free\_bytes*, uint32\_t *max\_length*) [static]**

Calculate the length of the buffer based on header offset and free bytes.

**Parameters:**

[in] *header\_offset* - buffer header offset (can be taken from [ezdp\\_frame\\_desc](#)) - max size is 255 bytes

[in] *free\_bytes* - buffer free bytes (can be taken from [ezdp\\_frame\\_desc](#)) - max size is 255 bytes

[in] *max\_length* - max length of buffer - max value 256 bytes

**Returns:**

buffer length - max 256 bytes

**static \_\_always\_inline uint32\_t ezdp\_lbd\_len (uint32\_t *data\_buf\_count*) [static]**

Calculate LBD buffer length according to BD count.

**Parameters:**

[in] *data\_buf\_count* - BD count from FD

**Returns:**

LBD buffer length in bytes

```
static __always_inline uint32_t ezdp_calc_header_offset (uint32_t min_header_offset,    bool  
prefer_128B_offset,    uint32_t buffer_length) [static]
```

Calculate optimized frame header offset.

**Parameters:**

[in] *min\_header\_offset* - min header offset (Input: 0, 16, 32, 48, ... 240 Bytes)

[in] *prefer\_128B\_offset* - Prefer header offset bigger than 128 byte

[in] *buffer\_length* - buffer length 1-256

**Returns:**

Calculated optimized header offset

```
static __always_inline void ezdp_inc_tm_imem_buf_ctr (uint32_t value) [static]
```

Increment TM IMEM buffer counter.

**Parameters:**

[in] *value* - value to increment

**Returns:**

none

```
static __always_inline void ezdp_inc_tm_imem_buf_ctr_async (uint32_t value) [static]
```

Non blocking version of `_ezdp_inc_tm_imem_buf_ctr`.

**Parameters:**

[in] *value* - value to increment

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete.

**Returns:**

none

```
static __always_inline void ezdp_dec_tm_imem_buf_ctr (uint32_t value) [static]
```

Decrement TM IMEM buffer counter.

**Parameters:**

[in] *value* - value to decrement

**Returns:**

none

```
static __always_inline void ezdp_dec_tm_imem_buf_ctr_async (uint32_t value) [static]
```

Non blocking version of `_ezdp_dec_tm_imem_buf_ctr`.

**Parameters:**

[in] *value* - value to decrement

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete.

**Returns:**

none

**static \_\_always\_inline uint32\_t ezdp\_read\_tm\_imem\_buf\_ctr (void) [static]**

Number of IMEM buffers used by frames currently being processed by TM.

**Returns:**

int32\_t - counter value

**Note:**

The return value can be negative due to delayed updates of the counter from the cpus

**static \_\_always\_inline int32\_t ezdp\_get\_first\_buf (struct [ezdp\\_frame\\_desc](#) \* fd, struct [ezdp\\_buffer\\_desc](#) \* buf\_desc, struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) \_\_cmem \* lbd\_line, [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#) \* iter\_st) [static]**

Gets first buffer from frame.

**Parameters:**

[in] *fd* - Pointer to frame descriptor

[out] *buf\_desc* - Pointer to buffer descriptor

[out] *lbd\_line* - Pointer in CMEM for LBD line

[out] *iter\_st* - Pointer to frame buffer iterator state

**Note:**

: LBD line is loaded only for extended frame type

**Returns:**

(-1) - no buffer other - length of data in buffer descriptor

**static \_\_always\_inline int32\_t ezdp\_get\_next\_buf (struct [ezdp\\_frame\\_desc](#) \* fd, struct [ezdp\\_buffer\\_desc](#) \* buf\_desc, struct [ezdp\\_linked\\_buffers\\_desc\\_line](#) \_\_cmem \* lbd\_line, [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#) \* iter\_st) [static]**

Get next buffer from frame.

**Parameters:**

[in] *fd* - Pointer to frame descriptor

[out] *buf\_desc* - Pointer to buffer descriptor

[in,out] *lbd\_line* - Pointer in CMEM for last LBD line

[in,out] *iter\_st* - Pointer to frame buffer iterator state

**Note:**

LBD line is re-loaded if needed you must call `ezdp_get_first_buf` before calling this function

**Returns:**

(-1) - no more buffers other - length of data in buffer descriptor

```
static __always_inline bool ezdp_init_frame (struct ezdp\_frame\_desc __cmem * frame_desc,
enum ezdp\_frame\_type frame_type, ezdp\_buffer\_desc\_t buf_desc, uint32_t buf_len, struct
ezdp\_linked\_buffers\_desc\_line __cmem * lbd_line, uint32_t header_offset, uint32_t
budget_id, ezdp\_frame\_buf\_iterator\_state\_t * iter_st) [static]
```

Create a new frame by setting its frame descriptor params and init frame iterator.

**Parameters:**

[in,out] *frame\_desc* - Pointer to frame descriptor  
[in] *frame\_type* - Frame type: standard or extended  
[in] *buf\_desc* - New buffer descriptor that will be added to frame  
[in] *buf\_len* - Buffer descriptor length (max length is 256 bytes)  
[in,out] *lbd\_line* - Pointer in CMEM for last LBD line  
[in] *header\_offset* - Header offset in 1st buffer data.  
[in] *budget\_id* - Buffer data budget id  
[in,out] *iter\_st* - Pointer to frame buffer iterator state

**Note:**

: Not all frame descriptor fields are initialized in this function. The following fields are the user's responsibility: class of service logical\_id multicast\_control job\_budget\_id transmit\_confirmation\_flag timestamp\_flag transmit\_keep\_buf\_flag gross\_checksum\_flag efa\_control

**Returns:**

true - success creating new frame false - fail in the process of creating new frame

```
static __always_inline void ezdp_append_buf (struct ezdp\_frame\_desc __cmem * frame_desc,
ezdp\_buffer\_desc\_t buf_desc, uint32_t buf_len, struct ezdp\_linked\_buffers\_desc\_line
__cmem * lbd_line, ezdp\_frame\_buf\_iterator\_state\_t * iter_st) [static]
```

Add newly allocated (by user) buffer to frame pointed by iterator.

**Parameters:**

[in,out] *frame\_desc* - Pointer to frame descriptor  
[in] *buf\_desc* - New buffer descriptor that will be added to frame  
[in] *buf\_len* - Buffer descriptor length (max length is 256 bytes)  
[in,out] *lbd\_line* - Pointer in CMEM for last LBD line  
[in,out] *iter\_st* - Pointer to frame buffer iterator state

**Note:**

Call `ezdp_init_frame` before using this function. Assumes that iterator points to the last buffer in the frame.

**Returns:**

void

```
static __always_inline void ezdp_sync_frame (struct ezdp\_frame\_desc __cmem * frame_desc,
struct ezdp\_linked\_buffers\_desc\_line __cmem * lbd_line, ezdp\_frame\_buf\_iterator\_state\_t *
iter_st, bool force) [static]
```

Store last LBD line to memory, if required.

**Parameters:**

[in] *frame\_desc* - Pointer to frame descriptor  
[in] *lbd\_line* - Pointer in CMEM for last LBD line  
[in] *iter\_st* - Pointer to frame buffer iterator state  
[in] *force* - Force store of the LBD line

**Returns:**

void

## dpe/dp/include/ezdp\_frame\_defs.h File Reference

### Data Structures

- struct [ezdp\\_buffer\\_desc](#)
- Buffer descriptor (BD) data structure. struct [ezdp\\_frame\\_desc](#)
- Frame descriptor data structure. struct [ezdp\\_2step\\_1588\\_header](#)
- 2-step 1588 header format definition struct [ezdp\\_1step\\_1588\\_header](#)
- 1-step 1588 header format definition struct [ezdp\\_buffer\\_info](#)
- Buffer descriptor info. struct [ezdp\\_1588\\_header](#)
- 1588 header format definition struct [ezdp\\_linked\\_buffers\\_desc\\_line](#)
- LBD Line data structure. struct [ezdp\\_linked\\_buffers\\_desc](#)
- A generic linked buffers descriptor. struct [ezdp\\_small\\_linked\\_buffers\\_desc](#)
- Small linked buffers descriptor. May hold up to 3 buffs. struct [ezdp\\_large\\_linked\\_buffers\\_desc](#)
- Large linked buffers descriptor. struct [ezdp\\_ext\\_linked\\_buffers\\_desc](#)

### Extended linked buffers descriptor. Defines

- #define [EZDP\\_BUFFER\\_DATA\\_SIZE](#) 256
- #define [EZDP\\_MEM\\_FRAME\\_DATA\\_BUFFER\\_SIZE](#) EZDP\_BUFFER\_DATA\_SIZE
- #define [EZDP\\_BUFFER\\_DESC\\_ID\\_SIZE](#) 28
- #define [EZDP\\_BUFFER\\_DESC\\_ID\\_OFFSET](#) 0
- #define [EZDP\\_BUFFER\\_DESC\\_RESERVED28\\_29\\_SIZE](#) 2
- #define [EZDP\\_BUFFER\\_DESC\\_RESERVED28\\_29\\_OFFSET](#) 28
- #define [EZDP\\_BUFFER\\_DESC\\_MEM\\_TYPE\\_SIZE](#) 1
- #define [EZDP\\_BUFFER\\_DESC\\_MEM\\_TYPE\\_OFFSET](#) 30
- #define [EZDP\\_BUFFER\\_DESC\\_MEM\\_TYPE\\_MASK](#) (1 << EZDP\_BUFFER\_DESC\_MEM\_TYPE\_OFFSET)
- #define [EZDP\\_BUFFER\\_DESC\\_VALID\\_DATA\\_BUF\\_SIZE](#) 1
- #define [EZDP\\_BUFFER\\_DESC\\_VALID\\_DATA\\_BUF\\_OFFSET](#) 31
- #define [EZDP\\_BUFFER\\_DESC\\_VALID\\_DATA\\_BUF\\_MASK](#) (1 << EZDP\_BUFFER\_DESC\_VALID\_DATA\_BUF\_OFFSET)
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_BUDGET\\_ID\\_SIZE](#) 10
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_BUDGET\\_ID\\_OFFSET](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_BUDGET\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_BUDGET\\_ID\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED10\\_11\\_SIZE](#) 2
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED10\\_11\\_OFFSET](#) 10
- #define [EZDP\\_FRAME\\_DESC\\_CLASS\\_OF\\_SERVICE\\_SIZE](#) 2
- #define [EZDP\\_FRAME\\_DESC\\_CLASS\\_OF\\_SERVICE\\_OFFSET](#) 12
- #define [EZDP\\_FRAME\\_DESC\\_CLASS\\_OF\\_SERVICE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_CLASS\\_OF\\_SERVICE\\_WORD\\_OFFSET](#) 12
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED14\\_15\\_SIZE](#) 2
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED14\\_15\\_OFFSET](#) 14
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED0\\_1\\_SIZE](#) 2
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED0\\_1\\_OFFSET](#) 16
- #define [EZDP\\_FRAME\\_DESC\\_GROSS\\_CHECKSUM\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_FRAME\\_DESC\\_GROSS\\_CHECKSUM\\_FLAG\\_OFFSET](#) 18
- #define [EZDP\\_FRAME\\_DESC\\_GROSS\\_CHECKSUM\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_GROSS\\_CHECKSUM\\_FLAG\\_WORD\\_OFFSET](#) 18
- #define [EZDP\\_FRAME\\_DESC\\_GROSS\\_CHECKSUM\\_FLAG\\_MASK](#) (1 << EZDP\_FRAME\_DESC\_GROSS\_CHECKSUM\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_KEEP\\_BUF\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_KEEP\\_BUF\\_FLAG\\_OFFSET](#) 19
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_KEEP\\_BUF\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_KEEP\\_BUF\\_FLAG\\_WORD\\_OFFSET](#) 19



- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_KEEP\\_BUF\\_FLAG\\_MASK](#) (1 << EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUF\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_FRAME\\_DESC\\_TIMESTAMP\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_FRAME\\_DESC\\_TIMESTAMP\\_FLAG\\_OFFSET](#) 20
- #define [EZDP\\_FRAME\\_DESC\\_TIMESTAMP\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_TIMESTAMP\\_FLAG\\_WORD\\_OFFSET](#) 20
- #define [EZDP\\_FRAME\\_DESC\\_TIMESTAMP\\_FLAG\\_MASK](#) (1 << EZDP\_FRAME\_DESC\_TIMESTAMP\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_CONFIRMATION\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_CONFIRMATION\\_FLAG\\_OFFSET](#) 21
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_CONFIRMATION\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_CONFIRMATION\\_FLAG\\_WORD\\_OFFSET](#) 21
- #define [EZDP\\_FRAME\\_DESC\\_TRANSMIT\\_CONFIRMATION\\_FLAG\\_MASK](#) (1 << EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATION\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_FRAME\\_DESC\\_TYPE\\_SIZE](#) 2
- #define [EZDP\\_FRAME\\_DESC\\_TYPE\\_OFFSET](#) 22
- #define [EZDP\\_FRAME\\_DESC\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_TYPE\\_WORD\\_OFFSET](#) 22
- #define [EZDP\\_FRAME\\_DESC\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_FRAME\\_DESC\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_FRAME\\_DESC\\_ECC\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_ECC\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_FRAME\\_DESC\\_HEADER\\_OFFSET\\_SIZE](#) 8
- #define [EZDP\\_FRAME\\_DESC\\_HEADER\\_OFFSET\\_OFFSET](#) 32
- #define [EZDP\\_FRAME\\_DESC\\_HEADER\\_OFFSET\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_FRAME\\_DESC\\_HEADER\\_OFFSET\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_DATA\\_BUF\\_COUNT\\_SIZE](#) 8
- #define [EZDP\\_FRAME\\_DESC\\_DATA\\_BUF\\_COUNT\\_OFFSET](#) 40
- #define [EZDP\\_FRAME\\_DESC\\_DATA\\_BUF\\_COUNT\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_FRAME\\_DESC\\_DATA\\_BUF\\_COUNT\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_FRAME\\_DESC\\_FRAME\\_LENGTH\\_SIZE](#) 16
- #define [EZDP\\_FRAME\\_DESC\\_FRAME\\_LENGTH\\_OFFSET](#) 48
- #define [EZDP\\_FRAME\\_DESC\\_FRAME\\_LENGTH\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_FRAME\\_DESC\\_FRAME\\_LENGTH\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_DESC\\_SIZE](#) 32
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_DESC\\_OFFSET](#) 64
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_DESC\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_FRAME\\_DESC\\_BUF\\_DESC\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_JOB\\_BUDGET\\_ID\\_SIZE](#) 10
- #define [EZDP\\_FRAME\\_DESC\\_JOB\\_BUDGET\\_ID\\_OFFSET](#) 96
- #define [EZDP\\_FRAME\\_DESC\\_JOB\\_BUDGET\\_ID\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_FRAME\\_DESC\\_JOB\\_BUDGET\\_ID\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED106\\_110\\_SIZE](#) 4
- #define [EZDP\\_FRAME\\_DESC\\_RESERVED106\\_110\\_OFFSET](#) 106
- #define [EZDP\\_FRAME\\_DESC\\_MULTICAST\\_CONTROL\\_SIZE](#) 2
- #define [EZDP\\_FRAME\\_DESC\\_MULTICAST\\_CONTROL\\_OFFSET](#) 110
- #define [EZDP\\_FRAME\\_DESC\\_MULTICAST\\_CONTROL\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_FRAME\\_DESC\\_MULTICAST\\_CONTROL\\_WORD\\_OFFSET](#) 14
- #define [EZDP\\_FRAME\\_DESC\\_LOGICAL\\_ID\\_SIZE](#) 8
- #define [EZDP\\_FRAME\\_DESC\\_LOGICAL\\_ID\\_OFFSET](#) 112
- #define [EZDP\\_FRAME\\_DESC\\_LOGICAL\\_ID\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_FRAME\\_DESC\\_LOGICAL\\_ID\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_FRAME\\_DESC\\_FREE\\_BYTES\\_SIZE](#) 8
- #define [EZDP\\_FRAME\\_DESC\\_FREE\\_BYTES\\_OFFSET](#) 120
- #define [EZDP\\_FRAME\\_DESC\\_FREE\\_BYTES\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_FRAME\\_DESC\\_FREE\\_BYTES\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_FRAME\\_DESC\\_WORD\\_COUNT](#) 4

- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED0\\_23\\_SIZE](#) 24
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED0\\_23\\_OFFSET](#) 0
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED24\\_SIZE](#) 1
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED24\\_OFFSET](#) 24
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED24\\_31\\_SIZE](#) 7
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED24\\_31\\_OFFSET](#) 25
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED32\\_63\\_SIZE](#) 32
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED32\\_63\\_OFFSET](#) 32
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_BUDGET\\_ID\\_SIZE](#) 10
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_BUDGET\\_ID\\_OFFSET](#) 64
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_BUDGET\\_ID\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_BUDGET\\_ID\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED74\\_75\\_SIZE](#) 2
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED74\\_75\\_OFFSET](#) 74
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_CLASS\\_OF\\_SERVICE\\_SIZE](#) 2
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_CLASS\\_OF\\_SERVICE\\_OFFSET](#) 76
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_CLASS\\_OF\\_SERVICE\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_CLASS\\_OF\\_SERVICE\\_WORD\\_OFFSET](#) 12
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED76\\_77\\_SIZE](#) 2
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_RESERVED76\\_77\\_OFFSET](#) 78
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_HEADER\\_OFFSET\\_SIZE](#) 8
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_HEADER\\_OFFSET\\_OFFSET](#) 80
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_HEADER\\_OFFSET\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_HEADER\\_OFFSET\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_FREE\\_BYTES\\_SIZE](#) 8
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_FREE\\_BYTES\\_OFFSET](#) 88
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_FREE\\_BYTES\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_FREE\\_BYTES\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_DESC\\_SIZE](#) 32
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_DESC\\_OFFSET](#) 96
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_DESC\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_BUF\\_DESC\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_2STEP\\_1588\\_HEADER\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_SIZE](#) 16
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_OFFSET](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_RESERVED16\\_23\\_SIZE](#) 8
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_RESERVED16\\_23\\_OFFSET](#) 16
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_RESERVED24\\_SIZE](#) 1
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_RESERVED24\\_OFFSET](#) 24
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_INJECT\\_CHECKSUM\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_INJECT\\_CHECKSUM\\_FLAG\\_OFFSET](#) 25
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_INJECT\\_CHECKSUM\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_INJECT\\_CHECKSUM\\_FLAG\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_INJECT\\_CHECKSUM\\_FLAG\\_MASK](#) (1 << EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_WRAP\\_AROUND\\_CONDITION\\_SIZE](#) 1
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_WRAP\\_AROUND\\_CONDITION\\_OFFSET](#) 26
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_WRAP\\_AROUND\\_CONDITION\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_WRAP\\_AROUND\\_CONDITION\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_WRAP\\_AROUND\\_CONDITION\\_MASK](#) (1 << EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_WORD\_OFFSET)
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_ODD\\_START\\_SIZE](#) 1
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_ODD\\_START\\_OFFSET](#) 27
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_ODD\\_START\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_ODD\\_START\\_WORD\\_OFFSET](#) 27

- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_ODD\\_START\\_MASK](#) (1 << EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_ODD\_START\_WORD\_OFFSET)
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_RESERVED28\\_31\\_SIZE](#) 4
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_RESERVED28\\_31\\_OFFSET](#) 28
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_OFFSET\\_SIZE](#) 16
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_OFFSET\\_OFFSET](#) 32
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_OFFSET\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_OFFSET\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_OFFSET\\_SIZE](#) 16
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_OFFSET\\_OFFSET](#) 48
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_OFFSET\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CHECKSUM\\_OFFSET\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_SIZE](#) 64
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_OFFSET](#) 64
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_CORRECTION\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_1STEP\\_1588\\_HEADER\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_LINKED\\_BUFFER\\_DESC\\_LINE\\_NUMBER\\_OF\\_BUFFERS\\_DESC](#) 3

## Typedefs

- typedef uint32\_t [ezdp\\_buffer\\_desc\\_t](#)
- typedef struct ezdp\_frame\_buf\_iterator\_state [ezdp\\_frame\\_buf\\_iterator\\_state\\_t](#)
- typedef struct ezdp\_extract\_frame\_tail\_working\_area [ezdp\\_extract\\_frame\\_tail\\_working\\_area\\_t](#)
- typedef struct ezdp\_convert\_std2ext\_working\_area [ezdp\\_convert\\_std2ext\\_working\\_area\\_t](#)
- typedef struct ezdp\_concat\_frames\_working\_area [ezdp\\_concat\\_frames\\_working\\_area\\_t](#)
- typedef struct ezdp\_trim\_frame\_head\_working\_area [ezdp\\_trim\\_frame\\_head\\_working\\_area\\_t](#)

## Enumerations

- enum [ezdp\\_buffer\\_mem\\_type](#) { [EZDP\\_EXT\\_MEM](#) = 0x0, [EZDP\\_INT\\_MEM](#) = 0x1 }  
*buffer mem type possible values.*
- enum [ezdp\\_frame\\_type](#) { [EZDP\\_NULL\\_FRAME](#) = 0x0, [EZDP\\_EXT\\_FRAME](#) = 0x1, [EZDP\\_STD\\_FRAME](#) = 0x2 }  
*frame type possible values.*
- enum [ezdp\\_multicast\\_control](#) { [EZDP\\_UNICAST](#) = 0x0, [EZDP\\_MULTICAST](#) = 0x1, [EZDP\\_BROADCAST](#) = 0x2, [EZDP\\_REPLICA](#) = 0x3 }  
*multicast control possible values.*
- enum [ezdp\\_1588\\_type](#) { [EZDP\\_2STEP](#) = 0, [EZDP\\_1STEP](#) = 1 }  
*job container command type possible values.*
- enum [ezdp\\_linked\\_buffers\\_desc\\_size](#) { [EZDP\\_SMALL\\_LBD](#) = 1, [EZDP\\_LARGE\\_LBD](#) = 2, [EZDP\\_EXTENDED\\_LBD](#) = 16 }  
*LBD definition.*

## Define Documentation

```
#define EZDP_BUFFER_DATA_SIZE 256

#define EZDP_MEM_FRAME_DATA_BUFFER_SIZE EZDP_BUFFER_DATA_SIZE

#define EZDP_BUFFER_DESC_ID_SIZE 28

#define EZDP_BUFFER_DESC_ID_OFFSET 0

#define EZDP_BUFFER_DESC_RESERVED28_29_SIZE 2

#define EZDP_BUFFER_DESC_RESERVED28_29_OFFSET 28

#define EZDP_BUFFER_DESC_MEM_TYPE_SIZE 1

#define EZDP_BUFFER_DESC_MEM_TYPE_OFFSET 30

#define EZDP_BUFFER_DESC_MEM_TYPE_MASK (1 <<
EZDP_BUFFER_DESC_MEM_TYPE_OFFSET)

#define EZDP_BUFFER_DESC_VALID_DATA_BUF_SIZE 1

#define EZDP_BUFFER_DESC_VALID_DATA_BUF_OFFSET 31

#define EZDP_BUFFER_DESC_VALID_DATA_BUF_MASK (1 <<
EZDP_BUFFER_DESC_VALID_DATA_BUF_OFFSET)

#define EZDP_FRAME_DESC_BUF_BUDGET_ID_SIZE 10

#define EZDP_FRAME_DESC_BUF_BUDGET_ID_OFFSET 0

#define EZDP_FRAME_DESC_BUF_BUDGET_ID_WORD_SELECT 0

#define EZDP_FRAME_DESC_BUF_BUDGET_ID_WORD_OFFSET 0

#define EZDP_FRAME_DESC_RESERVED10_11_SIZE 2

#define EZDP_FRAME_DESC_RESERVED10_11_OFFSET 10

#define EZDP_FRAME_DESC_CLASS_OF_SERVICE_SIZE 2

#define EZDP_FRAME_DESC_CLASS_OF_SERVICE_OFFSET 12

#define EZDP_FRAME_DESC_CLASS_OF_SERVICE_WORD_SELECT 0

#define EZDP_FRAME_DESC_CLASS_OF_SERVICE_WORD_OFFSET 12

#define EZDP_FRAME_DESC_RESERVED14_15_SIZE 2

#define EZDP_FRAME_DESC_RESERVED14_15_OFFSET 14

#define EZDP_FRAME_DESC_RESERVED0_1_SIZE 2
```

```
#define EZDP_FRAME_DESC_RESERVED0_1_OFFSET 16

#define EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_SIZE 1

#define EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_OFFSET 18

#define EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_WORD_SELECT 0

#define EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_WORD_OFFSET 18

#define EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_MASK (1 <<
EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_WORD_OFFSET)

#define EZDP_FRAME_DESC_TRANSMIT_KEEP_BUF_FLAG_SIZE 1

#define EZDP_FRAME_DESC_TRANSMIT_KEEP_BUF_FLAG_OFFSET 19

#define EZDP_FRAME_DESC_TRANSMIT_KEEP_BUF_FLAG_WORD_SELECT 0

#define EZDP_FRAME_DESC_TRANSMIT_KEEP_BUF_FLAG_WORD_OFFSET 19

#define EZDP_FRAME_DESC_TRANSMIT_KEEP_BUF_FLAG_MASK (1 <<
EZDP_FRAME_DESC_TRANSMIT_KEEP_BUF_FLAG_WORD_OFFSET)

#define EZDP_FRAME_DESC_TIMESTAMP_FLAG_SIZE 1

#define EZDP_FRAME_DESC_TIMESTAMP_FLAG_OFFSET 20

#define EZDP_FRAME_DESC_TIMESTAMP_FLAG_WORD_SELECT 0

#define EZDP_FRAME_DESC_TIMESTAMP_FLAG_WORD_OFFSET 20

#define EZDP_FRAME_DESC_TIMESTAMP_FLAG_MASK (1 <<
EZDP_FRAME_DESC_TIMESTAMP_FLAG_WORD_OFFSET)

#define EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_SIZE 1

#define EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_OFFSET 21

#define EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_WORD_SELECT 0

#define EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_WORD_OFFSET 21

#define EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_MASK (1 <<
EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_WORD_OFFSET)

#define EZDP_FRAME_DESC_TYPE_SIZE 2

#define EZDP_FRAME_DESC_TYPE_OFFSET 22

#define EZDP_FRAME_DESC_TYPE_WORD_SELECT 0

#define EZDP_FRAME_DESC_TYPE_WORD_OFFSET 22
```

```
#define EZDP_FRAME_DESC_ECC_SIZE 8

#define EZDP_FRAME_DESC_ECC_OFFSET 24

#define EZDP_FRAME_DESC_ECC_WORD_SELECT 0

#define EZDP_FRAME_DESC_ECC_WORD_OFFSET 24

#define EZDP_FRAME_DESC_HEADER_OFFSET_SIZE 8

#define EZDP_FRAME_DESC_HEADER_OFFSET_OFFSET 32

#define EZDP_FRAME_DESC_HEADER_OFFSET_WORD_SELECT 1

#define EZDP_FRAME_DESC_HEADER_OFFSET_WORD_OFFSET 0

#define EZDP_FRAME_DESC_DATA_BUF_COUNT_SIZE 8

#define EZDP_FRAME_DESC_DATA_BUF_COUNT_OFFSET 40

#define EZDP_FRAME_DESC_DATA_BUF_COUNT_WORD_SELECT 1

#define EZDP_FRAME_DESC_DATA_BUF_COUNT_WORD_OFFSET 8

#define EZDP_FRAME_DESC_FRAME_LENGTH_SIZE 16

#define EZDP_FRAME_DESC_FRAME_LENGTH_OFFSET 48

#define EZDP_FRAME_DESC_FRAME_LENGTH_WORD_SELECT 1

#define EZDP_FRAME_DESC_FRAME_LENGTH_WORD_OFFSET 16

#define EZDP_FRAME_DESC_BUF_DESC_SIZE 32

#define EZDP_FRAME_DESC_BUF_DESC_OFFSET 64

#define EZDP_FRAME_DESC_BUF_DESC_WORD_SELECT 2

#define EZDP_FRAME_DESC_BUF_DESC_WORD_OFFSET 0

#define EZDP_FRAME_DESC_JOB_BUDGET_ID_SIZE 10

#define EZDP_FRAME_DESC_JOB_BUDGET_ID_OFFSET 96

#define EZDP_FRAME_DESC_JOB_BUDGET_ID_WORD_SELECT 3

#define EZDP_FRAME_DESC_JOB_BUDGET_ID_WORD_OFFSET 0

#define EZDP_FRAME_DESC_RESERVED106_110_SIZE 4

#define EZDP_FRAME_DESC_RESERVED106_110_OFFSET 106
```

```
#define EZDP_FRAME_DESC_MULTICAST_CONTROL_SIZE 2

#define EZDP_FRAME_DESC_MULTICAST_CONTROL_OFFSET 110

#define EZDP_FRAME_DESC_MULTICAST_CONTROL_WORD_SELECT 3

#define EZDP_FRAME_DESC_MULTICAST_CONTROL_WORD_OFFSET 14

#define EZDP_FRAME_DESC_LOGICAL_ID_SIZE 8

#define EZDP_FRAME_DESC_LOGICAL_ID_OFFSET 112

#define EZDP_FRAME_DESC_LOGICAL_ID_WORD_SELECT 3

#define EZDP_FRAME_DESC_LOGICAL_ID_WORD_OFFSET 16

#define EZDP_FRAME_DESC_FREE_BYTES_SIZE 8

#define EZDP_FRAME_DESC_FREE_BYTES_OFFSET 120

#define EZDP_FRAME_DESC_FREE_BYTES_WORD_SELECT 3

#define EZDP_FRAME_DESC_FREE_BYTES_WORD_OFFSET 24

#define EZDP_FRAME_DESC_WORD_COUNT 4

#define EZDP_2STEP_1588_HEADER_RESERVED0_23_SIZE 24

#define EZDP_2STEP_1588_HEADER_RESERVED0_23_OFFSET 0

#define EZDP_2STEP_1588_HEADER_RESERVED24_SIZE 1

#define EZDP_2STEP_1588_HEADER_RESERVED24_OFFSET 24

#define EZDP_2STEP_1588_HEADER_RESERVED24_31_SIZE 7

#define EZDP_2STEP_1588_HEADER_RESERVED24_31_OFFSET 25

#define EZDP_2STEP_1588_HEADER_RESERVED32_63_SIZE 32

#define EZDP_2STEP_1588_HEADER_RESERVED32_63_OFFSET 32

#define EZDP_2STEP_1588_HEADER_BUF_BUDGET_ID_SIZE 10

#define EZDP_2STEP_1588_HEADER_BUF_BUDGET_ID_OFFSET 64

#define EZDP_2STEP_1588_HEADER_BUF_BUDGET_ID_WORD_SELECT 2

#define EZDP_2STEP_1588_HEADER_BUF_BUDGET_ID_WORD_OFFSET 0

#define EZDP_2STEP_1588_HEADER_RESERVED74_75_SIZE 2
```

```
#define EZDP_2STEP_1588_HEADER_RESERVED74_75_OFFSET 74

#define EZDP_2STEP_1588_HEADER_CLASS_OF_SERVICE_SIZE 2

#define EZDP_2STEP_1588_HEADER_CLASS_OF_SERVICE_OFFSET 76

#define EZDP_2STEP_1588_HEADER_CLASS_OF_SERVICE_WORD_SELECT 2

#define EZDP_2STEP_1588_HEADER_CLASS_OF_SERVICE_WORD_OFFSET 12

#define EZDP_2STEP_1588_HEADER_RESERVED76_77_SIZE 2

#define EZDP_2STEP_1588_HEADER_RESERVED76_77_OFFSET 78

#define EZDP_2STEP_1588_HEADER_HEADER_OFFSET_SIZE 8

#define EZDP_2STEP_1588_HEADER_HEADER_OFFSET_OFFSET 80

#define EZDP_2STEP_1588_HEADER_HEADER_OFFSET_WORD_SELECT 2

#define EZDP_2STEP_1588_HEADER_HEADER_OFFSET_WORD_OFFSET 16

#define EZDP_2STEP_1588_HEADER_FREE_BYTES_SIZE 8

#define EZDP_2STEP_1588_HEADER_FREE_BYTES_OFFSET 88

#define EZDP_2STEP_1588_HEADER_FREE_BYTES_WORD_SELECT 2

#define EZDP_2STEP_1588_HEADER_FREE_BYTES_WORD_OFFSET 24

#define EZDP_2STEP_1588_HEADER_BUF_DESC_SIZE 32

#define EZDP_2STEP_1588_HEADER_BUF_DESC_OFFSET 96

#define EZDP_2STEP_1588_HEADER_BUF_DESC_WORD_SELECT 3

#define EZDP_2STEP_1588_HEADER_BUF_DESC_WORD_OFFSET 0

#define EZDP_2STEP_1588_HEADER_WORD_COUNT 4

#define EZDP_1STEP_1588_HEADER_CHECKSUM_SIZE 16

#define EZDP_1STEP_1588_HEADER_CHECKSUM_OFFSET 0

#define EZDP_1STEP_1588_HEADER_CHECKSUM_WORD_SELECT 0

#define EZDP_1STEP_1588_HEADER_CHECKSUM_WORD_OFFSET 0

#define EZDP_1STEP_1588_HEADER_RESERVED16_23_SIZE 8

#define EZDP_1STEP_1588_HEADER_RESERVED16_23_OFFSET 16
```



```
#define EZDP_1STEP_1588_HEADER_RESERVED24_SIZE 1

#define EZDP_1STEP_1588_HEADER_RESERVED24_OFFSET 24

#define EZDP_1STEP_1588_HEADER_INJECT_CHECKSUM_FLAG_SIZE 1

#define EZDP_1STEP_1588_HEADER_INJECT_CHECKSUM_FLAG_OFFSET 25

#define EZDP_1STEP_1588_HEADER_INJECT_CHECKSUM_FLAG_WORD_SELECT 0

#define EZDP_1STEP_1588_HEADER_INJECT_CHECKSUM_FLAG_WORD_OFFSET 25

#define EZDP_1STEP_1588_HEADER_INJECT_CHECKSUM_FLAG_MASK (1 <<
EZDP_1STEP_1588_HEADER_INJECT_CHECKSUM_FLAG_WORD_OFFSET)

#define EZDP_1STEP_1588_HEADER_WRAP_AROUND_CONDITION_SIZE 1

#define EZDP_1STEP_1588_HEADER_WRAP_AROUND_CONDITION_OFFSET 26

#define EZDP_1STEP_1588_HEADER_WRAP_AROUND_CONDITION_WORD_SELECT 0

#define EZDP_1STEP_1588_HEADER_WRAP_AROUND_CONDITION_WORD_OFFSET 26

#define EZDP_1STEP_1588_HEADER_WRAP_AROUND_CONDITION_MASK (1 <<
EZDP_1STEP_1588_HEADER_WRAP_AROUND_CONDITION_WORD_OFFSET)

#define EZDP_1STEP_1588_HEADER_CORRECTION_ODD_START_SIZE 1

#define EZDP_1STEP_1588_HEADER_CORRECTION_ODD_START_OFFSET 27

#define EZDP_1STEP_1588_HEADER_CORRECTION_ODD_START_WORD_SELECT 0

#define EZDP_1STEP_1588_HEADER_CORRECTION_ODD_START_WORD_OFFSET 27

#define EZDP_1STEP_1588_HEADER_CORRECTION_ODD_START_MASK (1 <<
EZDP_1STEP_1588_HEADER_CORRECTION_ODD_START_WORD_OFFSET)

#define EZDP_1STEP_1588_HEADER_RESERVED28_31_SIZE 4

#define EZDP_1STEP_1588_HEADER_RESERVED28_31_OFFSET 28

#define EZDP_1STEP_1588_HEADER_CORRECTION_OFFSET_SIZE 16

#define EZDP_1STEP_1588_HEADER_CORRECTION_OFFSET_OFFSET 32

#define EZDP_1STEP_1588_HEADER_CORRECTION_OFFSET_WORD_SELECT 1

#define EZDP_1STEP_1588_HEADER_CORRECTION_OFFSET_WORD_OFFSET 0

#define EZDP_1STEP_1588_HEADER_CHECKSUM_OFFSET_SIZE 16

#define EZDP_1STEP_1588_HEADER_CHECKSUM_OFFSET_OFFSET 48
```

```

#define EZDP_1STEP_1588_HEADER_CHECKSUM_OFFSET_WORD_SELECT 1

#define EZDP_1STEP_1588_HEADER_CHECKSUM_OFFSET_WORD_OFFSET 16

#define EZDP_1STEP_1588_HEADER_CORRECTION_SIZE 64

#define EZDP_1STEP_1588_HEADER_CORRECTION_OFFSET 64

#define EZDP_1STEP_1588_HEADER_CORRECTION_WORD_SELECT 2

#define EZDP_1STEP_1588_HEADER_CORRECTION_WORD_OFFSET 0

#define EZDP_1STEP_1588_HEADER_WORD_COUNT 4

#define EZDP_LINKED_BUFFER_DESC_LINE_NUMBER_OF_BUFFERS_DESC 3

```

---

## Typedef Documentation

```

typedef uint32_t ezdp\_buffer\_desc\_t

typedef struct ezdp_frame_buf_iterator_state ezdp\_frame\_buf\_iterator\_state\_t

typedef struct ezdp_extract_frame_tail_working_area ezdp\_extract\_frame\_tail\_working\_area\_t

typedef struct ezdp_convert_std2ext_working_area ezdp\_convert\_std2ext\_working\_area\_t

typedef struct ezdp_concat_frames_working_area ezdp\_concat\_frames\_working\_area\_t

typedef struct ezdp_trim_frame_head_working_area ezdp\_trim\_frame\_head\_working\_area\_t

```

---

## Enumeration Type Documentation

enum [ezdp\\_buffer\\_mem\\_type](#)

buffer mem type possible values.

### Enumerator:

***EZDP\_EXT\_MEM*** Data is located in EMEM.

***EZDP\_INT\_MEM*** Data is located in IMEM.

enum [ezdp\\_frame\\_type](#)

frame type possible values.

### Enumerator:

***EZDP\_NULL\_FRAME*** Null frame.

Frame has no allocated buffers. All other ezdp\_frame fields are don't-care.

***EZDP\_EXT\_FRAME*** Multi-buffer frame with separate link buffer descriptor.

All frame data buffers (BDs) are listed in the link buffer descriptor when header buffer descriptor points to the link buffer descriptor.

***EZDP\_STD\_FRAME*** Multi-buffer frame data with embedded link buffer descriptor.

The buffer pointer has two meanings, differentiated through API usage. In data buffer API it points to the first frame data buffer (which starts after header\_offset bytes), while in the LBD buffer API it points to the LBD entries at the beginning of the buffer. The size of the embedded link buffer descriptor (length) may be empty (data\_buf\_count = 1), 16B (data\_buf\_count greater than 1 but less than 4) or 32B (data\_buf\_count greater than 4 but less than 7). The maximum length of the embedded LBD is limited to 32B.

#### enum [ezdp\\_multicast\\_control](#)

multicast control possible values.

##### Enumerator:

***EZDP\_UNICAST*** The frame is unicast.

Buffer multicast reference counter value is not applicable.

***EZDP\_MULTICAST*** The first buffer and LBD are not multicasted, the rest of the buffers are multicasted.

***EZDP\_BROADCAST*** All buffer are multicasted (Buffer multicast reference counter values are applicable to all buffers).

***EZDP\_REPLICA*** On receive path, the value indicates that the ezdp\_frame.

replication\_num field is valid and holds the current replication number. On transmit path to loopback port, this value is used as replication command. On transmit path to external port or in case of drop, this value is similar to UNICAST functionality.

#### enum [ezdp\\_1588\\_type](#)

job container command type possible values.

##### Enumerator:

***EZDP\_2STEP*** 2-step 1588 header.

***EZDP\_1STEP*** 1-step 1588 header.

#### enum [ezdp\\_linked\\_buffers\\_desc\\_size](#)

LBD definition.

##### Enumerator:

***EZDP\_SMALL\_LBD*** Small LBD may contain up to 3 buffs which are organized in 1 LBD line.

Applicable for EZDP\_STD\_FRAME frame type.

**EZDP\_LARGE\_LBD** Small LBD may contain up to 6 buffs which are organized in 2 LBD line  
Applicable for EZDP\_STD\_FRAME frame type.

**EZDP\_EXTENDED\_LBD** Extended LBD may contain up to 48 buffs which are organized in 16 LBD lines  
Applicable for EZDP\_EXT\_FRAME frame type.

## dpe/dp/include/ezdp\_job.h File Reference

### Defines

- #define [ezdp\\_notifier](#)(NAME) void ezdp\_notice\_handler(void) \_\_attribute\_\_((alias ( #NAME )))

### Register notifier function. Typedefs

- typedef void(\* [ezdp\\_notifier\\_t](#))(void)

### IPC notifier handle function. Functions

- static \_\_always\_inline [ezdp\\_job\\_id\\_t ezdp\\_alloc\\_job\\_id](#) (uint32\_t budget\_id)
- Allocate a new job from the PMU. static \_\_always\_inline void [ezdp\\_alloc\\_job\\_id\\_async](#) ([ezdp\\_job\\_id\\_t](#) \*jobh\_ptr, uint32\_t budget\_id)
- Non blocking version of [ezdp\\_alloc\\_job\\_id\(\)](#). static \_\_always\_inline [ezdp\\_job\\_id\\_t ezdp\\_alloc\\_multi\\_job\\_id](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobhs\_ptr, uint32\_t budget\_id, uint32\_t num\_of\_jobs)
- Allocate multiple new jobs from the PMU. static \_\_always\_inline void [ezdp\\_alloc\\_multi\\_job\\_id\\_async](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobhs\_ptr, uint32\_t budget\_id, uint32\_t num\_of\_jobs)
- Non blocking version of [ezdp\\_alloc\\_multi\\_job\\_id\(\)](#). static \_\_always\_inline bool [ezdp\\_job\\_alloc\\_failed](#) ([ezdp\\_job\\_id\\_t](#) ret)
- Check if allocation of the job failed. static \_\_always\_inline void [ezdp\\_free\\_job\\_id](#) ([ezdp\\_job\\_id\\_t](#) \* \_\_cmem jobh\_ptr, uint32\_t budget\_id)
- Recycle a job to the PMU. static \_\_always\_inline void [ezdp\\_free\\_job\\_id\\_async](#) ([ezdp\\_job\\_id\\_t](#) \* \_\_cmem jobh\_ptr, uint32\_t budget\_id)
- Non blocking version of [ezdp\\_free\\_job\\_id\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_read\\_free\\_job](#) (void)
- The number of jobs available to be obtained. static \_\_always\_inline void [ezdp\\_rebudget\\_job](#) (uint32\_t free\_budget\_id, uint32\_t free\_num\_of\_jobs, uint32\_t alloc\_budget\_id, uint32\_t alloc\_num\_of\_jobs)
- Update the budget to which jobs are credited. static \_\_always\_inline void [ezdp\\_rebudget\\_job\\_async](#) (uint32\_t free\_budget\_id, uint32\_t free\_num\_of\_jobs, uint32\_t alloc\_budget\_id, uint32\_t alloc\_num\_of\_jobs)
- Non blocking version of [ezdp\\_rebudget\\_job\(\)](#). static \_\_always\_inline void [ezdp\\_load\\_job](#) ([ezdp\\_job\\_id\\_t](#) \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr)
- Copy the job descriptor from IMEM to CMEM. static \_\_always\_inline void [ezdp\\_load\\_job\\_async](#) ([ezdp\\_job\\_id\\_t](#) \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr)
- Non blocking version of [ezdp\\_load\\_job\(\)](#). static \_\_always\_inline void [ezdp\\_store\\_job](#) ([ezdp\\_job\\_id\\_t](#) \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr)
- Copy the job descriptor from CMEM to IMEM. static \_\_always\_inline void [ezdp\\_store\\_job\\_async](#) ([ezdp\\_job\\_id\\_t](#) \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr)
- Non blocking version of [ezdp\\_store\\_job\(\)](#). static \_\_always\_inline void [ezdp\\_store\\_job\\_container](#) ([ezdp\\_job\\_id\\_t](#) \*jobh\_ptr, struct [ezdp\\_job\\_container\\_desc](#) \_\_cmem \*jd\_ptr)
- Copy the job container descriptor from CMEM to IMEM. static \_\_always\_inline void [ezdp\\_store\\_job\\_container\\_async](#) ([ezdp\\_job\\_id\\_t](#) \*jobh\_ptr, struct [ezdp\\_job\\_container\\_desc](#) \_\_cmem \*jd\_ptr)
- Non blocking version of [ezdp\\_store\\_job\\_container\(\)](#). static \_\_always\_inline void [ezdp\\_request\\_job\\_id](#) ([ezdp\\_job\\_id\\_t](#) \* \_\_cmem jobh\_ptr)
- Request a new job from the PMU. static \_\_always\_inline void [ezdp\\_wait\\_for\\_job\\_id](#) (void)
- Suspend execution until a job request completes. static \_\_always\_inline void [ezdp\\_cancel\\_job\\_request](#) (void)
- Cancel a job request from the PMU. static \_\_always\_inline void [ezdp\\_receive\\_job](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr, uint32\_t flags)
- Request a new job from the PMU and load it to CMEM. static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_to\\_queue](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, uint32\_t side, uint32\_t target\_queue, uint32\_t flags)
- Dispatch the job to another PMU queue. static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_to\\_queue\\_async](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, uint32\_t side, uint32\_t target\_queue, uint32\_t flags)
- Non blocking version of [ezdp\\_send\\_job\\_id\\_to\\_queue\(\)](#). static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_to\\_tm](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, uint32\_t side, uint32\_t flags)
- Transmit the job via the TM. static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_to\\_tm\\_async](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, uint32\_t side, uint32\_t flags)

- Non blocking version of [ezdp\\_send\\_job\\_id\\_to\\_tm\(\)](#). static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_to\\_interface](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, uint32\_t side, uint32\_t output\_channel, uint32\_t flags)
- Transmit the job directly to an output queue channel, bypassing the TM. static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_to\\_interface\\_async](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, uint32\_t side, uint32\_t output\_channel, uint32\_t flags)
- Non blocking version of [ezdp\\_send\\_job\\_id\\_to\\_interface\(\)](#). static \_\_always\_inline void [ezdp\\_update\\_job\\_id\\_queue](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, uint32\_t side, uint32\_t target\_queue, uint32\_t flags)
- Move the job to another PMU queue without dispatching it. static \_\_always\_inline void [ezdp\\_send\\_job\\_to\\_queue](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr, uint32\_t side, uint32\_t target\_queue, uint32\_t flags)
- Store the job descriptor and dispatch the job to another queue in the PMU. static \_\_always\_inline void [ezdp\\_send\\_job\\_to\\_tm](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr, uint32\_t side, uint32\_t flags)
- Store the job descriptor and transmit the job via the TM. static \_\_always\_inline void [ezdp\\_send\\_job\\_to\\_interface](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr, uint32\_t side, uint32\_t output\_channel, uint32\_t flags)
- Store the job descriptor and transmit the job directly to interface by bypassing the TM. static \_\_always\_inline void [ezdp\\_update\\_job\\_queue](#) ([ezdp\\_job\\_id\\_t](#) \* \_\_cmem jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr, uint32\_t side, uint32\_t target\_queue, uint32\_t flags)
- Store the job descriptor and move the job to another PMU queue without dispatching it. static \_\_always\_inline void [ezdp\\_discard\\_job\\_id](#) ([ezdp\\_job\\_id\\_t](#) \* \_\_cmem jobh\_ptr)
- Discard a job and all its associated frame resources. static \_\_always\_inline void [ezdp\\_discard\\_job\\_id\\_async](#) ([ezdp\\_job\\_id\\_t](#) \* \_\_cmem jobh\_ptr)
- Non blocking version of [ezdp\\_discard\\_job\\_id\(\)](#). static \_\_always\_inline void [ezdp\\_discard\\_job](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, struct [ezdp\\_job\\_desc](#) \_\_cmem \*jd\_ptr)
- Store the job descriptor and discard a job and all its associated frame resources. static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_container](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr)
- Send request to PMU to distribute the job container. static \_\_always\_inline void [ezdp\\_send\\_job\\_id\\_container\\_async](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr)
- Non blocking version of [ezdp\\_send\\_job\\_id\\_container\(\)](#). static \_\_always\_inline void [ezdp\\_send\\_job\\_container](#) ([ezdp\\_job\\_id\\_t](#) \_\_cmem \*jobh\_ptr, struct [ezdp\\_job\\_container\\_desc](#) \_\_cmem \*jd\_container\_ptr)
- Store the job container and send request to PMU to distribute it. static \_\_always\_inline uint32\_t [ezdp\\_container\\_job\\_count](#) ([ezdp\\_job\\_container\\_info\\_t](#) info)
- Return the number of the jobs that are in the job container. static \_\_always\_inline [ezdp\\_job\\_container\\_info\\_t](#) [ezdp\\_container\\_info](#) (uint32\_t job\_count)
- Set the info according to the number of jobs in the job container. static \_\_always\_inline void [ezdp\\_notify\\_cpu](#) (uint32\_t target\_cpu\_id)
- Notify target CPU. static \_\_always\_inline bool [ezdp\\_notice\\_pending](#) (void)
- Check if there is a new notice. static \_\_always\_inline void [ezdp\\_clear\\_notice](#) (void)
- Clear notice indication. static \_\_always\_inline void [ezdp\\_wait\\_for\\_notice](#) (void)
- Suspend execution until receiving new notification for CPU. static \_\_always\_inline bool [ezdp\\_check\\_notice](#) (void)
- Check if there is a new notice and clear new notice indication. static \_\_always\_inline void [ezdp\\_wait\\_for\\_event](#) (void)
- Suspend execution until job request completed or new notice received. static \_\_always\_inline void [ezdp\\_handle\\_notice](#) (void)
- Handle notice. static \_\_always\_inline [ezdp\\_congestion\\_status\\_t](#) [ezdp\\_read\\_congestion\\_status](#) (enum [ezdp\\_flow\\_control\\_node](#) node\_type, uint32\_t node\_id)
- Read priority drop congestion status. static \_\_always\_inline [ezdp\\_flow\\_control\\_status\\_t](#) [ezdp\\_read\\_flow\\_control\\_status](#) (enum [ezdp\\_budget\\_type](#) budget\_type, enum [ezdp\\_flow\\_control\\_node](#) node\_type, uint32\_t node\_id)
- Read flow control status. static \_\_always\_inline enum [ezdp\\_congestion\\_level](#) [ezdp\\_read\\_pmu\\_input\\_queue\\_congestion](#) (uint32\_t side, uint32\_t queue\_id)
- Read PMU input queue congestion level. static \_\_always\_inline uint32\_t [ezdp\\_read\\_global\\_budget](#) (enum [ezdp\\_budget\\_type](#) budget\_type)
- Read global budget counter value. static \_\_always\_inline void [ezdp\\_read\\_pmu\\_input\\_queue\\_status](#) (uint32\_t side, uint32\_t queue\_id, struct [ezdp\\_input\\_queue\\_status](#) \*queue\_status)

- Read PMU input queue status from system info. static \_\_always\_inline void [ezdp\\_read\\_pmu\\_tm\\_output\\_queue\\_status](#) (uint32\_t side, struct [ezdp\\_output\\_queue\\_status](#) \*queue\_status)
- Read PMU TM output queue status from system info. static \_\_always\_inline void [ezdp\\_read\\_pmu\\_discard\\_output\\_queue\\_status](#) (uint32\_t side, struct [ezdp\\_output\\_queue\\_status](#) \*queue\_status)
- Read PMU discard output queue status from system info. static \_\_always\_inline void [ezdp\\_read\\_pmu\\_tm\\_bypass\\_output\\_queue\\_status](#) (uint32\_t side, uint32\_t queue\_id, struct [ezdp\\_output\\_queue\\_status](#) \*queue\_status)
- Read PMU TM bypass output queue status from system info. static \_\_always\_inline void [ezdp\\_read\\_pmu\\_app\\_schlr\\_status](#) (uint32\_t side, uint32\_t app\_schlr\_id, struct [ezdp\\_app\\_schlr\\_status](#) \*app\_schlr\_status)
- Read PMU application scheduler status from system info. static \_\_always\_inline void [ezdp\\_read\\_pmu\\_group\\_schlr\\_status](#) (uint32\_t side, uint32\_t group\_schlr\_id, struct [ezdp\\_group\\_schlr\\_status](#) \*group\_schlr\_status)
- Read PMU group scheduler status from system info. static \_\_always\_inline void [ezdp\\_init\\_tm\\_reporting\\_desc](#) (uint32\_t side, uint32\_t level, ezdp\_tm\_queue\_depth\_desc\_t \*tm\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Initialize TM queue depth descriptor. static \_\_always\_inline ezdp\_tm\_queue\_depth\_handle\_t [ezdp\\_calc\\_tm\\_queue\\_depth\\_handle](#) (uint32\_t queue\_id, ezdp\_tm\_queue\_depth\_desc\_t \*tm\_desc)
- Calculate tm queue depth handle. static \_\_always\_inline uint32\_t [ezdp\\_get\\_tm\\_queue\\_depth](#) (ezdp\_tm\_queue\_depth\_handle\_t tm\_handle, enum [ezdp\\_report\\_size](#) report\_size)
- Get entity queue depth. static \_\_always\_inline bool [ezdp\\_valid\\_tm\\_queue\\_depth\\_handle](#) (ezdp\_tm\_queue\_depth\_handle\_t tm\_handle)

validate tm\_handle received from CP/DP API

---

## Define Documentation

```
#define ezdp_notifier(NAME) void ezdp_notice_handler(void) __attribute__((alias ( #NAME )))
```

Register notifier function.

### Returns:

none

---

## Typedef Documentation

```
typedef void(* ezdp\_notifier\_t)(void)
```

IPC notifier handle function.

---

## Function Documentation

```
static __always_inline ezdp\_job\_id\_t ezdp_alloc_job_id (uint32_t budget_id) [static]
```

Allocate a new job from the PMU.

### Parameters:

[in] *budget\_id* - budget id to charge

### Returns:

ezdp\_job\_id\_t, use ezdp\_job\_alloc\_failed API to check if allocation success or fail

---

```
static __always_inline void ezdp_alloc_job_id_async (ezdp\_job\_id\_t * jobh_ptr, uint32_t budget_id) [static]
```

Non blocking version of [ezdp\\_alloc\\_job\\_id\(\)](#).

**Parameters:**

[out] *jobh\_ptr* - pointer to the job id to write response to  
 [in] *budget\_id* - budget id to charge

**Returns:**

void, use [ezdp\\_job\\_alloc\\_failed](#) API to check if allocation success or fail

```
static __always_inline ezdp\_job\_id\_t ezdp_alloc_multi_job_id (ezdp\_job\_id\_t __cmem * jobhs_ptr,  
uint32_t budget_id, uint32_t num_of_jobs) [static]
```

Allocate multiple new jobs from the PMU.

**Parameters:**

[out] *jobhs\_ptr* - relative offset in CMEM to write response to  
 [in] *budget\_id* - budget id to charge  
 [in] *num\_of\_jobs* - the number of job IDs to allocate (possible values: 1-4)

**Returns:**

[ezdp\\_job\\_id\\_t](#), use [ezdp\\_job\\_alloc\\_failed](#) API on returned *job\_id* to check if allocation success or fail

```
static __always_inline void ezdp_alloc_multi_job_id_async (ezdp\_job\_id\_t __cmem * jobhs_ptr,  
uint32_t budget_id, uint32_t num_of_jobs) [static]
```

Non blocking version of [ezdp\\_alloc\\_multi\\_job\\_id\(\)](#).

**Parameters:**

[out] *jobhs\_ptr* - relative offset in CMEM to write response to  
 [in] *budget\_id* - budget id to charge  
 [in] *num\_of\_jobs* - the number of job IDs to allocate (possible values: 1-4)

**Returns:**

void, use [ezdp\\_job\\_alloc\\_failed](#) API on first allocated *job\_id* to check if allocation success or fail

```
static __always_inline bool ezdp_job_alloc_failed (ezdp\_job\_id\_t ret) [static]
```

Check if allocation of the job failed.

**Parameters:**

[in] *ret* - return value from job allocation API

**Returns:**

bool - true if allocation failed

```
static __always_inline void ezdp_free_job_id (ezdp\_job\_id\_t * __cmem jobh_ptr, uint32_t budget_id) [static]
```

Recycle a job to the PMU.



**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM  
 [in] *budget\_id* - budget id to charge

**Note:**

Any associated frame resources are not recycled.

**Returns:**

none

```
static __always_inline void ezdp_free_job_id_async (ezdp_job_id_t * __cmem jobh_ptr,  uint32_t
budget_id) [static]
```

Non blocking version of [ezdp\\_free\\_job\\_id\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM  
 [in] *budget\_id* - budget id to charge

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the job id was read from CMEM and the request was sent to the PMU.

**Returns:**

void

```
static __always_inline uint32_t ezdp_read_free_job (void) [static]
```

The number of jobs available to be obtained.

**Returns:**

uint32\_t - number of available/free jobs

```
static __always_inline void ezdp_rebudget_job (uint32_t free_budget_id,  uint32_t
free_num_of_jobs,  uint32_t alloc_budget_id,  uint32_t alloc_num_of_jobs) [static]
```

Update the budget to which jobs are credited.

Adds free\_num\_of\_indexes credits back into the budget selected by free\_budget\_id and subtracts alloc\_num\_of\_indexes credits from the budget selected by alloc\_budget\_id.

**Parameters:**

[in] *free\_budget\_id* - budget ID to reimburse  
 [in] *free\_num\_of\_jobs* - number of jobs to reimburse. Decrement operation (can be zero)  
 [in] *alloc\_budget\_id* - budget ID to charge  
 [in] *alloc\_num\_of\_jobs* - number of jobs to charge. Increment operation (can be zero)

**Returns:**

none

```
static __always_inline void ezdp_rebudget_job_async (uint32_t free_budget_id,  uint32_t
free_num_of_jobs,  uint32_t alloc_budget_id,  uint32_t alloc_num_of_jobs) [static]
```

Non blocking version of [ezdp\\_rebudget\\_job\(\)](#).

**Parameters:**

[in] *free\_budget\_id* - budget ID to reimburse  
 [in] *free\_num\_of\_jobs* - number of jobs to reimburse (can be zero)  
 [in] *alloc\_budget\_id* - budget ID to charge  
 [in] *alloc\_num\_of\_jobs* - number of jobs to charge (can be zero)

**Returns:**

none

```
static __always_inline void ezdp_load_job (ezdp_job_id_t * jobh_ptr,  struct ezdp_job_desc
__cmem * jd_ptr) [static]
```

Copy the job descriptor from IMEM to CMEM.

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id  
 [out] *jd\_ptr* - pointer to the job descriptor in CMEM

**Returns:**

none

```
static __always_inline void ezdp_load_job_async (ezdp_job_id_t * jobh_ptr,  struct
ezdp_job_desc __cmem * jd_ptr) [static]
```

Non blocking version of [ezdp\\_load\\_job\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id  
 [out] *jd\_ptr* - pointer to the job descriptor in CMEM

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the that job descriptor is ready in CMEM.

**Returns:**

none

```
static __always_inline void ezdp_store_job (ezdp_job_id_t * jobh_ptr,  struct ezdp_job_desc
__cmem * jd_ptr) [static]
```

Copy the job descriptor from CMEM to IMEM.

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id  
 [in] *jd\_ptr* - pointer to the job descriptor in CMEM

**Returns:**

none

```
static __always_inline void ezdp_store_job_async (ezdp_job_id_t * jobh_ptr,  struct
ezdp_job_desc __cmem * jd_ptr) [static]
```

Non blocking version of [ezdp\\_store\\_job\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id  
 [in] *jd\_ptr* - pointer to the job descriptor in CMEM

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the that job descriptor was written to IMEM.

**Returns:**

none

```
static __always_inline void ezdp_store_job_container (ezdp\_job\_id\_t * jobh_ptr, struct
ezdp\_job\_container\_desc __cmem * jd_ptr) [static]
```

Copy the job container descriptor from CMEM to IMEM.

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id  
 [in] *jd\_ptr* - pointer to the job container descriptor in CMEM

**Returns:**

none

```
static __always_inline void ezdp_store_job_container_async (ezdp\_job\_id\_t * jobh_ptr, struct
ezdp\_job\_container\_desc __cmem * jd_ptr) [static]
```

Non blocking version of [ezdp\\_store\\_job\\_container\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id  
 [in] *jd\_ptr* - pointer to the job descriptor in CMEM

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the that job descriptor was written to IMEM.

**Returns:**

none

```
static __always_inline void ezdp_request_job_id (ezdp\_job\_id\_t * __cmem jobh_ptr) [static]
```

Request a new job from the PMU.

**Parameters:**

[out] *jobh\_ptr* - pointer to the job id in CMEM

**Note:**

- The job request is non blocking. You must call [ezdp\\_wait\\_for\\_job\\_id\(\)](#) to wait for the request to complete before reading the response data in CMEM.

**Returns:**

none

```
static __always_inline void ezdp_wait_for_job_id (void) [static]
```

Suspend execution until a job request completes.

**Note:**

Should be called after [ezdp\\_request\\_job\\_id\(\)](#) to wait for the request to complete. When returning from this routine, the job id is ready in CMEM.

**Returns:**

none

```
static __always_inline void ezdp_cancel_job_request(void) [static]
```

Cancel a job request from the PMU.

**Returns:**

none

```
static __always_inline void ezdp_receive_job(ezdp\_job\_id\_t __cmem * jobh_ptr, struct
ezdp\_job\_desc __cmem * jd_ptr, uint32_t flags) [static]
```

Request a new job from the PMU and load it to CMEM.

Convenience function which requests a new job, waits for the job request to complete, and then loads the job descriptor.

**Parameters:**

[out] *jobh\_ptr* - pointer to the job id in CMEM  
 [out] *jd\_ptr* - pointer to the job descriptor in CMEM  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in *ezdp\_job\_flags* enum.

**Returns:**

none

```
static __always_inline void ezdp_send_job_id_to_queue(ezdp\_job\_id\_t __cmem * jobh_ptr,
uint32_t side, uint32_t target_queue, uint32_t flags) [static]
```

Dispatch the job to another PMU queue.

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM  
 [in] *side* - target PMU side  
 [in] *target\_queue* - target PMU queue  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in *ezdp\_job\_flags* enum.

**Returns:**

none

```
static __always_inline void ezdp_send_job_id_to_queue_async(ezdp\_job\_id\_t __cmem *
jobh_ptr, uint32_t side, uint32_t target_queue, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_send\\_job\\_id\\_to\\_queue\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM  
 [in] *side* - target PMU side  
 [in] *target\_queue* - target PMU queue

[in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in `ezdp_job_flags` enum.

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the job id was read from CMEM and the request was sent to the PMU.

**Returns:**

none

```
static __always_inline void ezdp_send_job_id_to_tm (ezdp_job_id_t __cmem * jobh_ptr,
uint32_t side,   uint32_t flags) [static]
```

Transmit the job via the TM.

Return the job back to the PMU and then transmit it via the TM.

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM

[in] *side* - target TM side

[in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in `ezdp_job_flags` enum.

**Returns:**

none

```
static __always_inline void ezdp_send_job_id_to_tm_async (ezdp_job_id_t __cmem * jobh_ptr,
uint32_t side,   uint32_t flags) [static]
```

Non blocking version of [ezdp\\_send\\_job\\_id\\_to\\_tm\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM

[in] *side* - target TM side

[in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in `ezdp_job_flags` enum.

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the job id was read from CMEM and the request was sent to the PMU.

**Returns:**

none

```
static __always_inline void ezdp_send_job_id_to_interface (ezdp_job_id_t __cmem * jobh_ptr,
uint32_t side,   uint32_t output_channel,   uint32_t flags) [static]
```

Transmit the job directly to an output queue channel, bypassing the TM.

Return the job back to the PMU and then transmit it directly to interface. The interface is selected using output channel and side.

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM

[in] *side* - transmission side for job

[in] *output\_channel* - output channel in selected side to transmit packet to (define destination port)

[in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in `ezdp_job_flags` enum.

**Returns:**

none

```
static __always_inline void ezdp_send_job_id_to_interface_async(ezdp\_job\_id\_t __cmem *  
jobh\_ptr, uint32_t side, uint32_t output\_channel, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_send\\_job\\_id\\_to\\_interface\(\)](#).

**Parameters:**

- [in] [jobh\\_ptr](#) - pointer to the job id in CMEM
- [in] [side](#) - transmission side for job
- [in] [output\\_channel](#) - output channel in selected side to transmit packet to (define destination port)
- [in] [flags](#) - execution flags. Bitwise OR of zero or more flags as defined in [ezdp\\_job\\_flags](#) enum.

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the job id was read from CMEM and the request was sent to the PMU.

**Returns:**

none

```
static __always_inline void ezdp_update_job_id_queue(ezdp\_job\_id\_t __cmem *jobh\_ptr,  
uint32_t side, uint32_t target\_queue, uint32_t flags) [static]
```

Move the job to another PMU queue without dispatching it.

When called, a request is sent to the PMU to move the job to another queue, thus removing it from the previous ordering and inserting it to a new queue with new ordering. However, the job is not dispatched and is still owned by the calling thread which can continue to work on it.

**Parameters:**

- [in] [jobh\\_ptr](#) - pointer to the job id in CMEM
- [in] [side](#) - target PMU side
- [in] [target\\_queue](#) - target PMU queue
- [in] [flags](#) - execution flags. Bitwise OR of zero or more flags as defined in [ezdp\\_job\\_flags](#) enum.

**Returns:**

none

```
static __always_inline void ezdp_send_job_to_queue(ezdp\_job\_id\_t __cmem *jobh\_ptr, struct  
ezdp\_job\_desc __cmem *jd\_ptr, uint32_t side, uint32_t target\_queue, uint32_t flags)  
[static]
```

Store the job descriptor and dispatch the job to another queue in the PMU.

Convenience function which stores the job descriptor and then dispatches the job.

**Parameters:**

- [in] [jobh\\_ptr](#) - pointer to the job id in CMEM
- [in] [jd\\_ptr](#) - pointer to the job descriptor in CMEM
- [in] [side](#) - target PMU side
- [in] [target\\_queue](#) - target PMU queue
- [in] [flags](#) - execution flags. Bitwise OR of zero or more flags as defined in [ezdp\\_job\\_flags](#) enum.

**Returns:**

none

```
static __always_inline void ezdp_send_job_to_tm(ezdp\_job\_id\_t __cmem *jobh\_ptr, struct  
ezdp\_job\_desc __cmem *jd\_ptr, uint32_t side, uint32_t flags) [static]
```

Store the job descriptor and transmit the job via the TM.

Return the job back to the PMU and then transmit it via the TM. Convenience function which stores the job descriptor and then forwards the job to the TM.

**Parameters:**

- [in] *jobh\_ptr* - pointer to the job id in CMEM
- [in] *jd\_ptr* - pointer to the job descriptor in CMEM
- [in] *side* - target TM side
- [in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in `ezdp_job_flags` enum.

**Returns:**

none

```
static __always_inline void ezdp_send_job_to_interface (ezdp_job_id_t __cmem * jobh_ptr,
struct ezdp_job_desc __cmem * jd_ptr,  uint32_t side,  uint32_t output_channel,  uint32_t
flags) [static]
```

Store the job descriptor and transmit the job directly to interface by bypassing the TM.

Return the job back to the PMU and then transmit directly to output channel. Convenience function which stores the job descriptor and then transmit the job directly to interface. The interface is selected using output channel and side

**Parameters:**

- [in] *jobh\_ptr* - pointer to the job id in CMEM
- [in] *jd\_ptr* - pointer to the job descriptor in CMEM
- [in] *side* - the side job should be transmitted
- [in] *output\_channel* - output channel for transmit; used to map to destination port
- [in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in `ezdp_job_flags` enum.

**Returns:**

none

```
static __always_inline void ezdp_update_job_queue (ezdp_job_id_t * __cmem jobh_ptr,  struct
ezdp_job_desc __cmem * jd_ptr,  uint32_t side,  uint32_t target_queue,  uint32_t flags)
[static]
```

Store the job descriptor and move the job to another PMU queue without dispatching it.

When called, a request is sent to the PMU to move the job to another queue, thus removing it from the previous ordering and inserting it to a new queue with new ordering. However, the job is not dispatched and is still owned by the calling thread which can continue to work on it.

**Parameters:**

- [in] *jobh\_ptr* - pointer to the job id in CMEM
- [in] *jd\_ptr* - pointer to the job descriptor in CMEM
- [in] *side* - target PMU side
- [in] *target\_queue* - target PMU queue
- [in] *flags* - execution flags. Bitwise OR of zero or more flags as defined in `ezdp_job_flags` enum.

**Returns:**

none

```
static __always_inline void ezdp_discard_job_id (ezdp_job_id_t * __cmem jobh_ptr) [static]
```

Discard a job and all its associated frame resources.

**Parameters:**

- [in] *jobh\_ptr* - pointer to the job id in CMEM

**Note:**

All associated frame resources are recycled. TKB flag is implied to be zero (resources are recycled regardless of FD[TKB] state). Discard job with NULL frame or non valid\_data\_buf is not supported

**Returns:**

none

```
static __always_inline void ezdp_discard_job_id_async (ezdp\_job\_id\_t * __cmem jobh_ptr)
[static]
```

Non blocking version of [ezdp\\_discard\\_job\\_id\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the job id was read from CMEM and the request was sent to the PMU.

**Returns:**

void

```
static __always_inline void ezdp_discard_job (ezdp\_job\_id\_t __cmem * jobh_ptr, struct
ezdp\_job\_desc __cmem * jd_ptr) [static]
```

Store the job descriptor and discard a job and all its associated frame resources.

Convenience function which stores the job descriptor and then forwards the job back to PMU for discarding.

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM

[in] *jd\_ptr* - pointer to the job descriptor in CMEM

**Returns:**

none

```
static __always_inline void ezdp_send_job_id_container (ezdp\_job\_id\_t __cmem * jobh_ptr)
[static]
```

Send request to PMU to distribute the job container.

When called, the job container is send to the PMU. It distribute the container job requests while preserving order with other jobs in the queue and recycles the job id after reading all job requests. The job container may contain a link to an extension container, which gets further expanded and recycled in a similar manner.

\*

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM, which points to the job container

**Note:**

job\_id must point to the job container, which contains multiple jobs

**Returns:**

none



```
static __always_inline void ezdp_send_job_id_container_async (ezdp\_job\_id\_t __cmem *  
jobh_ptr) [static]
```

Non blocking version of [ezdp\\_send\\_job\\_id\\_container\(\)](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM, which point to the job container

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the job id was read from CMEM and the request was sent to the PMU.

**Returns:**

none

```
static __always_inline void ezdp_send_job_container (ezdp\_job\_id\_t __cmem * jobh_ptr, struct  
ezdp\_job\_container\_desc __cmem * jd_container_ptr) [static]
```

Store the job container and send request to PMU to distribute it.

Convenience function which stores the job descriptor and then call [ezdp\\_send\\_job\\_id\\_container](#).

**Parameters:**

[in] *jobh\_ptr* - pointer to the job id in CMEM

[in] *jd\_container\_ptr* - pointer to the job container in CMEM

**Returns:**

none

```
static __always_inline uint32_t ezdp_container_job_count (ezdp\_job\_container\_info\_t info)  
[static]
```

Return the number of the jobs that are in the job container.

**Parameters:**

[in] *info* - job container info

**Returns:**

1 - 7

```
static __always_inline ezdp\_job\_container\_info\_t ezdp_container_info (uint32_t job_count)  
[static]
```

Set the info according to the number of jobs in the job container.

**Parameters:**

[in] *job\_count* - job count in the job container, include extension

**Returns:**

[ezdp\\_job\\_container\\_info\\_t](#)

```
static __always_inline void ezdp_notify_cpu (uint32_t target_cpu_id) [static]
```

Notify target CPU.

Turn on notice flag in target CPU and select this CPU available for execution

**Parameters:**

[in] *target\_cpu\_id* - The ID of the CPU

**Note:**

Used to implement IPC

**Returns:**

none

**static \_\_always\_inline bool ezdp\_notice\_pending (void) [static]**

Check if there is a new notice.

**Returns:**

bool - true when having new notice indication

**static \_\_always\_inline void ezdp\_clear\_notice (void) [static]**

Clear notice indication.

**Returns:**

none

**static \_\_always\_inline void ezdp\_wait\_for\_notice (void) [static]**

Suspend execution until receiving new notification for CPU.

**Note:**

New notice indication is cleared automatically

**Returns:**

none

**static \_\_always\_inline bool ezdp\_check\_notice (void) [static]**

Check if there is a new notice and clear new notice indication.

**Returns:**

bool - true when having new notice indication

**static \_\_always\_inline void ezdp\_wait\_for\_event (void) [static]**

Suspend execution until job request completed or new notice received.

**Note:**

Use `ezdp_check_notice` or `ezdp_notice_pending` to check if job or notice received.

**Returns:**

none

```
static __always_inline void ezdp_handle_notice(void) [static]
```

Handle notice.

If there is a new notice, call registered notifier function.

**Returns:**

none

```
static __always_inline ezdp\_congestion\_status\_t ezdp_read_congestion_status (enum ezdp\_flow\_control\_node node_type, uint32_t node_id) [static]
```

Read priority drop congestion status.

**Parameters:**

[in] *node\_type* - flow control node type

[in] *node\_id* - index of the node For the channel nodes: 0 - 255 For the port nodes: 0 - 127 For group nodes: 0 - 15 For global node: 0/value is ignored

**Returns:**

[ezdp\\_congestion\\_status\\_t](#) - according to struct [ezdp\\_congestion\\_status](#)

```
static __always_inline ezdp\_flow\_control\_status\_t ezdp_read_flow_control_status (enum ezdp\_budget\_type budget_type, enum ezdp\_flow\_control\_node node_type, uint32_t node_id) [static]
```

Read flow control status.

**Parameters:**

[in] *budget\_type* - flow control budget type

[in] *node\_type* - flow control node type

[in] *node\_id* - index of the node For the channel nodes: 0 - 255 For the port nodes: 0 - 127 For group nodes: 0 - 15 For global node: 0/value is ignored

**Returns:**

[ezdp\\_flow\\_control\\_status\\_t](#) - according to struct [ezdp\\_flow\\_control\\_status](#)

```
static __always_inline enum ezdp\_congestion\_level ezdp_read_pmu_input_queue_congestion (uint32_t side, uint32_t queue_id) [static]
```

Read PMU input queue congestion level.

**Parameters:**

[in] *side* - PMU side {0,1}

[in] *queue\_id* - pmu queue id {0 - 127}

**Returns:**

enum [ezdp\\_congestion\\_level](#) - congestion level

```
static __always_inline uint32_t ezdp_read_global_budget (enum ezdp\_budget\_type budget_type) [static]
```

Read global budget counter value.

**Parameters:**

[in] *budget\_type* - flow control budget type

**Returns:**

uint32\_t - value of the budget

```
static __always_inline void ezdp_read_pmu_input_queue_status (uint32_t side,    uint32_t
queue_id,    struct ezdp\_input\_queue\_status * queue_status) [static]
```

Read PMU input queue status from system info.

**Parameters:**

[in] *side* - PMU side {0,1}

[in] *queue\_id* - pmu queue id {0 - 127}

[in] *queue\_status* - point to write queue status into

**Returns:**

none

```
static __always_inline void ezdp_read_pmu_tm_output_queue_status (uint32_t side,    struct
ezdp\_output\_queue\_status * queue_status) [static]
```

Read PMU TM output queue status from system info.

**Parameters:**

[in] *side* - PMU side {0,1}

[in] *queue\_status* - point to write queue status into

**Returns:**

none

```
static __always_inline void ezdp_read_pmu_discard_output_queue_status (uint32_t side,    struct
ezdp\_output\_queue\_status * queue_status) [static]
```

Read PMU discard output queue status from system info.

**Parameters:**

[in] *side* - PMU side {0,1}

[in] *queue\_status* - point to write queue status into

**Returns:**

none

```
static __always_inline void ezdp_read_pmu_tm_bypass_output_queue_status (uint32_t side,
uint32_t queue_id,    struct ezdp\_output\_queue\_status * queue_status) [static]
```

Read PMU TM bypass output queue status from system info.

**Parameters:**

[in] *side* - PMU side {0,1}

[in] *queue\_id* - queue id {0-3}

[in] *queue\_status* - point to write queue status into

**Returns:**

none

```
static __always_inline void ezdp_read_pmu_app_schlr_status (uint32_t side,   uint32_t
app_schlr_id,   struct ezdp\_app\_schlr\_status * app_schlr_status) [static]
```

Read PMU application scheduler status from system info.

**Parameters:**

[in] *side* - PMU side {0,1}

[in] *app\_schlr\_id* - application scheduler id {0-7}

[in] *app\_schlr\_status* - point to write status into

**Returns:**

none

```
static __always_inline void ezdp_read_pmu_group_schlr_status (uint32_t side,   uint32_t
group_schlr_id,   struct ezdp\_group\_schlr\_status * group_schlr_status) [static]
```

Read PMU group scheduler status from system info.

**Parameters:**

[in] *side* - PMU side {0,1}

[in] *group\_schlr\_id* - group scheduler id {0-15}

[in] *group\_schlr\_status* - point to write status into

**Returns:**

none

```
static __always_inline void ezdp_init_tm_reporting_desc (uint32_t side,   uint32_t level,
ezdp_tm_queue_depth_desc_t * tm_desc,   char __cmem * work_area_ptr,   uint32_t
work_area_size) [static]
```

Initialize TM queue depth descriptor.

**Parameters:**

[in] *side* - TM side (0 / 1)

[in] *level* - TM level (0..4)

[out] *tm\_desc* - TM Queue depth descriptor

[in] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_TM\_REPORT\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Returns:**

void

```
static __always_inline ezdp_tm_queue_depth_handle_t ezdp_calc_tm_queue_depth_handle
(uint32_t queue_id,   ezdp_tm_queue_depth_desc_t * tm_desc) [static]
```

Calculate tm queue depth handle.

**Parameters:**

[in] *queue\_id* - TM Queue-ID

[in] *tm\_desc* - TM reporting description

**Note:**

For better performance, use CP calculation instead of dp calculation

**Returns:**

ezdp\_tm\_queue\_depth\_handle\_t

```
static __always_inline uint32_t ezdp_get_tm_queue_depth (ezdp_tm_queue_depth_handle_t  
tm_handle, enum ezdp\_report\_size report_size) [static]
```

Get entity queue depth.

**Parameters:**

[in] *tm\_handle* - 32bit TM handle value for queue and level received from CP/DP API

[in] *report\_size* - TM reporting size as configured by CP (1B/2B/4B)

**Note:**

API relevant only if report is enabled for the requested queue, etc valid *tm\_handle*

**Returns:**

entity queue depth

```
static __always_inline bool ezdp_valid_tm_queue_depth_handle (ezdp_tm_queue_depth_handle_t  
tm_handle) [static]
```

validate *tm\_handle* received from CP/DP API

**Parameters:**

[in] *tm\_handle* - 32bit TM handle value for queue and level received from CP/DP API

**Returns:**

bool - true if *tm\_handle* is valid

## dpe/dp/include/ezdp\_job\_defs.h File Reference

### Data Structures

- struct [ezdp\\_job\\_rx\\_interface\\_info](#)
- Info field for incoming job from external RX interfaces. struct [ezdp\\_job\\_rx\\_loopback\\_info](#)
- Info field for incoming job from loopback ports. struct [ezdp\\_job\\_rx\\_confirmation\\_info](#)
- Info field for incoming job from TX confirmation ports. struct [ezdp\\_job\\_rx\\_timer\\_info](#)
- Info field for incoming timer job (PMU Timer). struct [ezdp\\_job\\_rx\\_user\\_info](#)
- Info field for incoming frame job from generic user forwarding. struct [ezdp\\_job\\_rx\\_info](#)
- Job receive info. struct [ezdp\\_job\\_tx\\_info](#)
- Info field for transmitting frame job (TM mode is full or tm qos bypass). struct [ezdp\\_job\\_queue\\_cmd\\_info](#)
- Job container send to queue request info. struct [ezdp\\_job\\_transmit\\_cmd\\_info](#)
- Job container send out request info. struct [ezdp\\_job\\_discard\\_cmd\\_info](#)
- Job container discard request info. struct [ezdp\\_congestion\\_status](#)
- System priority drop congestion status. struct [ezdp\\_flow\\_control\\_status](#)
- Flow control status. struct [ezdp\\_input\\_queue\\_status](#)
- PMU physical input queue status definition (based on PMU system info). struct [ezdp\\_output\\_queue\\_status](#)
- PMU output queue status definition (based on PMU system info). struct [ezdp\\_app\\_schlr\\_status](#)
- PMU application scheduler status (based on PMU system info). struct [ezdp\\_group\\_schlr\\_status](#)
- PMU group scheduler status (based on PMU system info). struct [ezdp\\_job\\_container\\_cmd\\_desc](#)
- Job container request. struct [ezdp\\_job\\_container\\_desc](#)
- Job container descriptor. struct [ezdp\\_job\\_desc](#)

### job descriptor data structure Defines

- #define [EZDP\\_TM\\_REPORT\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_init\_tm\_report\_wa)
- Work area minimal required size definitions. #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_SEC\\_SIZE](#) 8
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_SEC\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_SEC\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_SEC\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_CHECKED\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_CHECKED\\_FLAG\\_OFFSET](#) 8
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_CHECKED\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_CHECKED\\_FLAG\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_CHECKED\\_FLAG\\_MASK](#) (1 << EZDP\_JOB\_RX\_INTERFACE\_INFO\_CRC\_CHECKED\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_OK\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_OK\\_FLAG\\_OFFSET](#) 9
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_OK\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_OK\\_FLAG\\_WORD\\_OFFSET](#) 9
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_CRC\\_OK\\_FLAG\\_MASK](#) (1 << EZDP\_JOB\_RX\_INTERFACE\_INFO\_CRC\_OK\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED10\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED10\\_OFFSET](#) 10
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED11\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED11\\_OFFSET](#) 11
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TRUNCATION\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TRUNCATION\\_FLAG\\_OFFSET](#) 12
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TRUNCATION\\_FLAG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TRUNCATION\\_FLAG\\_WORD\\_OFFSET](#) 12
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TRUNCATION\\_FLAG\\_MASK](#) (1 << EZDP\_JOB\_RX\_INTERFACE\_INFO\_TRUNCATION\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_ICU\\_SUCC\\_PARSING\\_FLAG\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_ICU\\_SUCC\\_PARSING\\_FLAG\\_OFFSET](#) 13
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_ICU\\_SUCC\\_PARSING\\_FLAG\\_WORD\\_SELECT](#) 0

- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_ICU\\_SUCC\\_PARSING\\_FLAG\\_WORD\\_OFFSET](#) 13
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_ICU\\_SUCC\\_PARSING\\_FLAG\\_MASK](#) (1 << EZDP\_JOB\_RX\_INTERFACE\_INFO\_ICU\_SUCC\_PARSING\_FLAG\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED14\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED14\\_OFFSET](#) 14
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED15\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_RESERVED15\\_OFFSET](#) 15
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_COUNT\\_SIZE](#) 6
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_COUNT\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_COUNT\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_COUNT\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_GLOBAL\\_CONGESTION\\_LEVEL\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_GLOBAL\\_CONGESTION\\_LEVEL\\_OFFSET](#) 22
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_GLOBAL\\_CONGESTION\\_LEVEL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_GLOBAL\\_CONGESTION\\_LEVEL\\_WORD\\_OFFSET](#) 22
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_CONGESTION\\_LEVEL\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_CONGESTION\\_LEVEL\\_OFFSET](#) 24
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_CONGESTION\\_LEVEL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_IMEM\\_BUF\\_CONGESTION\\_LEVEL\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_EMEM\\_BUF\\_CONGESTION\\_LEVEL\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_EMEM\\_BUF\\_CONGESTION\\_LEVEL\\_OFFSET](#) 26
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_EMEM\\_BUF\\_CONGESTION\\_LEVEL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_EMEM\\_BUF\\_CONGESTION\\_LEVEL\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_JOB\\_CONGESTION\\_LEVEL\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_JOB\\_CONGESTION\\_LEVEL\\_OFFSET](#) 28
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_JOB\\_CONGESTION\\_LEVEL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_JOB\\_CONGESTION\\_LEVEL\\_WORD\\_OFFSET](#) 28
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_PMU\\_QUEUE\\_CONGESTION\\_LEVEL\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_PMU\\_QUEUE\\_CONGESTION\\_LEVEL\\_OFFSET](#) 30
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_PMU\\_QUEUE\\_CONGESTION\\_LEVEL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_PMU\\_QUEUE\\_CONGESTION\\_LEVEL\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_NSEC\\_SIZE](#) 32
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_NSEC\\_OFFSET](#) 32
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_NSEC\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_TIMESTAMP\\_NSEC\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_INTERFACE\\_INFO\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_RESERVED0\\_15\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_RESERVED0\\_15\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_REPLICATION\\_ID\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_REPLICATION\\_ID\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_REPLICATION\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_REPLICATION\\_ID\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_RESERVED32\\_63\\_SIZE](#) 32
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_RESERVED32\\_63\\_OFFSET](#) 32
- #define [EZDP\\_JOB\\_RX\\_LOOPBACK\\_INFO\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_SEC\\_SIZE](#) 8
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_SEC\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_SEC\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_SEC\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_RESERVED8\\_31\\_SIZE](#) 24
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_RESERVED8\\_31\\_OFFSET](#) 8
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_NSEC\\_SIZE](#) 32
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_NSEC\\_OFFSET](#) 32
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_NSEC\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_TIMESTAMP\\_NSEC\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_CONFIRMATION\\_INFO\\_WORD\\_COUNT](#) 2



- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_RESERVED0\\_8\\_SIZE](#) 8
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_RESERVED0\\_8\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_TIMER\\_ID\\_SIZE](#) 8
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_TIMER\\_ID\\_OFFSET](#) 8
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_TIMER\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_TIMER\\_ID\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_RESERVED16\\_31\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_RESERVED16\\_31\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_EVENT\\_ID\\_SIZE](#) 32
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_EVENT\\_ID\\_OFFSET](#) 32
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_EVENT\\_ID\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_EVENT\\_ID\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_TIMER\\_INFO\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO0\\_SIZE](#) 32
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO0\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO0\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO0\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO1\\_SIZE](#) 32
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO1\\_OFFSET](#) 32
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO1\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_USER\\_DATA\\_INFO1\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_USER\\_INFO\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_JOB\\_RX\\_INFO\\_GROSS\\_CHECKSUM\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_RX\\_INFO\\_GROSS\\_CHECKSUM\\_OFFSET](#) 64
- #define [EZDP\\_JOB\\_RX\\_INFO\\_GROSS\\_CHECKSUM\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_JOB\\_RX\\_INFO\\_GROSS\\_CHECKSUM\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_INFO\\_RESERVED112\\_127\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_RX\\_INFO\\_RESERVED112\\_127\\_OFFSET](#) 80
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SOURCE\\_QUEUE\\_SIZE](#) 7
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SOURCE\\_QUEUE\\_OFFSET](#) 96
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SOURCE\\_QUEUE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SOURCE\\_QUEUE\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SIDE\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SIDE\\_OFFSET](#) 103
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SIDE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SIDE\\_WORD\\_OFFSET](#) 7
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SIDE\\_MASK](#) (1 << EZDP\_JOB\_RX\_INFO\_SIDE\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_RX\\_INFO\\_RESERVED104\\_107\\_SIZE](#) 4
- #define [EZDP\\_JOB\\_RX\\_INFO\\_RESERVED104\\_107\\_OFFSET](#) 104
- #define [EZDP\\_JOB\\_RX\\_INFO\\_RESERVED108\\_109\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_RX\\_INFO\\_RESERVED108\\_109\\_OFFSET](#) 108
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_VALID\\_OFFSET](#) 110
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_VALID\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_VALID\\_WORD\\_OFFSET](#) 14
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_VALID\\_MASK](#) (1 << EZDP\_JOB\_RX\_INFO\_SEQ\_NUMBER\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_RX\\_INFO\\_IS\\_SERVICE\\_READY\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_RX\\_INFO\\_IS\\_SERVICE\\_READY\\_OFFSET](#) 111
- #define [EZDP\\_JOB\\_RX\\_INFO\\_IS\\_SERVICE\\_READY\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_RX\\_INFO\\_IS\\_SERVICE\\_READY\\_WORD\\_OFFSET](#) 15
- #define [EZDP\\_JOB\\_RX\\_INFO\\_IS\\_SERVICE\\_READY\\_MASK](#) (1 << EZDP\_JOB\_RX\_INFO\_IS\_SERVICE\_READY\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_OFFSET](#) 112
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_RX\\_INFO\\_SEQ\\_NUMBER\\_WORD\\_OFFSET](#) 16

- #define [EZDP\\_JOB\\_RX\\_INFO\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_ID\\_SELECT\\_SIZE](#) 9
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_ID\\_SELECT\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_ID\\_SELECT\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_ID\\_SELECT\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED25\\_29\\_SIZE](#) 4
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED25\\_29\\_OFFSET](#) 25
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_COLOR\\_SIZE](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_COLOR\\_OFFSET](#) 29
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_COLOR\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_COLOR\\_WORD\\_OFFSET](#) 29
- #define [EZDP\\_JOB\\_TX\\_INFO\\_FLOW\\_ID\\_SIZE](#) 19
- #define [EZDP\\_JOB\\_TX\\_INFO\\_FLOW\\_ID\\_OFFSET](#) 32
- #define [EZDP\\_JOB\\_TX\\_INFO\\_FLOW\\_ID\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_FLOW\\_ID\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_TX\\_INFO\\_SIDE\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_SIDE\\_OFFSET](#) 51
- #define [EZDP\\_JOB\\_TX\\_INFO\\_SIDE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_SIDE\\_WORD\\_OFFSET](#) 19
- #define [EZDP\\_JOB\\_TX\\_INFO\\_SIDE\\_MASK](#) (1 << EZDP\_JOB\_TX\_INFO\_SIDE\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED52\\_55\\_SIZE](#) 4
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED52\\_55\\_OFFSET](#) 52
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_MODE\\_SIZE](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_MODE\\_OFFSET](#) 56
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_MODE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_PACKET\\_SWITCH\\_MODE\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED59\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED59\\_OFFSET](#) 59
- #define [EZDP\\_JOB\\_TX\\_INFO\\_QOS\\_BYPASS\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_QOS\\_BYPASS\\_OFFSET](#) 60
- #define [EZDP\\_JOB\\_TX\\_INFO\\_QOS\\_BYPASS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_QOS\\_BYPASS\\_WORD\\_OFFSET](#) 28
- #define [EZDP\\_JOB\\_TX\\_INFO\\_QOS\\_BYPASS\\_MASK](#) (1 << EZDP\_JOB\_TX\_INFO\_QOS\_BYPASS\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED61\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED61\\_OFFSET](#) 61
- #define [EZDP\\_JOB\\_TX\\_INFO\\_DROP\\_MODE\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_TX\\_INFO\\_DROP\\_MODE\\_OFFSET](#) 62
- #define [EZDP\\_JOB\\_TX\\_INFO\\_DROP\\_MODE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_DROP\\_MODE\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_STREAM\\_ID\\_SIZE](#) 24
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_STREAM\\_ID\\_OFFSET](#) 64
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_STREAM\\_ID\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_STREAM\\_ID\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED88\\_90\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED88\\_90\\_OFFSET](#) 88
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE2\\_SIZE](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE2\\_OFFSET](#) 90
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE2\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE2\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE1\\_SIZE](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE1\\_OFFSET](#) 93
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE1\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_JOB\\_TX\\_INFO\\_STAT\\_CODE\\_PROFILE1\\_WORD\\_OFFSET](#) 29
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_SIZE](#) 5
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_OFFSET](#) 96
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_WORD\\_SELECT](#) 3

- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_CONTROL\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_CONTROL\\_OFFSET](#) 101
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_CONTROL\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_CONTROL\\_WORD\\_OFFSET](#) 5
- #define [EZDP\\_JOB\\_TX\\_INFO\\_INTER\\_PACKET\\_GAP\\_CONTROL\\_MASK](#) (1 << EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CONTROL\_WORD\_OFFSET)
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED102\\_103\\_SIZE](#) 2
- #define [EZDP\\_JOB\\_TX\\_INFO\\_RESERVED102\\_103\\_OFFSET](#) 102
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_SCALE\\_PROFILE\\_SIZE](#) 8
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_SCALE\\_PROFILE\\_OFFSET](#) 104
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_SCALE\\_PROFILE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_SCALE\\_PROFILE\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_SCALE\\_PROFILE\\_SIZE](#) 8
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_SCALE\\_PROFILE\\_OFFSET](#) 112
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_SCALE\\_PROFILE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_SCALE\\_PROFILE\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_TEMPLATE\\_PROFILE\\_SIZE](#) 4
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_TEMPLATE\\_PROFILE\\_OFFSET](#) 120
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_TEMPLATE\\_PROFILE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_CLASS\\_TEMPLATE\\_PROFILE\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_TEMPLATE\\_PROFILE\\_SIZE](#) 4
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_TEMPLATE\\_PROFILE\\_OFFSET](#) 124
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_TEMPLATE\\_PROFILE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WRED\\_FLOW\\_TEMPLATE\\_PROFILE\\_WORD\\_OFFSET](#) 28
- #define [EZDP\\_JOB\\_TX\\_INFO\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_JOB\\_QUEUE\\_CMD\\_INFO\\_TARGET\\_QUEUE\\_SIZE](#) 7
- #define [EZDP\\_JOB\\_QUEUE\\_CMD\\_INFO\\_TARGET\\_QUEUE\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_QUEUE\\_CMD\\_INFO\\_SIDE\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_QUEUE\\_CMD\\_INFO\\_SIDE\\_OFFSET](#) 7
- #define [EZDP\\_JOB\\_QUEUE\\_CMD\\_INFO\\_SIDE\\_MASK](#) (1 << EZDP\_JOB\_QUEUE\_CMD\_INFO\_SIDE\_OFFSET)
- #define [EZDP\\_JOB\\_QUEUE\\_CMD\\_INFO\\_RESERVED8\\_15\\_SIZE](#) 8
- #define [EZDP\\_JOB\\_QUEUE\\_CMD\\_INFO\\_RESERVED8\\_15\\_OFFSET](#) 8
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_OUTPUT\\_CHANNEL\\_SIZE](#) 10
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_OUTPUT\\_CHANNEL\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_SIDE\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_SIDE\\_OFFSET](#) 10
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_SIDE\\_MASK](#) (1 << EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_OFFSET)
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_DESTINATION\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_DESTINATION\\_OFFSET](#) 11
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_DESTINATION\\_MASK](#) (1 << EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_DESTINATION\_OFFSET)
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_RESERVED12\\_15\\_SIZE](#) 4
- #define [EZDP\\_JOB\\_TRANSMIT\\_CMD\\_INFO\\_RESERVED12\\_15\\_OFFSET](#) 12
- #define [EZDP\\_JOB\\_DISCARD\\_CMD\\_INFO\\_RESERVED0\\_9\\_SIZE](#) 10
- #define [EZDP\\_JOB\\_DISCARD\\_CMD\\_INFO\\_RESERVED0\\_9\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_DISCARD\\_CMD\\_INFO\\_SIDE\\_SIZE](#) 1
- #define [EZDP\\_JOB\\_DISCARD\\_CMD\\_INFO\\_SIDE\\_OFFSET](#) 10
- #define [EZDP\\_JOB\\_DISCARD\\_CMD\\_INFO\\_SIDE\\_MASK](#) (1 << EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_OFFSET)
- #define [EZDP\\_JOB\\_DISCARD\\_CMD\\_INFO\\_RESERVED11\\_15\\_SIZE](#) 5
- #define [EZDP\\_JOB\\_DISCARD\\_CMD\\_INFO\\_RESERVED11\\_15\\_OFFSET](#) 11
- #define [EZDP\\_CONGESTION\\_STATUS\\_IMEM\\_BUF\\_CONGESTION\\_LEVEL\\_SIZE](#) 2
- #define [EZDP\\_CONGESTION\\_STATUS\\_IMEM\\_BUF\\_CONGESTION\\_LEVEL\\_OFFSET](#) 0
- #define [EZDP\\_CONGESTION\\_STATUS\\_IMEM\\_BUF\\_GUARANTEE\\_SIZE](#) 1

```

• #define EZDP_CONGESTION_STATUS_IMEM_BUF_GUARANTEE_OFFSET 2
• #define EZDP_CONGESTION_STATUS_IMEM_BUF_GUARANTEE_MASK (1 <<
EZDP_CONGESTION_STATUS_IMEM_BUF_GUARANTEE_OFFSET)
• #define EZDP_CONGESTION_STATUS_RESERVED3_SIZE 1
• #define EZDP_CONGESTION_STATUS_RESERVED3_OFFSET 3
• #define EZDP_CONGESTION_STATUS_EMEM_BUF_CONGESTION_LEVEL_SIZE 2
• #define EZDP_CONGESTION_STATUS_EMEM_BUF_CONGESTION_LEVEL_OFFSET 4
• #define EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_SIZE 1
• #define EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_OFFSET 6
• #define EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_MASK (1 <<
EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_OFFSET)
• #define EZDP_CONGESTION_STATUS_RESERVED7_SIZE 1
• #define EZDP_CONGESTION_STATUS_RESERVED7_OFFSET 7
• #define EZDP_CONGESTION_STATUS_JOB_CONGESTION_LEVEL_SIZE 2
• #define EZDP_CONGESTION_STATUS_JOB_CONGESTION_LEVEL_OFFSET 8
• #define EZDP_CONGESTION_STATUS_JOB_GUARANTEE_SIZE 1
• #define EZDP_CONGESTION_STATUS_JOB_GUARANTEE_OFFSET 10
• #define EZDP_CONGESTION_STATUS_JOB_GUARANTEE_MASK (1 <<
EZDP_CONGESTION_STATUS_JOB_GUARANTEE_OFFSET)
• #define EZDP_CONGESTION_STATUS_RESERVED11_SIZE 1
• #define EZDP_CONGESTION_STATUS_RESERVED11_OFFSET 11
• #define EZDP_CONGESTION_STATUS_PORT_CONGESTION_LEVEL_SIZE 2
• #define EZDP_CONGESTION_STATUS_PORT_CONGESTION_LEVEL_OFFSET 12
• #define EZDP_CONGESTION_STATUS_RESERVED14_15_SIZE 2
• #define EZDP_CONGESTION_STATUS_RESERVED14_15_OFFSET 14
• #define EZDP_FLOW_CONTROL_STATUS_CONGESTION_LEVEL_SIZE 3
• #define EZDP_FLOW_CONTROL_STATUS_CONGESTION_LEVEL_OFFSET 0
• #define EZDP_FLOW_CONTROL_STATUS_ENABLE_SIZE 1
• #define EZDP_FLOW_CONTROL_STATUS_ENABLE_OFFSET 3
• #define EZDP_FLOW_CONTROL_STATUS_ENABLE_MASK (1 <<
EZDP_FLOW_CONTROL_STATUS_ENABLE_OFFSET)
• #define EZDP_FLOW_CONTROL_STATUS_RESERVED4_7_SIZE 4
• #define EZDP_FLOW_CONTROL_STATUS_RESERVED4_7_OFFSET 4
• #define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_SIZE 16
• #define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_OFFSET 0
• #define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_WORD_SELECT 0
• #define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_WORD_OFFSET 0
• #define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_SIZE 2
• #define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_OFFSET 16
• #define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_WORD_SELECT 0
• #define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_WORD_OFFSET 16
• #define EZDP_INPUT_QUEUE_STATUS_READY_SIZE 1
• #define EZDP_INPUT_QUEUE_STATUS_READY_OFFSET 18
• #define EZDP_INPUT_QUEUE_STATUS_READY_WORD_SELECT 0
• #define EZDP_INPUT_QUEUE_STATUS_READY_WORD_OFFSET 18
• #define EZDP_INPUT_QUEUE_STATUS_READY_MASK (1 <<
EZDP_INPUT_QUEUE_STATUS_READY_WORD_OFFSET)
• #define EZDP_INPUT_QUEUE_STATUS_RESERVED19_31_SIZE 13
• #define EZDP_INPUT_QUEUE_STATUS_RESERVED19_31_OFFSET 19
• #define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_SIZE 16
• #define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_OFFSET 32
• #define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_WORD_SELECT 1
• #define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_WORD_OFFSET 0
• #define EZDP_INPUT_QUEUE_STATUS_SIZE_SIZE 16
• #define EZDP_INPUT_QUEUE_STATUS_SIZE_OFFSET 48
• #define EZDP_INPUT_QUEUE_STATUS_SIZE_WORD_SELECT 1
• #define EZDP_INPUT_QUEUE_STATUS_SIZE_WORD_OFFSET 16

```

- #define [EZDP\\_INPUT\\_QUEUE\\_STATUS\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_SIZE\\_SIZE](#) 16
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_SIZE\\_OFFSET](#) 0
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_READY\\_SIZE](#) 1
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_READY\\_OFFSET](#) 16
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_READY\\_MASK](#) (1 << EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_OFFSET)
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_CONGESTION\\_SIZE](#) 1
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_CONGESTION\\_OFFSET](#) 17
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_CONGESTION\\_MASK](#) (1 << EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_OFFSET)
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_RESERVED18\\_31\\_SIZE](#) 14
- #define [EZDP\\_OUTPUT\\_QUEUE\\_STATUS\\_RESERVED18\\_31\\_OFFSET](#) 18
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_DISPATCHED\\_JOB\\_SIZE](#) 13
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_DISPATCHED\\_JOB\\_OFFSET](#) 0
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_RESERVED13\\_SIZE](#) 1
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_RESERVED13\\_OFFSET](#) 13
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_BUSY\\_SIZE](#) 1
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_BUSY\\_OFFSET](#) 14
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_BUSY\\_MASK](#) (1 << EZDP\_APP\_SCHLR\_STATUS\_BUSY\_OFFSET)
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_ENABLE\\_SIZE](#) 1
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_ENABLE\\_OFFSET](#) 15
- #define [EZDP\\_APP\\_SCHLR\\_STATUS\\_ENABLE\\_MASK](#) (1 << EZDP\_APP\_SCHLR\_STATUS\_ENABLE\_OFFSET)
- #define [EZDP\\_GROUP\\_SCHLR\\_STATUS\\_DISPATCHED\\_JOB\\_SIZE](#) 13
- #define [EZDP\\_GROUP\\_SCHLR\\_STATUS\\_DISPATCHED\\_JOB\\_OFFSET](#) 0
- #define [EZDP\\_GROUP\\_SCHLR\\_STATUS\\_RESERVED13\\_15\\_SIZE](#) 3
- #define [EZDP\\_GROUP\\_SCHLR\\_STATUS\\_RESERVED13\\_15\\_OFFSET](#) 13
- #define [EZDP\\_JOB\\_CONTAINER\\_CMD\\_DESC\\_JOB\\_ID\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_CONTAINER\\_CMD\\_DESC\\_JOB\\_ID\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_CONTAINER\\_CMD\\_DESC\\_RESERVED0\\_11\\_SIZE](#) 12
- #define [EZDP\\_JOB\\_CONTAINER\\_CMD\\_DESC\\_RESERVED0\\_11\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_CONTAINER\\_CMD\\_DESC\\_COMMAND\\_SIZE](#) 4
- #define [EZDP\\_JOB\\_CONTAINER\\_CMD\\_DESC\\_COMMAND\\_OFFSET](#) 12
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_RESERVED0\\_15\\_SIZE](#) 16
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_RESERVED0\\_15\\_OFFSET](#) 0
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_JOB\\_BUDGET\\_ID\\_SIZE](#) 10
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_JOB\\_BUDGET\\_ID\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_JOB\\_BUDGET\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_JOB\\_BUDGET\\_ID\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_INFO\\_SIZE](#) 3
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_INFO\\_OFFSET](#) 26
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_INFO\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_INFO\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_RESERVED29\\_31\\_SIZE](#) 3
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_RESERVED29\\_31\\_OFFSET](#) 29
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_WORD\\_COUNT](#) 8
- #define [EZDP\\_JOB\\_CONTAINER\\_DESC\\_MAX\\_NUM\\_OF\\_JOBS](#) 7

## Typedefs

- typedef uint32\_t [ezdp\\_job\\_id\\_t](#)
- *Job id struct definition.* typedef uint16\_t [ezdp\\_job\\_queue\\_cmd\\_info\\_t](#)
- typedef uint16\_t [ezdp\\_job\\_transmit\\_cmd\\_info\\_t](#)
- typedef uint16\_t [ezdp\\_job\\_discard\\_cmd\\_info\\_t](#)
- typedef uint16\_t [ezdp\\_congestion\\_status\\_t](#)
- typedef uint8\_t [ezdp\\_flow\\_control\\_status\\_t](#)



- typedef uint32\_t [ezdp\\_output\\_queue\\_status\\_t](#)
- typedef uint16\_t [ezdp\\_app\\_schlr\\_status\\_t](#)
- typedef uint16\_t [ezdp\\_group\\_schlr\\_status\\_t](#)
- typedef uint32\_t [ezdp\\_job\\_container\\_cmd\\_desc\\_t](#)
- typedef uint32\_t [ezdp\\_job\\_container\\_info\\_t](#)

## Enumerations

- enum [ezdp\\_flow\\_control\\_node](#) { [EZDP\\_CHANNEL\\_NODE](#) = 0x0, [EZDP\\_PORT\\_NODE](#) = 0x1, [EZDP\\_GROUP\\_NODE](#) = 0x2, [EZDP\\_GLOBAL\\_NODE](#) = 0x3 }
- Flow control congestion node types.*
- enum [ezdp\\_budget\\_type](#) { [EZDP\\_INT\\_MEM\\_BUF\\_BUDGET](#) = 0x0, [EZDP\\_EXT\\_MEM\\_BUF\\_BUDGET](#) = 0x1, [EZDP\\_JOB\\_BUDGET](#) = 0x2 }
- Flow control budget types.*
- enum [ezdp\\_congestion\\_level](#) { [EZDP\\_LOW\\_LEVEL](#) = 0x0, [EZDP\\_MEDIUM\\_LEVEL](#) = 0x1, [EZDP\\_HIGH\\_LEVEL](#) = 0x2, [EZDP\\_CRITICAL\\_LEVEL](#) = 0x3 }
- congestion level possible values.*
- enum [ezdp\\_tx\\_packet\\_switch\\_mode](#) { [EZDP\\_TOPOLOGY](#) = 0x0, [EZDP\\_FIXED\\_BASE](#) = 0x1, [EZDP\\_EXPLICIT\\_PSID](#) = 0x2 }
- tx packet switch mode possible values.*
- enum [ezdp\\_tx\\_drop\\_mode](#) { [EZDP\\_CAN\\_DROP](#) = 0x0, [EZDP\\_DONT\\_DROP](#) = 0x2, [EZDP\\_NEVER\\_DROP](#) = 0x3 }
- tx drop mode possible values.*
- enum [ezdp\\_job\\_transmit\\_dest](#) { [EZDP\\_TM\\_DEST](#) = EZASM\_JOB\_TRANSMIT\_MODE\_TM, [EZDP\\_INTERFACE\\_DEST](#) = EZASM\_JOB\_TRANSMIT\_MODE\_TM\_BYPASS }
- job transmit dest possible values.*
- enum [ezdp\\_flow\\_control\\_congestion\\_level](#) { [EZDP\\_CONGESTION\\_LEVEL\\_0](#) = 0x0, [EZDP\\_CONGESTION\\_LEVEL\\_1](#) = 0x4, [EZDP\\_CONGESTION\\_LEVEL\\_2](#) = 0x5, [EZDP\\_CONGESTION\\_LEVEL\\_3](#) = 0x6, [EZDP\\_CONGESTION\\_LEVEL\\_4](#) = 0x7 }
- flow control congestion level possible values.*
- enum [ezdp\\_job\\_container\\_cmd](#) { [EZDP\\_TRANSMIT](#) = EZASM\_JOB\_NEXT\_DST\_TRANSMIT, [EZDP\\_FREE](#) = EZASM\_JOB\_NEXT\_DST\_TERMINATE, [EZDP\\_DISCARD](#) = EZASM\_JOB\_NEXT\_DST\_DISCARD, [EZDP\\_QUEUE](#) = EZASM\_JOB\_NEXT\_DST\_PMU, [EZDP\\_QUEUE\\_WITH\\_SEQ\\_NUM](#) = EZASM\_JOB\_NEXT\_DST\_PMU\_WITH\_PQ\_SERVICE }
- job container command type possible values.*
- enum [ezdp\\_job\\_flags](#) { [EZDP\\_ALLOW\\_REORDER](#) = 0x1, [EZDP\\_SET\\_SEQ\\_NUM](#) = 0x2, [EZDP\\_HANDLE\\_NOTICE](#) = 0x4 }
- job flags.*
- enum [ezdp\\_report\\_size](#) { [EZDP\\_8BITS\\_REPORT](#) = EZDP\_DRIVER\_REPORT\_SIZE\_TYPE\_8BITS\_REPORT, [EZDP\\_16BITS\\_REPORT](#) = EZDP\_DRIVER\_REPORT\_SIZE\_TYPE\_16BITS\_REPORT, [EZDP\\_32BITS\\_REPORT](#) = EZDP\_DRIVER\_REPORT\_SIZE\_TYPE\_32BITS\_REPORT }
- report size possible values.*

## Define Documentation

```
#define EZDP_TM_REPORT_WORK_AREA_SIZE  sizeof(struct ezdp_init_tm_report_wa)
```

Work area minimal required size definitions.

```
#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_SEC_SIZE 8

#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_SEC_OFFSET 0

#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_SEC_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_SEC_WORD_OFFSET 0

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_OFFSET 8

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_WORD_OFFSET 8

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_MASK (1 <<
EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_WORD_OFFSET)

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_OFFSET 9

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_WORD_OFFSET 9

#define EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_MASK (1 <<
EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_WORD_OFFSET)

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED10_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED10_OFFSET 10

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED11_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED11_OFFSET 11

#define EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_OFFSET 12

#define EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_WORD_OFFSET 12

#define EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_MASK (1 <<
EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_WORD_OFFSET)

#define EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_OFFSET 13
```

```
#define EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_WORD_OFFSET 13

#define EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_MASK (1 <<
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_WORD_OFFSET)

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED14_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED14_OFFSET 14

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED15_SIZE 1

#define EZDP_JOB_RX_INTERFACE_INFO_RESERVED15_OFFSET 15

#define EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_SIZE 6

#define EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_OFFSET 16

#define EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_WORD_OFFSET 16

#define EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_SIZE 2

#define EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_OFFSET 22

#define EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_WORD_OFFSET 22

#define EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_SIZE 2

#define EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_OFFSET 24

#define EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_WORD_SELECT 0

#define
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_WORD_OFFSET 24

#define EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_SIZE 2

#define EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_OFFSET 26

#define
EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_WORD_SELECT 0

#define
EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_WORD_OFFSET 26

#define EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_SIZE 2

#define EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_OFFSET 28
```



```
#define EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_WORD_SELECT 0

#define EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_WORD_OFFSET 28

#define EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_SIZE 2

#define EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_OFFSET 30

#define
EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_WORD_SELECT 0

#define
EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_WORD_OFFSET 30

#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_SIZE 32

#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_OFFSET 32

#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_WORD_SELECT 1

#define EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_WORD_OFFSET 0

#define EZDP_JOB_RX_INTERFACE_INFO_WORD_COUNT 2

#define EZDP_JOB_RX_LOOPBACK_INFO_RESERVED0_15_SIZE 16

#define EZDP_JOB_RX_LOOPBACK_INFO_RESERVED0_15_OFFSET 0

#define EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_SIZE 16

#define EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_OFFSET 16

#define EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_WORD_SELECT 0

#define EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_WORD_OFFSET 16

#define EZDP_JOB_RX_LOOPBACK_INFO_RESERVED32_63_SIZE 32

#define EZDP_JOB_RX_LOOPBACK_INFO_RESERVED32_63_OFFSET 32

#define EZDP_JOB_RX_LOOPBACK_INFO_WORD_COUNT 2

#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_SEC_SIZE 8

#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_SEC_OFFSET 0

#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_SEC_WORD_SELECT 0

#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_SEC_WORD_OFFSET 0

#define EZDP_JOB_RX_CONFIRMATION_INFO_RESERVED8_31_SIZE 24

#define EZDP_JOB_RX_CONFIRMATION_INFO_RESERVED8_31_OFFSET 8
```

```
#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_NSEC_SIZE 32

#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_NSEC_OFFSET 32

#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_NSEC_WORD_SELECT 1

#define EZDP_JOB_RX_CONFIRMATION_INFO_TIMESTAMP_NSEC_WORD_OFFSET 0

#define EZDP_JOB_RX_CONFIRMATION_INFO_WORD_COUNT 2

#define EZDP_JOB_RX_TIMER_INFO_RESERVED0_8_SIZE 8

#define EZDP_JOB_RX_TIMER_INFO_RESERVED0_8_OFFSET 0

#define EZDP_JOB_RX_TIMER_INFO_TIMER_ID_SIZE 8

#define EZDP_JOB_RX_TIMER_INFO_TIMER_ID_OFFSET 8

#define EZDP_JOB_RX_TIMER_INFO_TIMER_ID_WORD_SELECT 0

#define EZDP_JOB_RX_TIMER_INFO_TIMER_ID_WORD_OFFSET 8

#define EZDP_JOB_RX_TIMER_INFO_RESERVED16_31_SIZE 16

#define EZDP_JOB_RX_TIMER_INFO_RESERVED16_31_OFFSET 16

#define EZDP_JOB_RX_TIMER_INFO_EVENT_ID_SIZE 32

#define EZDP_JOB_RX_TIMER_INFO_EVENT_ID_OFFSET 32

#define EZDP_JOB_RX_TIMER_INFO_EVENT_ID_WORD_SELECT 1

#define EZDP_JOB_RX_TIMER_INFO_EVENT_ID_WORD_OFFSET 0

#define EZDP_JOB_RX_TIMER_INFO_WORD_COUNT 2

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO0_SIZE 32

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO0_OFFSET 0

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO0_WORD_SELECT 0

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO0_WORD_OFFSET 0

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO1_SIZE 32

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO1_OFFSET 32

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO1_WORD_SELECT 1

#define EZDP_JOB_RX_USER_INFO_USER_DATA_INFO1_WORD_OFFSET 0
```

```
#define EZDP_JOB_RX_USER_INFO_WORD_COUNT 2

#define EZDP_JOB_RX_INFO_GROSS_CHECKSUM_SIZE 16

#define EZDP_JOB_RX_INFO_GROSS_CHECKSUM_OFFSET 64

#define EZDP_JOB_RX_INFO_GROSS_CHECKSUM_WORD_SELECT 2

#define EZDP_JOB_RX_INFO_GROSS_CHECKSUM_WORD_OFFSET 0

#define EZDP_JOB_RX_INFO_RESERVED112_127_SIZE 16

#define EZDP_JOB_RX_INFO_RESERVED112_127_OFFSET 80

#define EZDP_JOB_RX_INFO_SOURCE_QUEUE_SIZE 7

#define EZDP_JOB_RX_INFO_SOURCE_QUEUE_OFFSET 96

#define EZDP_JOB_RX_INFO_SOURCE_QUEUE_WORD_SELECT 3

#define EZDP_JOB_RX_INFO_SOURCE_QUEUE_WORD_OFFSET 0

#define EZDP_JOB_RX_INFO_SIDE_SIZE 1

#define EZDP_JOB_RX_INFO_SIDE_OFFSET 103

#define EZDP_JOB_RX_INFO_SIDE_WORD_SELECT 3

#define EZDP_JOB_RX_INFO_SIDE_WORD_OFFSET 7

#define EZDP_JOB_RX_INFO_SIDE_MASK (1 << EZDP_JOB_RX_INFO_SIDE_WORD_OFFSET)

#define EZDP_JOB_RX_INFO_RESERVED104_107_SIZE 4

#define EZDP_JOB_RX_INFO_RESERVED104_107_OFFSET 104

#define EZDP_JOB_RX_INFO_RESERVED108_109_SIZE 2

#define EZDP_JOB_RX_INFO_RESERVED108_109_OFFSET 108

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_SIZE 1

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_OFFSET 110

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_WORD_SELECT 3

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_WORD_OFFSET 14

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_MASK (1 <<
EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_WORD_OFFSET)

#define EZDP_JOB_RX_INFO_IS_SERVICE_READY_SIZE 1
```

```
#define EZDP_JOB_RX_INFO_IS_SERVICE_READY_OFFSET 111

#define EZDP_JOB_RX_INFO_IS_SERVICE_READY_WORD_SELECT 3

#define EZDP_JOB_RX_INFO_IS_SERVICE_READY_WORD_OFFSET 15

#define EZDP_JOB_RX_INFO_IS_SERVICE_READY_MASK (1 <<
EZDP_JOB_RX_INFO_IS_SERVICE_READY_WORD_OFFSET)

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_SIZE 16

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_OFFSET 112

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_WORD_SELECT 3

#define EZDP_JOB_RX_INFO_SEQ_NUMBER_WORD_OFFSET 16

#define EZDP_JOB_RX_INFO_WORD_COUNT 4

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_SIZE 9

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_OFFSET 16

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_WORD_SELECT 0

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_WORD_OFFSET 16

#define EZDP_JOB_TX_INFO_RESERVED25_29_SIZE 4

#define EZDP_JOB_TX_INFO_RESERVED25_29_OFFSET 25

#define EZDP_JOB_TX_INFO_WRED_COLOR_SIZE 3

#define EZDP_JOB_TX_INFO_WRED_COLOR_OFFSET 29

#define EZDP_JOB_TX_INFO_WRED_COLOR_WORD_SELECT 0

#define EZDP_JOB_TX_INFO_WRED_COLOR_WORD_OFFSET 29

#define EZDP_JOB_TX_INFO_FLOW_ID_SIZE 19

#define EZDP_JOB_TX_INFO_FLOW_ID_OFFSET 32

#define EZDP_JOB_TX_INFO_FLOW_ID_WORD_SELECT 1

#define EZDP_JOB_TX_INFO_FLOW_ID_WORD_OFFSET 0

#define EZDP_JOB_TX_INFO_SIDE_SIZE 1

#define EZDP_JOB_TX_INFO_SIDE_OFFSET 51

#define EZDP_JOB_TX_INFO_SIDE_WORD_SELECT 1
```

```
#define EZDP_JOB_TX_INFO_SIDE_WORD_OFFSET 19

#define EZDP_JOB_TX_INFO_SIDE_MASK (1 << EZDP_JOB_TX_INFO_SIDE_WORD_OFFSET)

#define EZDP_JOB_TX_INFO_RESERVED52_55_SIZE 4

#define EZDP_JOB_TX_INFO_RESERVED52_55_OFFSET 52

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_SIZE 3

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_OFFSET 56

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_WORD_SELECT 1

#define EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_WORD_OFFSET 24

#define EZDP_JOB_TX_INFO_RESERVED59_SIZE 1

#define EZDP_JOB_TX_INFO_RESERVED59_OFFSET 59

#define EZDP_JOB_TX_INFO_QOS_BYPASS_SIZE 1

#define EZDP_JOB_TX_INFO_QOS_BYPASS_OFFSET 60

#define EZDP_JOB_TX_INFO_QOS_BYPASS_WORD_SELECT 1

#define EZDP_JOB_TX_INFO_QOS_BYPASS_WORD_OFFSET 28

#define EZDP_JOB_TX_INFO_QOS_BYPASS_MASK (1 <<
EZDP_JOB_TX_INFO_QOS_BYPASS_WORD_OFFSET)

#define EZDP_JOB_TX_INFO_RESERVED61_SIZE 1

#define EZDP_JOB_TX_INFO_RESERVED61_OFFSET 61

#define EZDP_JOB_TX_INFO_DROP_MODE_SIZE 2

#define EZDP_JOB_TX_INFO_DROP_MODE_OFFSET 62

#define EZDP_JOB_TX_INFO_DROP_MODE_WORD_SELECT 1

#define EZDP_JOB_TX_INFO_DROP_MODE_WORD_OFFSET 30

#define EZDP_JOB_TX_INFO_STAT_STREAM_ID_SIZE 24

#define EZDP_JOB_TX_INFO_STAT_STREAM_ID_OFFSET 64

#define EZDP_JOB_TX_INFO_STAT_STREAM_ID_WORD_SELECT 2

#define EZDP_JOB_TX_INFO_STAT_STREAM_ID_WORD_OFFSET 0

#define EZDP_JOB_TX_INFO_RESERVED88_90_SIZE 2
```

```
#define EZDP_JOB_TX_INFO_RESERVED88_90_OFFSET 88

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_SIZE 3

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_OFFSET 90

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_WORD_SELECT 2

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_WORD_OFFSET 26

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_SIZE 3

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_OFFSET 93

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_WORD_SELECT 2

#define EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_WORD_OFFSET 29

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_SIZE 5

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_OFFSET 96

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_WORD_SELECT 3

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_WORD_OFFSET 0

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_SIZE 1

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_OFFSET 101

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_WORD_SELECT 3

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_WORD_OFFSET 5

#define EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_MASK (1 <<
EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_WORD_OFFSET)

#define EZDP_JOB_TX_INFO_RESERVED102_103_SIZE 2

#define EZDP_JOB_TX_INFO_RESERVED102_103_OFFSET 102

#define EZDP_JOB_TX_INFO_WRED_CLASS_SCALE_PROFILE_SIZE 8

#define EZDP_JOB_TX_INFO_WRED_CLASS_SCALE_PROFILE_OFFSET 104

#define EZDP_JOB_TX_INFO_WRED_CLASS_SCALE_PROFILE_WORD_SELECT 3

#define EZDP_JOB_TX_INFO_WRED_CLASS_SCALE_PROFILE_WORD_OFFSET 8

#define EZDP_JOB_TX_INFO_WRED_FLOW_SCALE_PROFILE_SIZE 8

#define EZDP_JOB_TX_INFO_WRED_FLOW_SCALE_PROFILE_OFFSET 112
```

```
#define EZDP_JOB_TX_INFO_WRED_FLOW_SCALE_PROFILE_WORD_SELECT 3

#define EZDP_JOB_TX_INFO_WRED_FLOW_SCALE_PROFILE_WORD_OFFSET 16

#define EZDP_JOB_TX_INFO_WRED_CLASS_TEMPLATE_PROFILE_SIZE 4

#define EZDP_JOB_TX_INFO_WRED_CLASS_TEMPLATE_PROFILE_OFFSET 120

#define EZDP_JOB_TX_INFO_WRED_CLASS_TEMPLATE_PROFILE_WORD_SELECT 3

#define EZDP_JOB_TX_INFO_WRED_CLASS_TEMPLATE_PROFILE_WORD_OFFSET 24

#define EZDP_JOB_TX_INFO_WRED_FLOW_TEMPLATE_PROFILE_SIZE 4

#define EZDP_JOB_TX_INFO_WRED_FLOW_TEMPLATE_PROFILE_OFFSET 124

#define EZDP_JOB_TX_INFO_WRED_FLOW_TEMPLATE_PROFILE_WORD_SELECT 3

#define EZDP_JOB_TX_INFO_WRED_FLOW_TEMPLATE_PROFILE_WORD_OFFSET 28

#define EZDP_JOB_TX_INFO_WORD_COUNT 4

#define EZDP_JOB_QUEUE_CMD_INFO_TARGET_QUEUE_SIZE 7

#define EZDP_JOB_QUEUE_CMD_INFO_TARGET_QUEUE_OFFSET 0

#define EZDP_JOB_QUEUE_CMD_INFO_SIDE_SIZE 1

#define EZDP_JOB_QUEUE_CMD_INFO_SIDE_OFFSET 7

#define EZDP_JOB_QUEUE_CMD_INFO_SIDE_MASK (1 <<
EZDP_JOB_QUEUE_CMD_INFO_SIDE_OFFSET)

#define EZDP_JOB_QUEUE_CMD_INFO_RESERVED8_15_SIZE 8

#define EZDP_JOB_QUEUE_CMD_INFO_RESERVED8_15_OFFSET 8

#define EZDP_JOB_TRANSMIT_CMD_INFO_OUTPUT_CHANNEL_SIZE 10

#define EZDP_JOB_TRANSMIT_CMD_INFO_OUTPUT_CHANNEL_OFFSET 0

#define EZDP_JOB_TRANSMIT_CMD_INFO_SIDE_SIZE 1

#define EZDP_JOB_TRANSMIT_CMD_INFO_SIDE_OFFSET 10

#define EZDP_JOB_TRANSMIT_CMD_INFO_SIDE_MASK (1 <<
EZDP_JOB_TRANSMIT_CMD_INFO_SIDE_OFFSET)

#define EZDP_JOB_TRANSMIT_CMD_INFO_DESTINATION_SIZE 1

#define EZDP_JOB_TRANSMIT_CMD_INFO_DESTINATION_OFFSET 11
```

```
#define EZDP_JOB_TRANSMIT_CMD_INFO_DESTINATION_MASK (1 <<
EZDP_JOB_TRANSMIT_CMD_INFO_DESTINATION_OFFSET)

#define EZDP_JOB_TRANSMIT_CMD_INFO_RESERVED12_15_SIZE 4

#define EZDP_JOB_TRANSMIT_CMD_INFO_RESERVED12_15_OFFSET 12

#define EZDP_JOB_DISCARD_CMD_INFO_RESERVED0_9_SIZE 10

#define EZDP_JOB_DISCARD_CMD_INFO_RESERVED0_9_OFFSET 0

#define EZDP_JOB_DISCARD_CMD_INFO_SIDE_SIZE 1

#define EZDP_JOB_DISCARD_CMD_INFO_SIDE_OFFSET 10

#define EZDP_JOB_DISCARD_CMD_INFO_SIDE_MASK (1 <<
EZDP_JOB_DISCARD_CMD_INFO_SIDE_OFFSET)

#define EZDP_JOB_DISCARD_CMD_INFO_RESERVED11_15_SIZE 5

#define EZDP_JOB_DISCARD_CMD_INFO_RESERVED11_15_OFFSET 11

#define EZDP_CONGESTION_STATUS_IMEM_BUF_CONGESTION_LEVEL_SIZE 2

#define EZDP_CONGESTION_STATUS_IMEM_BUF_CONGESTION_LEVEL_OFFSET 0

#define EZDP_CONGESTION_STATUS_IMEM_BUF_GUARANTEE_SIZE 1

#define EZDP_CONGESTION_STATUS_IMEM_BUF_GUARANTEE_OFFSET 2

#define EZDP_CONGESTION_STATUS_IMEM_BUF_GUARANTEE_MASK (1 <<
EZDP_CONGESTION_STATUS_IMEM_BUF_GUARANTEE_OFFSET)

#define EZDP_CONGESTION_STATUS_RESERVED3_SIZE 1

#define EZDP_CONGESTION_STATUS_RESERVED3_OFFSET 3

#define EZDP_CONGESTION_STATUS_EMEM_BUF_CONGESTION_LEVEL_SIZE 2

#define EZDP_CONGESTION_STATUS_EMEM_BUF_CONGESTION_LEVEL_OFFSET 4

#define EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_SIZE 1

#define EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_OFFSET 6

#define EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_MASK (1 <<
EZDP_CONGESTION_STATUS_EMEM_BUF_GUARANTEE_OFFSET)

#define EZDP_CONGESTION_STATUS_RESERVED7_SIZE 1

#define EZDP_CONGESTION_STATUS_RESERVED7_OFFSET 7

#define EZDP_CONGESTION_STATUS_JOB_CONGESTION_LEVEL_SIZE 2
```



```
#define EZDP_CONGESTION_STATUS_JOB_CONGESTION_LEVEL_OFFSET 8

#define EZDP_CONGESTION_STATUS_JOB_GUARANTEE_SIZE 1

#define EZDP_CONGESTION_STATUS_JOB_GUARANTEE_OFFSET 10

#define EZDP_CONGESTION_STATUS_JOB_GUARANTEE_MASK (1 <<
EZDP_CONGESTION_STATUS_JOB_GUARANTEE_OFFSET)

#define EZDP_CONGESTION_STATUS_RESERVED11_SIZE 1

#define EZDP_CONGESTION_STATUS_RESERVED11_OFFSET 11

#define EZDP_CONGESTION_STATUS_PORT_CONGESTION_LEVEL_SIZE 2

#define EZDP_CONGESTION_STATUS_PORT_CONGESTION_LEVEL_OFFSET 12

#define EZDP_CONGESTION_STATUS_RESERVED14_15_SIZE 2

#define EZDP_CONGESTION_STATUS_RESERVED14_15_OFFSET 14

#define EZDP_FLOW_CONTROL_STATUS_CONGESTION_LEVEL_SIZE 3

#define EZDP_FLOW_CONTROL_STATUS_CONGESTION_LEVEL_OFFSET 0

#define EZDP_FLOW_CONTROL_STATUS_ENABLE_SIZE 1

#define EZDP_FLOW_CONTROL_STATUS_ENABLE_OFFSET 3

#define EZDP_FLOW_CONTROL_STATUS_ENABLE_MASK (1 <<
EZDP_FLOW_CONTROL_STATUS_ENABLE_OFFSET)

#define EZDP_FLOW_CONTROL_STATUS_RESERVED4_7_SIZE 4

#define EZDP_FLOW_CONTROL_STATUS_RESERVED4_7_OFFSET 4

#define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_SIZE 16

#define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_OFFSET 0

#define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_WORD_SELECT 0

#define EZDP_INPUT_QUEUE_STATUS_DISPATCHED_JOB_WORD_OFFSET 0

#define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_SIZE 2

#define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_OFFSET 16

#define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_WORD_SELECT 0

#define EZDP_INPUT_QUEUE_STATUS_CONGESTION_LEVEL_WORD_OFFSET 16

#define EZDP_INPUT_QUEUE_STATUS_READY_SIZE 1
```

```
#define EZDP_INPUT_QUEUE_STATUS_READY_OFFSET 18

#define EZDP_INPUT_QUEUE_STATUS_READY_WORD_SELECT 0

#define EZDP_INPUT_QUEUE_STATUS_READY_WORD_OFFSET 18

#define EZDP_INPUT_QUEUE_STATUS_READY_MASK (1 <<
EZDP_INPUT_QUEUE_STATUS_READY_WORD_OFFSET)

#define EZDP_INPUT_QUEUE_STATUS_RESERVED19_31_SIZE 13

#define EZDP_INPUT_QUEUE_STATUS_RESERVED19_31_OFFSET 19

#define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_SIZE 16

#define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_OFFSET 32

#define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_WORD_SELECT 1

#define EZDP_INPUT_QUEUE_STATUS_OUTSTANDING_JOB_WORD_OFFSET 0

#define EZDP_INPUT_QUEUE_STATUS_SIZE_SIZE 16

#define EZDP_INPUT_QUEUE_STATUS_SIZE_OFFSET 48

#define EZDP_INPUT_QUEUE_STATUS_SIZE_WORD_SELECT 1

#define EZDP_INPUT_QUEUE_STATUS_SIZE_WORD_OFFSET 16

#define EZDP_INPUT_QUEUE_STATUS_WORD_COUNT 2

#define EZDP_OUTPUT_QUEUE_STATUS_SIZE_SIZE 16

#define EZDP_OUTPUT_QUEUE_STATUS_SIZE_OFFSET 0

#define EZDP_OUTPUT_QUEUE_STATUS_READY_SIZE 1

#define EZDP_OUTPUT_QUEUE_STATUS_READY_OFFSET 16

#define EZDP_OUTPUT_QUEUE_STATUS_READY_MASK (1 <<
EZDP_OUTPUT_QUEUE_STATUS_READY_OFFSET)

#define EZDP_OUTPUT_QUEUE_STATUS_CONGESTION_SIZE 1

#define EZDP_OUTPUT_QUEUE_STATUS_CONGESTION_OFFSET 17

#define EZDP_OUTPUT_QUEUE_STATUS_CONGESTION_MASK (1 <<
EZDP_OUTPUT_QUEUE_STATUS_CONGESTION_OFFSET)

#define EZDP_OUTPUT_QUEUE_STATUS_RESERVED18_31_SIZE 14

#define EZDP_OUTPUT_QUEUE_STATUS_RESERVED18_31_OFFSET 18
```

```
#define EZDP_APP_SCHLR_STATUS_DISPATCHED_JOB_SIZE 13

#define EZDP_APP_SCHLR_STATUS_DISPATCHED_JOB_OFFSET 0

#define EZDP_APP_SCHLR_STATUS_RESERVED13_SIZE 1

#define EZDP_APP_SCHLR_STATUS_RESERVED13_OFFSET 13

#define EZDP_APP_SCHLR_STATUS_BUSY_SIZE 1

#define EZDP_APP_SCHLR_STATUS_BUSY_OFFSET 14

#define EZDP_APP_SCHLR_STATUS_BUSY_MASK (1 <<
EZDP_APP_SCHLR_STATUS_BUSY_OFFSET)

#define EZDP_APP_SCHLR_STATUS_ENABLE_SIZE 1

#define EZDP_APP_SCHLR_STATUS_ENABLE_OFFSET 15

#define EZDP_APP_SCHLR_STATUS_ENABLE_MASK (1 <<
EZDP_APP_SCHLR_STATUS_ENABLE_OFFSET)

#define EZDP_GROUP_SCHLR_STATUS_DISPATCHED_JOB_SIZE 13

#define EZDP_GROUP_SCHLR_STATUS_DISPATCHED_JOB_OFFSET 0

#define EZDP_GROUP_SCHLR_STATUS_RESERVED13_15_SIZE 3

#define EZDP_GROUP_SCHLR_STATUS_RESERVED13_15_OFFSET 13

#define EZDP_JOB_CONTAINER_CMD_DESC_JOB_ID_SIZE 16

#define EZDP_JOB_CONTAINER_CMD_DESC_JOB_ID_OFFSET 16

#define EZDP_JOB_CONTAINER_CMD_DESC_RESERVED0_11_SIZE 12

#define EZDP_JOB_CONTAINER_CMD_DESC_RESERVED0_11_OFFSET 0

#define EZDP_JOB_CONTAINER_CMD_DESC_COMMAND_SIZE 4

#define EZDP_JOB_CONTAINER_CMD_DESC_COMMAND_OFFSET 12

#define EZDP_JOB_CONTAINER_DESC_RESERVED0_15_SIZE 16

#define EZDP_JOB_CONTAINER_DESC_RESERVED0_15_OFFSET 0

#define EZDP_JOB_CONTAINER_DESC_JOB_BUDGET_ID_SIZE 10

#define EZDP_JOB_CONTAINER_DESC_JOB_BUDGET_ID_OFFSET 16

#define EZDP_JOB_CONTAINER_DESC_JOB_BUDGET_ID_WORD_SELECT 0

#define EZDP_JOB_CONTAINER_DESC_JOB_BUDGET_ID_WORD_OFFSET 16
```

```
#define EZDP_JOB_CONTAINER_DESC_INFO_SIZE 3

#define EZDP_JOB_CONTAINER_DESC_INFO_OFFSET 26

#define EZDP_JOB_CONTAINER_DESC_INFO_WORD_SELECT 0

#define EZDP_JOB_CONTAINER_DESC_INFO_WORD_OFFSET 26

#define EZDP_JOB_CONTAINER_DESC_RESERVED29_31_SIZE 3

#define EZDP_JOB_CONTAINER_DESC_RESERVED29_31_OFFSET 29

#define EZDP_JOB_CONTAINER_DESC_WORD_COUNT 8

#define EZDP_JOB_CONTAINER_DESC_MAX_NUM_OF_JOBS 7
```

---

## Typedef Documentation

typedef uint32\_t [ezdp\\_job\\_id\\_t](#)

Job id struct definition.

typedef uint16\_t [ezdp\\_job\\_queue\\_cmd\\_info\\_t](#)

typedef uint16\_t [ezdp\\_job\\_transmit\\_cmd\\_info\\_t](#)

typedef uint16\_t [ezdp\\_job\\_discard\\_cmd\\_info\\_t](#)

typedef uint16\_t [ezdp\\_congestion\\_status\\_t](#)

typedef uint8\_t [ezdp\\_flow\\_control\\_status\\_t](#)

typedef uint32\_t [ezdp\\_output\\_queue\\_status\\_t](#)

typedef uint16\_t [ezdp\\_app\\_schlr\\_status\\_t](#)

typedef uint16\_t [ezdp\\_group\\_schlr\\_status\\_t](#)

typedef uint32\_t [ezdp\\_job\\_container\\_cmd\\_desc\\_t](#)

typedef uint32\_t [ezdp\\_job\\_container\\_info\\_t](#)

---

## Enumeration Type Documentation

enum [ezdp\\_flow\\_control\\_node](#)

Flow control congestion node types.

**Enumerator:**

***EZDP\_CHANNEL\_NODE*** Lowest level - channel.

Each incoming/outgoing packet is accompanied a channel (AKA source tag). The max number of channel nodes is 256

***EZDP\_PORT\_NODE*** Second level - port.

Usually represent port in the system (physical ports, AUX ports, Loop-back, user defined) The max number of port nodes is 128

***EZDP\_GROUP\_NODE*** Third level - group.

Group can aggregate few port nodes The max number of port nodes is 16

***EZDP\_GLOBAL\_NODE*** The last level - global.

Only 1 global node exist in the system.

#### enum [ezdp\\_budget\\_type](#)

Flow control budget types.

##### Enumerator:

***EZDP\_INT\_MEM\_BUF\_BUDGET*** Internal memory frame buffer budget.

***EZDP\_EXT\_MEM\_BUF\_BUDGET*** External memory frame buffer budget.

***EZDP\_JOB\_BUDGET*** Job budget.

#### enum [ezdp\\_congestion\\_level](#)

congestion level possible values.

##### Enumerator:

***EZDP\_LOW\_LEVEL*** First threshold not crossed.

***EZDP\_MEDIUM\_LEVEL*** Crossing first threshold.

***EZDP\_HIGH\_LEVEL*** Crossing second threshold.

***EZDP\_CRITICAL\_LEVEL*** Crossing max threshold.

#### enum [ezdp\\_tx\\_packet\\_switch\\_mode](#)

tx packet switch mode possible values.

##### Enumerator:

***EZDP\_TOPOLOGY*** Topology based entry selection.

L0/L1 entity selects base table entry according to TM queue topology, 3 lsb of packet\_switch\_id\_select selects offset, and the table entry provides an output channel. Applicable only for TM Mode = FULL.

***EZDP\_FIXED\_BASE*** Preconfigured fixed base entry selection.

TM packet switch id table base is taken from TM configuration and 3 lsb of packet\_switch\_id\_select selects an offset in that entry. This method bypasses TM topological

L0/L1 queue selection when addressing the port switching table. Typically this flow is used by implicit TM loopback flows to one of eight PMU loopback channels (channel IDs 128-143). It can also be used for specific transmission path with eight selectable output channels. Applicable for TM Modes FULL and BYPASS.

***EZDP\_EXPLICIT\_PSID*** Explicit PSID selection.

In this mode, SW puts an explicit PSID in the 16-bit INFO fields carried in the TM packet header. Explicitly selects an entry from the packet switch id table. Applicable for TM Modes FULL and BYPASS.

#### enum [ezdp\\_tx\\_drop\\_mode](#)

tx drop mode possible values.

##### Enumerator:

***EZDP\_CAN\_DROP*** Standard TM drop policy.

The packet can be dropped from TM queues by policer WRED, or by other TM scheduling levels.

***EZDP\_DONT\_DROP*** No Drop Packet.

The packet is protected from getting dropped along TM data paths (either by policer WRED or other TM scheduling levels). The packet could be dropped if the TM queue it belongs to is being flushed through configuration access.

***EZDP\_NEVER\_DROP*** Never Drop Packet (SYNC).

In addition to not being dropped by TM data paths, the NEVER\_DROP packet is also not dropped on a flushed queue (the NEVER\_DROP packet is used to mark the end of the queue flush).

#### enum [ezdp\\_job\\_transmit\\_dest](#)

job transmit dest possible values.

##### Enumerator:

***EZDP\_TM\_DEST*** Forward to TM.

***EZDP\_INTERFACE\_DEST*** Forward directly to output interface.

#### enum [ezdp\\_flow\\_control\\_congestion\\_level](#)

flow control congestion level possible values.

##### Enumerator:

***EZDP\_CONGESTION\_LEVEL\_0*** First threshold not crossed.

***EZDP\_CONGESTION\_LEVEL\_1*** Crossing first threshold.

***EZDP\_CONGESTION\_LEVEL\_2*** Crossing second threshold.

***EZDP\_CONGESTION\_LEVEL\_3*** Crossing third threshold.

***EZDP\_CONGESTION\_LEVEL\_4*** Crossing fourth threshold.

enum [ezdp\\_job\\_container\\_cmd](#)

job container command type possible values.

**Enumerator:**

***EZDP\_TRANSMIT*** Transmit the job either via TM or directly to output channel bypassing the TM.

***EZDP\_FREE*** Recycle a job to the PMU.

NOTE: Any associated frame resources are not recycle.

***EZDP\_DISCARD*** Discard a job.

NOTE: All associated frame resources are recycled.

***EZDP\_QUEUE*** Dispatch the job to another PMU queue.

***EZDP\_QUEUE\_WITH\_SEQ\_NUM*** Dispatch the job to another PMU queue and request target queue sequence numbering service.

enum [ezdp\\_job\\_flags](#)

job flags.

**Enumerator:**

***EZDP\_ALLOW\_REORDER*** Allow jobs to be unordered.

***EZDP\_SET\_SEQ\_NUM*** Request sequence numbering - applicable only when changing the PMU queue.

***EZDP\_HANDLE\_NOTICE*** Allow notice handling, by calling to ezdp\_notice\_handler function, before receiving the job.

enum [ezdp\\_report\\_size](#)

report size possible values.

**Enumerator:**

***EZDP\_8BITS\_REPORT*** 8 bits report size  
***EZDP\_16BITS\_REPORT*** 16 bits report size  
***EZDP\_32BITS\_REPORT*** 32 bits report size



## dpe/dp/include/ezdp\_lock.h File Reference

### Functions

- static \_\_always\_inline uint32\_t [ezdp\\_init\\_spinlock\\_ext\\_addr](#) ([ezdp\\_spinlock\\_t](#) \*spinlock\_ref, struct [ezdp\\_ext\\_addr](#) \*addr)
  - *Initialize resources required for a spin lock.* static \_\_always\_inline uint32\_t [ezdp\\_init\\_spinlock\\_sum\\_addr](#) ([ezdp\\_spinlock\\_t](#) \*spinlock\_ref, [ezdp\\_sum\\_addr\\_t](#) addr)
  - *Initialize resources required for a spin lock.* static \_\_always\_inline uint32\_t [ezdp\\_lock\\_spinlock](#) ([ezdp\\_spinlock\\_t](#) \*spinlock\_ref)
  - *Lock a spin lock.* static \_\_always\_inline uint32\_t [ezdp\\_try\\_lock\\_spinlock](#) ([ezdp\\_spinlock\\_t](#) \*spinlock\_ref)
  - *Lock a spin lock with limited number of attempts.* static \_\_always\_inline uint32\_t [ezdp\\_unlock\\_spinlock](#) ([ezdp\\_spinlock\\_t](#) \*spinlock\_ref)
  - *Release a spin lock which was locked.* static \_\_always\_inline void [ezdp\\_init\\_qlock](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref, [ezdp\\_sum\\_addr\\_t](#) addr, [ezdp\\_mem\\_pool\\_t](#) \*link\_pool)
  - *Initialize queue lock structure.* static \_\_always\_inline [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_destroy\\_qlock](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref)
  - *Get queue lock address.* static \_\_always\_inline [ezdp\\_qlock\\_slot\\_t](#) [ezdp\\_alloc\\_qlock\\_slot](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref)
  - *allocate queue lock slot.* static \_\_always\_inline void [ezdp\\_free\\_qlock\\_slot](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref, [ezdp\\_qlock\\_slot\\_t](#) qlock\_slot)
  - *free queue lock slot.* static \_\_always\_inline bool [ezdp\\_lock\\_qlock](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref, [ezdp\\_qlock\\_slot\\_t](#) qlock\_slot, char \*work\_area\_ptr, uint32\_t work\_area\_size)
  - *Try to lock queue lock.* static \_\_always\_inline bool [ezdp\\_order\\_lock\\_qlock](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref, [ezdp\\_qlock\\_slot\\_t](#) qlock\_slot, uint16\_t \*order\_lock, uint16\_t sequence, char \*work\_area\_ptr, uint32\_t work\_area\_size)
  - *Try to lock queue lock (with order).* static \_\_always\_inline void [ezdp\\_enqueue\\_qlock](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref, [ezdp\\_qlock\\_slot\\_t](#) qlock\_slot, uint8\_t \*data, uint8\_t data\_size, char \*work\_area\_ptr, uint32\_t work\_area\_size)
  - *enqueue data to queue lock.* static \_\_always\_inline bool [ezdp\\_dequeue\\_qlock](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref, uint8\_t \*data, uint8\_t data\_size, char \*work\_area\_ptr, uint32\_t work\_area\_size)
  - *dequeue data from queue lock.* static \_\_always\_inline bool [ezdp\\_try\\_unlock\\_qlock](#) ([ezdp\\_qlock\\_t](#) \*qlock\_ref)
- Try to unlock queue lock.*

### Function Documentation

static \_\_always\_inline uint32\_t [ezdp\\_init\\_spinlock\\_ext\\_addr](#) ([ezdp\\_spinlock\\_t](#) \* spinlock\_ref, struct [ezdp\\_ext\\_addr](#) \* addr) [static]

Initialize resources required for a spin lock.

#### Parameters:

[out] *spinlock\_ref* - Reference to spinlock object  
 [in] *addr* - address of spinlock (1 byte)

#### Returns:

0 - success

static \_\_always\_inline uint32\_t [ezdp\\_init\\_spinlock\\_sum\\_addr](#) ([ezdp\\_spinlock\\_t](#) \* spinlock\_ref, [ezdp\\_sum\\_addr\\_t](#) addr) [static]

Initialize resources required for a spin lock.

#### Parameters:

[out] *spinlock\_ref* - Reference to spinlock object

[in] *addr* - address of spinlock (1 byte)

**Returns:**

0 - success

```
static __always_inline uint32_t ezdp_lock_spinlock (ezdp\_spinlock\_t * spinlock_ref) [static]
```

Lock a spin lock.

Calling thread shall acquire the lock if it is not held by another thread. Otherwise, the thread shall spin until the lock becomes available.

**Parameters:**

[in] *spinlock\_ref* - Reference to spinlock object

**Returns:**

0 - success

```
static __always_inline uint32_t ezdp_try_lock_spinlock (ezdp\_spinlock\_t * spinlock_ref) [static]
```

Lock a spin lock with limited number of attempts.

Calling thread shall acquire the lock if it is not held by another thread. Otherwise, the thread shall spin until the lock becomes available or number of tries reach maximum.

**Parameters:**

[in] *spinlock\_ref* - Reference to spinlock object

**Returns:**

0 - success 1 - try lock num\_of\_tries with no success

```
static __always_inline uint32_t ezdp_unlock_spinlock (ezdp\_spinlock\_t * spinlock_ref) [static]
```

Release a spin lock which was locked.

**Parameters:**

[in] *spinlock\_ref* - Reference to spinlock object

**Returns:**

0 - success 2 - try to unlock twice

```
static __always_inline void ezdp_init_qlock (ezdp\_qlock\_t * qlock_ref, ezdp\_sum\_addr\_t addr, ezdp\_mem\_pool\_t * link_pool) [static]
```

Initialize queue lock structure.

**Parameters:**

[out] *qlock\_ref* - Reference to queue lock object

[in] *addr* - address of queue lock (4B)

[in] *link\_pool* - memory pool for queue links (16B each)

**Returns:**

void

```
static __always_inline ezdp\_sum\_addr\_t ezdp_destroy_qlock (ezdp\_qlock\_t * qlock_ref) [static]
```

Get queue lock address.

**Parameters:**

[in] *qlock\_ref* - Reference to queue lock object

**Returns:**

Address of queue lock

```
static __always_inline ezdp\_qlock\_slot\_t ezdp_alloc_qlock_slot (ezdp\_qlock\_t * qlock_ref)
[static]
```

allocate queue lock slot.

allocate queue lock slot. slot should be validated with `ezdp_is_null_qlock_slot()` before usage.

**Parameters:**

[in] *qlock\_ref* - Reference to queue lock object

**Returns:**

queue lock slot to be used in `ezdp_try_to_lock_qlock()` and [ezdp\\_enqueue\\_qlock\(\)](#)

```
static __always_inline void ezdp_free_qlock_slot (ezdp\_qlock\_t * qlock_ref, ezdp\_qlock\_slot\_t
qlock_slot) [static]
```

free queue lock slot.

**Parameters:**

[in] *qlock\_ref* - Reference to queue lock object

[in] *qlock\_slot* - queue lock slot received from [ezdp\\_alloc\\_qlock\\_slot\(\)](#)

**Returns:**

queue lock slot to be used in `ezdp_try_to_lock_qlock()` and [ezdp\\_enqueue\\_qlock\(\)](#)

```
static __always_inline bool ezdp_lock_qlock (ezdp\_qlock\_t * qlock_ref, ezdp\_qlock\_slot\_t
qlock_slot, char * work_area_ptr, uint32_t work_area_size) [static]
```

Try to lock queue lock.

**Parameters:**

[in] *qlock\_ref* - Reference to queue lock object

[in] *qlock\_slot* - queue lock slot received from [ezdp\\_alloc\\_qlock\\_slot\(\)](#)

[in] *work\_area\_ptr* - work area pointer (temporary memory to be used by the function). The size of the temporary memory is determined by `EZDP_QLOCK_WORK_AREA_SIZE`

[in] *work\_area\_size* - size of work area pointer

**Returns:**

true iff lock queue was locked.

```
static __always_inline bool ezdp_order_lock_qlock (ezdp\_qlock\_t * qlock_ref,
ezdp\_qlock\_slot\_t qlock_slot, uint16_t * order_lock, uint16_t sequence, char *
work_area_ptr, uint32_t work_area_size) [static]
```

Try to lock queue lock (with order).

**Parameters:**

- [in] *qlock\_ref* - Reference to queue lock object
- [in] *qlock\_slot* - queue lock slot received from [ezdp\\_alloc\\_qlock\\_slot\(\)](#)
- [in] *order\_lock* - order lock
- [in] *sequence* - sequence
- [in] *work\_area\_ptr* - work area pointer (temporary memory to be used by the function). The size of the temporary memory is determined by EZDP\_QLOCK\_WORK\_AREA\_SIZE
- [in] *work\_area\_size* - size of work area pointer

**Note:**

The API is experimental

**Returns:**

true if lock queue was locked.

```
static __always_inline void ezdp_enqueue_qlock(ezdp\_qlock\_t * qlock_ref, ezdp\_qlock\_slot\_t
qlock_slot, uint8\_t * data, uint8\_t data_size, char * work_area_ptr, uint32\_t
work_area_size) [static]
```

enqueue data to queue lock.

**Parameters:**

- [in] *qlock\_ref* - Reference to queue lock object
- [in] *qlock\_slot* - queue lock slot received from [ezdp\\_alloc\\_qlock\\_slot\(\)](#)
- [in] *data* - data (up to 12 bytes)
- [in] *data\_size* - data size
- [in] *work\_area\_ptr* - work area pointer (temporary memory to be used by the function). The size of the temporary memory is determined by EZDP\_QLOCK\_WORK\_AREA\_SIZE
- [in] *work\_area\_size* - size of work area pointer

**Returns:**

void

```
static __always_inline bool ezdp_dequeue_qlock(ezdp\_qlock\_t * qlock_ref, uint8\_t * data,
uint8\_t data_size, char * work_area_ptr, uint32\_t work_area_size) [static]
```

dequeue data from queue lock.

**Parameters:**

- [in] *qlock\_ref* - Reference to queue lock object
- [out] *data* - data (up to 12 bytes)
- [in] *data\_size* - data size
- [in] *work\_area\_ptr* - work area pointer (temporary memory to be used by the function). The size of the temporary memory is determined by EZDP\_QLOCK\_WORK\_AREA\_SIZE
- [in] *work\_area\_size* - size of work area pointer

**Returns:**

true iff next data is ready in data out argument.

```
static __always_inline bool ezdp_try_unlock_qlock(ezdp\_qlock\_t * qlock_ref) [static]
```

Try to unlock queue lock.

**Parameters:**

- [in] *qlock\_ref* - Reference to queue lock object

**Returns:**

true if queue lock was unlocked. false if queue is not empty.

## dpe/dp/include/ezdp\_lock\_defs.h File Reference

### Defines

- #define [EZDP\\_QLOCK\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_qlock\_working\_area)
- *Work area minimal required size definitions.* #define [EZDP\\_NULL\\_QLOCK\\_SLOT](#) EZDP\_NULL\_SUM\_ADDR

### Typedefs

- typedef struct ezdp\_spinlock [ezdp\\_spinlock\\_t](#)
  - typedef struct ezdp\_qlock [ezdp\\_qlock\\_t](#)
  - typedef [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_qlock\\_slot\\_t](#)
- 

### Define Documentation

**#define EZDP\_QLOCK\_WORK\_AREA\_SIZE** sizeof(struct ezdp\_qlock\_working\_area)

Work area minimal required size definitions.

**#define EZDP\_NULL\_QLOCK\_SLOT** EZDP\_NULL\_SUM\_ADDR

---

### Typedef Documentation

typedef struct ezdp\_spinlock [ezdp\\_spinlock\\_t](#)

typedef struct ezdp\_qlock [ezdp\\_qlock\\_t](#)

typedef [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_qlock\\_slot\\_t](#)

## dpe/dp/include/ezdp\_math.h File Reference

### Enumerations

- enum [ezdp\\_bit\\_mode](#) { [EZDP\\_BIT\\_MODE\\_VALUE](#) = EZASM\_BM\_BIT\_MODE\_VALUE, [EZDP\\_BIT\\_MODE\\_INVERSE](#) = EZASM\_BM\_BIT\_MODE\_INVERSE, [EZDP\\_BIT\\_MODE\\_FALSE](#) = EZASM\_BM\_BIT\_MODE\_FALSE, [EZDP\\_BIT\\_MODE\\_TRUE](#) = EZASM\_BM\_BIT\_MODE\_TRUE }  
*Bit manipulation mode possible values.*
- enum [ezdp\\_reflect\\_resolution](#) { [EZDP\\_REFLECT\\_RESOLUTION\\_1\\_BYTE](#) = EZASM\_RFLT\_RESOLUTION\_1\_BYTE, [EZDP\\_REFLECT\\_RESOLUTION\\_2\\_BYTE](#) = EZASM\_RFLT\_RESOLUTION\_2\_BYTE, [EZDP\\_REFLECT\\_RESOLUTION\\_4\\_BYTE](#) = EZASM\_RFLT\_RESOLUTION\_4\_BYTE }  
*Bit manipulation mode possible values.*
- enum [ezdp\\_hash\\_base\\_matrix](#) { [EZDP\\_HASH\\_BASE\\_MATRIX\\_HASH\\_BASE\\_MATRIX\\_0](#) = 0x0, [EZDP\\_HASH\\_BASE\\_MATRIX\\_HASH\\_BASE\\_MATRIX\\_1](#) = 0x1 }  
*Hash base matrix possible values.*
- enum [ezdp\\_hash\\_permutation](#) { [EZDP\\_HASH\\_PERMUTATION\\_0](#) = 0x0, [EZDP\\_HASH\\_PERMUTATION\\_1](#) = 0x1, [EZDP\\_HASH\\_PERMUTATION\\_2](#) = 0x2, [EZDP\\_HASH\\_PERMUTATION\\_3](#) = 0x3 }  
*Hash permutation possible values.*

### Functions

- static \_\_always\_inline uint32\_t [ezdp\\_add](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Add selected bits of src1 to selected bits of src2.* static \_\_always\_inline uint32\_t [ezdp\\_sub](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Subtract selected bits of src2 from selected bits of src1.* static \_\_always\_inline uint32\_t [ezdp\\_and](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Perform logical 'AND' between selected bits of src1 and src2.* static \_\_always\_inline uint32\_t [ezdp\\_or](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Perform logical 'OR' between selected bits of src1 and src2.* static \_\_always\_inline uint32\_t [ezdp\\_not](#) (uint32\_t src, uint32\_t src\_pos, uint32\_t size)
- Perform logical 'NOT' between selected bits of src1 and src2.* static \_\_always\_inline uint32\_t [ezdp\\_xor](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Perform logical 'XOR' between selected bits of src1 and src2.* static \_\_always\_inline uint32\_t [ezdp\\_fxor8](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Apply an 8 bit 'folded xor' operation on selected bits of src1 and src2.* static \_\_always\_inline uint32\_t [ezdp\\_fxor16](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Apply a 16 bit 'folded xor' operation on selected bits of src1 and selected bits of src2.* static \_\_always\_inline uint32\_t [ezdp\\_shift\\_left](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Perform a 'shift left' operation on a set of bits in src1, with shift size selected by 5 adjacent bits of src2.* static \_\_always\_inline uint32\_t [ezdp\\_shift\\_right](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos, uint32\_t size)
- Perform a 'shift right' operation on a set of bits in src1, with shift size selected by 5 adjacent bits of src2.* static \_\_always\_inline uint32\_t [ezdp\\_count\\_bits](#) (uint32\_t src, uint32\_t src\_pos, uint32\_t size)
- Count the number of bits with value of '1' in a set of bits in src.* static \_\_always\_inline uint32\_t [ezdp\\_div](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos)
- Divide 8 selected bits of src1 by 4 selected bits of src2.* static \_\_always\_inline uint32\_t [ezdp\\_mod](#) (uint32\_t src1, uint32\_t src2, uint32\_t src1\_pos, uint32\_t src2\_pos)
- Modulus 8 selected bits of src1 by 4 selected bits of src2.* static \_\_always\_inline uint32\_t [ezdp\\_pow\\_of\\_2](#) (uint32\_t exp, uint32\_t dst\_pos)
- Calculate the value of 2<sup>exp</sup> and place into any position in dst.* static \_\_always\_inline uint32\_t [ezdp\\_merge\\_pow\\_of\\_2](#) (uint32\_t src, uint32\_t exp, uint32\_t dst\_pos, uint32\_t size)
- Calculate the value of 2<sup>exp</sup> and merge into any position in src.* static \_\_always\_inline uint32\_t [ezdp\\_set\\_bit](#) (uint32\_t src, uint32\_t idx, uint32\_t dst\_pos, uint32\_t size)

- Set a single bit in *src* to 'one'. static \_\_always\_inline uint32\_t [ezdp\\_clear\\_bit](#) (uint32\_t *src*, uint32\_t *idx*, uint32\_t *dst\_pos*, uint32\_t *size*)
- Clear a single bit in *src* (sets to 'zero'). static \_\_always\_inline uint32\_t [ezdp\\_find\\_first\\_one](#) (uint32\_t *src*, uint32\_t *src\_pos*, uint32\_t *src\_size*)
- Find the position of the first 'one' in the range *src[src\_pos+size-1 : src\_pos]* (from lsb to msb). static \_\_always\_inline uint32\_t [ezdp\\_find\\_first\\_zero](#) (uint32\_t *src*, uint32\_t *src\_pos*, uint32\_t *src\_size*)
- Find the position of the first 'zero' in the range *src[src\_pos+size-1 : src\_pos]* (from lsb to msb). static \_\_always\_inline uint32\_t [ezdp\\_get\\_bitfield](#) (uint32\_t *src*, uint32\_t *dest\_pos*, uint32\_t *src\_pos*, uint32\_t *size*)
- Get a set of adjacent bits from *src* and place into any position in *dst*. static \_\_always\_inline uint32\_t [ezdp\\_merge\\_bitfield](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos*, uint32\_t *src2\_pos*, uint32\_t *size*)
- Get a set of adjacent bits from *src2* and merge into any position in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_get\\_2\\_bitfields](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos1*, uint32\_t *src1\_pos*, uint32\_t *size1*, uint32\_t *dest\_pos2*, uint32\_t *src2\_pos*, uint32\_t *size2*)
- Get 2 sets of adjacent bits from *src1* and *src2* and place into two locations in *dst*. static \_\_always\_inline uint32\_t [ezdp\\_merge\\_2\\_bitfields](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos1*, uint32\_t *src1\_pos*, uint32\_t *size1*, uint32\_t *dest\_pos2*, uint32\_t *src2\_pos*, uint32\_t *size2*)
- Get 2 sets of adjacent bits from *src1* and *src2* and merge into two locations in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_get\\_bit](#) (uint32\_t *src*, uint32\_t *dest\_pos*, uint32\_t *src\_pos*)
- Get any bit from *src* and place in any position in *dst*. static \_\_always\_inline uint32\_t [ezdp\\_merge\\_bit](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos*, uint32\_t *src2\_pos*)
- Get any bit from *src2* and merge it any position in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_get\\_2\\_bits](#) (uint32\_t *src*, uint32\_t *dest\_pos1*, [ezdp\\_bit\\_mode](#) mode1, uint32\_t *src\_pos1*, uint32\_t *dest\_pos2*, [ezdp\\_bit\\_mode](#) mode2, uint32\_t *src\_pos2*)
- Get two separate bits from *src* and place in two separate locations in *dst*. static \_\_always\_inline uint32\_t [ezdp\\_merge\\_2\\_bits](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos1*, [ezdp\\_bit\\_mode](#) mode1, uint32\_t *src2\_pos1*, uint32\_t *dest\_pos2*, [ezdp\\_bit\\_mode](#) mode2, uint32\_t *src2\_pos2*)
- Get two separate bits from *src2* and merge into two separate locations in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_get\\_3\\_bits](#) (uint32\_t *src*, uint32\_t *dest\_pos1*, [ezdp\\_bit\\_mode](#) mode1, uint32\_t *src\_pos1*, uint32\_t *dest\_pos2*, [ezdp\\_bit\\_mode](#) mode2, uint32\_t *src\_pos2*, uint32\_t *dest\_pos3*, [ezdp\\_bit\\_mode](#) mode3, uint32\_t *src\_pos3*)
- Get three separate bits from *src* and place in three separate locations in *dst*. static \_\_always\_inline uint32\_t [ezdp\\_merge\\_3\\_bits](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos1*, [ezdp\\_bit\\_mode](#) mode1, uint32\_t *src2\_pos1*, uint32\_t *dest\_pos2*, [ezdp\\_bit\\_mode](#) mode2, uint32\_t *src2\_pos2*, uint32\_t *dest\_pos3*, [ezdp\\_bit\\_mode](#) mode3, uint32\_t *src2\_pos3*)
- Get three separate bits from *src2* and merge into three separate locations in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_get\\_4\\_bits](#) (uint32\_t *src*, uint32\_t *dest\_pos1*, [ezdp\\_bit\\_mode](#) mode1, uint32\_t *src\_pos1*, uint32\_t *dest\_pos2*, [ezdp\\_bit\\_mode](#) mode2, uint32\_t *src\_pos2*, uint32\_t *dest\_pos3*, [ezdp\\_bit\\_mode](#) mode3, uint32\_t *src\_pos3*, uint32\_t *dest\_pos4*, [ezdp\\_bit\\_mode](#) mode4, uint32\_t *src\_pos4*)
- Get four separate bits from *src* and place in four separate locations in *dst*. static \_\_always\_inline uint32\_t [ezdp\\_merge\\_4\\_bits](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos1*, [ezdp\\_bit\\_mode](#) mode1, uint32\_t *src2\_pos1*, uint32\_t *dest\_pos2*, [ezdp\\_bit\\_mode](#) mode2, uint32\_t *src2\_pos2*, uint32\_t *dest\_pos3*, [ezdp\\_bit\\_mode](#) mode3, uint32\_t *src2\_pos3*, uint32\_t *dest\_pos4*, [ezdp\\_bit\\_mode](#) mode4, uint32\_t *src2\_pos4*)
- Get four separate bits from *src2* and merge into four separate locations in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_combine\\_4\\_bits](#) (uint32\_t *src*, uint32\_t *dest\_pos*, uint32\_t *src\_pos1*, uint32\_t *src\_pos2*, uint32\_t *src\_pos3*, uint32\_t *src\_pos4*)
- Get four separate bits from *src* and place into 4 adjacent bits in *dst*. static \_\_always\_inline uint32\_t [ezdp\\_combine\\_merge\\_4\\_bits](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos*, uint32\_t *src2\_pos1*, uint32\_t *src2\_pos2*, uint32\_t *src2\_pos3*, uint32\_t *src2\_pos4*)
- Get four separate bits from *src2* to merge into 4 adjacent bits in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_split\\_4\\_bits](#) (uint32\_t *src*, uint32\_t *dest\_pos1*, uint32\_t *dest\_pos2*, uint32\_t *dest\_pos3*, uint32\_t *dest\_pos4*, uint32\_t *src\_pos*)
- Get four adjacent bits from *src* and place in four separate positions in destination. static \_\_always\_inline uint32\_t [ezdp\\_split\\_merge\\_4\\_bits](#) (uint32\_t *src1*, uint32\_t *src2*, uint32\_t *dest\_pos1*, uint32\_t *dest\_pos2*, uint32\_t *dest\_pos3*, uint32\_t *dest\_pos4*, uint32\_t *src2\_pos*)
- Get four adjacent bits from *src2* and merge into four separate positions in *src1*. static \_\_always\_inline uint32\_t [ezdp\\_get\\_4\\_bytes](#) (uint32\_t *src1*, uint32\_t *src2*, ezasm\_src\_pos\_index index0, ezasm\_src\_pos\_index index1, ezasm\_src\_pos\_index index2, ezasm\_src\_pos\_index index3)
- Extract any four bytes from *src1* and *src2*. static \_\_always\_inline uint32\_t [ezdp\\_reflect\\_bits](#) (uint32\_t *data*, [ezdp\\_reflect\\_resolution](#) resolution)



- Perform bit swap in resolution of 1, 2 or 4 bytes. static \_\_always\_inline uint32\_t [ezdp\\_hash](#) (uint32\_t src1, uint32\_t src2, uint32\_t hash\_size, uint32\_t input\_size, uint32\_t input\_offset, [ezdp\\_hash\\_base\\_matrix](#) base\_matrix, const [ezdp\\_hash\\_permutation](#) perm)
- General purpose hash function. static \_\_always\_inline uint32\_t [ezdp\\_hash32](#) (uint32\_t src, uint32\_t hash\_size, uint32\_t permut\_id, uint32\_t base\_matrix)
- General purpose hash function for 32-bit input. static \_\_always\_inline uint32\_t [ezdp\\_hash64](#) (uint32\_t src1, uint32\_t src2, uint32\_t hash\_size, uint32\_t permut\_id, uint32\_t base\_matrix)
- General purpose hash function for 64-bit input. static \_\_always\_inline uint32\_t [ezdp\\_bulk\\_hash](#) (uint8\_t \_\_cmem \*data, uint32\_t size)
- General purpose hash function for up to 64-byte input. static \_\_always\_inline uint32\_t [ezdp\\_calc\\_crc16](#) (uint32\_t crc\_value, uint8\_t input\_value, bool input\_value\_bit\_rflt)
- Perform CRC16 calculation. static \_\_always\_inline uint32\_t [ezdp\\_calc\\_crc32](#) (uint32\_t crc\_value, uint8\_t input\_value, bool input\_value\_bit\_rflt)
- Perform CRC32 calculation. static \_\_always\_inline uint32\_t [ezdp\\_add\\_checksum](#) (uint32\_t checksum\_value, uint32\_t add\_value)
- Add value to checksum. static \_\_always\_inline uint32\_t [ezdp\\_sub\\_checksum](#) (uint32\_t checksum\_value, uint32\_t sub\_value)

Subtract value from checksum.

## Enumeration Type Documentation

### enum [ezdp\\_bit\\_mode](#)

Bit manipulation mode possible values.

#### Enumerator:

- EZDP\_BIT\_MODE\_VALUE** Copy value of bit.  
**EZDP\_BIT\_MODE\_INVERSE** Copy inverse of bit.  
**EZDP\_BIT\_MODE\_FALSE** Copy false to bit - Clear bit.  
**EZDP\_BIT\_MODE\_TRUE** Copy true to bit - Set bit.

### enum [ezdp\\_reflect\\_resolution](#)

Bit manipulation mode possible values.

#### Enumerator:

- EZDP\_REFLECT\_RESOLUTION\_1\_BYTE** Reflect 1 byte resolution.  
**EZDP\_REFLECT\_RESOLUTION\_2\_BYTE** Reflect 2 byte resolution.  
**EZDP\_REFLECT\_RESOLUTION\_4\_BYTE** Reflect 4 byte resolution.

### enum [ezdp\\_hash\\_base\\_matrix](#)

Hash base matrix possible values.

#### Enumerator:

- EZDP\_HASH\_BASE\_MATRIX\_HASH\_BASE\_MATRIX\_0** The base matrix for the hashing will be 0.  
**EZDP\_HASH\_BASE\_MATRIX\_HASH\_BASE\_MATRIX\_1** The base matrix for the hashing will be 1.

**enum [ezdp\\_hash\\_permutation](#)**

Hash permutation possible values.

**Enumerator:**

- EZDP\_HASH\_PERMUTATION\_0*** The hashing function will use permutation 0.  
***EZDP\_HASH\_PERMUTATION\_1*** The hashing function will use permutation 1.  
***EZDP\_HASH\_PERMUTATION\_2*** The hashing function will use permutation 2.  
***EZDP\_HASH\_PERMUTATION\_3*** The hashing function will use permutation 3.

**Function Documentation**

**static \_\_always\_inline uint32\_t ezdp\_add (uint32\_t src1,   uint32\_t src2,   uint32\_t src1\_pos,  
uint32\_t src2\_pos,   uint32\_t size) [static]**

Add selected bits of src1 to selected bits of src2.

**Parameters:**

- [in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to add

**Returns:**

The function returns: *src1[src1\_pos+size-1 : src1\_pos] + src2[src2\_pos+size-1 : src2\_pos]*

**static \_\_always\_inline uint32\_t ezdp\_sub (uint32\_t src1,   uint32\_t src2,   uint32\_t src1\_pos,  
uint32\_t src2\_pos,   uint32\_t size) [static]**

Subtract selected bits of src2 from selected bits of src1.

**Parameters:**

- [in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to subtract

**Returns:**

The function returns: *src1[src1\_pos+size-1 : src1\_pos] - src2[src2\_pos+size-1 : src2\_pos]*

**static \_\_always\_inline uint32\_t ezdp\_and (uint32\_t src1,   uint32\_t src2,   uint32\_t src1\_pos,  
uint32\_t src2\_pos,   uint32\_t size) [static]**

Perform logical 'AND' between selected bits of src1 and src2.

**Parameters:**

- [in] *src1* - source 1

[in] *src2* - source 2  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply 'and' to

**Returns:**

The function returns: *src1*[*src1\_pos*+*size*-1 : *src1\_pos*] & *src2*[*src2\_pos*+*size*-1 : *src2\_pos*]

```
static __always_inline uint32_t ezdp_or (uint32_t src1,    uint32_t src2,    uint32_t src1_pos,
uint32_t src2_pos,    uint32_t size) [static]
```

Perform logical 'OR' between selected bits of *src1* and *src2*.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply 'or' to

**Returns:**

The function returns: *src1*[*src1\_pos*+*size*-1 : *src1\_pos*] | *src2*[*src2\_pos*+*size*-1 : *src2\_pos*]

```
static __always_inline uint32_t ezdp_not (uint32_t src,    uint32_t src_pos,    uint32_t size)
[static]
```

Perform logical 'NOT' between selected bits of *src1* and *src2*.

**Parameters:**

[in] *src* - source 2  
 [in] *src\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply 'not' to

**Returns:**

The function returns: ~ *src2*[*src2\_pos*+*size*-1 : *src2\_pos*]

```
static __always_inline uint32_t ezdp_xor (uint32_t src1,    uint32_t src2,    uint32_t src1_pos,
uint32_t src2_pos,    uint32_t size) [static]
```

Perform logical 'XOR' between selected bits of *src1* and *src2*.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply 'xor' to

**Returns:**

The function returns: *src1*[*src1\_pos*+*size*-1 : *src1\_pos*] ^ *src2*[*src2\_pos*+*size*-1 : *src2\_pos*]

```
static __always_inline uint32_t ezdp_fxor8 (uint32_t src1,    uint32_t src2,    uint32_t src1_pos,
uint32_t src2_pos,    uint32_t size) [static]
```

Apply an 8 bit 'folded xor' operation on selected bits of *src1* and *src2*.

The selected bits from sources are padded with zeros to 32-bits.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply operation on

**Returns:**

The function returns: `folded_xor8(src1[src1_pos+size-1 : src1_pos], src2[src2_pos+size-1 : src2_pos])`

```
static __always_inline uint32_t ezdp_fxor16 (uint32_t src1,    uint32_t src2,    uint32_t src1_pos,
uint32_t src2_pos,    uint32_t size) [static]
```

Apply a 16 bit 'folded xor' operation on selected bits of src1 and selected bits of src2.

The selected bits from sources are padded with zeros to 32-bits.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply operation on

**Returns:**

The function returns: `folded_xor16(src1[src1_pos+size-1 : src1_pos], src2[src2_pos+size-1 : src2_pos])`

```
static __always_inline uint32_t ezdp_shift_left (uint32_t src1,    uint32_t src2,    uint32_t
src1_pos,    uint32_t src2_pos,    uint32_t size) [static]
```

Perform a 'shift left' operation on a set of bits in src1, with shift size selected by 5 adjacent bits of src2.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2 (only 4 bits are used)  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply 'shift left' on

**Note:**

only 5 bits of the src2 are used.

**Returns:**

The function returns: `src1[src1_pos+size-1 : src1_pos] << src2[src2_pos+4 : src2_pos]`

```
static __always_inline uint32_t ezdp_shift_right (uint32_t src1,    uint32_t src2,    uint32_t
src1_pos,    uint32_t src2_pos,    uint32_t size) [static]
```

Perform a 'shift right' operation on a set of bits in src1, with shift size selected by 5 adjacent bits of src2.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2 (only 4 bits are used)  
 [in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32  
 [in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32  
 [in] *size* - number of bits to apply 'shift right' on

**Note:**

only 5 bits of the src2 are used.

**Returns:**

The function returns: `src1[src1_pos+size-1 : src1_pos] >> src2[src2_pos+4 : src2_pos]`

```
static __always_inline uint32_t ezdp_count_bits (uint32_t src,    uint32_t src_pos,    uint32_t size)
[static]
```

Count the number of bits with value of '1' in a set of bits in src.

**Parameters:**

[in] *src* - source

[in] *src\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32

[in] *size* - number of bits to count '1' bits in

**Returns:**

The function returns the number of '1' bits in `src2[src2_pos+size-1 : src2_pos]`

```
static __always_inline uint32_t ezdp_div (uint32_t src1,    uint32_t src2,    uint32_t src1_pos,
uint32_t src2_pos) [static]
```

Divide 8 selected bits of src1 by 4 selected bits of src2.

**Parameters:**

[in] *src1* - source 1

[in] *src2* - source 2

[in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32

[in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32

**Returns:**

The function returns: `src1[src1_pos+7 : src1_pos] / src2[src2_pos+3 : src2_pos]`

```
static __always_inline uint32_t ezdp_mod (uint32_t src1,    uint32_t src2,    uint32_t src1_pos,
uint32_t src2_pos) [static]
```

Modulus 8 selected bits of src1 by 4 selected bits of src2.

**Parameters:**

[in] *src1* - source 1

[in] *src2* - source 2

[in] *src1\_pos* - source 1 starting bit position possible values are: 0, 8, 16, 32

[in] *src2\_pos* - source 2 starting bit position possible values are: 0, 8, 16, 32

**Returns:**

The function returns: `src1[src1_pos+7 : src1_pos] % src2[src2_pos+3 : src2_pos]`

```
static __always_inline uint32_t ezdp_pow_of_2 (uint32_t exp,    uint32_t dst_pos) [static]
```

Calculate the value of  $2^{\text{exp}}$  and place into any position in dst.

**Parameters:**

[in] *exp* - exponent value  $\text{exp} < 32$

[in] *dst\_pos* - the starting bit position of the field in the destination

**Returns:**

The function returns: result[31 : 0] = 0; result[ dst\_pos + exp ] = 1;

```
static __always_inline uint32_t ezdp_merge_pow_of_2 (uint32_t src,    uint32_t exp,    uint32_t  
dst_pos,    uint32_t size) [static]
```

Calculate the value of 2^exp and merge into any position in src.

**Parameters:**

[in] *src* - source

[in] *exp* - exponent value exp < size

[in] *dst\_pos* - the starting bit position of the field in the destination 0 <= dst\_pos <= 31

[in] *size* - the size of the field 1 <= size <= 32

**Returns:**

The function returns: result[31 : 0] = src; result[size-1+dst\_pos : dst\_pos] = 0; // the field is clears  
result[dst\_pos + exp] = 1; // the bit is set

```
static __always_inline uint32_t ezdp_set_bit (uint32_t src,    uint32_t idx,    uint32_t dst_pos,  
uint32_t size) [static]
```

Set a single bit in src to 'one'.

**Parameters:**

[in] *src* - source

[in] *idx* - bit to set in the field idx < size

[in] *dst\_pos* - the starting bit position of the field in the destination 0 <= dst\_pos <= 31

[in] *size* - the size of the field 1 <= size <= 32

**Returns:**

The function returns: result[31 : 0] = src; result[dst\_pos + idx] = 1; // the bit is set

```
static __always_inline uint32_t ezdp_clear_bit (uint32_t src,    uint32_t idx,    uint32_t dst_pos,  
uint32_t size) [static]
```

Clear a single bit in src (sets to 'zero').

**Parameters:**

[in] *src* - source

[in] *idx* - bit to clear in the field idx < size

[in] *dst\_pos* - the starting bit position of the field in the destination 0 <= dst\_pos <= 31

[in] *size* - the size of the field 1 <= size <= 32

**Returns:**

The function returns: result[31 : 0] = src; result[dst\_pos + idx] = 0; // the bit is cleared

```
static __always_inline uint32_t ezdp_find_first_one (uint32_t src,    uint32_t src_pos,    uint32_t  
src_size) [static]
```

Find the position of the first 'one' in the range src[src\_pos+size-1 : src\_pos] (from lsb to msb).

**Note:**

The index returned is relative to src\_pos, counting from lsb to msb. If no 'one' is present, -1 is returned.

**Parameters:**

[in] *src* - source  
 [in] *src\_pos* - source starting bit position  
 [in] *src\_size* - number of bits to count from *src*

**Returns:**

The function returns the result calculated in the following way:  $K = \text{find\_first\_one}(\text{src2}[\text{src2\_pos} + \text{size} - 1 : \text{src2\_pos}])$ ; result = {0..0,K};

```
static __always_inline uint32_t ezdp_find_first_zero (uint32_t src,   uint32_t src_pos,   uint32_t src_size) [static]
```

Find the position of the first 'zero' in the range *src*[*src\_pos*+size-1 : *src\_pos*] (from lsb to msb).

**Note:**

The index returned is relative to *src\_pos*, counting from lsb to msb. If no 'zero' is present, -1 is returned.

**Parameters:**

[in] *src* - source  
 [in] *src\_pos* - source starting bit position  
 [in] *src\_size* - number of bits to count from *src*

**Returns:**

The function returns the result calculated in the following way:  $K = \text{find\_first\_zero}(\text{src2}[\text{src2\_pos} + \text{size} - 1 : \text{src2\_pos}])$ ; result = {0..0,K};

```
static __always_inline uint32_t ezdp_get_bitfield (uint32_t src,   uint32_t dest_pos,   uint32_t src_pos,   uint32_t size) [static]
```

Get a set of adjacent bits from *src* and place into any position in *dst*.

**Parameters:**

[in] *src* - source  
 [in] *dest\_pos* - destination starting bit position  
 [in] *src\_pos* - source starting bit position  
 [in] *size* - number of bits to extract

**Returns:**

The function returns the result calculated in the following way: result = 0; result[*dest\_pos*+size-1 : *dest\_pos*] = *src*[*src\_pos*+size-1 : *src\_pos*]

```
static __always_inline uint32_t ezdp_merge_bitfield (uint32_t src1,   uint32_t src2,   uint32_t dest_pos,   uint32_t src2_pos,   uint32_t size) [static]
```

Get a set of adjacent bits from *src2* and merge into any position in *src1*.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *dest\_pos* - destination starting bit position  
 [in] *src2\_pos* - source 2 starting bit position  
 [in] *size* - number of bits to merge

**Returns:**

The function returns the result calculated in the following way: result = *src1*; result[*dest\_pos*+size-1 : *dest\_pos*] = *src2*[*src2\_pos*+size-1 : *src2\_pos*]

```
static __always_inline uint32_t ezdp_get_2_bitfields(uint32_t src1, uint32_t src2, uint32_t
dest_pos1, uint32_t src1_pos, uint32_t size1, uint32_t dest_pos2, uint32_t src2_pos,
uint32_t size2) [static]
```

Get 2 sets of adjacent bits from src1 and src2 and place into two locations in dst.

**Parameters:**

[in] *src1* - source 1  
[in] *src2* - source 2  
[in] *dest\_pos1* - destination starting bit position to put bits from source 1  
[in] *src1\_pos* - source 1 starting bit position  
[in] *size1* - number of bits to extract from src1  
[in] *dest\_pos2* - destination starting bit position to put bits from source 2  
[in] *src2\_pos* - source 2 starting bit position  
[in] *size2* - number of bits to extract from src2

**Returns:**

The function returns the result calculated in the following way: result = 0; result[dest\_pos1+size1-1 : dest\_pos1] = src1[src1\_pos+size1-1 : src1\_pos] result[dest\_pos2+size2-1 : dest\_pos2] = src2[src2\_pos+size1-1 : src2\_pos]

```
static __always_inline uint32_t ezdp_merge_2_bitfields(uint32_t src1, uint32_t src2, uint32_t
dest_pos1, uint32_t src1_pos, uint32_t size1, uint32_t dest_pos2, uint32_t src2_pos,
uint32_t size2) [static]
```

Get 2 sets of adjacent bits from src1 and src2 and merge into two locations in src1.

**Parameters:**

[in] *src1* - source 1  
[in] *src2* - source 2  
[in] *dest\_pos1* - destination starting bit position to put bits from source 1  
[in] *src1\_pos* - source 1 starting bit position  
[in] *size1* - number of bits to extract from src1  
[in] *dest\_pos2* - destination starting bit position to put bits from source 2  
[in] *src2\_pos* - source 2 starting bit position  
[in] *size2* - number of bits to extract from src2

**Returns:**

The function returns the result calculated in the following way: result = src1; result[dest\_pos1+size1-1 : dest\_pos1] = src1[src1\_pos+size1-1 : src1\_pos] result[dest\_pos2+size2-1 : dest\_pos2] = src2[src2\_pos+size1-1 : src2\_pos]

```
static __always_inline uint32_t ezdp_get_bit(uint32_t src, uint32_t dest_pos, uint32_t
src_pos) [static]
```

Get any bit from src and place in any position in dst.

**Parameters:**

[in] *src* - source  
[in] *dest\_pos* - destination bit position  
[in] *src\_pos* - source bit position

**Returns:**

The function returns the result calculated in the following way: result = 0; result[dst\_pos] = src[src\_pos]



```
static __always_inline uint32_t ezdp_merge_bit (uint32_t src1,  uint32_t src2,  uint32_t
dest_pos,  uint32_t src2_pos) [static]
```

Get any bit from src2 and merge it any position in src1.

**Parameters:**

[in] *src1* - source 1  
[in] *src2* - source 2  
[in] *dest\_pos* - destination bit position  
[in] *src2\_pos* - source 2 bit position

**Returns:**

The function returns the result calculated in the following way: result = src1; result[dst\_pos] = src2[src2\_pos]

```
static __always_inline uint32_t ezdp_get_2_bits (uint32_t src,  uint32_t dest_pos1,
ezdp_bit_mode mode1,  uint32_t src_pos1,  uint32_t dest_pos2,  ezdp_bit_mode mode2,
uint32_t src_pos2) [static]
```

Get two separate bits from src and place in two separate locations in dst.

The value of each bit can be copied as is, inverted, or set to either true (1) or false (0).

**Parameters:**

[in] *src* - source  
[in] *dest\_pos1* - destination bit 1 position  
[in] *mode1* - how to copy bit 1 (value, inverse, true, false)  
[in] *src\_pos1* - source bit 1 position  
[in] *dest\_pos2* - destination bit 2 position  
[in] *mode2* - how to copy bit 2 (value, inverse, true, false)  
[in] *src\_pos2* - source bit 2 position

**Returns:**

The function returns the result calculated in the following way: result = 0; result[dest\_pos1] = { src[src\_pos1] or ~ src[src\_pos1] or 1 or 0 } - based on mode1 result[dest\_pos2] = { src[src\_pos2] or ~ src[src\_pos2] or 1 or 0 } - based on mode2

```
static __always_inline uint32_t ezdp_merge_2_bits (uint32_t src1,  uint32_t src2,  uint32_t
dest_pos1,  ezdp_bit_mode mode1,  uint32_t src2_pos1,  uint32_t dest_pos2,
ezdp_bit_mode mode2,  uint32_t src2_pos2) [static]
```

Get two separate bits from src2 and merge into two separate locations in src1.

The value of each bit can be copied as is, inverted, or set to either true (1) or false (0).

**Parameters:**

[in] *src1* - source 1  
[in] *src2* - source 2  
[in] *dest\_pos1* - destination bit 1 position  
[in] *mode1* - how to copy bit 1 (value, inverse, true, false)  
[in] *src2\_pos1* - source 2 bit 1 position  
[in] *dest\_pos2* - destination bit 2 position  
[in] *mode2* - how to copy bit 2 (value, inverse, true, false)  
[in] *src2\_pos2* - source 2 bit 2 position

**Returns:**

The function returns the result calculated in the following way: result = src1; result[dest\_pos1] = { src[src2\_pos1] or ~ src[src2\_pos1] or 1 or 0 } - based on mode1 result[dest\_pos2] = { src[src2\_pos2] or ~ src[src2\_pos2] or 1 or 0 } - based on mode2

```
static __always_inline uint32_t ezdp_get_3_bits(uint32_t src, uint32_t dest_pos1,
ezdp_bit_mode mode1, uint32_t src_pos1, uint32_t dest_pos2, ezdp_bit_mode mode2,
uint32_t src_pos2, uint32_t dest_pos3, ezdp_bit_mode mode3, uint32_t src_pos3)
[static]
```

Get three separate bits from src and place in three separate locations in dst.

The value of each bit can be copied as is, inverted, or set to either true (1) or false (0).

**Parameters:**

[in] *src* - source  
[in] *dest\_pos1* - destination bit1 position  
[in] *mode1* - how to copy bit 1 (value, inverse, true, false)  
[in] *src\_pos1* - source bit 1 position  
[in] *dest\_pos2* - destination bit 2 position  
[in] *mode2* - how to copy bit 2 (value, inverse, true, false)  
[in] *src\_pos2* - source bit 2 position  
[in] *dest\_pos3* - destination bit 3 position  
[in] *mode3* - how to copy bit 3 (value, inverse, true, false)  
[in] *src\_pos3* - source bit 3 position

**Returns:**

The function returns the result calculated in the following way: result = 0; result[dest\_pos1] = { src[src\_pos1] or ~ src[src\_pos1] or 1 or 0 } - based on mode1 result[dest\_pos2] = { src[src\_pos2] or ~ src[src\_pos2] or 1 or 0 } - based on mode2 result[dest\_pos3] = { src[src\_pos3] or ~ src[src\_pos3] or 1 or 0 } - based on mode3

```
static __always_inline uint32_t ezdp_merge_3_bits(uint32_t src1, uint32_t src2, uint32_t
dest_pos1, ezdp_bit_mode mode1, uint32_t src2_pos1, uint32_t dest_pos2,
ezdp_bit_mode mode2, uint32_t src2_pos2, uint32_t dest_pos3, ezdp_bit_mode mode3,
uint32_t src2_pos3) [static]
```

Get three separate bits from src2 and merge into three separate locations in src1.

The value of each bit can be copied as is, inverted, or set to either true (1) or false (0).

**Parameters:**

[in] *src1* - source 1  
[in] *src2* - source 2  
[in] *dest\_pos1* - destination bit 1 position  
[in] *mode1* - how to copy bit 1 (value, inverse, true, false)  
[in] *src2\_pos1* - source 2 bit 1 position  
[in] *dest\_pos2* - destination bit 2 position  
[in] *mode2* - how to copy bit 2 (value, inverse, true, false)  
[in] *src2\_pos2* - source 2 bit 2 position  
[in] *dest\_pos3* - destination bit 3 position  
[in] *mode3* - how to copy bit 3 (value, inverse, true, false)  
[in] *src2\_pos3* - source 2 bit 3 position

**Returns:**

The function returns the result calculated in the following way: result = src1; result[dest\_pos1] = { src[src2\_pos1] or ~ src[src2\_pos1] or 1 or 0 } - based on mode1 result[dest\_pos2] = { src[src2\_pos2] or ~ src[src2\_pos2] or 1 or 0 } - based on mode2 result[dest\_pos3] = { src[src2\_pos3] or ~ src[src2\_pos3] or 1 or 0 } - based on mode3

```
static __always_inline uint32_t ezdp_get_4_bits(uint32_t src, uint32_t dest_pos1,
ezdp_bit_mode mode1, uint32_t src_pos1, uint32_t dest_pos2, ezdp_bit_mode mode2,
uint32_t src_pos2, uint32_t dest_pos3, ezdp_bit_mode mode3, uint32_t src_pos3,
uint32_t dest_pos4, ezdp_bit_mode mode4, uint32_t src_pos4) [static]
```

Get four separate bits from src and place in four separate locations in dst.

The value of each bit can be copied as is, inverted, or set to either true (1) or false (0).

#### Parameters:

[in] *src* - source  
 [in] *dest\_pos1* - destination bit 1 position  
 [in] *mode1* - how to copy bit 1 (value, inverse, true, false)  
 [in] *src\_pos1* - source bit 1 position  
 [in] *dest\_pos2* - destination bit2 position  
 [in] *mode2* - how to copy bit 2 (value, inverse, true, false)  
 [in] *src\_pos2* - source bit 2 position  
 [in] *dest\_pos3* - destination bit3 position  
 [in] *mode3* - how to copy bit 3 (value, inverse, true, false)  
 [in] *src\_pos3* - source bit 3 position  
 [in] *dest\_pos4* - destination bit4 position  
 [in] *mode4* - how to copy bit 4 (value, inverse, true, false)  
 [in] *src\_pos4* - source bit 4 position

#### Returns:

The function return result calculated in the following way: result = 0; result[dest\_pos1] = { src[src\_pos1] or ~ src[src\_pos1] or 1 or 0 } - based on mode1 result[dest\_pos2] = { src[src\_pos2] or ~ src[src\_pos2] or 1 or 0 } - based on mode2 result[dest\_pos3] = { src[src\_pos3] or ~ src[src\_pos3] or 1 or 0 } - based on mode3 result[dest\_pos4] = { src[src\_pos4] or ~ src[src\_pos4] or 1 or 0 } - based on mode4

```
static __always_inline uint32_t ezdp_merge_4_bits(uint32_t src1, uint32_t src2, uint32_t
dest_pos1, ezdp\_bit\_mode mode1, uint32_t src2_pos1, uint32_t dest_pos2,
ezdp\_bit\_mode mode2, uint32_t src2_pos2, uint32_t dest_pos3, ezdp\_bit\_mode mode3,
uint32_t src2_pos3, uint32_t dest_pos4, ezdp\_bit\_mode mode4, uint32_t src2_pos4)
[static]
```

Get four separate bits from src2 and merge into four separate locations in src1.

The value of each bit can be copied as is, inverted, or set to either true (1) or false (0).

#### Parameters:

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *dest\_pos1* - destination bit 1 position  
 [in] *mode1* - how to copy bit 1 (value, inverse, true, false)  
 [in] *src2\_pos1* - source 2 bit1 position  
 [in] *dest\_pos2* - destination bit 2 position  
 [in] *mode2* - how to copy bit 2 (value, inverse, true, false)  
 [in] *src2\_pos2* - source 2 bit 2 position  
 [in] *dest\_pos3* - destination bit 3 position  
 [in] *mode3* - how to copy bit 3 (value, inverse, true, false)  
 [in] *src2\_pos3* - source 2 bit 3 position  
 [in] *dest\_pos4* - destination bit 4 position  
 [in] *mode4* - how to copy bit 4 (value, inverse, true, false)  
 [in] *src2\_pos4* - source 2 bit 4 position

#### Returns:

The function return result calculated in the following way: result = src1; result[dest\_pos1] = { src[src2\_pos1] or ~ src[src2\_pos1] or 1 or 0 } - based on mode1 result[dest\_pos2] = { src[src2\_pos2] or ~ src[src2\_pos2] or 1 or 0 } - based on mode2 result[dest\_pos3] = { src[src2\_pos3] or ~ src[src2\_pos3] or 1 or 0 } - based on mode3 result[dest\_pos4] = { src[src2\_pos4] or ~ src[src2\_pos4] or 1 or 0 } - based on mode4

```
static __always_inline uint32_t ezdp_combine_4_bits(uint32_t src, uint32_t dest_pos,
uint32_t src_pos1, uint32_t src_pos2, uint32_t src_pos3, uint32_t src_pos4) [static]
```

Get four separate bits from src and place into 4 adjacent bits in dst.

**Parameters:**

[in] *src* - source  
 [in] *dest\_pos* - destination bit position  
 [in] *src\_pos1* - source bit 1 position  
 [in] *src\_pos2* - source bit 2 position  
 [in] *src\_pos3* - source bit 3 position  
 [in] *src\_pos4* - source bit 4 position

**Returns:**

The function returns the result calculated in the following way: result = 0; result[dest\_pos] = src[src\_pos1] result[dest\_pos+1] = src[src\_pos2] result[dest\_pos+2] = src[src\_pos3] result[dest\_pos+3] = src[src\_pos4]

```
static __always_inline uint32_t ezdp_combine_merge_4_bits (uint32_t src1,  uint32_t src2,
uint32_t dest_pos,  uint32_t src2_pos1,  uint32_t src2_pos2,  uint32_t src2_pos3,  uint32_t
src2_pos4) [static]
```

Get four separate bits from src2 to merge into 4 adjacent bits in src1.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *dest\_pos* - destination bit position  
 [in] *src2\_pos1* - source 2 bit 1 position  
 [in] *src2\_pos2* - source 2 bit 2 position  
 [in] *src2\_pos3* - source 2 bit 3 position  
 [in] *src2\_pos4* - source 2 bit 4 position

**Returns:**

The function returns the result calculated in the following way: result = src1; result[dest\_pos] = src[src2\_pos1] result[dest\_pos+1] = src[src2\_pos2] result[dest\_pos+2] = src[src2\_pos3] result[dest\_pos+3] = src[src2\_pos4]

```
static __always_inline uint32_t ezdp_split_4_bits (uint32_t src,  uint32_t dest_pos1,  uint32_t
dest_pos2,  uint32_t dest_pos3,  uint32_t dest_pos4,  uint32_t src_pos) [static]
```

Get four adjacent bits from src and place in four separate positions in destination.

**Parameters:**

[in] *src* - source  
 [in] *dest\_pos1* - destination bit 1 position  
 [in] *dest\_pos2* - destination bit 2 position  
 [in] *dest\_pos3* - destination bit 3 position  
 [in] *dest\_pos4* - destination bit 4 position  
 [in] *src\_pos* - source bit position

**Returns:**

The function returns the result calculated in the following way: result = 0; result[dest\_pos1] = src[src\_pos] result[dest\_pos2] = src[src\_pos+1] result[dest\_pos3] = src[src\_pos+2] result[dest\_pos4] = src[src\_pos+3]

```
static __always_inline uint32_t ezdp_split_merge_4_bits (uint32_t src1,  uint32_t src2,
uint32_t dest_pos1,  uint32_t dest_pos2,  uint32_t dest_pos3,  uint32_t dest_pos4,
uint32_t src2_pos) [static]
```

Get four adjacent bits from src2 and merge into four separate positions in src1.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *dest\_pos1* - destination bit1 position  
 [in] *dest\_pos2* - destination bit2 position  
 [in] *dest\_pos3* - destination bit3 position  
 [in] *dest\_pos4* - destination bit4 position  
 [in] *src2\_pos* - source 2 bit position

**Returns:**

The function returns the result calculated in the following way: result = src1; result[dest\_pos1] = src2[src2\_pos1] result[dest\_pos2] = src2[src2\_pos2] result[dest\_pos3] = src2[src2\_pos3] result[dest\_pos4] = src2[src2\_pos4]

```
static __always_inline uint32_t ezdp_get_4_bytes (uint32_t src1,  uint32_t src2,
ezasm_src_pos_index index0,  ezasm_src_pos_index index1,  ezasm_src_pos_index index2,
ezasm_src_pos_index index3) [static]
```

Extract any four bytes from src1 and src2.

Each byte of result register gets the relevant byte from either src1 or src2 as indicated by the index fields.

**Parameters:**

[in] *src1* - source 1  
 [in] *src2* - source 2  
 [in] *index0* - the index of byte 0 in the result register  
 [in] *index1* - the index of byte 1 in the result register  
 [in] *index2* - the index of byte 2 in the result register  
 [in] *index3* - the index of byte 3 in the result register

**Note:**

index3 must be taken from src2 (index3 >= 4).

**Returns:**

The function returns the extracted 4 bytes

```
static __always_inline uint32_t ezdp_reflect_bits (uint32_t data,  ezdp\_reflect\_resolution
resolution) [static]
```

Perform bit swap in resolution of 1, 2 or 4 bytes.

Example: data = 0x00 00 00 01 resolution = 1: Result 0x00 00 00 80 data = 0x00 00 00 01 resolution = 2: Result 0x00 00 80 00 data = 0x00 00 00 01 resolution = 4: Result 0x80 00 00 00

**Note:**

In case of invalid resolution parameter, no reflection will be made.

**Parameters:**

[in] *data* - data for reflection  
 [in] *resolution* - reflection resolution: 1B, 2B or 4B.

**Returns:**

reflected data

```
static __always_inline uint32_t ezdp_hash (uint32_t src1,  uint32_t src2,  uint32_t hash_size,
uint32_t input_size,  uint32_t input_offset,  ezdp\_hash\_base\_matrix base_matrix,  const
ezdp\_hash\_permutation perm) [static]
```

General purpose hash function.

Different hash values can be obtained by using different base matrixes. The hashing will be applied to a window of bytes from the input src1.src2 (concatenated). The window is defined as input\_size bytes, starting from the input\_offset byte (starting the count from the MSB). Example: src1.src2 = [b0, b1, b2, b3].[b4, b5, b6, b7] input\_size = 5 input\_offset = 2 the window will be: [b2, b3, b4, b5, b6]

**Parameters:**

- [in] *src1* - source 1 value (input byte 0 to 3)
- [in] *src2* - source 2 value (input byte 4 to 7)
- [in] *hash\_size* - output hash size in bits 1 <= hash\_size <= 32
- [in] *input\_size* - input size in bytes 1 <= input\_size <= 8
- [in] *input\_offset* - input offset in bytes (starting from the MSB) 0 <= input\_offset <= 3
- [in] *base\_matrix* - base matrix to use
- [in] *perm* - permutation to use

**Returns:**

the hash result

```
static __always_inline uint32_t ezdp_hash32 (uint32_t src,  uint32_t hash_size,  uint32_t
permut_id,  uint32_t base_matrix) [static]
```

General purpose hash function for 32-bit input.

Different hash value can be obtained by using different permutations and base matrixes.

**Parameters:**

- [in] *src* - source value
- [in] *hash\_size* - output hash size
- [in] *permut\_id* - permutation id
- [in] *base\_matrix* - base matrix to use

**Returns:**

32 bit hash value

```
static __always_inline uint32_t ezdp_hash64 (uint32_t src1,  uint32_t src2,  uint32_t hash_size,
uint32_t permut_id,  uint32_t base_matrix) [static]
```

General purpose hash function for 64-bit input.

Different hash value can be obtained by using different permutations and base matrixes.

**Note:**

Input of less than 64 bits should be aligned to lsb of src2

**Parameters:**

- [in] *src1* - source 1 value
- [in] *src2* - source 2 value
- [in] *hash\_size* - output hash size
- [in] *permut\_id* - permutation number
- [in] *base\_matrix* - base matrix to use

**Returns:**

32 bit hash result

```
static __always_inline uint32_t ezdp_bulk_hash (uint8_t __cmem * data,  uint32_t size)
[static]
```

General purpose hash function for up to 64-byte input.

**Parameters:**

[in] *data* - pointer to the data on CMEM  
 [in] *size* - size of the data

**Returns:**

32 bit hash result

```
static __always_inline uint32_t ezdp_calc_crc16 (uint32_t crc_value,  uint8_t input_value,  bool
input_value_bit_rflt) [static]
```

Perform CRC16 calculation.

The calculation is using the standard polynomial with the following coefficients: 0x1021.

**Parameters:**

[in] *crc\_value* - 16 LSB are the initial or previous CRC16 value  
 [in] *input\_value* - next CRC input byte  
 [in] *input\_value\_bit\_rflt* - whether the input value is reflected before the CRC calculation or not

**Returns:**

uint32\_t - the 16 LSB represents the result of the CRC16 calculation

```
static __always_inline uint32_t ezdp_calc_crc32 (uint32_t crc_value,  uint8_t input_value,  bool
input_value_bit_rflt) [static]
```

Perform CRC32 calculation.

The calculation is using the standard polynomial with the following coefficients: 0x04C11DB7.

**Parameters:**

[in] *crc\_value* - initial or previous CRC32 value  
 [in] *input\_value* - next CRC input byte  
 [in] *input\_value\_bit\_rflt* - whether the input value is reflected before the CRC calculation or not

**Returns:**

uint32\_t - the result of the CRC32 calculation

```
static __always_inline uint32_t ezdp_add_checksum (uint32_t checksum_value,  uint32_t
add_value) [static]
```

Add value to checksum.

**Parameters:**

[in] *checksum\_value* - initial or intermediate checksum value  
 [in] *add\_value* - value to be added to checksum

**Note:**

The checksum calculation assumes an even (2 byte aligned) offset. If the pointer is odd, the checksum result should be swapped.

**Returns:**

uint32\_t - 16 bit new checksum result

```
static __always_inline uint32_t ezdp_sub_checksum (uint32_t checksum_value,   uint32_t  
sub_value) [static]
```

Subtract value from checksum.

**Parameters:**

[in] *checksum\_value* - initial or intermediate checksum value

[in] *sub\_value* - value to be subtract from checksum

**Note:**

The checksum calculation assumes an even (2 byte aligned) offset. If the pointer is odd, the checksum result should be swapped.

**Returns:**

uint32\_t - 16 bit new checksum result



## dpe/dp/include/ezdp\_memory.h File Reference

### Functions

- static \_\_always\_inline uint32\_t [ezdp\\_calc\\_checksum\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags)
- Calculate checksum of data on extended address. static \_\_always\_inline uint32\_t [ezdp\\_calc\\_checksum](#) (void \_\_cmem \*ptr, uint32\_t size)
- Calculate checksum on a block of memory in CMEM. static \_\_always\_inline bool [ezdp\\_is\\_null\\_sum\\_addr](#) ([ezdp\\_sum\\_addr\\_t](#) sum\_addr)
- Check if a summarized address is null. static \_\_always\_inline [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_calc\\_sum\\_addr](#) ([ezdp\\_sum\\_addr\\_table\\_desc\\_t](#) addr\_desc, uint32\_t entry\_size, uint32\_t key)
- Calculate summarized address from address descriptor and key. static \_\_always\_inline void [ezdp\\_sum\\_addr\\_to\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*ext\_addr, [ezdp\\_sum\\_addr\\_t](#) sum\_addr, uint32\_t entry\_size)
- Calculate extended address from summarized address. static \_\_always\_inline [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_ext\\_addr\\_to\\_sum\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*ext\_addr, uint32\_t entry\_size)
- Calculate summarized address from extended address. static \_\_always\_inline uint32\_t [ezdp\\_calc\\_sum\\_addr\\_offset](#) (struct [ezdp\\_ext\\_addr](#) \*ext\_addr, uint32\_t entry\_size)
- Calculate offset of a summarized address from extended address. static \_\_always\_inline [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_scramble\\_sum\\_addr](#) ([ezdp\\_sum\\_addr\\_t](#) addr, const uint32\_t entry\_size)
- Scramble given summarized address. static \_\_always\_inline void [ezdp\\_scramble\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \*addr, const uint32\_t entry\_size)

Scramble given extended address.

---

### Function Documentation

static \_\_always\_inline uint32\_t [ezdp\\_calc\\_checksum\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t flags) [static]

Calculate checksum of data on extended address.

#### Parameters:

- [in] *src\_ptr* - pointer in CMEM to the extended address
- [in] *size* - the amount of data to sum in byte
- [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

#### Note:

The checksum calculation assumes an even (2 byte aligned) offset. If the pointer is odd, the checksum result should be swapped.

#### Returns:

uint32\_t - the calculated checksum value

static \_\_always\_inline uint32\_t [ezdp\\_calc\\_checksum](#) (void \_\_cmem \*ptr, uint32\_t size) [static]

Calculate checksum on a block of memory in CMEM.

#### Parameters:

- [in] *ptr* - pointer to CMEM to calculate checksum on
- [in] *size* - number of bytes to sum

**Note:**

The checksum calculation assumes an even (2 byte aligned) offset. If the pointer is odd, the checksum result should be swapped.

**Returns:**

uint32\_t - 16 bit checksum value

```
static __always_inline bool ezdp_is_null_sum_addr (ezdp\_sum\_addr\_t sum_addr) [static]
```

Check if a summarized address is null.

**Parameters:**

[in] *sum\_addr* - summarized address

**Returns:**

bool - true iff the *sum\_addr* == EZDP\_NULL\_SUM\_ADDR

```
static __always_inline ezdp\_sum\_addr\_t ezdp_calc_sum_addr (ezdp\_sum\_addr\_table\_desc\_t addr_desc, uint32_t entry_size, uint32_t key) [static]
```

Calculate summarized address from address descriptor and key.

**Parameters:**

[in] *addr\_desc* - table summarized address descriptor

[in] *entry\_size* - size of a single entry

[in] *key* - index into table

**Returns:**

uint32\_t - the calculated summarized address

```
static __always_inline void ezdp_sum_addr_to_ext_addr (struct ezdp\_ext\_addr * ext_addr, ezdp\_sum\_addr\_t sum_addr, uint32_t entry_size) [static]
```

Calculate extended address from summarized address.

**Parameters:**

[out] *ext\_addr* - pointer to the extended address

[in] *sum\_addr* - summarized address

[in] *entry\_size* - size of entry

**Returns:**

void

```
static __always_inline ezdp\_sum\_addr\_t ezdp_ext_addr_to_sum_addr (struct ezdp\_ext\_addr * ext_addr, uint32_t entry_size) [static]
```

Calculate summarized address from extended address.

**Parameters:**

[in] *ext\_addr* - pointer to extended address

[in] *entry\_size* - size of entry

**Note:**

If the extended address is not a multiple of the entry\_size, call [ezdp\\_calc\\_sum\\_addr\\_offset\(\)](#) to calculate the offset of the summarized address.

**Returns:**

ezdp\_sum\_addr\_t - the calculated summarized address

```
static __always_inline uint32_t ezdp_calc_sum_addr_offset (struct ezdp\_ext\_addr * ext_addr,
uint32_t entry_size) [static]
```

Calculate offset of a summarized address from extended address.

**Parameters:**

[in] *ext\_addr* - pointer to extended address  
[in] *entry\_size* - size of entry

**Returns:**

uint32\_t - the offset of the summarized address

```
static __always_inline ezdp\_sum\_addr\_t ezdp_scramble_sum_addr (ezdp\_sum\_addr\_t addr,
const uint32_t entry_size) [static]
```

Scramble given summarized address.

**Parameters:**

[in] *addr* - summarized address  
[in] *entry\_size* - size of entry (power of 2)

**Note:**

For entry size equal/bigger than 64 or equal/smaller than 8, function will return the same address (no scrambling will be done) For such sizes scrambling DDR memory scatter is not required (done internally by HW )

**Returns:**

scrambled ezdp\_sum\_addr\_t

```
static __always_inline void ezdp_scramble_ext_addr (struct ezdp\_ext\_addr * addr, const
uint32_t entry_size) [static]
```

Scramble given extended address.

**Parameters:**

[in,out] *addr* - extended address  
[in] *entry\_size* - size of entry (power of 2)

**Note:**

For entry size equal/bigger than 64 or equal/smaller than 8, function will return the same address (no scrambling will be done) For such sizes scrambling DDR memory scatter is not required (done internally by HW )

**Returns:**

void

## dpe/dp/include/ezdp\_memory\_defs.h File Reference

### Data Structures

- struct [ezdp\\_ext\\_addr](#)
- *Extended address definition.* struct [ezdp\\_sum\\_addr](#)
- *Summarized Address data structure.* struct [ezdp\\_pci\\_addr](#)
- *PCI Address data structure.* struct [ezdp\\_sum\\_addr\\_table\\_desc](#)
- *Structure definition table entry data structure.* struct [ezdp\\_dual\\_add32\\_result](#)
- *The result of the atomic dual add32 instruction.* struct [ezdp\\_dual\\_add64\\_result](#)

### The result of the atomic dual add64 instruction. Defines

- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_MSB\\_SIZE](#) 4
- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_MSB\\_OFFSET](#) 0
- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_MSB\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_MSB\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_EXT\\_ADDR\\_RESERVED4\\_7\\_SIZE](#) 4
- #define [EZDP\\_EXT\\_ADDR\\_RESERVED4\\_7\\_OFFSET](#) 4
- #define [EZDP\\_EXT\\_ADDR\\_MSID\\_SIZE](#) 5
- #define [EZDP\\_EXT\\_ADDR\\_MSID\\_OFFSET](#) 8
- #define [EZDP\\_EXT\\_ADDR\\_MSID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_EXT\\_ADDR\\_MSID\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_EXT\\_ADDR\\_MEM\\_TYPE\\_SIZE](#) 1
- #define [EZDP\\_EXT\\_ADDR\\_MEM\\_TYPE\\_OFFSET](#) 13
- #define [EZDP\\_EXT\\_ADDR\\_MEM\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_EXT\\_ADDR\\_MEM\\_TYPE\\_WORD\\_OFFSET](#) 13
- #define [EZDP\\_EXT\\_ADDR\\_MEM\\_TYPE\\_MASK](#) (1 << EZDP\_EXT\_ADDR\_MEM\_TYPE\_WORD\_OFFSET)
- #define [EZDP\\_EXT\\_ADDR\\_RESERVED14\\_15\\_SIZE](#) 2
- #define [EZDP\\_EXT\\_ADDR\\_RESERVED14\\_15\\_OFFSET](#) 14
- #define [EZDP\\_EXT\\_ADDR\\_RESERVED16\\_31\\_SIZE](#) 16
- #define [EZDP\\_EXT\\_ADDR\\_RESERVED16\\_31\\_OFFSET](#) 16
- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_SIZE](#) 32
- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_OFFSET](#) 32
- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_EXT\\_ADDR\\_ADDRESS\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_EXT\\_ADDR\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_SUM\\_ADDR\\_ELEMENT\\_INDEX\\_SIZE](#) 27
- #define [EZDP\\_SUM\\_ADDR\\_ELEMENT\\_INDEX\\_OFFSET](#) 0
- #define [EZDP\\_SUM\\_ADDR\\_MSID\\_SIZE](#) 4
- #define [EZDP\\_SUM\\_ADDR\\_MSID\\_OFFSET](#) 27
- #define [EZDP\\_SUM\\_ADDR\\_MEM\\_TYPE\\_SIZE](#) 1
- #define [EZDP\\_SUM\\_ADDR\\_MEM\\_TYPE\\_OFFSET](#) 31
- #define [EZDP\\_SUM\\_ADDR\\_MEM\\_TYPE\\_MASK](#) (1 << EZDP\_SUM\_ADDR\_MEM\_TYPE\_OFFSET)
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_MSB\\_SIZE](#) 4
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_MSB\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_MSB\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_MSB\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_RESERVED4\\_7\\_SIZE](#) 4
- #define [EZDP\\_PCI\\_ADDR\\_RESERVED4\\_7\\_OFFSET](#) 4
- #define [EZDP\\_PCI\\_ADDR\\_MSID\\_SIZE](#) 5
- #define [EZDP\\_PCI\\_ADDR\\_MSID\\_OFFSET](#) 8
- #define [EZDP\\_PCI\\_ADDR\\_MSID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_MSID\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_PCI\\_ADDR\\_MEM\\_TYPE\\_SIZE](#) 1
- #define [EZDP\\_PCI\\_ADDR\\_MEM\\_TYPE\\_OFFSET](#) 13

- #define [EZDP\\_PCI\\_ADDR\\_MEM\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_MEM\\_TYPE\\_WORD\\_OFFSET](#) 13
- #define [EZDP\\_PCI\\_ADDR\\_MEM\\_TYPE\\_MASK](#) (1 << EZDP\_PCI\_ADDR\_MEM\_TYPE\_WORD\_OFFSET)
- #define [EZDP\\_PCI\\_ADDR\\_RESERVED14\\_15\\_SIZE](#) 2
- #define [EZDP\\_PCI\\_ADDR\\_RESERVED14\\_15\\_OFFSET](#) 14
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_SIZE](#) 8
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_OFFSET](#) 16
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_PCI\\_ADDR\\_PHY\\_FUNC\\_SIZE](#) 4
- #define [EZDP\\_PCI\\_ADDR\\_PHY\\_FUNC\\_OFFSET](#) 24
- #define [EZDP\\_PCI\\_ADDR\\_PHY\\_FUNC\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_PHY\\_FUNC\\_WORD\\_OFFSET](#) 24
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_EN\\_SIZE](#) 1
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_EN\\_OFFSET](#) 28
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_EN\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_EN\\_WORD\\_OFFSET](#) 28
- #define [EZDP\\_PCI\\_ADDR\\_VIRT\\_FUNC\\_EN\\_MASK](#) (1 << EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_WORD\_OFFSET)
- #define [EZDP\\_PCI\\_ADDR\\_RESERVED29\\_30\\_SIZE](#) 2
- #define [EZDP\\_PCI\\_ADDR\\_RESERVED29\\_30\\_OFFSET](#) 29
- #define [EZDP\\_PCI\\_ADDR\\_ADDR\\_TYPE\\_SIZE](#) 1
- #define [EZDP\\_PCI\\_ADDR\\_ADDR\\_TYPE\\_OFFSET](#) 31
- #define [EZDP\\_PCI\\_ADDR\\_ADDR\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_ADDR\\_TYPE\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_PCI\\_ADDR\\_ADDR\\_TYPE\\_MASK](#) (1 << EZDP\_PCI\_ADDR\_ADDR\_TYPE\_WORD\_OFFSET)
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_OFFSET](#) 32
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_PCI\\_ADDR\\_ADDRESS\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_ADDR\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_BASE\\_INDEX\\_SIZE](#) 16
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_BASE\\_INDEX\\_OFFSET](#) 0
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_MSID\\_SIZE](#) 4
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_MSID\\_OFFSET](#) 16
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_KEY\\_SIZE\\_SIZE](#) 5
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_KEY\\_SIZE\\_OFFSET](#) 20
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_RESERVED25\\_26\\_SIZE](#) 2
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_RESERVED25\\_26\\_OFFSET](#) 25
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_MEM\\_TYPE\\_SIZE](#) 1
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_MEM\\_TYPE\\_OFFSET](#) 27
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_MEM\\_TYPE\\_MASK](#) (1 << EZDP\_SUM\_ADDR\_TABLE\_DESC\_MEM\_TYPE\_OFFSET)
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_KEY\\_SHUFF\\_BITS\\_SIZE](#) 3
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_KEY\\_SHUFF\\_BITS\\_OFFSET](#) 28
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_KEY\\_SHUFF\\_EN\\_SIZE](#) 1
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_KEY\\_SHUFF\\_EN\\_OFFSET](#) 31
- #define [EZDP\\_SUM\\_ADDR\\_TABLE\\_DESC\\_KEY\\_SHUFF\\_EN\\_MASK](#) (1 << EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_OFFSET)
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE2\\_SIZE](#) 32
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE2\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE2\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE2\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE1\\_SIZE](#) 32
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE1\\_OFFSET](#) 32

- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE1\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_ORIGINAL\\_VALUE1\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_ADD32\\_RESULT\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE2\\_SIZE](#) 64
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE2\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE2\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE2\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE1\\_SIZE](#) 64
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE1\\_OFFSET](#) 64
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE1\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_ORIGINAL\\_VALUE1\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DUAL\\_ADD64\\_RESULT\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_NULL\\_SUM\\_ADDR](#) 0x00000000

## Typedefs

- typedef uint32\_t [ezdp\\_sum\\_addr\\_t](#)
- typedef uint32\_t [ezdp\\_sum\\_addr\\_table\\_desc\\_t](#)

## Enumerations

- enum [ezdp\\_internal\\_mem\\_space](#) { [EZDP\\_PRIVATE\\_DATA](#) = 0x0, [EZDP\\_STACK](#) = 0x1, [EZDP\\_HALF\\_CLUSTER\\_DATA](#) = 0x2, [EZDP\\_HALF\\_CLUSTER\\_CODE](#) = 0x3, [EZDP\\_1\\_CLUSTER\\_DATA](#) = 0x4, [EZDP\\_1\\_CLUSTER\\_CODE](#) = 0x5, [EZDP\\_2\\_CLUSTER\\_DATA](#) = 0x6, [EZDP\\_2\\_CLUSTER\\_CODE](#) = 0x7, [EZDP\\_4\\_CLUSTER\\_DATA](#) = 0x8, [EZDP\\_4\\_CLUSTER\\_CODE](#) = 0x9, [EZDP\\_16\\_CLUSTER\\_DATA](#) = 0xa, [EZDP\\_16\\_CLUSTER\\_CODE](#) = 0xb, [EZDP\\_ALL\\_CLUSTER\\_DATA](#) = 0xc, [EZDP\\_ALL\\_CLUSTER\\_CODE](#) = 0xd, [EZDP\\_ALL\\_CLUSTER\\_IO](#) = 0xe, [EZDP\\_ALL\\_CLUSTER\\_DATA\\_EXT\\_MEM](#) = 0xf }

*Internal memory space types.*

- enum [ezdp\\_mem\\_space\\_type](#) { [EZDP\\_INTERNAL\\_MS](#) = 0x0, [EZDP\\_EXTERNAL\\_MS](#) = 0x1 }

*mem space type possible values.*

- enum [ezdp\\_pci\\_addr\\_type](#) { [EZDP\\_PCI\\_ADDR\\_TYPE\\_UNTRANSLATED](#) = 0x0, [EZDP\\_PCI\\_ADDR\\_TYPE\\_TRANSLATED](#) = 0x1 }

*pci addr type possible values.*

- enum [ezdp\\_dma\\_flags](#) { [EZDP\\_MEMORY\\_FLAG\\_CLASS\\_0](#) = 0, [EZDP\\_MEMORY\\_FLAG\\_CLASS\\_1](#) = [EZDP\\_MEMORY\\_IMP\\_FLAG\\_CLASS\\_BIT\\_0](#), [EZDP\\_MEMORY\\_FLAG\\_CLASS\\_2](#) = [EZDP\\_MEMORY\\_IMP\\_FLAG\\_CLASS\\_BIT\\_1](#), [EZDP\\_MEMORY\\_FLAG\\_CLASS\\_3](#) = [EZDP\\_MEMORY\\_IMP\\_FLAG\\_CLASS\\_BIT\\_0](#) | [EZDP\\_MEMORY\\_IMP\\_FLAG\\_CLASS\\_BIT\\_1](#), [EZDP\\_MEMORY\\_FLAG\\_UNCACHED](#) = [EZDP\\_MEMORY\\_IMP\\_FLAG\\_UNCACHED](#), [EZDP\\_MEMORY\\_FLAG\\_OVERWRITE](#) = [EZDP\\_MEMORY\\_IMP\\_FLAG\\_OVERWRITE](#), [EZDP\\_PCI\\_FLAG\\_NO\\_SNOOP](#) = [EZDP\\_PCI\\_IMP\\_FLAG\\_NO\\_SNOOP](#), [EZDP\\_PCI\\_FLAG\\_RELEX\\_ORDERED](#) = [EZDP\\_PCI\\_IMP\\_FLAG\\_RELEX\\_ORDERED](#), [EZDP\\_PCI\\_FLAG\\_ATU\\_BYPASS](#) = [EZDP\\_PCI\\_IMP\\_FLAG\\_ATU\\_BYPASS](#) }

*dma flags.*

## Define Documentation

```
#define EZDP_EXT_ADDR_ADDRESS_MSB_SIZE 4

#define EZDP_EXT_ADDR_ADDRESS_MSB_OFFSET 0

#define EZDP_EXT_ADDR_ADDRESS_MSB_WORD_SELECT 0

#define EZDP_EXT_ADDR_ADDRESS_MSB_WORD_OFFSET 0

#define EZDP_EXT_ADDR_RESERVED4_7_SIZE 4

#define EZDP_EXT_ADDR_RESERVED4_7_OFFSET 4

#define EZDP_EXT_ADDR_MSID_SIZE 5

#define EZDP_EXT_ADDR_MSID_OFFSET 8

#define EZDP_EXT_ADDR_MSID_WORD_SELECT 0

#define EZDP_EXT_ADDR_MSID_WORD_OFFSET 8

#define EZDP_EXT_ADDR_MEM_TYPE_SIZE 1

#define EZDP_EXT_ADDR_MEM_TYPE_OFFSET 13

#define EZDP_EXT_ADDR_MEM_TYPE_WORD_SELECT 0

#define EZDP_EXT_ADDR_MEM_TYPE_WORD_OFFSET 13

#define EZDP_EXT_ADDR_MEM_TYPE_MASK (1 <<
EZDP_EXT_ADDR_MEM_TYPE_WORD_OFFSET)

#define EZDP_EXT_ADDR_RESERVED14_15_SIZE 2

#define EZDP_EXT_ADDR_RESERVED14_15_OFFSET 14

#define EZDP_EXT_ADDR_RESERVED16_31_SIZE 16

#define EZDP_EXT_ADDR_RESERVED16_31_OFFSET 16

#define EZDP_EXT_ADDR_ADDRESS_SIZE 32

#define EZDP_EXT_ADDR_ADDRESS_OFFSET 32

#define EZDP_EXT_ADDR_ADDRESS_WORD_SELECT 1

#define EZDP_EXT_ADDR_ADDRESS_WORD_OFFSET 0

#define EZDP_EXT_ADDR_WORD_COUNT 2

#define EZDP_SUM_ADDR_ELEMENT_INDEX_SIZE 27
```

```
#define EZDP_SUM_ADDR_ELEMENT_INDEX_OFFSET 0

#define EZDP_SUM_ADDR_MSID_SIZE 4

#define EZDP_SUM_ADDR_MSID_OFFSET 27

#define EZDP_SUM_ADDR_MEM_TYPE_SIZE 1

#define EZDP_SUM_ADDR_MEM_TYPE_OFFSET 31

#define EZDP_SUM_ADDR_MEM_TYPE_MASK (1 << EZDP_SUM_ADDR_MEM_TYPE_OFFSET)

#define EZDP_PCI_ADDR_ADDRESS_MSB_SIZE 4

#define EZDP_PCI_ADDR_ADDRESS_MSB_OFFSET 0

#define EZDP_PCI_ADDR_ADDRESS_MSB_WORD_SELECT 0

#define EZDP_PCI_ADDR_ADDRESS_MSB_WORD_OFFSET 0

#define EZDP_PCI_ADDR_RESERVED4_7_SIZE 4

#define EZDP_PCI_ADDR_RESERVED4_7_OFFSET 4

#define EZDP_PCI_ADDR_MSID_SIZE 5

#define EZDP_PCI_ADDR_MSID_OFFSET 8

#define EZDP_PCI_ADDR_MSID_WORD_SELECT 0

#define EZDP_PCI_ADDR_MSID_WORD_OFFSET 8

#define EZDP_PCI_ADDR_MEM_TYPE_SIZE 1

#define EZDP_PCI_ADDR_MEM_TYPE_OFFSET 13

#define EZDP_PCI_ADDR_MEM_TYPE_WORD_SELECT 0

#define EZDP_PCI_ADDR_MEM_TYPE_WORD_OFFSET 13

#define EZDP_PCI_ADDR_MEM_TYPE_MASK (1 << EZDP_PCI_ADDR_MEM_TYPE_WORD_OFFSET)

#define EZDP_PCI_ADDR_RESERVED14_15_SIZE 2

#define EZDP_PCI_ADDR_RESERVED14_15_OFFSET 14

#define EZDP_PCI_ADDR_VIRT_FUNC_SIZE 8

#define EZDP_PCI_ADDR_VIRT_FUNC_OFFSET 16

#define EZDP_PCI_ADDR_VIRT_FUNC_WORD_SELECT 0
```



```
#define EZDP_PCI_ADDR_VIRT_FUNC_WORD_OFFSET 16

#define EZDP_PCI_ADDR_PHY_FUNC_SIZE 4

#define EZDP_PCI_ADDR_PHY_FUNC_OFFSET 24

#define EZDP_PCI_ADDR_PHY_FUNC_WORD_SELECT 0

#define EZDP_PCI_ADDR_PHY_FUNC_WORD_OFFSET 24

#define EZDP_PCI_ADDR_VIRT_FUNC_EN_SIZE 1

#define EZDP_PCI_ADDR_VIRT_FUNC_EN_OFFSET 28

#define EZDP_PCI_ADDR_VIRT_FUNC_EN_WORD_SELECT 0

#define EZDP_PCI_ADDR_VIRT_FUNC_EN_WORD_OFFSET 28

#define EZDP_PCI_ADDR_VIRT_FUNC_EN_MASK (1 <<
EZDP_PCI_ADDR_VIRT_FUNC_EN_WORD_OFFSET)

#define EZDP_PCI_ADDR_RESERVED29_30_SIZE 2

#define EZDP_PCI_ADDR_RESERVED29_30_OFFSET 29

#define EZDP_PCI_ADDR_ADDR_TYPE_SIZE 1

#define EZDP_PCI_ADDR_ADDR_TYPE_OFFSET 31

#define EZDP_PCI_ADDR_ADDR_TYPE_WORD_SELECT 0

#define EZDP_PCI_ADDR_ADDR_TYPE_WORD_OFFSET 31

#define EZDP_PCI_ADDR_ADDR_TYPE_MASK (1 <<
EZDP_PCI_ADDR_ADDR_TYPE_WORD_OFFSET)

#define EZDP_PCI_ADDR_ADDRESS_SIZE 32

#define EZDP_PCI_ADDR_ADDRESS_OFFSET 32

#define EZDP_PCI_ADDR_ADDRESS_WORD_SELECT 1

#define EZDP_PCI_ADDR_ADDRESS_WORD_OFFSET 0

#define EZDP_PCI_ADDR_WORD_COUNT 2

#define EZDP_SUM_ADDR_TABLE_DESC_BASE_INDEX_SIZE 16

#define EZDP_SUM_ADDR_TABLE_DESC_BASE_INDEX_OFFSET 0

#define EZDP_SUM_ADDR_TABLE_DESC_MSID_SIZE 4

#define EZDP_SUM_ADDR_TABLE_DESC_MSID_OFFSET 16
```

```
#define EZDP_SUM_ADDR_TABLE_DESC_KEY_SIZE_SIZE 5

#define EZDP_SUM_ADDR_TABLE_DESC_KEY_SIZE_OFFSET 20

#define EZDP_SUM_ADDR_TABLE_DESC_RESERVED25_26_SIZE 2

#define EZDP_SUM_ADDR_TABLE_DESC_RESERVED25_26_OFFSET 25

#define EZDP_SUM_ADDR_TABLE_DESC_MEM_TYPE_SIZE 1

#define EZDP_SUM_ADDR_TABLE_DESC_MEM_TYPE_OFFSET 27

#define EZDP_SUM_ADDR_TABLE_DESC_MEM_TYPE_MASK (1 <<
EZDP_SUM_ADDR_TABLE_DESC_MEM_TYPE_OFFSET)

#define EZDP_SUM_ADDR_TABLE_DESC_KEY_SHUFF_BITS_SIZE 3

#define EZDP_SUM_ADDR_TABLE_DESC_KEY_SHUFF_BITS_OFFSET 28

#define EZDP_SUM_ADDR_TABLE_DESC_KEY_SHUFF_EN_SIZE 1

#define EZDP_SUM_ADDR_TABLE_DESC_KEY_SHUFF_EN_OFFSET 31

#define EZDP_SUM_ADDR_TABLE_DESC_KEY_SHUFF_EN_MASK (1 <<
EZDP_SUM_ADDR_TABLE_DESC_KEY_SHUFF_EN_OFFSET)

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE2_SIZE 32

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE2_OFFSET 0

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE2_WORD_SELECT 0

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE2_WORD_OFFSET 0

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE1_SIZE 32

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE1_OFFSET 32

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE1_WORD_SELECT 1

#define EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALUE1_WORD_OFFSET 0

#define EZDP_DUAL_ADD32_RESULT_WORD_COUNT 2

#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE2_SIZE 64

#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE2_OFFSET 0

#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE2_WORD_SELECT 0

#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE2_WORD_OFFSET 0

#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE1_SIZE 64
```

```
#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE1_OFFSET 64

#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE1_WORD_SELECT 2

#define EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALUE1_WORD_OFFSET 0

#define EZDP_DUAL_ADD64_RESULT_WORD_COUNT 4

#define EZDP_NULL_SUM_ADDR 0x00000000
```

---

## Typedef Documentation

```
typedef uint32_t ezdp\_sum\_addr\_t

typedef uint32_t ezdp\_sum\_addr\_table\_desc\_t
```

---

## Enumeration Type Documentation

```
enum ezdp\_internal\_mem\_space
```

Internal memory space types.

### Enumerator:

***EZDP\_PRIVATE\_DATA*** Thread private data memory space (PDMEM).

***EZDP\_STACK*** PDEMEM is an extension to PDMEM with the ability to access/cache private data on EMEM with programmed window size mapped to LPMEM, providing low miss penalty on selected area (PDEMEM).

***EZDP\_HALF\_CLUSTER\_DATA*** Sub cluster data memory space type (LDMEM).

***EZDP\_HALF\_CLUSTER\_CODE*** Sub cluster code memory space type (LCMEM).

***EZDP\_1\_CLUSTER\_DATA*** Single cluster data memory space type (IDMEM1).

***EZDP\_1\_CLUSTER\_CODE*** Single cluster code memory space type (ICMEM1).

***EZDP\_2\_CLUSTER\_DATA*** Dual cluster data memory space type (IDMEM2).

***EZDP\_2\_CLUSTER\_CODE*** Dual cluster code memory space type (ICMEM2).

***EZDP\_4\_CLUSTER\_DATA*** Quad cluster data memory space type (IDMEM4).

***EZDP\_4\_CLUSTER\_CODE*** Quad cluster code memory space type (ICMEM4).

***EZDP\_16\_CLUSTER\_DATA*** 16 cluster data memory space type (IDMEM16).

***EZDP\_16\_CLUSTER\_CODE*** 16 cluster code memory space type (ICMEM16).

***EZDP\_ALL\_CLUSTER\_DATA*** All cluster data memory space type (IDMEMG).

***EZDP\_ALL\_CLUSTER\_CODE*** All cluster code memory space type (ICMEMG).

***EZDP\_ALL\_CLUSTER\_IO*** Entire SOC shared IO data (seen by all SOC threads and IO peripherals) (IDMEMGIO).

***EZDP\_ALL\_CLUSTER\_DATA\_EXT\_MEM*** External memory shared data (seen by all SOC threads) (EDMEMG).

#### enum [ezdp\\_mem\\_space\\_type](#)

mem space type possible values.

##### Enumerator:

***EZDP\_INTERNAL\_MS*** Internal memory space.

msid is according to type of the internal memory space. See `ezdp_internal_mem_space`.

***EZDP\_EXTERNAL\_MS*** External memory space.

msid is map, according to the configuration, to one of the configured MSID.

#### enum [ezdp\\_pci\\_addr\\_type](#)

pci addr type possible values.

##### Enumerator:

***EZDP\_PCI\_ADDR\_TYPE\_UNTRANSLATED*** PCIe untranslated address.

***EZDP\_PCI\_ADDR\_TYPE\_TRANSLATED*** PCIe translated address.

#### enum [ezdp\\_dma\\_flags](#)

dma flags.

##### Enumerator:

***EZDP\_MEMORY\_FLAG\_CLASS\_0*** Class 0 (Default class).

***EZDP\_MEMORY\_FLAG\_CLASS\_1*** Class 1.

***EZDP\_MEMORY\_FLAG\_CLASS\_2*** Class 2.

***EZDP\_MEMORY\_FLAG\_CLASS\_3*** Class 3.

**EZDP\_MEMORY\_FLAG\_UNCACHED** Do not allocate new cache entry.

**EZDP\_MEMORY\_FLAG\_OVERWRITE** Calculate the ECC without first reading the current ECC, then override the current ECC with the result.

**EZDP\_PCI\_FLAG\_NO\_SNOOP** PCI no snoop.

**EZDP\_PCI\_FLAG\_RELEX\_ORDERED** PCI relaxed order enable.

**EZDP\_PCI\_FLAG\_ATU\_BYPASS** PCI Address Translation Unit bypass enable.

## dpe/dp/include/ezdp\_pci.h File Reference

### Functions

- static `__always_inline` void [ezdp\\_init\\_pci\\_queue\\_desc](#) (uint32\_t endpoint, uint32\_t queue\_id, [ezdp\\_pci\\_queue\\_desc\\_t](#) \*pci\_queue\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Initialize and get PCI message queue description. static `__always_inline` void [ezdp\\_get\\_pci\\_msg](#) ([ezdp\\_pci\\_queue\\_desc\\_t](#) \*pci\_queue\_desc, uint32\_t msg\_index, struct [ezdp\\_pci\\_msg](#) \_\_cmem \*msg)
- Get message from PCI queue according to given index. static `__always_inline` void [ezdp\\_set\\_pci\\_msgq\\_read\\_index](#) ([ezdp\\_pci\\_queue\\_desc\\_t](#) \*pci\_queue\_desc, uint32\_t value, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Set read index of PCI message queue. static `__always_inline` void [ezdp\\_set\\_pci\\_msgq\\_read\\_index\\_async](#) ([ezdp\\_pci\\_queue\\_desc\\_t](#) \*pci\_queue\_desc, uint32\_t value, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Set read index of PCI message queue. static `__always_inline` uint32\_t [ezdp\\_get\\_pci\\_msgq\\_write\\_index](#) ([ezdp\\_pci\\_queue\\_desc\\_t](#) \*pci\_queue\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Get write index of PCI message queue. static `__always_inline` uint32\_t [ezdp\\_get\\_pci\\_msgq\\_read\\_index](#) ([ezdp\\_pci\\_queue\\_desc\\_t](#) \*pci\_queue\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Get read index of PCI message queue. static `__always_inline` uint32\_t [ezdp\\_copy\\_frame\\_data\\_to\\_pci](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*dst\_pci\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Copy data from a frame buffer to PCI address. static `__always_inline` void [ezdp\\_copy\\_frame\\_data\\_to\\_pci\\_async](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*dst\_pci\_ptr, struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*src\_bd\_ptr, uint32\_t src\_bd\_offset, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_frame\\_data\\_to\\_pci\(\)](#). static `__always_inline` void [ezdp\\_copy\\_frame\\_data\\_from\\_pci](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*src\_pci\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Copy data from PCI address to a frame buffer. static `__always_inline` void [ezdp\\_copy\\_frame\\_data\\_from\\_pci\\_async](#) (struct [ezdp\\_buffer\\_desc](#) \_\_cmem \*dst\_bd\_ptr, uint32\_t dst\_bd\_offset, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*src\_pci\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_frame\\_data\\_from\\_pci\(\)](#). static `__always_inline` void [ezdp\\_load\\_data\\_from\\_pci](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Copy data from a PCI address to CMEM. static `__always_inline` void [ezdp\\_load\\_data\\_from\\_pci\\_async](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Non blocking version of [ezdp\\_load\\_data\\_from\\_pci\(\)](#). static `__always_inline` uint32\_t [ezdp\\_store\\_data\\_to\\_pci](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Copy data from CMEM to PCI address. static `__always_inline` void [ezdp\\_store\\_data\\_to\\_pci\\_async](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Non blocking version of [ezdp\\_store\\_data\\_to\\_pci\(\)](#). static `__always_inline` uint32\_t [ezdp\\_copy\\_pci\\_data\\_to\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*src\_pci\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Copy PCI data to extended addresses. static `__always_inline` void [ezdp\\_copy\\_pci\\_data\\_to\\_ext\\_addr\\_async](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*src\_pci\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_pci\\_data\\_to\\_ext\\_addr\(\)](#). static `__always_inline` uint32\_t [ezdp\\_copy\\_pci\\_data\\_from\\_ext\\_addr](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*dst\_pci\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Copy extended address data to PCI. static `__always_inline` void [ezdp\\_copy\\_pci\\_data\\_from\\_ext\\_addr\\_async](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*dst\_pci\_ptr, struct [ezdp\\_ext\\_addr](#) \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t flags)
- Non blocking version of [ezdp\\_copy\\_pci\\_data\\_from\\_ext\\_addr\(\)](#). static `__always_inline` uint32\_t [ezdp\\_translate\\_pci\\_addr](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, uint32\_t traffic\_class, uint32\_t flags)
- PCI address translation request; result will be saved in CMEM. static `__always_inline` void [ezdp\\_translate\\_pci\\_addr\\_async](#) (void \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, uint32\_t traffic\_class, uint32\_t flags)

- Non blocking version of [ezdp\\_translate\\_pci\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_translate\\_pci\\_addr\\_to\\_ext\\_addr](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*src\_pci\_ptr, uint32\_t traffic\_class, uint32\_t flags)
- Copy PCI data to extended addresses. static \_\_always\_inline void [ezdp\\_translate\\_pci\\_addr\\_to\\_ext\\_addr\\_async](#) (struct [ezdp\\_ext\\_addr](#) \_\_cmem \*dst\_ptr, struct [ezdp\\_pci\\_addr](#) \_\_cmem \*src\_pci\_ptr, uint32\_t traffic\_class, uint32\_t flags)
- Non blocking version of [ezdp\\_translate\\_pci\\_addr\\_to\\_ext\\_addr\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_send\\_message\\_to\\_pci](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t msg\_code, uint32\_t flags)
- Copy data from CMEM to PCI address. static \_\_always\_inline void [ezdp\\_send\\_message\\_to\\_pci\\_async](#) (struct [ezdp\\_pci\\_addr](#) \_\_cmem \*pci\_ptr, void \_\_cmem \*src\_ptr, uint32\_t size, uint32\_t traffic\_class, uint32\_t msg\_code, uint32\_t flags)
- Non blocking version of [ezdp\\_send\\_message\\_to\\_pci\(\)](#). static \_\_always\_inline void [ezdp\\_send\\_interrupt\\_to\\_pci](#) (uint32\_t endpoint, uint32\_t msid, uint32\_t phy\_func, uint32\_t virt\_func, bool virt\_func\_en, uint32\_t msix\_vec\_id, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Send interrupt message to PCI. static \_\_always\_inline void [ezdp\\_send\\_interrupt\\_to\\_pci\\_async](#) (uint32\_t endpoint, uint32\_t msid, uint32\_t phy\_func, uint32\_t virt\_func, bool virt\_func\_en, uint32\_t msix\_vec\_id, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Non blocking version of [ezdp\\_send\\_interrupt\\_to\\_pci\(\)](#). uint32\_t [ezdp\\_get\\_pci\\_ctrl\\_reg](#) (uint32\_t endpoint, uint32\_t pf, uint32\_t reg\_id, uint32\_t \*value)
- Get the value of the PCIe vendor specific configuration space register. uint32\_t [ezdp\\_set\\_pci\\_ctrl\\_reg](#) (uint32\_t endpoint, uint32\_t pf, uint32\_t reg\_id, uint32\_t value)

Set the value of the PCIe vendor specific configuration space register.

## Function Documentation

```
static __always_inline void ezdp_init_pci_queue_desc (uint32_t endpoint, uint32_t queue_id,
ezdp\_pci\_queue\_desc\_t *pci_queue_desc, char __cmem *work_area_ptr, uint32_t
work_area_size) [static]
```

Initialize and get PCI message queue description.

### Parameters:

- [in] *endpoint* - PCI end point number
- [in] *queue\_id* - PCI queue ID (0/1/2/3)
- [out] *pci\_queue\_desc* - message queue description
- [in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_INIT\_PCI\_QUEUE\_DESC\_WORK\_AREA\_SIZE
- [in] *work\_area\_size* - size of work area pointer

### Returns:

none

```
static __always_inline void ezdp_get_pci_msg (ezdp\_pci\_queue\_desc\_t *pci_queue_desc,
uint32_t msg_index, struct ezdp\_pci\_msg __cmem *msg) [static]
```

Get message from PCI queue according to given index.

### Parameters:

- [in] *pci\_queue\_desc* - message queue description
- [out] *msg\_index* - message index
- [out] *msg* - message from queue

### Returns:

none

```
static __always_inline void ezdp_set_pci_msgq_read_index (ezdp\_pci\_queue\_desc\_t *  
pci_queue_desc,  uint32_t value,  char __cmem * work_area_ptr,  uint32_t work_area_size)  
[static]
```

Set read index of PCI message queue.

**Parameters:**

- [in] *pci\_queue\_desc* - message queue description
- [in] *value* - new read index value
- [in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_PCI\_RW\_INDEX\_WORK\_AREA\_SIZE
- [in] *work\_area\_size* - size of work area pointer

**Returns:**

none

```
static __always_inline void ezdp_set_pci_msgq_read_index_async (ezdp\_pci\_queue\_desc\_t *  
pci_queue_desc,  uint32_t value,  char __cmem * work_area_ptr,  uint32_t work_area_size)  
[static]
```

Set read index of PCI message queue.

**Parameters:**

- [in] *pci\_queue\_desc* - message queue description
- [in] *value* - new read index value
- [in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_PCI\_RW\_INDEX\_WORK\_AREA\_SIZE
- [in] *work\_area\_size* - size of work area pointer

**Returns:**

none

```
static __always_inline uint32_t ezdp_get_pci_msgq_write_index (ezdp\_pci\_queue\_desc\_t *  
pci_queue_desc,  char __cmem * work_area_ptr,  uint32_t work_area_size) [static]
```

Get write index of PCI message queue.

**Parameters:**

- [in] *pci\_queue\_desc* - message queue description
- [in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_PCI\_RW\_INDEX\_WORK\_AREA\_SIZE
- [in] *work\_area\_size* - size of work area pointer

**Returns:**

uint32\_t write index value

```
static __always_inline uint32_t ezdp_get_pci_msgq_read_index (ezdp\_pci\_queue\_desc\_t *  
pci_queue_desc,  char __cmem * work_area_ptr,  uint32_t work_area_size) [static]
```

Get read index of PCI message queue.

**Parameters:**

- [in] *pci\_queue\_desc* - message queue description



[in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_PCI\_RW\_INDEX\_WORK\_AREA\_SIZE  
[in] *work\_area\_size* - size of work area pointer

**Returns:**

uint32\_t read index value

```
static __always_inline uint32_t ezdp_copy_frame_data_to_pci (struct ezdp\_pci\_addr __cmem *
dst_pci_ptr, struct ezdp\_buffer\_desc __cmem * src_bd_ptr, uint32_t src_bd_offset,
uint32_t size, uint32_t traffic_class, uint32_t flags) [static]
```

Copy data from a frame buffer to PCI address.

**Parameters:**

[in] *dst\_pci\_ptr* - pointer to destination PCI address (in CMEM in 8 byte alignment)  
[in] *src\_bd\_ptr* - pointer to source frame describing the source buffer  
[in] *src\_bd\_offset* - offset in the source buffer to copy data from  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Returns:**

uint32\_t - 16 bit checksum value

```
static __always_inline void ezdp_copy_frame_data_to_pci_async (struct ezdp\_pci\_addr __cmem *
dst_pci_ptr, struct ezdp\_buffer\_desc __cmem * src_bd_ptr, uint32_t src_bd_offset,
uint32_t size, uint32_t traffic_class, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_frame\\_data\\_to\\_pci\(\)](#).

**Parameters:**

[in] *dst\_pci\_ptr* - pointer to destination PCI address (in CMEM in 8 byte alignment)  
[in] *src\_bd\_ptr* - pointer to source frame describing the source buffer  
[in] *src\_bd\_offset* - offset in the source buffer to copy data from  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_copy_frame_data_from_pci (struct ezdp\_buffer\_desc __cmem *
dst_bd_ptr, uint32_t dst_bd_offset, struct ezdp\_pci\_addr __cmem * src_pci_ptr, uint32_t
size, uint32_t traffic_class, uint32_t flags) [static]
```

Copy data from PCI address to a frame buffer.

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to frame describing the destination buffer  
 [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to  
 [in] *src\_pci\_ptr* - pointer to source PCI address (in CMEM in 8 byte alignment)  
 [in] *size* - number of bytes to copy  
 [in] *traffic\_class* - PCI traffic class  
 [in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable  
 EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Returns:**

none

```
static __always_inline void ezdp_copy_frame_data_from_pci_async (struct ezdp\_buffer\_desc
__cmem * dst_bd_ptr,  uint32_t dst_bd_offset,  struct ezdp\_pci\_addr __cmem * src_pci_ptr,
uint32_t size,  uint32_t traffic_class,  uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_frame\\_data\\_from\\_pci\(\)](#).

**Parameters:**

[in] *dst\_bd\_ptr* - pointer to frame describing the destination buffer  
 [in] *dst\_bd\_offset* - offset in the destination buffer to copy data to  
 [in] *src\_pci\_ptr* - pointer to source PCI address (in CMEM in 8 byte alignment)  
 [in] *size* - number of bytes to copy  
 [in] *traffic\_class* - PCI traffic class  
 [in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable  
 EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline void ezdp_load_data_from_pci (void __cmem * dst_ptr,  struct
ezdp\_pci\_addr __cmem * pci_ptr,  uint32_t size,  uint32_t traffic_class,  uint32_t flags)
[static]
```

Copy data from a PCI address to CMEM.

**Parameters:**

[out] *dst\_ptr* - pointer to the destination array in CMEM where the content is to be copied  
 [in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment) to be copied  
 [in] *size* - number of bytes to copy  
 [in] *traffic\_class* - PCI traffic class  
 [in] *flags* - execution memory flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
 EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable  
 EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Returns:**

none

```
static __always_inline void ezdp_load_data_from_pci_async(void __cmem * dst_ptr, struct
ezdp_pci_addr __cmem * pci_ptr, uint32_t size, uint32_t traffic_class, uint32_t flags)
[static]
```

Non blocking version of [ezdp\\_load\\_data\\_from\\_pci\(\)](#).

#### Parameters:

[out] *dst\_ptr* - pointer in CMEM to copy data to  
[in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment)  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution memory flags. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

#### Note:

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

#### Returns:

none

```
static __always_inline uint32_t ezdp_store_data_to_pci(struct ezdp_pci_addr __cmem * pci_ptr,
void __cmem * src_ptr, uint32_t size, uint32_t traffic_class, uint32_t flags) [static]
```

Copy data from CMEM to PCI address.

#### Parameters:

[in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment)  
[in] *src\_ptr* - pointer in CMEM to copy data from  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

#### Returns:

uint32\_t - 16 bit checksum value

```
static __always_inline void ezdp_store_data_to_pci_async(struct ezdp_pci_addr __cmem *
pci_ptr, void __cmem * src_ptr, uint32_t size, uint32_t traffic_class, uint32_t flags)
[static]
```

Non blocking version of [ezdp\\_store\\_data\\_to\\_pci\(\)](#).

#### Parameters:

[in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment)  
[in] *src\_ptr* - pointer in CMEM to copy data from  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry

EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

non

```
static __always_inline uint32_t ezdp_copy_pci_data_to_ext_addr (struct ezdp\_ext\_addr __cmem *
dst_ptr, struct ezdp\_pci\_addr __cmem * src_pci_ptr, uint32_t size, uint32_t traffic_class,
uint32_t flags) [static]
```

Copy PCI data to extended addresses.

**Parameters:**

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
[in] *src\_pci\_ptr* - pointer to source PCI address (in CMEM in 8 byte alignment)  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_copy_pci_data_to_ext_addr_async (struct ezdp\_ext\_addr
__cmem * dst_ptr, struct ezdp\_pci\_addr __cmem * src_pci_ptr, uint32_t size, uint32_t
traffic_class, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_pci\\_data\\_to\\_ext\\_addr\(\)](#).

**Parameters:**

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
[in] *src\_pci\_ptr* - pointer to source PCI address (in CMEM in 8 byte alignment)  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_copy_pci_data_from_ext_addr (struct ezdp\_pci\_addr
__cmem * dst_pci_ptr, struct ezdp\_ext\_addr __cmem * src_ptr, uint32_t size, uint32_t
traffic_class, uint32_t flags) [static]
```

Copy extended address data to PCI.

**Parameters:**

- [in] *dst\_pci\_ptr* - pointer to destination PCI address (in CMEM in 8 byte alignment)
- [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)
- [in] *size* - number of bytes to copy
- [in] *traffic\_class* - PCI traffic class
- [in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
 EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
 order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_copy_pci_data_from_ext_addr_async (struct ezdp\_pci\_addr
__cmem * dst_pci_ptr, struct ezdp\_ext\_addr __cmem * src_ptr, uint32_t size, uint32_t
traffic_class, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_copy\\_pci\\_data\\_from\\_ext\\_addr\(\)](#).

**Parameters:**

- [in] *dst\_pci\_ptr* - pointer to destination PCI address (in CMEM in 8 byte alignment)
- [in] *src\_ptr* - pointer to source extended address (in CMEM in 8 byte alignment)
- [in] *size* - number of bytes to copy
- [in] *traffic\_class* - PCI traffic class
- [in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
 EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
 order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_translate_pci_addr (void __cmem * dst_ptr, struct
ezdp\_pci\_addr __cmem * pci_ptr, uint32_t traffic_class, uint32_t flags) [static]
```

PCI address translation request; result will be saved in CMEM.

**Parameters:**

- [out] *dst\_ptr* - pointer to the destination array in CMEM where the content is to be copied
- [in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment) to be copied
- [in] *traffic\_class* - PCI traffic class
- [in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
 EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
 order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_translate_pci_addr_async (void __cmem * dst_ptr, struct
ezdp_pci_addr __cmem * pci_ptr, uint32_t traffic_class, uint32_t flags) [static]
```

Non blocking version of [ezdp\\_translate\\_pci\\_addr\(\)](#).

#### Parameters:

[out] *dst\_ptr* - pointer in CMEM to copy data to  
[in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment)  
[in] *traffic\_class* - The PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable  
EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

#### Note:

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

#### Returns:

none

```
static __always_inline uint32_t ezdp_translate_pci_addr_to_ext_addr (struct ezdp_ext_addr
__cmem * dst_ptr, struct ezdp_pci_addr __cmem * src_pci_ptr, uint32_t traffic_class,
uint32_t flags) [static]
```

Copy PCI data to extended addresses.

#### Parameters:

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
[in] *src\_pci\_ptr* - pointer to source PCI address (in CMEM in 8 byte alignment)  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable  
EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

#### Returns:

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_translate_pci_addr_to_ext_addr_async (struct ezdp_ext_addr
__cmem * dst_ptr, struct ezdp_pci_addr __cmem * src_pci_ptr, uint32_t traffic_class,
uint32_t flags) [static]
```

Non blocking version of [ezdp\\_translate\\_pci\\_addr\\_to\\_ext\\_addr\(\)](#).

#### Parameters:

[in] *dst\_ptr* - pointer to destination extended address (in CMEM in 8 byte alignment)  
[in] *src\_pci\_ptr* - pointer to source PCI address (in CMEM in 8 byte alignment)  
[in] *traffic\_class* - PCI traffic class  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed order enable  
EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
static __always_inline uint32_t ezdp_send_message_to_pci (struct ezdp\_pci\_addr __cmem *
pci_ptr, void __cmem * src_ptr, uint32_t size, uint32_t traffic_class, uint32_t msg_code,
uint32_t flags) [static]
```

Copy data from CMEM to PCI address.

**Parameters:**

[in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment)  
[in] *src\_ptr* - pointer in CMEM to copy data from  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *msg\_code* - PCI Message Code  
[in] *flags* - execution flags for memory. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Returns:**

uint32\_t - first 4 bytes of the copied data

```
static __always_inline void ezdp_send_message_to_pci_async (struct ezdp\_pci\_addr __cmem *
pci_ptr, void __cmem * src_ptr, uint32_t size, uint32_t traffic_class, uint32_t msg_code,
uint32_t flags) [static]
```

Non blocking version of [ezdp\\_send\\_message\\_to\\_pci\(\)](#).

**Parameters:**

[in] *pci\_ptr* - pointer to PCI address (in CMEM in 8 byte alignment)  
[in] *src\_ptr* - pointer in CMEM to copy data from  
[in] *size* - number of bytes to copy  
[in] *traffic\_class* - PCI traffic class  
[in] *msg\_code* - PCI Message Code  
[in] *flags* - execution flags for pci. Bitwise OR of zero or more flags out of the following:  
EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)  
EZDP\_MEMORY\_FLAG\_UNCACHED - do not allocate new cache entry  
EZDP\_PCI\_FLAG\_NO\_SNOOP - PCI no snoop EZDP\_PCI\_FLAG\_RELEX\_ORDERED - PCI relaxed  
order enable EZDP\_PCI\_FLAG\_ATU\_BYPASS - PCI Address Translation Unit bypass enable

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

non

```
static __always_inline void ezdp_send_interrupt_to_pci (uint32_t endpoint, uint32_t msid,
uint32_t phy_func, uint32_t virt_func, bool virt_func_en, uint32_t msix_vec_id, char
__cmem * work_area_ptr, uint32_t work_area_size) [static]
```



Send interrupt message to PCI.

**Parameters:**

[in] *endpoint* - PCI endpoint number  
 [in] *msid* - MSID of the endpoint  
 [in] *phy\_func* - physical function number  
 [in] *virt\_func* - virtual function number  
 [in] *virt\_func\_en* - enable virtual function usage  
 [in] *msix\_vec\_id* - MSI-X vector ID  
 [in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_PCI\_INTERRUPT\_WORK\_AREA\_SIZE  
 [in] *work\_area\_size* - size of work area pointer

**Returns:**

none

```
static __always_inline void ezdp_send_interrupt_to_pci_async (uint32_t endpoint,  uint32_t msid,  uint32_t phy_func,  uint32_t virt_func,  bool virt_func_en,  uint32_t msix_vec_id,  char __cmem * work_area_ptr,  uint32_t work_area_size) [static]
```

Non blocking version of [ezdp\\_send\\_interrupt\\_to\\_pci\(\)](#).

**Parameters:**

[in] *endpoint* - PCI endpoint number  
 [in] *msid* - MSID of the endpoint  
 [in] *phy\_func* - physical function number  
 [in] *virt\_func* - virtual function number  
 [in] *virt\_func\_en* - enable virtual function usage  
 [in] *msix\_vec\_id* - MSI-X vector ID  
 [in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_PCI\_INTERRUPT\_WORK\_AREA\_SIZE  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was copied to the destination.

**Returns:**

none

```
uint32_t ezdp_get_pci_ctrl_reg (uint32_t endpoint,  uint32_t pf,  uint32_t reg_id,  uint32_t * value)
```

Get the value of the PCIe vendor specific configuration space register.

**Parameters:**

[in] *endpoint* - PCI endpoint number  
 [in] *pf* - physical function number  
 [in] *reg\_id* - register ID (0..3)  
 [out] *value* - register value

**Note:**

This function performs a Linux system call. Avoid calling it from packet processing code



**Returns:**

- 0 (operation success), EIO (OS failed). The function may also fail for any of the errors specified for routines: open,iocctl,close. In such case the errno value of the failure routine is returned. Use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

**uint32\_t ezdp\_set\_pci\_ctrl\_reg (uint32\_t endpoint, uint32\_t pf, uint32\_t reg\_id, uint32\_t value)**

Set the value of the PCIe vendor specific configuration space register.

**Parameters:**

[in] *endpoint* - PCI end point number  
[in] *pf* - physical function number  
[in] *reg\_id* - register ID (0..3)  
[in] *value* - register value

**Note:**

This function performs a Linux system call. Avoid calling it from packet processing code

**Returns:**

- 0 (operation success) EIO (OS failed). The function may also fail for any of the errors specified for routines open,iocctl,close. In such case the errno value of the failure routine is returned Use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

## dpe/dp/include/ezdp\_pci\_defs.h File Reference

### Data Structures

- struct [ezdp\\_pci\\_info](#)
- *PCI info for describing to which endpoint, physical function, virtual function and queue the frame is to be sent.* struct [ezdp\\_pci\\_msg\\_ctrl](#)
- *PCI message control.* struct [ezdp\\_pci\\_msg\\_payload\\_elbi](#)
- *PCI ELBI message payload.* struct [ezdp\\_pci\\_msg\\_payload\\_ats](#)
- *PCI ATS message payload.* struct [ezdp\\_pci\\_msg\\_payload\\_msix](#)
- *PCI MSIX message payload.* struct [ezdp\\_pci\\_msg](#)
- *Message from PCI queue.* struct [ezdp\\_driver\\_desc\\_flags](#)
- *TX/RX descriptor flags structure.* struct [ezdp\\_driver\\_desc](#)

### TX/RX descriptor. Defines

- #define [EZDP\\_PCI\\_VERSION\\_MAJOR](#) 2
- #define [EZDP\\_PCI\\_VERSION\\_MINOR](#) 1
- #define [EZDP\\_PCI\\_INFO\\_VIRT\\_FUNC\\_SIZE](#) 7
- #define [EZDP\\_PCI\\_INFO\\_VIRT\\_FUNC\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_INFO\\_ENDPOINT\\_SIZE](#) 1
- #define [EZDP\\_PCI\\_INFO\\_ENDPOINT\\_OFFSET](#) 7
- #define [EZDP\\_PCI\\_INFO\\_ENDPOINT\\_MASK](#) (1 << EZDP\_PCI\_INFO\_ENDPOINT\_OFFSET)
- #define [EZDP\\_PCI\\_INFO\\_PHYS\\_FUNC\\_SIZE](#) 2
- #define [EZDP\\_PCI\\_INFO\\_PHYS\\_FUNC\\_OFFSET](#) 8
- #define [EZDP\\_PCI\\_INFO\\_QUEUE\\_SIZE](#) 5
- #define [EZDP\\_PCI\\_INFO\\_QUEUE\\_OFFSET](#) 10
- #define [EZDP\\_PCI\\_INFO\\_VIRT\\_FUNC\\_EN\\_SIZE](#) 1
- #define [EZDP\\_PCI\\_INFO\\_VIRT\\_FUNC\\_EN\\_OFFSET](#) 15
- #define [EZDP\\_PCI\\_INFO\\_VIRT\\_FUNC\\_EN\\_MASK](#) (1 << EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_OFFSET)
- #define [EZDP\\_PCI\\_INFO\\_RESERVED16\\_32\\_SIZE](#) 16
- #define [EZDP\\_PCI\\_INFO\\_RESERVED16\\_32\\_OFFSET](#) 16
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_VIRT\\_FUNC\\_SIZE](#) 7
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_VIRT\\_FUNC\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_RESERVED8\\_SIZE](#) 1
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_RESERVED8\\_OFFSET](#) 7
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_PHY\\_FUNC\\_SIZE](#) 2
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_PHY\\_FUNC\\_OFFSET](#) 8
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_RESERVED10\\_11\\_SIZE](#) 2
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_RESERVED10\\_11\\_OFFSET](#) 10
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_BAR\\_NUM\\_SIZE](#) 3
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_BAR\\_NUM\\_OFFSET](#) 12
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_VIRT\\_FUNC\\_EN\\_SIZE](#) 1
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_VIRT\\_FUNC\\_EN\\_OFFSET](#) 15
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_VIRT\\_FUNC\\_EN\\_MASK](#) (1 << EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_OFFSET)
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_RESERVED\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_RESERVED\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_ADDRESS\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_ADDRESS\\_OFFSET](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_ADDRESS\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_ADDRESS\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_DATA\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_DATA\\_OFFSET](#) 64
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_DATA\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ELBI\\_WORD\\_COUNT](#) 3

- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_RESERVED\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_RESERVED\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_MSB\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_MSB\\_OFFSET](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_MSB\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_MSB\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_LSB\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_LSB\\_OFFSET](#) 64
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_LSB\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_DATA\\_LSB\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_ATS\\_WORD\\_COUNT](#) 3
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_RESERVED0\\_31\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_RESERVED0\\_31\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_RESERVED\\_32\\_63\\_SIZE](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_RESERVED\\_32\\_63\\_OFFSET](#) 32
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_VECTOR\\_INDEX\\_SIZE](#) 2
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_VECTOR\\_INDEX\\_OFFSET](#) 64
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_VECTOR\\_INDEX\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_VECTOR\\_INDEX\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_RESERVED66\\_95\\_SIZE](#) 30
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_RESERVED66\\_95\\_OFFSET](#) 66
- #define [EZDP\\_PCI\\_MSG\\_PAYLOAD\\_MSIX\\_WORD\\_COUNT](#) 3
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_SIZE](#) 16
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_MSG\\_CTRL\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_PCI\\_MSG\\_MSG\\_SIZE](#) 8
- #define [EZDP\\_PCI\\_MSG\\_MSG\\_OFFSET](#) 16
- #define [EZDP\\_PCI\\_MSG\\_MSG\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_PCI\\_MSG\\_MSG\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_PCI\\_MSG\\_ECC\\_SIZE](#) 8
- #define [EZDP\\_PCI\\_MSG\\_ECC\\_OFFSET](#) 24
- #define [EZDP\\_PCI\\_MSG\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_DATA\\_SIZE](#) 1
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_DATA\\_OFFSET](#) 0
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_DATA\\_MASK](#) (1 << EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_OFFSET)
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_OWNER\\_SIZE](#) 1
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_OWNER\\_OFFSET](#) 1
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_OWNER\\_MASK](#) (1 << EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_OFFSET)
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_ERROR\\_OFFSET](#) 2
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_ERROR\\_MASK](#) (1 << EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_OFFSET)
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_TYPE\\_SIZE](#) 5
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_TYPE\\_OFFSET](#) 3
- #define [EZDP\\_DRIVER\\_DESC\\_BUF\\_DATA\\_ADDR\\_SIZE](#) 64
- #define [EZDP\\_DRIVER\\_DESC\\_BUF\\_DATA\\_ADDR\\_OFFSET](#) 0
- #define [EZDP\\_DRIVER\\_DESC\\_BUF\\_DATA\\_ADDR\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_DRIVER\\_DESC\\_BUF\\_DATA\\_ADDR\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DRIVER\\_DESC\\_LEN\\_SIZE](#) 32
- #define [EZDP\\_DRIVER\\_DESC\\_LEN\\_OFFSET](#) 64
- #define [EZDP\\_DRIVER\\_DESC\\_LEN\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_DRIVER\\_DESC\\_LEN\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DRIVER\\_DESC\\_TOTAL\\_SIZE](#) 8
- #define [EZDP\\_DRIVER\\_DESC\\_TOTAL\\_OFFSET](#) 96

- #define [EZDP\\_DRIVER\\_DESC\\_TOTAL\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_DRIVER\\_DESC\\_TOTAL\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_SIZE](#) 8
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_OFFSET](#) 104
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_DRIVER\\_DESC\\_FLAGS\\_WORD\\_OFFSET](#) 8
- #define [EZDP\\_DRIVER\\_DESC\\_SUB\\_TYPE\\_SIZE](#) 16
- #define [EZDP\\_DRIVER\\_DESC\\_SUB\\_TYPE\\_OFFSET](#) 112
- #define [EZDP\\_DRIVER\\_DESC\\_SUB\\_TYPE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_DRIVER\\_DESC\\_SUB\\_TYPE\\_WORD\\_OFFSET](#) 16
- #define [EZDP\\_DRIVER\\_DESC\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_PCI\\_INTERRUPT\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_send\_pci\_interrupt\_working\_area)
- *Send MSI-X to PCI working area.* #define [EZDP\\_INIT\\_PCI\\_QUEUE\\_DESC\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_init\_msgq\_desc\_working\_area)
- *Initialized PCI queue descriptor working area.* #define [EZDP\\_PCI\\_RW\\_INDEX\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_ctrl\_line)

### Get PCI queue read/write index working area. Typedefs

- typedef uint32\_t [ezdp\\_pci\\_info\\_t](#)
- typedef uint16\_t [ezdp\\_pci\\_msg\\_ctrl\\_t](#)
- typedef uint8\_t [ezdp\\_driver\\_desc\\_flags\\_t](#)
- typedef enum ezdp\_pci\_queue\_type [ezdp\\_pci\\_queue\\_type\\_t](#)
- *PCI queue type.* typedef struct ezdp\_pci\_msgq\_desc [ezdp\\_pci\\_queue\\_desc\\_t](#)
- *PCI message queue descriptor.* typedef struct ezdp\_init\_msgq\_desc\_working\_area [ezdp\\_init\\_pci\\_queue\\_desc\\_working\\_area\\_t](#)

### Initialized PCI queue descriptor working area. Enumerations

- enum [ezdp\\_pci\\_msg\\_type](#) { [EZDP\\_PCI\\_MSG\\_ERROR](#) = 0x0, [EZDP\\_PCI\\_MSG\\_RESET\\_REQUEST](#) = 0x1, [EZDP\\_PCI\\_MSG\\_FUNCTION\\_LEVEL\\_RESET](#) = 0x2, [EZDP\\_PCI\\_MSG\\_PM](#) = 0x3, [EZDP\\_PCI\\_MSG\\_OBFF\\_IDLE](#) = 0x4, [EZDP\\_PCI\\_MSG\\_OBFF\\_STATE](#) = 0x5, [EZDP\\_PCI\\_MSG\\_OBFF\\_ACTIVE](#) = 0x6, [EZDP\\_PCI\\_MSG\\_ATS\\_INVALID](#) = 0x7, [EZDP\\_PCI\\_MSG\\_ELBI](#) = 0x8, [EZDP\\_PCI\\_MSG\\_VPD\\_0](#) = 0x9, [EZDP\\_PCI\\_MSG\\_VPD\\_1](#) = 0xa, [EZDP\\_PCI\\_MSG\\_VPD\\_2](#) = 0xb, [EZDP\\_PCI\\_MSG\\_VPD\\_3](#) = 0xc, [EZDP\\_PCI\\_MSG\\_MSIX](#) = 0xd, [EZDP\\_PCI\\_MSG\\_NONE](#) = 0xe }

*pci msg type possible values.*

## Define Documentation

```
#define EZDP_PCI_VERSION_MAJOR 2

#define EZDP_PCI_VERSION_MINOR 1

#define EZDP_PCI_INFO_VIRT_FUNC_SIZE 7

#define EZDP_PCI_INFO_VIRT_FUNC_OFFSET 0

#define EZDP_PCI_INFO_ENDPOINT_SIZE 1

#define EZDP_PCI_INFO_ENDPOINT_OFFSET 7

#define EZDP_PCI_INFO_ENDPOINT_MASK (1 << EZDP_PCI_INFO_ENDPOINT_OFFSET)

#define EZDP_PCI_INFO_PHYS_FUNC_SIZE 2

#define EZDP_PCI_INFO_PHYS_FUNC_OFFSET 8

#define EZDP_PCI_INFO_QUEUE_SIZE 5

#define EZDP_PCI_INFO_QUEUE_OFFSET 10

#define EZDP_PCI_INFO_VIRT_FUNC_EN_SIZE 1

#define EZDP_PCI_INFO_VIRT_FUNC_EN_OFFSET 15

#define EZDP_PCI_INFO_VIRT_FUNC_EN_MASK (1 << EZDP_PCI_INFO_VIRT_FUNC_EN_OFFSET)

#define EZDP_PCI_INFO_RESERVED16_32_SIZE 16

#define EZDP_PCI_INFO_RESERVED16_32_OFFSET 16

#define EZDP_PCI_MSG_CTRL_VIRT_FUNC_SIZE 7

#define EZDP_PCI_MSG_CTRL_VIRT_FUNC_OFFSET 0

#define EZDP_PCI_MSG_CTRL_RESERVED8_SIZE 1

#define EZDP_PCI_MSG_CTRL_RESERVED8_OFFSET 7

#define EZDP_PCI_MSG_CTRL_PHY_FUNC_SIZE 2

#define EZDP_PCI_MSG_CTRL_PHY_FUNC_OFFSET 8

#define EZDP_PCI_MSG_CTRL_RESERVED10_11_SIZE 2

#define EZDP_PCI_MSG_CTRL_RESERVED10_11_OFFSET 10

#define EZDP_PCI_MSG_CTRL_BAR_NUM_SIZE 3
```

```
#define EZDP_PCI_MSG_CTRL_BAR_NUM_OFFSET 12

#define EZDP_PCI_MSG_CTRL_VIRT_FUNC_EN_SIZE 1

#define EZDP_PCI_MSG_CTRL_VIRT_FUNC_EN_OFFSET 15

#define EZDP_PCI_MSG_CTRL_VIRT_FUNC_EN_MASK (1 <<
EZDP_PCI_MSG_CTRL_VIRT_FUNC_EN_OFFSET)

#define EZDP_PCI_MSG_PAYLOAD_ELBI_RESERVED_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_ELBI_RESERVED_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_ELBI_ADDRESS_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_ELBI_ADDRESS_OFFSET 32

#define EZDP_PCI_MSG_PAYLOAD_ELBI_ADDRESS_WORD_SELECT 1

#define EZDP_PCI_MSG_PAYLOAD_ELBI_ADDRESS_WORD_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_ELBI_DATA_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_ELBI_DATA_OFFSET 64

#define EZDP_PCI_MSG_PAYLOAD_ELBI_DATA_WORD_SELECT 2

#define EZDP_PCI_MSG_PAYLOAD_ELBI_DATA_WORD_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_ELBI_WORD_COUNT 3

#define EZDP_PCI_MSG_PAYLOAD_ATS_RESERVED_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_ATS_RESERVED_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_MSB_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_MSB_OFFSET 32

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_MSB_WORD_SELECT 1

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_MSB_WORD_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_LSB_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_LSB_OFFSET 64

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_LSB_WORD_SELECT 2

#define EZDP_PCI_MSG_PAYLOAD_ATS_DATA_LSB_WORD_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_ATS_WORD_COUNT 3
```

```
#define EZDP_PCI_MSG_PAYLOAD_MSIX_RESERVED0_31_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_MSIX_RESERVED0_31_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_MSIX_RESERVED_32_63_SIZE 32

#define EZDP_PCI_MSG_PAYLOAD_MSIX_RESERVED_32_63_OFFSET 32

#define EZDP_PCI_MSG_PAYLOAD_MSIX_VECTOR_INDEX_SIZE 2

#define EZDP_PCI_MSG_PAYLOAD_MSIX_VECTOR_INDEX_OFFSET 64

#define EZDP_PCI_MSG_PAYLOAD_MSIX_VECTOR_INDEX_WORD_SELECT 2

#define EZDP_PCI_MSG_PAYLOAD_MSIX_VECTOR_INDEX_WORD_OFFSET 0

#define EZDP_PCI_MSG_PAYLOAD_MSIX_RESERVED66_95_SIZE 30

#define EZDP_PCI_MSG_PAYLOAD_MSIX_RESERVED66_95_OFFSET 66

#define EZDP_PCI_MSG_PAYLOAD_MSIX_WORD_COUNT 3

#define EZDP_PCI_MSG_CTRL_SIZE 16

#define EZDP_PCI_MSG_CTRL_OFFSET 0

#define EZDP_PCI_MSG_CTRL_WORD_SELECT 0

#define EZDP_PCI_MSG_CTRL_WORD_OFFSET 0

#define EZDP_PCI_MSG_MSG_SIZE 8

#define EZDP_PCI_MSG_MSG_OFFSET 16

#define EZDP_PCI_MSG_MSG_WORD_SELECT 0

#define EZDP_PCI_MSG_MSG_WORD_OFFSET 16

#define EZDP_PCI_MSG_ECC_SIZE 8

#define EZDP_PCI_MSG_ECC_OFFSET 24

#define EZDP_PCI_MSG_WORD_COUNT 4

#define EZDP_DRIVER_DESC_FLAGS_DATA_SIZE 1

#define EZDP_DRIVER_DESC_FLAGS_DATA_OFFSET 0

#define EZDP_DRIVER_DESC_FLAGS_DATA_MASK (1 <<
EZDP_DRIVER_DESC_FLAGS_DATA_OFFSET)

#define EZDP_DRIVER_DESC_FLAGS_OWNER_SIZE 1
```

```
#define EZDP_DRIVER_DESC_FLAGS_OWNER_OFFSET 1

#define EZDP_DRIVER_DESC_FLAGS_OWNER_MASK (1 <<
EZDP_DRIVER_DESC_FLAGS_OWNER_OFFSET)

#define EZDP_DRIVER_DESC_FLAGS_ERROR_SIZE 1

#define EZDP_DRIVER_DESC_FLAGS_ERROR_OFFSET 2

#define EZDP_DRIVER_DESC_FLAGS_ERROR_MASK (1 <<
EZDP_DRIVER_DESC_FLAGS_ERROR_OFFSET)

#define EZDP_DRIVER_DESC_FLAGS_TYPE_SIZE 5

#define EZDP_DRIVER_DESC_FLAGS_TYPE_OFFSET 3

#define EZDP_DRIVER_DESC_BUF_DATA_ADDR_SIZE 64

#define EZDP_DRIVER_DESC_BUF_DATA_ADDR_OFFSET 0

#define EZDP_DRIVER_DESC_BUF_DATA_ADDR_WORD_SELECT 0

#define EZDP_DRIVER_DESC_BUF_DATA_ADDR_WORD_OFFSET 0

#define EZDP_DRIVER_DESC_LEN_SIZE 32

#define EZDP_DRIVER_DESC_LEN_OFFSET 64

#define EZDP_DRIVER_DESC_LEN_WORD_SELECT 2

#define EZDP_DRIVER_DESC_LEN_WORD_OFFSET 0

#define EZDP_DRIVER_DESC_TOTAL_SIZE 8

#define EZDP_DRIVER_DESC_TOTAL_OFFSET 96

#define EZDP_DRIVER_DESC_TOTAL_WORD_SELECT 3

#define EZDP_DRIVER_DESC_TOTAL_WORD_OFFSET 0

#define EZDP_DRIVER_DESC_FLAGS_SIZE 8

#define EZDP_DRIVER_DESC_FLAGS_OFFSET 104

#define EZDP_DRIVER_DESC_FLAGS_WORD_SELECT 3

#define EZDP_DRIVER_DESC_FLAGS_WORD_OFFSET 8

#define EZDP_DRIVER_DESC_SUB_TYPE_SIZE 16

#define EZDP_DRIVER_DESC_SUB_TYPE_OFFSET 112

#define EZDP_DRIVER_DESC_SUB_TYPE_WORD_SELECT 3
```



```
#define EZDP_DRIVER_DESC_SUB_TYPE_WORD_OFFSET 16
```

```
#define EZDP_DRIVER_DESC_WORD_COUNT 4
```

```
#define EZDP_PCI_INTERRUPT_WORK_AREA_SIZE sizeof(struct  
ezdp_send_pci_interrupt_working_area)
```

Send MSI-X to PCI working area.

```
#define EZDP_INIT_PCI_QUEUE_DESC_WORK_AREA_SIZE sizeof(struct  
ezdp_init_msgq_desc_working_area)
```

Initialized PCI queue descriptor working area.

```
#define EZDP_PCI_RW_INDEX_WORK_AREA_SIZE sizeof(struct ezdp_ctrl_line)
```

Get PCI queue read/write index working area.

## Typedef Documentation

```
typedef uint32_t ezdp\_pci\_info\_t
```

```
typedef uint16_t ezdp\_pci\_msg\_ctrl\_t
```

```
typedef uint8_t ezdp\_driver\_desc\_flags\_t
```

```
typedef enum ezdp_pci_queue_type ezdp\_pci\_queue\_type\_t
```

PCI queue type.

```
typedef struct ezdp_pci_msgq_desc ezdp\_pci\_queue\_desc\_t
```

PCI message queue descriptor.

```
typedef struct ezdp_init_msgq_desc_working_area ezdp\_init\_pci\_queue\_desc\_working\_area\_t
```

Initialized PCI queue descriptor working area.

## Enumeration Type Documentation

```
enum ezdp\_pci\_msg\_type
```

pci msg type possible values.

**Enumerator:**

***EZDP\_PCI\_MSG\_ERROR*** Error event.

***EZDP\_PCI\_MSG\_RESET\_REQUEST*** Reset Request Due link down.

***EZDP\_PCI\_MSG\_FUNCTION\_LEVEL\_RESET*** Function Level Reset.

***EZDP\_PCI\_MSG\_PM*** Power Management.

***EZDP\_PCI\_MSG\_OBFF\_IDLE*** Optimized Buffer Flush and Fill idle.

***EZDP\_PCI\_MSG\_OBFF\_STATE*** Optimized Buffer Flush and Fill message.

***EZDP\_PCI\_MSG\_OBFF\_ACTIVE*** Optimized Buffer Flush and Fill cpu active.

***EZDP\_PCI\_MSG\_ATS\_INVALID*** Address Translate Service invalid.

***EZDP\_PCI\_MSG\_ELBI*** External Local Bus Interface.

***EZDP\_PCI\_MSG\_VPD\_0*** Vital product data capability 0.

***EZDP\_PCI\_MSG\_VPD\_1*** Vital product data capability 1.

***EZDP\_PCI\_MSG\_VPD\_2*** Vital product data capability 2.

***EZDP\_PCI\_MSG\_VPD\_3*** Vital product data capability 3.

***EZDP\_PCI\_MSG\_MSIX*** Message Interrupt X.

***EZDP\_PCI\_MSG\_NONE*** No message in queue.

## dpe/dp/include/ezdp\_pool.h File Reference

### Functions

- static \_\_always\_inline uint32\_t [ezdp\\_alloc\\_index](#) (uint32\_t pool\_id)
- *Allocate a single index from an index pool.* static \_\_always\_inline void [ezdp\\_free\\_index](#) (uint32\_t pool\_id, uint32\_t free\_index)
- *Free a single index from an index pool.* static \_\_always\_inline void [ezdp\\_free\\_index\\_async](#) (uint32\_t pool\_id, uint32\_t free\_index)
- *Non blocking version of [ezdp\\_free\\_index\(\)](#).* static \_\_always\_inline uint32\_t [ezdp\\_alloc\\_multi\\_index](#) (uint32\_t pool\_id, uint32\_t num\_of\_indexes, uint32\_t \_\_cmem \*index\_arrays\_ptr)
- *Allocate multiple indexes from an index pool.* static \_\_always\_inline void [ezdp\\_alloc\\_multi\\_index\\_async](#) (uint32\_t pool\_id, uint32\_t num\_of\_indexes, uint32\_t \_\_cmem \*index\_arrays\_ptr)
- *Non blocking version of [ezdp\\_alloc\\_multi\\_index\(\)](#).* static \_\_always\_inline void [ezdp\\_free\\_multi\\_index](#) (uint32\_t pool\_id, uint32\_t num\_of\_indexes, uint32\_t \_\_cmem \*index\_arrays\_ptr)
- *Free multiple indexes from an index pool.* static \_\_always\_inline void [ezdp\\_free\\_multi\\_index\\_async](#) (uint32\_t pool\_id, uint32\_t num\_of\_indexes, uint32\_t \_\_cmem \*index\_arrays\_ptr)
- *Non blocking version of [ezdp\\_free\\_multi\\_index\(\)](#).* static \_\_always\_inline uint32\_t [ezdp\\_read\\_free\\_indexes](#) (uint32\_t pool\_id)
- *The number of indexes available to be obtained.* static \_\_always\_inline void [ezdp\\_init\\_memory\\_pool](#) ([ezdp\\_mem\\_pool\\_t](#) \*memory\_pool, struct [ezdp\\_mem\\_pool\\_config](#) \*config)
- *Initialize a memory pool.* static \_\_always\_inline [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_alloc\\_obj](#) ([ezdp\\_mem\\_pool\\_t](#) \*memory\_pool)
- *Allocate a single object from a memory pool.* static \_\_always\_inline void [ezdp\\_free\\_obj](#) ([ezdp\\_mem\\_pool\\_t](#) \*memory\_pool, [ezdp\\_sum\\_addr\\_t](#) free\_obj)
- *Free a single object from a memory pool.* static \_\_always\_inline [ezdp\\_sum\\_addr\\_t](#) [ezdp\\_get\\_obj](#) ([ezdp\\_mem\\_pool\\_t](#) \*memory\_pool, uint32\_t object\_id)
- *Get object based on object id.* static \_\_always\_inline uint32\_t [ezdp\\_read\\_free\\_objs](#) ([ezdp\\_mem\\_pool\\_t](#) \*memory\_pool)

*The number of objects available to be obtained.*

---

### Function Documentation

**static \_\_always\_inline uint32\_t ezdp\_alloc\_index (uint32\_t pool\_id) [static]**

Allocate a single index from an index pool.

#### Parameters:

[in] *pool\_id* - the pool to allocate from

#### Returns:

allocated index EZDP\_NULL\_INDEX in case of error

**static \_\_always\_inline void ezdp\_free\_index (uint32\_t pool\_id, uint32\_t free\_index) [static]**

Free a single index from an index pool.

#### Parameters:

[in] *pool\_id* - the pool to recycle to

[in] *free\_index* - index to release

#### Returns:

none

```
static __always_inline void ezdp_free_index_async (uint32_t pool_id,   uint32_t free_index)  
[static]
```

Non blocking version of [ezdp\\_free\\_index\(\)](#).

**Parameters:**

[in] *pool\_id* - the pool to recycle to  
[in] *free\_index* - index to release

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete.

**Returns:**

none

```
static __always_inline uint32_t ezdp_alloc_multi_index (uint32_t pool_id,   uint32_t  
num_of_indexes,   uint32_t __cmem * index_arrays_ptr) [static]
```

Allocate multiple indexes from an index pool.

Allocate up to 3 indexes from the same pool. The allocated indexes are written to the CMEM. In addition, the first allocated index is returned. The operation either succeeds to allocate all requested resources or fails without allocating any resources.

**Parameters:**

[in] *pool\_id* - the pool to allocate from  
[in] *num\_of\_indexes* - number of indexes to allocate (1-3)  
[out] *index\_arrays\_ptr* - pointer to CMEM to write response to

**Returns:**

First allocated index EZDP\_NULL\_INDEX in case of error

```
static __always_inline void ezdp_alloc_multi_index_async (uint32_t pool_id,   uint32_t  
num_of_indexes,   uint32_t __cmem * index_arrays_ptr) [static]
```

Non blocking version of [ezdp\\_alloc\\_multi\\_index\(\)](#).

**Parameters:**

[in] *pool\_id* - the pool to allocate from  
[in] *num\_of\_indexes* - number of indexes to allocate (1-3)  
[out] *index\_arrays\_ptr* - pointer to CMEM to write response to

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete.

**Returns:**

First allocated index EZDP\_NULL\_INDEX in case of error

```
static __always_inline void ezdp_free_multi_index (uint32_t pool_id,   uint32_t num_of_indexes,  
uint32_t __cmem * index_arrays_ptr) [static]
```

Free multiple indexes from an index pool.

Up to 8 indexes can be freed in one command.

**Parameters:**

[in] *pool\_id* - the pool to recycle to  
[in] *num\_of\_indexes* - number of indexes to recycle (1-8)

[in] *index\_arrays\_ptr* - pointer to array of indexes (in CMEM) to free

**Returns:**

non

```
static __always_inline void ezdp_free_multi_index_async (uint32_t pool_id,   uint32_t
num_of_indexes,   uint32_t __cmem * index_arrays_ptr) [static]
```

Non blocking version of [ezdp\\_free\\_multi\\_buf\(\)](#).

**Parameters:**

[in] *pool\_id* - the pool to recycle to

[in] *num\_of\_indexes* - number of indexes to recycle (1-8)

[in] *index\_arrays\_ptr* - pointer to array of indexes (in CMEM) to free

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete.

**Returns:**

none

```
static __always_inline uint32_t ezdp_read_free_indexes (uint32_t pool_id) [static]
```

The number of indexes available to be obtained.

**Parameters:**

[in] *pool\_id* - pool id {0-63}

**Returns:**

uint32\_t - number of available/free indexes

```
static __always_inline void ezdp_init_memory_pool (ezdp\_mem\_pool\_t * memory_pool,   struct
ezdp\_mem\_pool\_config * config) [static]
```

Initialize a memory pool.

**Parameters:**

[out] *memory\_pool* - the pool to initialize

[in] *config* - the configuration information for initialization

**Returns:**

void

```
static __always_inline ezdp\_sum\_addr\_t ezdp_alloc_obj (ezdp\_mem\_pool\_t * memory_pool)
[static]
```

Allocate a single object from a memory pool.

**Parameters:**

[in] *memory\_pool* - the pool to allocate from

**Returns:**

Summarized address ([ezdp\\_sum\\_addr\\_t](#)) of the allocated object. In case of error, null summarized address is returned. Use [ezdp\\_is\\_null\\_sum\\_addr](#) API to check it.

```
static __always_inline void ezdp_free_obj (ezdp_mem_pool_t * memory_pool,  
ezdp_sum_addr_t free_obj) [static]
```

Free a single object from a memory pool.

**Parameters:**

[in] *memory\_pool* - the pool to recycle to  
[in] *free\_obj* - object to release

**Returns:**

none

```
static __always_inline ezdp_sum_addr_t ezdp_get_obj (ezdp_mem_pool_t * memory_pool,  
uint32_t object_id) [static]
```

Get object based on object id.

**Parameters:**

[in] *memory\_pool* - the pool to calculate block address from  
[in] *object\_id* - block index

**Returns:**

Summarized address (ezdp\_sum\_addr\_t) of the block. In case of error, null summarized address is returned.  
Use ezdp\_is\_null\_sum\_addr API to check it.

```
static __always_inline uint32_t ezdp_read_free_objs (ezdp_mem_pool_t * memory_pool)  
[static]
```

The number of objects available to be obtained.

**Parameters:**

[in] *memory\_pool* - the pool

**Returns:**

uint32\_t - number of available/free indexes

## dpe/dp/include/ezdp\_pool\_defs.h File Reference

### Data Structures

- struct [ezdp\\_mem\\_pool\\_config](#)

### memory pool configuration data structure Defines

- #define [EZDP\\_NULL\\_INDEX](#) 0xFFFFFFFF

### Typedefs

- typedef struct ezdp\_memory\_pool [ezdp\\_mem\\_pool\\_t](#)
- 

### Define Documentation

#define EZDP\_NULL\_INDEX 0xFFFFFFFF

---

### Typedef Documentation

typedef struct ezdp\_memory\_pool [ezdp\\_mem\\_pool\\_t](#)

## dpe/dp/include/ezdp\_processor.h File Reference

### Defines

- #define [EZDP\\_MAX\\_HW\\_THREADS](#) 16
- #define [EZDP\\_MAX\\_HW\\_CORES](#) 16
- #define [EZDP\\_MAX\\_HW\\_CLUSTERS](#) 16
- #define [EZDP\\_MAX\\_CPUS\\_ID](#) (EZDP\_MAX\_THREADS \* EZDP\_MAX\_CORES \* EZDP\_MAX\_CLUSTERS)

### Functions

- static uint32\_t [ezdp\\_get\\_cpu\\_id](#) (void)
- *Get the logical id of the processor that the process is running on (0-4095).* static uint32\_t [ezdp\\_get\\_thread\\_id](#) (void)
- *Get the id of the thread (within the core) that the process is running on (0-15).* static uint32\_t [ezdp\\_get\\_core\\_id](#) (void)
- *Get the ID of the core (within the cluster) that the process is running on (0-15).* static uint32\_t [ezdp\\_get\\_cluster\\_id](#) (void)
- *Get the ID of the cluster that the process is running on (0-15).* static uint32\_t [ezdp\\_calc\\_cpu\\_id](#) (uint8\_t hw\_cluster\_id, uint8\_t hw\_core\_id, uint8\_t hw\_thread\_id)
- *Calculate the logical ID of a processor.* static void [ezdp\\_sync](#) (void)
- *Relinquish the execution unit until all outstanding transactions complete.* static void [ezdp\\_rsync](#) (void)
- *Relinquish the execution unit until all outstanding read transactions complete.* static void [ezdp\\_mb](#) (void)
- *Wait until all outstanding memory accesses complete.* static void [ezdp\\_rmb](#) (void)
- *Wait until all outstanding memory read accesses complete.* static void [ezdp\\_wmb](#) (void)

*Wait until all outstanding memory write accesses complete.*

---

### Define Documentation

**#define EZDP\_MAX\_HW\_THREADS 16**

**#define EZDP\_MAX\_HW\_CORES 16**

**#define EZDP\_MAX\_HW\_CLUSTERS 16**

**#define EZDP\_MAX\_CPUS\_ID (EZDP\_MAX\_THREADS \* EZDP\_MAX\_CORES \* EZDP\_MAX\_CLUSTERS)**

---

### Function Documentation

**static uint32\_t ezdp\_get\_cpu\_id (void) [inline, static]**

Get the logical id of the processor that the process is running on (0-4095).

#### Returns:

The logical processor id (0-4095)

**static uint32\_t ezdp\_get\_thread\_id (void) [inline, static]**

Get the id of the thread (within the core) that the process is running on (0-15).



**Returns:**

The thread ID (0-15)

```
static uint32_t ezdp_get_core_id(void) [inline, static]
```

Get the ID of the core (within the cluster) that the process is running on (0-15).

**Returns:**

The core ID (0-15)

```
static uint32_t ezdp_get_cluster_id(void) [inline, static]
```

Get the ID of the cluster that the process is running on (0-15).

**Returns:**

The cluster ID (0-15)

```
static uint32_t ezdp_calc_cpu_id(uint8_t hw_cluster_id,    uint8_t hw_core_id,    uint8_t  
hw_thread_id) [inline, static]
```

Calculate the logical ID of a processor.

**Parameters:**

[in] *hw\_cluster\_id* - hardware ID of the cluster

[in] *hw\_core\_id* - hardware ID of the core (within the cluster)

[in] *hw\_thread\_id* - hardware ID of the thread (within the core)

**Returns:**

Logical processor ID (0-4095)

```
static void ezdp_sync(void) [inline, static]
```

Relinquish the execution unit until all outstanding transactions complete.

When called, the hardware performs a context switch to another eligible thread to utilize the execution unit while the existing thread waits for its transaction to complete.

**Returns:**

none

```
static void ezdp_rsync(void) [inline, static]
```

Relinquish the execution unit until all outstanding read transactions complete.

When called, the hardware performs a context switch to another eligible thread to utilize the execution unit while the existing thread waits for its transaction to complete.

**Returns:**

none

```
static void ezdp_mb(void) [inline, static]
```

Wait until all outstanding memory accesses complete.

**Returns:**

none

**static void ezdp\_rmb (void) [inline, static]**

Wait until all outstanding memory read accesses complete.

**Returns:**

none

**static void ezdp\_wmb (void) [inline, static]**

Wait until all outstanding memory write accesses complete.

**Returns:**

none

## dpe/dp/include/ezdp\_queue.h File Reference

### Functions

- static \_\_always\_inline bool [ezdp\\_init\\_ring](#) ([ezdp\\_ring\\_t](#) \*ring, struct [ezdp\\_ring\\_cfg](#) \*ring\_config)
- Initialize ring. static \_\_always\_inline bool [ezdp\\_ring\\_empty](#) ([ezdp\\_ring\\_t](#) \*ring)
- Check if ring is empty. static \_\_always\_inline bool [ezdp\\_ring\\_full](#) ([ezdp\\_ring\\_t](#) \*ring)
- Check if array\_queue is full. static \_\_always\_inline uint32\_t [ezdp\\_ring\\_length](#) ([ezdp\\_ring\\_t](#) \*ring)
- Return the number of entries in ring. static \_\_always\_inline bool [ezdp\\_enqueue\\_ring](#) ([ezdp\\_ring\\_t](#) \*ring, void \_\_cmem \*data, uint32\_t size, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size, uint32\_t flags)
- Insert new entry into ring. static \_\_always\_inline bool [ezdp\\_dequeue\\_ring](#) ([ezdp\\_ring\\_t](#) \*ring, void \_\_cmem \*data, uint32\_t size, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size, uint32\_t flags)
- Remove a head entry from ring. static \_\_always\_inline bool [ezdp\\_init\\_list](#) ([ezdp\\_list\\_t](#) \*list, struct [ezdp\\_list\\_cfg](#) \*list\_config, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Initialize list. static \_\_always\_inline bool [ezdp\\_list\\_empty](#) ([ezdp\\_list\\_t](#) \*list)
- Check if a list is empty. static \_\_always\_inline bool [ezdp\\_enqueue\\_list](#) ([ezdp\\_list\\_t](#) \*list, void \*data, uint8\_t size, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size, uint32\_t flags)
- Insert a new entry into a list. static \_\_always\_inline bool [ezdp\\_dequeue\\_list](#) ([ezdp\\_list\\_t](#) \*list, void \*data, uint8\_t size, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size, uint32\_t flags)
- Remove a head entry from a list. static \_\_always\_inline bool [ezdp\\_peek\\_list](#) ([ezdp\\_list\\_t](#) \*list, void \*data, uint8\_t size, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size, uint32\_t flags)
- Peek at head entry of a list. static \_\_always\_inline bool [ezdp\\_destroy\\_list](#) ([ezdp\\_list\\_t](#) \*list)

Destroy a list.

---

### Function Documentation

**static \_\_always\_inline bool ezdp\_init\_ring ([ezdp\\_ring\\_t](#) \* ring, struct [ezdp\\_ring\\_cfg](#) \* ring\_config) [static]**

Initialize ring.

#### Parameters:

[out] *ring* - pointer to ring object  
 [in] *ring\_config* - ring configuration

#### Note:

ring\_config->control\_addr address must be in resolution of 16 bytes

#### Returns:

true- success false - failure

**static \_\_always\_inline bool ezdp\_ring\_empty ([ezdp\\_ring\\_t](#) \* ring) [static]**

Check if ring is empty.

#### Parameters:

[in] *ring* - pointer to ring object

#### Returns:

true - queue is empty false - queue is not empty

**static \_\_always\_inline bool ezdp\_ring\_full ([ezdp\\_ring\\_t](#) \* ring) [static]**

Check if array\_queue is full.

**Parameters:**

[in] *ring* - pointer to ring object

**Returns:**

true - ring is full, false - ring is not full

```
static __always_inline uint32_t ezdp_ring_length (ezdp_ring_t * ring) [static]
```

Return the number of entries in ring.

**Parameters:**

[in] *ring* - pointer to ring object

**Returns:**

uint32\_t

```
static __always_inline bool ezdp_enqueue_ring (ezdp_ring_t * ring, void __cmem * data,
uint32_t size, char __cmem * work_area_ptr, uint32_t work_area_size, uint32_t flags)
[static]
```

Insert new entry into ring.

**Parameters:**

[in] *ring* - pointer to ring object

[in] *data* - entry data

[in] *size* - entry data size

[in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_RING\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:

EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

true - data inserted to queue successfully false - unable to insert data to queue

```
static __always_inline bool ezdp_dequeue_ring (ezdp_ring_t * ring, void __cmem * data,
uint32_t size, char __cmem * work_area_ptr, uint32_t work_area_size, uint32_t flags)
[static]
```

Remove a head entry from ring.

**Parameters:**

[in] *ring* - pointer ring object

[out] *data* - entry data

[in] *size* - entry data size

[in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_RING\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:

EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

true - return & remove entry from queue successfully false - unable to return & remove entry from queue

```
static __always_inline bool ezdp_init_list (ezdp_list_t * list, struct ezdp_list_cfg * list_config,
char __cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Initialize list.

**Parameters:**

[out] *list* - a pointer to list object to initialize

[in] *list\_config* - configuration information

[in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_LIST\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Returns:**

true - success initialization

```
static __always_inline bool ezdp_list_empty (ezdp_list_t * list) [static]
```

Check if a list is empty.

**Parameters:**

[in] *list* - pointer to list object

**Returns:**

true - list is empty false - list is not empty

```
static __always_inline bool ezdp_enqueue_list (ezdp_list_t * list, void * data, uint8_t size,
char __cmem * work_area_ptr, uint32_t work_area_size, uint32_t flags) [static]
```

Insert a new entry into a list.

**Parameters:**

[in] *list* - pointer to list object

[in] *data* - data to enqueue

[in] *size* - size of data to enqueue (limited to 12B)

[in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_LIST\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

[in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:

EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

true - data inserted to queue successfully false - unable to insert data to queue

```
static __always_inline bool ezdp_dequeue_list (ezdp_list_t * list, void * data, uint8_t size,
char __cmem * work_area_ptr, uint32_t work_area_size, uint32_t flags) [static]
```

Remove a head entry from a list.

**Parameters:**

[in] *list* - pointer to the list object

[out] *data* - data to dequeue

[in] *size* - size of data to dequeue (limited to 12B)

[in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_LIST\_WORK\_AREA\_SIZE  
 [in] *work\_area\_size* - size of work area pointer  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

true - return & remove entry from queue successfully false - unable to return & remove entry from queue.

```
static __always_inline bool ezdp_peek_list(ezdp\_list\_t * list, void * data, uint8_t size, char
__cmem * work_area_ptr, uint32_t work_area_size, uint32_t flags) [static]
```

Peek at head entry of a list.

**Parameters:**

[in] *list* - pointer to the list object  
 [out] *data* - data on peeked head  
 [in] *size* - size of data to peek (limited to 12B)  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_LIST\_WORK\_AREA\_SIZE  
 [in] *work\_area\_size* - size of work area pointer  
 [in] *flags* - execution flags. Bitwise OR of zero or more flags out of the following:  
 EZDP\_MEMORY\_FLAG\_OVERWRITE - override ECC (do not merge)

**Returns:**

true - return & remove entry from list successfully false - unable to return & remove entry from queue.

```
static __always_inline bool ezdp_destroy_list(ezdp\_list\_t * list) [static]
```

Destroy a list.

**Parameters:**

[out] *list* - pointer to the list object

**Returns:**

true - operation succeeded. false - queue is not empty,

## dpe/dp/include/ezdp\_queue\_defs.h File Reference

### Data Structures

- struct [ezdp\\_ring\\_cfg](#)
- *ring (array queue) configuration data structure* struct [ezdp\\_list\\_cfg](#)

### *list queue configuration data structure* Defines

- #define [EZDP\\_RING\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_ring\_working\_area)
- *Work area minimal required size definitions.* #define [EZDP\\_LIST\\_WORK\\_AREA\\_SIZE](#) sizeof(struct ezdp\_list\_working\_area)

### Typedefs

- typedef struct ezdp\_ring [ezdp\\_ring\\_t](#)
  - typedef struct ezdp\_list [ezdp\\_list\\_t](#)
- 

### Define Documentation

**#define EZDP\_RING\_WORK\_AREA\_SIZE** sizeof(struct ezdp\_ring\_working\_area)

Work area minimal required size definitions.

**#define EZDP\_LIST\_WORK\_AREA\_SIZE** sizeof(struct ezdp\_list\_working\_area)

---

### Typedef Documentation

typedef struct ezdp\_ring [ezdp\\_ring\\_t](#)

typedef struct ezdp\_list [ezdp\\_list\\_t](#)

## dpe/dp/include/ezdp\_search.h File Reference

### Functions

- static \_\_always\_inline uint32\_t [ezdp\\_init\\_table\\_struct\\_desc](#) (uint32\_t struct\_id, [ezdp\\_table\\_struct\\_desc\\_t](#) \*table\_struct\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Initialize the structure descriptor for a table structure. static \_\_always\_inline uint32\_t [ezdp\\_validate\\_table\\_struct\\_desc](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*table\_struct\_desc, uint32\_t entry\_size)
- Validate the table structure parameters. static \_\_always\_inline uint32\_t [ezdp\\_lookup\\_table\\_entry](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t key, void \_\_cmem \*entry\_ptr, uint32\_t entry\_ptr\_size, uint32\_t flags)
- Lookup an entry in a table structure. static \_\_always\_inline uint32\_t [ezdp\\_add\\_table\\_entry](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t key, void \_\_cmem \*entry\_ptr, uint32\_t entry\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Add an entry in a table structure. static \_\_always\_inline uint32\_t [ezdp\\_modify\\_table\\_entry](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t key, void \_\_cmem \*entry\_ptr, uint32\_t entry\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Modify an existing entry in a table structure. static \_\_always\_inline uint32\_t [ezdp\\_update\\_table\\_entry](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t key, void \_\_cmem \*entry\_ptr, uint32\_t entry\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Update an entry in a table structure. static \_\_always\_inline uint32\_t [ezdp\\_delete\\_table\\_entry](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t key, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Delete an entry from a table structure. static \_\_always\_inline uint32\_t [ezdp\\_init\\_hash\\_struct\\_desc](#) (uint32\_t struct\_id, [ezdp\\_hash\\_struct\\_desc\\_t](#) \*hash\_struct\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Initialize the structure descriptor for a hash structure. static \_\_always\_inline uint32\_t [ezdp\\_validate\\_hash\\_struct\\_desc](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*hash\_struct\_desc, bool single\_cycle, uint32\_t key\_size, uint32\_t result\_size, uint32\_t entry\_size)
- Validate the hash structure parameters. static \_\_always\_inline uint32\_t [ezdp\\_lookup\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, void \_\_cmem \*key\_ptr, uint32\_t key\_ptr\_size, void \*\*result\_ptr, uint32\_t \*result\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Lookup an entry in a hash structure. static \_\_always\_inline uint32\_t [ezdp\\_add\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, void \_\_cmem \*key\_ptr, uint32\_t key\_ptr\_size, void \_\_cmem \*result\_ptr, uint32\_t result\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Add an entry in a hash structure. static \_\_always\_inline uint32\_t [ezdp\\_modify\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, void \_\_cmem \*key\_ptr, uint32\_t key\_ptr\_size, void \_\_cmem \*result\_ptr, uint32\_t result\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Modify an existing entry in a hash structure. static \_\_always\_inline uint32\_t [ezdp\\_update\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, void \_\_cmem \*key\_ptr, uint32\_t key\_ptr\_size, void \_\_cmem \*result\_ptr, uint32\_t result\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Update an entry in a hash structure. static \_\_always\_inline uint32\_t [ezdp\\_delete\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, void \_\_cmem \*key\_ptr, uint32\_t key\_ptr\_size, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Delete an entry from a hash structure. static \_\_always\_inline uint32\_t [ezdp\\_scan\\_hash\\_slot](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t slot\_num, [ezdp\\_scan\\_entry\\_cb](#) scan\_cb, uintptr\_t user\_data, uint32\_t flags, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Scan a hash slot. static \_\_always\_inline void [ezdp\\_get\\_hash\\_entry\\_key](#) (uint32\_t key\_size, uint32\_t result\_size, uint32\_t entry\_size, void \*entry\_ptr, void \*key\_ptr)
- Get hash key from entry. static \_\_always\_inline uint32\_t [ezdp\\_init\\_ultra\\_ip\\_struct\\_desc](#) (uint32\_t struct\_id, [ezdp\\_ultra\\_ip\\_struct\\_desc\\_t](#) \*uip\_struct\_desc, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Initialize the structure descriptor for a UltraIP structure. static \_\_always\_inline uint32\_t [ezdp\\_validate\\_ultra\\_ip\\_struct\\_desc](#) ([ezdp\\_ultra\\_ip\\_struct\\_desc\\_t](#) \*uip\_struct\_desc, uint32\_t key\_size)
- Validate the UltraIP structure parameters. static \_\_always\_inline uint32\_t [ezdp\\_lookup\\_ultra\\_ip\\_entry](#) ([ezdp\\_ultra\\_ip\\_struct\\_desc\\_t](#) \*struct\_desc, void \_\_cmem \*key\_ptr, uint32\_t key\_size, void \*result\_ptr, uint32\_t result\_size, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)



- *Lookup an entry in an UltraIP structure.* static \_\_always\_inline [ezdp\\_lookup\\_int\\_tcam\\_retval\\_t](#) [ezdp\\_lookup\\_int\\_tcam](#) (uint32\_t side, uint32\_t lookup\_profile, void \*\_\_cmem key\_ptr, uint32\_t key\_size, struct [ezdp\\_lookup\\_int\\_tcam\\_result](#) \*\_\_cmem result\_ptr)
- *Lookup an entry in an internal TCAM.* static \_\_always\_inline void [ezdp\\_lookup\\_int\\_tcam\\_async](#) (uint32\_t side, uint32\_t lookup\_profile, void \*\_\_cmem key\_ptr, uint32\_t key\_size, struct [ezdp\\_lookup\\_int\\_tcam\\_result](#) \*\_\_cmem result\_ptr)
- *Non blocking version of [ezdp\\_lookup\\_int\\_tcam\(\)](#).* static \_\_always\_inline [ezdp\\_lookup\\_ext\\_tcam\\_retval\\_t](#) [ezdp\\_lookup\\_ext\\_tcam](#) (uint32\_t side, uint32\_t lookup\_profile, void \*\_\_cmem key\_ptr, uint32\_t key\_size, char \*\_\_cmem result\_ptr, uint32\_t result\_len)
- *Lookup an entry in an external TCAM.* static \_\_always\_inline void [ezdp\\_lookup\\_ext\\_tcam\\_async](#) (uint32\_t side, uint32\_t lookup\_profile, void \*\_\_cmem key\_ptr, uint32\_t key\_size, char \*\_\_cmem result\_ptr, uint32\_t result\_len)
- *Non blocking version of [ezdp\\_lookup\\_ext\\_tcam\(\)](#).* static \_\_always\_inline uint32\_t [ezdp\\_init\\_alg\\_tcam\\_struct\\_desc](#) (uint32\_t struct\_id, [ezdp\\_alg\\_tcam\\_struct\\_desc\\_t](#) \*alg\_tcam\_struct\_desc, char \*\_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- *Initialize the structure descriptor for an algorithmic TCAM structure.* static \_\_always\_inline uint32\_t [ezdp\\_validate\\_alg\\_tcam\\_struct\\_desc](#) ([ezdp\\_alg\\_tcam\\_struct\\_desc\\_t](#) \*alg\_tcam\_struct\_desc, uint32\_t key\_size)
- *Validate the algorithmic TCAM structure parameters.* static \_\_always\_inline uint32\_t [ezdp\\_lookup\\_alg\\_tcam](#) ([ezdp\\_alg\\_tcam\\_struct\\_desc\\_t](#) \*struct\_desc, uint8\_t \*\_\_cmem \*key\_ptr, uint32\_t key\_size, uint32\_t \*priority\_ptr, char \*\_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)

*Lookup an entry in an algorithmic TCAM structure.*

---

## Function Documentation

```
static __always_inline uint32_t ezdp_init_table_struct_desc (uint32_t struct_id,
ezdp\_table\_struct\_desc\_t * table_struct_desc, char __cmem * work_area_ptr, uint32_t
work_area_size) [static]
```

Initialize the structure descriptor for a table structure.

### Parameters:

- [in] *struct\_id* - search struct id
- [out] *table\_struct\_desc* - table struct descriptor
- [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_TABLE\_LOW\_LEVEL\_WORK\_AREA\_SIZE or [EZDP\\_TABLE\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(entry\\_size\)](#).
- [in] *work\_area\_size* - size of work area pointer

### Returns:

0 (success), EINVAL (failure) use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_validate_table_struct_desc (ezdp\_table\_struct\_desc\_t *
table_struct_desc, uint32_t entry_size) [static]
```

Validate the table structure parameters.

Check that table structure descriptor ,initialized by ezdp\_init\_table\_struct\_desc, is match the parameters of the dp application

### Parameters:

- [in] *table\_struct\_desc* - table struct descriptor
- [in] *entry\_size* - size of the table entry (result)

### Returns:

0 (success), EINVAL (invalid argument) use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_lookup_table_entry(ezdp\_table\_struct\_desc\_t * struct_desc,
uint32_t key, void __cmem * entry_ptr, uint32_t entry_ptr_size, uint32_t flags) [static]
```

Lookup an entry in a table structure.

The lookup result is written to CMEM. Retry if there is memory error

**Parameters:**

[in] *struct\_desc* - table structure description  
[in] *key* - index into table  
[out] *entry\_ptr* - pointer (in CMEM) to write table entry (result)  
[in] *entry\_ptr\_size* - size of the table entry (result)  
[in] *flags* - flags

**Note:**

When the entry is not found (no match), *entry\_ptr* is not valid.

**Returns:**

0 (found), ENOENT (not found), EIO (memory error)

```
static __always_inline uint32_t ezdp_add_table_entry(ezdp\_table\_struct\_desc\_t * struct_desc,
uint32_t key, void __cmem * entry_ptr, uint32_t entry_ptr_size, uint32_t flags, char
__cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Add an entry in a table structure.

The function fail if entry with the same key already exist. The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if not found add entry, unlock.

**Parameters:**

[in] *struct\_desc* - table structure description  
[in] *key* - index into table  
[out] *entry\_ptr* - pointer to updated result (in CMEM)  
[in] *entry\_ptr\_size* - size of the table entry (result)  
[in] *flags* - flags {EZDP\_OPPORTUNISTIC, EZDP\_UNCONDITIONAL}  
[in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_TABLE\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(entry\\_size\)](#).  
[in] *work\_area\_size* - size of work area pointer

**Note:**

When unconditional flag is ON: add entry without lock and without lookup. When opportunistic flag is ON: if already locked or memory error, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock. if found already, will return with failure.

**Returns:**

0 (success), EEXIST (already exist), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_modify_table_entry(ezdp\_table\_struct\_desc\_t * struct_desc,
uint32_t key, void __cmem * entry_ptr, uint32_t entry_ptr_size, uint32_t flags, char
__cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Modify an existing entry in a table structure.

The function fail if entry doesn't exist. The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if found modify entry, unlock.

**Parameters:**

[in] *struct\_desc* - table structure description  
[in] *key* - index into table  
[out] *entry\_ptr* - pointer to updated result (in CMEM)

[in] *entry\_ptr\_size* - size of the table entry (result)  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC, EZDP\_UNCONDITIONAL}  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_TABLE\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(entry\\_size\)](#).  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When unconditional flag is ON: modify entry without lock and without lookup. When opportunistic flag is ON: if already locked or memory error, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock. if not found, will return with failure.

**Returns:**

0 (success), ENOENT (not found), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_update_table_entry(ezdp\_table\_struct\_desc\_t * struct_desc,
uint32_t key, void __cmem * entry_ptr, uint32_t entry_ptr_size, uint32_t flags, char
__cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Update an entry in a table structure.

The function add entry if it doesn't exist and update it if entry with the same key was found. The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if found modify otherwise add entry, unlock.

**Parameters:**

[in] *struct\_desc* - table structure description  
 [in] *key* - index into table  
 [out] *entry\_ptr* - pointer to updated result (in CMEM)  
 [in] *entry\_ptr\_size* - size of the table entry (result)  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC, EZDP\_UNCONDITIONAL}  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_TABLE\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(entry\\_size\)](#).  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When unconditional flag is ON: update entry without lock and without lookup. When opportunistic flag is ON: if already locked or memory error, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock.

**Returns:**

0 (success), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_delete_table_entry(ezdp\_table\_struct\_desc\_t * struct_desc,
uint32_t key, uint32_t flags, char __cmem * work_area_ptr, uint32_t work_area_size)
[static]
```

Delete an entry from a table structure.

The function fail if entry doesn't exist. The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if found delete entry, unlock.

**Parameters:**

[in] *struct\_desc* - table structure description  
 [in] *key* - index into table  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC, EZDP\_UNCONDITIONAL}  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_TABLE\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(entry\\_size\)](#).

[in] *work\_area\_size* - size of work area pointer

#### Note:

When unconditional flag is ON: delete entry without lock and without lookup. When opportunistic flag is ON: if already locked or memory error, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock. if not found, will return with failure.

#### Returns:

0 (success), ENOENT (not found), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_init_hash_struct_desc (uint32_t struct_id,
ezdp\_hash\_struct\_desc\_t * hash_struct_desc,  char __cmem * work_area_ptr,  uint32_t
work_area_size) [static]
```

Initialize the structure descriptor for a hash structure.

#### Parameters:

[in] *struct\_id* - search struct id

[out] *hash\_struct\_desc* - hash struct descriptor

[in] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_HASH\_LOW\_LEVEL\_WORK\_AREA\_SIZE or [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(result\\_size, key\\_size\)](#)

[in] *work\_area\_size* - size of work area pointer

#### Returns:

0 (success), EINVAL (failure) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_validate_hash_struct_desc (ezdp\_hash\_struct\_desc\_t *
hash_struct_desc,  bool single_cycle,  uint32_t key_size,  uint32_t result_size,  uint32_t
entry_size) [static]
```

Validate the hash structure parameters.

Check that hash structure descriptor ,initialized by `ezdp_init_hash_struct_desc`, is match the parameters of the dp application

#### Parameters:

[in] *hash\_struct\_desc* - hash struct descriptor

[in] *single\_cycle* - single-cycle or non-single-cycle hash

[in] *key\_size* - size of the key

[in] *result\_size* - size of the result (for best performance should be a multiple of 4)

[in] *entry\_size* - size of the entry

#### Returns:

0 (success), EINVAL (invalid argument) use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_lookup_hash_entry (ezdp\_hash\_struct\_desc\_t * struct_desc,
void __cmem * key_ptr,  uint32_t key_ptr_size,  void ** result_ptr,  uint32_t * result_ptr_size,
uint32_t flags,  char __cmem * work_area_ptr,  uint32_t work_area_size) [static]
```

Lookup an entry in a hash structure.

Handle memory error. The lookup result is written to CMEM. WARNING! For best performance result\_size should be a multiple of 4

#### Parameters:

[in] *struct\_desc* - hash struct descriptor

[in] *key\_ptr* - pointer to key (in CMEM)  
 [in] *key\_ptr\_size* - key size  
 [out] *result\_ptr* - pointer to returned result on work area  
 [out] *result\_ptr\_size* - returned result size  
 [in] *flags* - flags  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(result\\_size, key\\_size\)](#)  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When the entry is not found (no match), *result\_ptr* is not valid.  
*result\_ptr* point to the data in work area. If you want to reuse the work area but still want to have the result you have to copy the data from work area to other place.

**Returns:**

0 (found), ENOENT (not found), EIO (memory error)

```
static __always_inline uint32_t ezdp_add_hash_entry(ezdp\_hash\_struct\_desc\_t * struct_desc,
void __cmem * key_ptr,  uint32_t key_ptr_size,  void __cmem * result_ptr,  uint32_t
result_ptr_size,  uint32_t flags,  char __cmem * work_area_ptr,  uint32_t work_area_size)
[static]
```

Add an entry in a hash structure.

The function fail if entry already exist. The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if not found add entry, unlock.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *key\_ptr* - pointer to key (in CMEM)  
 [in] *key\_ptr\_size* - key size  
 [in] *result\_ptr* - pointer to updated result (in CMEM)  
 [in] *result\_ptr\_size* - result size  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC}  
 [out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(result\\_size, key\\_size\)](#)  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When opportunistic flag is ON: if already locked or memory error or hash is full, will return with success.  
 When opportunistic flag is OFF: unlimited memory error handling and waiting for lock. if found already or hash is full, will return with failure.

**Returns:**

0 (success), EEXIST (already exist), ENOMEM (hash is full), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_modify_hash_entry(ezdp\_hash\_struct\_desc\_t * struct_desc,
void __cmem * key_ptr,  uint32_t key_ptr_size,  void __cmem * result_ptr,  uint32_t
result_ptr_size,  uint32_t flags,  char __cmem * work_area_ptr,  uint32_t work_area_size)
[static]
```

Modify an existing entry in a hash structure.

The function fail if entry doesn't exist. The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if found modify entry, unlock.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *key\_ptr* - pointer to key (in CMEM)

[in] *key\_ptr\_size* - key size  
 [in] *result\_ptr* - pointer to updated result (in CMEM)  
 [in] *result\_ptr\_size* - result size  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC}  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(result\\_size, key\\_size\)](#)  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When opportunistic flag is ON: if already locked or memory error, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock. if not found, will return with failure.

**Returns:**

0 (success), ENOENT (not found), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_update_hash_entry(ezdp\_hash\_struct\_desc\_t * struct_desc,
void __cmem * key_ptr,  uint32_t key_ptr_size,  void __cmem * result_ptr,  uint32_t
result_ptr_size,  uint32_t flags,  char __cmem * work_area_ptr,  uint32_t work_area_size)
[static]
```

Update an entry in a hash structure.

The function add entry if it doesn't exist or modify if entry found The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if found modify entry otherwise add entry, unlock.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *key\_ptr* - pointer to key (in CMEM)  
 [in] *key\_ptr\_size* - key size  
 [in] *result\_ptr* - pointer to updated result (in CMEM)  
 [in] *result\_ptr\_size* - result size  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC}  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(result\\_size, key\\_size\)](#)  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When opportunistic flag is ON: if already locked or memory error or hash is full, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock. if hash is full, will return with failure.

**Returns:**

0 (success), ENOMEM (hash is full), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_delete_hash_entry(ezdp\_hash\_struct\_desc\_t * struct_desc,
void __cmem * key_ptr,  uint32_t key_ptr_size,  uint32_t flags,  char __cmem *
work_area_ptr,  uint32_t work_area_size) [static]
```

Delete an entry from a hash structure.

The function fail if entry doesn't exist. The procedure: Lock table line, lookup entry, handle memory error (if no flag is set), if found delete entry, unlock.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *key\_ptr* - pointer to key (in CMEM)



[in] *key\_ptr\_size* - key size  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC}  
 [in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(result\\_size, key\\_size\)](#)  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When opportunistic flag is ON: if already locked or memory error, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock.

**Returns:**

0 (success), ENOENT (not found), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_scan_hash_slot(ezdp\_hash\_struct\_desc\_t * struct_desc,
uint32_t slot_num, ezdp\_scan\_entry\_cb scan_cb, uintptr_t user_data, uint32_t flags,
char __cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Scan a hash slot.

Lock hash slot, go over all the entries, handle memory error (if EZDP\_OPPORTUNISTIC flag is not set), apply scan\_cb on each entry and act according to the returned answer, compress (if EZDP\_COMPRESS set), unlock.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *slot\_num* - slot number to scan  
 [in] *scan\_cb* - function to call on each entry in slot  
 [in] *user\_data* - additional data provided by the user for scan\_cb  
 [in] *flags* - flags {EZDP\_OPPORTUNISTIC, EZDP\_COMPRESS}  
 [out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE\(result\\_size, key\\_size\)](#)  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

When opportunistic flag is ON: if already locked or memory error, will return with success. When opportunistic flag is OFF: unlimited memory error handling and waiting for lock.

**Returns:**

0 (success), EIO (memory error) In debug mode use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline void ezdp_get_hash_entry_key(uint32_t key_size, uint32_t result_size,
uint32_t entry_size, void * entry_ptr, void * key_ptr) [static]
```

Get hash key from entry.

The key of the hash entry is copied to the key\_ptr.

**Parameters:**

[in] *key\_size* - size of the key  
 [in] *result\_size* - size of the result  
 [in] *entry\_size* - size of the entry  
 [in] *entry\_ptr* - pointer to entry for internal use  
 [out] *key\_ptr* - pointer to return key

**Returns:**

void

```
static __always_inline uint32_t ezdp_init_ultra_ip_struct_desc (uint32_t struct_id,
ezdp_ultra_ip_struct_desc_t __cmem * uip_struct_desc, char __cmem * work_area_ptr,
uint32_t work_area_size) [static]
```

Initialize the structure descriptor for a UltraIP structure.

#### Parameters:

[in] *struct\_id* - search struct id.  
[out] *uip\_struct\_desc* - UltraIP struct descriptor  
[in,out] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_ULTRA\_IP\_WORK\_AREA\_SIZE.  
[in] *work\_area\_size* - size of work area pointer

#### Returns:

0 (success), EINVAL (failure) use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_validate_ultra_ip_struct_desc (ezdp_ultra_ip_struct_desc_t *
uip_struct_desc, uint32_t key_size) [static]
```

Validate the UltraIP structure parameters.

Check that UltraIP structure descriptor, initialized by *ezdp\_init\_ultra\_ip\_struct\_desc*, is match the parameters of the dp application

#### Parameters:

[in] *uip\_struct\_desc* - UltraIP struct descriptor  
[in] *key\_size* - size of the key

#### Returns:

0 (success), EINVAL (invalid argument) use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_lookup_ultra_ip_entry (ezdp_ultra_ip_struct_desc_t *
struct_desc, void __cmem * key_ptr, uint32_t key_size, void * result_ptr, uint32_t
result_size, char __cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Lookup an entry in an UltraIP structure.

This function also handle memory error.

#### Parameters:

[in] *struct\_desc* - main table summarized address descriptor  
[in] *key\_ptr* - pointer to key (in CMEM)  
[in] *key\_size* - size of key  
[out] *result\_ptr* - pointer to write the result  
[in] *result\_size* - size of result  
[in,out] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_ULTRA\_IP\_WORK\_AREA\_SIZE  
[in] *work\_area\_size* - size of work area pointer

#### Returns:

0 (found), ENOENT (not found), EIO (memory error)

```
static __always_inline ezdp\_lookup\_int\_tcam\_retval\_t ezdp_lookup_int_tcam (uint32_t side,
uint32_t lookup_profile, void * __cmem key_ptr, uint32_t key_size, struct
ezdp\_lookup\_int\_tcam\_result * __cmem result_ptr) [static]
```

Lookup an entry in an internal TCAM.



The lookup is according to the TCAM profile. The lookup result is written to CMEM. In addition, the first 4 bytes of the lookup result are returned.

**Parameters:**

- [in] *side* - NPS-400 has two internal TCAM engines. This field select which one to use (the iTCAM on side 0 or on side 1).
- [in] *lookup\_profile* - lookup profile id. select the search definition, which defines up to 4 profiles
- [in] *key\_ptr* - pointer to key (in CMEM)
- [in] *key\_size* - size of the key; limited to 10, 20, 30, 40, 50, 60, 70, 80 bytes
- [out] *result\_ptr* - pointer to result (in CMEM)

**Returns:**

ezdp\_lookup\_int\_tcam\_retval\_t - according to [ezdp\\_lookup\\_int\\_tcam\\_retval](#)

```
static __always_inline void ezdp_lookup_int_tcam_async (uint32_t side,    uint32_t
lookup_profile,    void *__cmem key_ptr,    uint32_t key_size,    struct
ezdp\_lookup\_int\_tcam\_result *__cmem result_ptr) [static]
```

Non blocking version of [ezdp\\_lookup\\_int\\_tcam\(\)](#).

**Parameters:**

- [in] *side* - NPS-400 has two internal TCAM engines. This field select which one to use (the iTCAM on side 0 or on side 1).
- [in] *lookup\_profile* - lookup profile id. select the search definition, which defines up to 4 profiles
- [in] *key\_ptr* - pointer to key (in CMEM)
- [in] *key\_size* - size of the key; limited to 10, 20, 30, 40, 50, 60, 70, 80 bytes
- [out] *result\_ptr* - pointer to result (in CMEM)

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the lookup result is ready in CMEM.

**Returns:**

none

```
static __always_inline ezdp\_lookup\_ext\_tcam\_retval\_t ezdp_lookup_ext_tcam (uint32_t side,
uint32_t lookup_profile,    void *__cmem key_ptr,    uint32_t key_size,    char *__cmem
result_ptr,    uint32_t result_len) [static]
```

Lookup an entry in an external TCAM.

The lookup is according to the TCAM profile The lookup result is written to CMEM. RFLAGS are returned.

**Parameters:**

- [in] *side* - NPS-400 supports two external TCAM engines. This field select which one to use (the eTCAM on side 0 or on side 1).
- [in] *lookup\_profile* - lookup profile id. select the search definition profile
- [in] *key\_ptr* - pointer to key (in CMEM)
- [in] *key\_size* - size of the key; limited to 8, 10, 16, 20, 24, 30, 32, 40 and 80 bytes
- [out] *result\_ptr* - pointer to result (in CMEM) which consists of up to 6 result elements, based on lookup\_profile definition. a result element may be one of the following:  
[ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element](#) [ezdp\\_lookup\\_ext\\_tcam\\_index\\_4B\\_data\\_result\\_element](#)  
[ezdp\\_lookup\\_ext\\_tcam\\_index\\_8B\\_data\\_result\\_element](#)  
[ezdp\\_lookup\\_ext\\_tcam\\_index\\_16B\\_data\\_result\\_element](#)  
[ezdp\\_lookup\\_ext\\_tcam\\_index\\_32B\\_data\\_result\\_element](#) [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element](#)  
[ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element](#) [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element](#)  
[ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element](#)
- [in] *result\_len* - the lookup result length limit in bytes; 16, 32, 48 and 64 bytes

**Returns:**

ezdp\_lookup\_ext\_tcam\_retval\_t - according to [ezdp\\_lookup\\_ext\\_tcam\\_retval](#)

```
static __always_inline void ezdp_lookup_ext_tcam_async (uint32_t side,   uint32_t
lookup_profile,   void *__cmem key_ptr,   uint32_t key_size,   char *__cmem result_ptr,
uint32_t result_len) [static]
```

Non blocking version of [ezdp\\_lookup\\_ext\\_tcam\(\)](#).

**Parameters:**

[in] *side* - NPS-400 supports two external TCAM engines. This field select which one to use (the eTCAM on side 0 or on side 1).

[in] *lookup\_profile* - lookup profile id. select the search definition profile

[in] *key\_ptr* - pointer to key (in CMEM)

[in] *key\_size* - size of the key; limited to 8, 10, 16, 20, 24, 30, 32, 40 and 80 bytes

[out] *result\_ptr* - pointer to result (in CMEM) which consists of up to 6 result elements, based on lookup\_profile definition. a result element may be one of the following:

[ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element](#) [ezdp\\_lookup\\_ext\\_tcam\\_index\\_4B\\_data\\_result\\_element](#)

[ezdp\\_lookup\\_ext\\_tcam\\_index\\_8B\\_data\\_result\\_element](#)

[ezdp\\_lookup\\_ext\\_tcam\\_index\\_16B\\_data\\_result\\_element](#)

[ezdp\\_lookup\\_ext\\_tcam\\_index\\_32B\\_data\\_result\\_element](#) [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element](#)

[ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element](#) [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element](#)

[ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element](#)

[in] *result\_len* - the lookup result length limit in bytes; 16, 32, 48 and 64 bytes

**Returns:**

void

```
static __always_inline uint32_t ezdp_init_alg_tcam_struct_desc (uint32_t struct_id,
ezdp_alg_tcam_struct_desc_t * alg_tcam_struct_desc,   char *__cmem * work_area_ptr,
uint32_t work_area_size) [static]
```

Initialize the structure descriptor for an algorithmic TCAM structure.

**Parameters:**

[in] *struct\_id* - search struct id.

[out] *alg\_tcam\_struct\_desc* - algorithmic TCAM struct descriptor

[in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_ALG\_TCAM\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Returns:**

0 (success), EINVAL (failure). use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_validate_alg_tcam_struct_desc
(ezdp_alg_tcam_struct_desc_t * alg_tcam_struct_desc,   uint32_t key_size) [static]
```

Validate the algorithmic TCAM structure parameters.

Check that alg TCAM structure descriptor, initialized by `ezdp_init_alg_tcam_struct_desc`, is match the parameters of the dp application

**Parameters:**

[in] *alg\_tcam\_struct\_desc* - algorithmic TCAM struct descriptor

[in] *key\_size* - size of the key

**Returns:**

0 (success), EINVAL (invalid argument) use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline uint32_t ezdp_lookup_alg_tcam (ezdp\_alg\_tcam\_struct\_desc\_t *  
struct_desc, uint8_t __cmem * key_ptr, uint32_t key_size, uint32_t * priority_ptr, char  
__cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Lookup an entry in an algorithmic TCAM structure.

Handle memory error.

**Parameters:**

[in] *struct\_desc* - algo tcam struct descriptor

[in] *key\_ptr* - pointer to the key in CMEM

[in] *key\_size* - size of the key

[out] *priority\_ptr* - pointer to returned priority

[in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_ALG\_TCAM\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Returns:**

errno: 0 (found), ENOENT (not found), EIO (memory error).

## dpe/dp/include/ezdp\_search\_defs.h File Reference

### Data Structures

- struct [ezdp\\_lookup\\_retval](#)
- Lookup return value. struct [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result](#)
- Lookup internal tcam standard result. struct [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result](#)
- Lookup internal tcam 4 byte associated data result. struct [ezdp\\_lookup\\_int\\_tcam\\_8B\\_data\\_result](#)
- Lookup internal tcam 8 byte associated data result. struct [ezdp\\_lookup\\_int\\_tcam\\_12B\\_data\\_result](#)
- Lookup internal tcam 12 byte associated data result. struct [ezdp\\_lookup\\_int\\_tcam\\_16B\\_data\\_result](#)
- Lookup internal tcam 16 byte associated data result. struct [ezdp\\_lookup\\_ext\\_tcam\\_retval](#)
- Lookup external tcam return value. struct [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element](#)
- Lookup external tcam index result element. struct [ezdp\\_lookup\\_ext\\_tcam\\_index\\_4B\\_data\\_result\\_element](#)
- Lookup external tcam index result with 4 Byte associated data. struct [ezdp\\_lookup\\_ext\\_tcam\\_index\\_8B\\_data\\_result\\_element](#)
- Lookup external tcam index result with 8 Byte associated data. struct [ezdp\\_lookup\\_ext\\_tcam\\_index\\_16B\\_data\\_result\\_element](#)
- Lookup external tcam index result with 16 Byte associated data. struct [ezdp\\_lookup\\_ext\\_tcam\\_index\\_32B\\_data\\_result\\_element](#)
- Lookup external tcam index result with 32 Byte associated data. struct [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element](#)
- Lookup external tcam 4 Byte associated data only result. struct [ezdp\\_lookup\\_ext\\_tcam\\_8B\\_data\\_result\\_element](#)
- Lookup external tcam 8 Byte associated data only result. struct [ezdp\\_lookup\\_ext\\_tcam\\_16B\\_data\\_result\\_element](#)
- Lookup external tcam 16 Byte associated data only result. struct [ezdp\\_lookup\\_ext\\_tcam\\_32B\\_data\\_result\\_element](#)
- Lookup external tcam 32 Byte associated data only result. struct [ezdp\\_lookup\\_int\\_tcam\\_result](#)
- Lookup ITCAM result definition. struct [ezdp\\_lookup\\_int\\_tcam\\_retval](#)

### Lookup ITCAM retval definition. Defines

- #define [EZDP\\_LOOKUP\\_VERSION\\_MAJOR](#) 2
- #define [EZDP\\_LOOKUP\\_VERSION\\_MINOR](#) 1
- #define [EZDP\\_ALG\\_TCAM\\_MAX\\_KEY\\_SIZE](#) 128
- #define [EZDP\\_PAD\\_ALG\\_TCAM\\_WORKING\\_AREA](#)(key\_size)
- #define [EZDP\\_PAD\\_HASH\\_ENTRY](#)(result\_size, key\_size) uint8\_t  
\_\_pad[\_EZDP\_HASH\_CALC\_ENTRY\_PADDING\_SIZE(result\_size, key\_size)]
- #define [EZDP\\_PAD\\_HASH\\_WORKING\\_AREA](#)(result\_size, key\_size) uint8\_t  
\_\_pad[\_EZDP\_LOOKUP\_HASH\_CALC\_ENTRY\_SIZE(result\_size, key\_size) + sizeof(ezdp\_hash\_op\_ctx\_t)]
- #define  
[EZDP\\_TABLE\\_LOW\\_LEVEL\\_WORK\\_AREA\\_SIZE](#) \_EZDP\_TABLE\_LOW\_LEVEL\_WORK\_AREA\_SIZE
- Work area minimal required size definitions. #define  
[EZDP\\_TABLE\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE](#)(entry\_size) \_EZDP\_TABLE\_HIGH\_LEVEL\_WORK  
\_AREA\_SIZE(entry\_size)
- #define  
[EZDP\\_HASH\\_LOW\\_LEVEL\\_WORK\\_AREA\\_SIZE](#) \_EZDP\_HASH\_LOW\_LEVEL\_WORK\_AREA\_SIZE
- #define [EZDP\\_HASH\\_HIGH\\_LEVEL\\_WORK\\_AREA\\_SIZE](#)(result\_size,  
key\_size) \_EZDP\_HASH\_HIGH\_LEVEL\_WORK\_AREA\_SIZE(result\_size, key\_size)
- #define [EZDP\\_ULTRA\\_IP\\_WORK\\_AREA\\_SIZE](#) \_EZDP\_ULTRA\_IP\_WORK\_AREA\_SIZE
- #define  
[EZDP\\_ALG\\_TCAM\\_WORK\\_AREA\\_SIZE](#)(max\_key\_size) \_EZDP\_ALG\_TCAM\_WORK\_AREA\_SIZE(max\_key\_size)
- #define [EZDP\\_LOOKUP\\_PARITY\\_BITS\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_RESERVED\\_BITS\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_RETVAL\\_DATA\\_SIZE](#) 28
- #define [EZDP\\_LOOKUP\\_RETVAL\\_DATA\\_OFFSET](#) 0

```

• #define EZDP_LOOKUP_RETVAL_MATCH_SIZE 1
• #define EZDP_LOOKUP_RETVAL_MATCH_OFFSET 28
• #define EZDP_LOOKUP_RETVAL_MATCH_MASK (1 <<
EZDP_LOOKUP_RETVAL_MATCH_OFFSET)
• #define EZDP_LOOKUP_RETVAL_SUCCESS_SIZE 1
• #define EZDP_LOOKUP_RETVAL_SUCCESS_OFFSET 29
• #define EZDP_LOOKUP_RETVAL_SUCCESS_MASK (1 <<
EZDP_LOOKUP_RETVAL_SUCCESS_OFFSET)
• #define EZDP_LOOKUP_RETVAL_INFO_SIZE 1
• #define EZDP_LOOKUP_RETVAL_INFO_OFFSET 30
• #define EZDP_LOOKUP_RETVAL_INFO_MASK (1 << EZDP_LOOKUP_RETVAL_INFO_OFFSET)
• #define EZDP_LOOKUP_RETVAL_MEM_ERROR_SIZE 1
• #define EZDP_LOOKUP_RETVAL_MEM_ERROR_OFFSET 31
• #define EZDP_LOOKUP_RETVAL_MEM_ERROR_MASK (1 <<
EZDP_LOOKUP_RETVAL_MEM_ERROR_OFFSET)
• #define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_INDEX_SIZE 15
• #define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_INDEX_OFFSET 0
• #define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_RESERVED0_15_SIZE 16
• #define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_RESERVED0_15_OFFSET 15
• #define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_SIZE 1
• #define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_OFFSET 31
• #define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_OFFSET)
• #define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_DATA_SIZE 31
• #define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_DATA_OFFSET 0
• #define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_SIZE 1
• #define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_OFFSET 31
• #define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_OFFSET)
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_SIZE 31
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_OFFSET 0
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_WORD_SELECT 0
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_WORD_OFFSET 0
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_SIZE 1
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_OFFSET 31
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_WORD_SELECT 0
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_WORD_OFFSET 31
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_WORD_OFFSET)
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_SIZE 32
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_OFFSET 32
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_WORD_SELECT 1
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_WORD_OFFSET 0
• #define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_WORD_COUNT 2
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_SIZE 31
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_OFFSET 0
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_WORD_SELECT 0
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_WORD_OFFSET 0
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_SIZE 1
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_OFFSET 31
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_WORD_SELECT 0
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_WORD_OFFSET 31
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_WORD_OFFSET)
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA1_SIZE 32
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA1_OFFSET 32
• #define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA1_WORD_SELECT 1

```

- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_12B\\_DATA\\_RESULT\\_DATA1\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_12B\\_DATA\\_RESULT\\_DATA2\\_SIZE](#) 32
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_12B\\_DATA\\_RESULT\\_DATA2\\_OFFSET](#) 64
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_12B\\_DATA\\_RESULT\\_DATA2\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_12B\\_DATA\\_RESULT\\_DATA2\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_12B\\_DATA\\_RESULT\\_WORD\\_COUNT](#) 3
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA0\\_SIZE](#) 31
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA0\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA0\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA0\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_MATCH\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_MATCH\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_MATCH\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT\_MATCH\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA1\\_SIZE](#) 32
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA1\\_OFFSET](#) 32
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA1\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA1\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA2\\_SIZE](#) 32
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA2\\_OFFSET](#) 64
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA2\\_WORD\\_SELECT](#) 2
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA2\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA3\\_SIZE](#) 32
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA3\\_OFFSET](#) 96
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA3\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_DATA3\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_16B\\_DATA\\_RESULT\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_NO\\_CONTEXT\\_MATCH\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_NO\\_CONTEXT\\_MATCH\\_ERROR\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_NO\\_CONTEXT\\_MATCH\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONTEXT\_MATCH\_ERROR\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_MAC\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_MAC\\_ERROR\\_OFFSET](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_MAC\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ERROR\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_DEVICE\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_DEVICE\\_ERROR\\_OFFSET](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_DEVICE\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_ERROR\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_TIME\\_OUT\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_TIME\\_OUT\\_ERROR\\_OFFSET](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_TIME\\_OUT\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_OUT\_ERROR\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_ANY\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_ANY\\_MATCH\\_OFFSET](#) 4
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_ANY\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MATCH\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_MULTI\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_MULTI\\_MATCH\\_OFFSET](#) 5
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_MULTI\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_MATCH\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_TRUNCATED\\_OFFSET](#) 6
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNCATED\_OFFSET)



- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_LOOKUP\\_ERROR\\_OFFSET](#) 7
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_ERROR\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_RESERVED\\_BIT8\\_31\\_SIZE](#) 24
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_RETVAL\\_RESERVED\\_BIT8\\_31\\_OFFSET](#) 8
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_INDEX\\_SIZE](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_INDEX\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_RESERVED23\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_RESERVED23\\_OFFSET](#) 23
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_ANY\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_ANY\\_MATCH\\_OFFSET](#) 24
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_ANY\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_MATCH\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_VALID\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_SIZE](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_OFFSET](#) 23
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 25

- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_SIZE](#) 32
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_OFFSET](#) 32
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_COUNT](#) 4
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_SELECT](#) 1



- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_SIZE](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_OFFSET](#) 23
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1

- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_SIZE](#) 64
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_OFFSET](#) 32
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_COUNT](#) 8
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_WORD\\_COUNT](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_SIZE](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_INDEX\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_OFFSET](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 <<

```

EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET)
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE 1
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET 26
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT 0
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET 26
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK (1 << EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET)
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE 3
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET 27
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT 0
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET 27
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE 1
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET 30
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT 0
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET 30
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK (1 << EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET)
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE 1
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET 31
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT 0
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET 31
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK (1 << EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET)
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE 128
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET 32
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT 16
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT 1
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET 0
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT 5
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE 21
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_OFFSET 0
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_SELECT 0
• #define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSET 0

```

- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_DEVICE\\_ID\\_WORD\\_OFFSET](#) 21
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_SIZE](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED23\\_24\\_OFFSET](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 31

- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_SIZE](#) 256
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_OFFSET](#) 32
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_COUNT](#) 32
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_INDEX\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_WORD\\_COUNT](#) 9
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_SIZE](#) 24
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_COUNT](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_OFFSET](#) 24
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_4B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_SIZE](#) 56
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_COUNT](#) 7
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_OFFSET](#) 56
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 57
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_OFFSET](#) 25

- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 58
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 59
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 62
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 63
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_SELECT](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_8B\\_DATA\\_RESULT\\_ELEMENT\\_WORD\\_COUNT](#) 2
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_SIZE](#) 120
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_COUNT](#) 15
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_OFFSET](#) 120
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 121
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 122
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET)



- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 123
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 126
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 127
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_SELECT](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_16B\\_DATA\\_RESULT\\_ELEMENT\\_WORD\\_COUNT](#) 4
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_SIZE](#) 248
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_COUNT](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_SELECT](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_ASSOC\\_DATA\\_WORD\\_OFFSET](#) 0
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_RESERVED24\\_OFFSET](#) 248
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_OFFSET](#) 249
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_SELECT](#) 7
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_WORD\\_OFFSET](#) 25
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TRUNCATED\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_OFFSET](#) 250
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_SELECT](#) 7
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_WORD\\_OFFSET](#) 26
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_LOOKUP\\_ERROR\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_SIZE](#) 3
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_OFFSET](#) 251
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_SELECT](#) 7
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_TYPE\\_WORD\\_OFFSET](#) 27
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_OFFSET](#) 254
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_SELECT](#) 7
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_WORD\\_OFFSET](#) 30
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_MATCH\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_SIZE](#) 1
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_OFFSET](#) 255

- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_SELECT](#) 7
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_WORD\\_OFFSET](#) 31
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_VALID\\_MASK](#) (1 << EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET)
- #define [EZDP\\_LOOKUP\\_EXT\\_TCAM\\_32B\\_DATA\\_RESULT\\_ELEMENT\\_WORD\\_COUNT](#) 8
- #define [EZDP\\_LOOKUP\\_INT\\_TCAM\\_STANDARD\\_RESULT\\_MAX\\_NUM](#) 4

## Typedefs

- typedef uint32\_t [ezdp\\_hashed\\_key\\_t](#)
- typedef uint32\_t [ezdp\\_lookup\\_retval\\_t](#)
- typedef uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result\\_t](#)
- typedef uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result\\_t](#)
- typedef uint32\_t [ezdp\\_lookup\\_ext\\_tcam\\_retval\\_t](#)
- typedef uint32\_t [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element\\_t](#)
- typedef uint32\_t [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element\\_t](#)
- typedef struct ezdp\_table\_struct\_desc [ezdp\\_table\\_struct\\_desc\\_t](#)
- typedef struct ezdp\_hash\_struct\_desc [ezdp\\_hash\\_struct\\_desc\\_t](#)
- typedef struct ezdp\_alg\_tcam\_struct\_desc [ezdp\\_alg\\_tcam\\_struct\\_desc\\_t](#)
- typedef [ezdp\\_sum\\_addr\\_table\\_desc\\_t](#) [ezdp\\_search\\_base\\_addr\\_t](#)
- typedef uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_retval\\_t](#)

## Enumerations

- enum [ezdp\\_ext\\_tcam\\_result\\_element\\_type](#) { [EZDP\\_INDEX](#) = 0x0, [EZDP\\_INDEX\\_4B\\_DATA](#) = 0x1, [EZDP\\_INDEX\\_8B\\_DATA](#) = 0x2, [EZDP\\_INDEX\\_16B\\_DATA](#) = 0x3, [EZDP\\_INDEX\\_32B\\_DATA](#) = 0x4, [EZDP\\_USER\\_DEFINED\\_ASSOC\\_DATA1](#) = 0x5, [EZDP\\_USER\\_DEFINED\\_ASSOC\\_DATA2](#) = 0x6, [EZDP\\_USER\\_DEFINED\\_ASSOC\\_DATA3](#) = 0x7 }
- ext tcam result element type possible values.*
- enum [ezdp\\_search\\_hash\\_flags](#) { [EZDP\\_NONE](#) = 0x0, [EZDP\\_OPPORTUNISTIC](#) = 0x1, [EZDP\\_COMPRESS](#) = 0x2, [EZDP\\_UNCONDITIONAL](#) = 0x4 }
- search hash flags.*
- enum [ezdp\\_scan\\_hash\\_slot\\_action](#) { [EZDP\\_ACCEPT\\_ENTRY](#) = 0x0, [EZDP\\_DELETE\\_ENTRY](#) = 0x1, [EZDP\\_UPDATE\\_ENTRY](#) = 0x2 }
- scan hash slot callback function possible return values.*

## Variables

- enum [ezdp\\_scan\\_hash\\_slot\\_action](#)(\* [ezdp\\_scan\\_entry\\_cb](#))(void \_\_cmem \*result\_ptr, struct ezdp\_hash\_struct\_desc \*hash\_desc, uintptr\_t user\_data)

## Define Documentation

**#define EZDP\_LOOKUP\_VERSION\_MAJOR 2**

**#define EZDP\_LOOKUP\_VERSION\_MINOR 1**

**#define EZDP\_ALG\_TCAM\_MAX\_KEY\_SIZE 128**

**#define EZDP\_PAD\_ALG\_TCAM\_WORKING\_AREA(key\_size)**

```
Value: union \
{ \
    uint8_t
    acl_line[EZDP_ALG_TCAM_WORKING_AREA_ACL_LINE_COUNT]; \
    struct ezdp_driver_algtcam_main_table_line main_table_line; \
}
```



```

}; \

\
union \
{ \
    struct ezdp\_lookup\_int\_tcam\_result          tcam_result; \
    struct ezdp\_sum\_addr                        memcmp_sd; \
    uint8_t
    memcmp_array[_EZDP_ALG_TCAM_WORKING_AREA_CALC_PADDING_SIZE(key_size)]; \
};

```

```

#define EZDP_PAD_HASH_ENTRY(result_size,  key_size) uint8_t
__pad[_EZDP_HASH_CALC_ENTRY_PADDING_SIZE(result_size, key_size)]

```

```

#define EZDP_PAD_HASH_WORKING_AREA(result_size,  key_size) uint8_t
__pad[_EZDP_LOOKUP_HASH_CALC_ENTRY_SIZE(result_size, key_size) +
sizeof(ezdp_hash_op_ctx_t)]

```

```

#define
EZDP_TABLE_LOW_LEVEL_WORK_AREA_SIZE  _EZDP_TABLE_LOW_LEVEL_WORK_AREA_SI
ZE

```

Work area minimal required size definitions.

```
#define
EZDP_TABLE_HIGH_LEVEL_WORK_AREA_SIZE(entry_size) _EZDP_TABLE_HIGH_LEVEL_WOR
K_AREA_SIZE(entry_size)

#define
EZDP_HASH_LOW_LEVEL_WORK_AREA_SIZE _EZDP_HASH_LOW_LEVEL_WORK_AREA_SIZE

#define EZDP_HASH_HIGH_LEVEL_WORK_AREA_SIZE(result_size,
key_size) _EZDP_HASH_HIGH_LEVEL_WORK_AREA_SIZE(result_size, key_size)

#define EZDP_ULTRA_IP_WORK_AREA_SIZE _EZDP_ULTRA_IP_WORK_AREA_SIZE

#define
EZDP_ALG_TCAM_WORK_AREA_SIZE(max_key_size) _EZDP_ALG_TCAM_WORK_AREA_SIZE(
max_key_size)

#define EZDP_LOOKUP_PARITY_BITS_SIZE 3

#define EZDP_LOOKUP_RESERVED_BITS_SIZE 1

#define EZDP_LOOKUP_RETVAL_DATA_SIZE 28

#define EZDP_LOOKUP_RETVAL_DATA_OFFSET 0

#define EZDP_LOOKUP_RETVAL_MATCH_SIZE 1

#define EZDP_LOOKUP_RETVAL_MATCH_OFFSET 28

#define EZDP_LOOKUP_RETVAL_MATCH_MASK (1 <<
EZDP_LOOKUP_RETVAL_MATCH_OFFSET)

#define EZDP_LOOKUP_RETVAL_SUCCESS_SIZE 1

#define EZDP_LOOKUP_RETVAL_SUCCESS_OFFSET 29

#define EZDP_LOOKUP_RETVAL_SUCCESS_MASK (1 <<
EZDP_LOOKUP_RETVAL_SUCCESS_OFFSET)

#define EZDP_LOOKUP_RETVAL_INFO_SIZE 1

#define EZDP_LOOKUP_RETVAL_INFO_OFFSET 30

#define EZDP_LOOKUP_RETVAL_INFO_MASK (1 << EZDP_LOOKUP_RETVAL_INFO_OFFSET)

#define EZDP_LOOKUP_RETVAL_MEM_ERROR_SIZE 1

#define EZDP_LOOKUP_RETVAL_MEM_ERROR_OFFSET 31

#define EZDP_LOOKUP_RETVAL_MEM_ERROR_MASK (1 <<
EZDP_LOOKUP_RETVAL_MEM_ERROR_OFFSET)

#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_INDEX_SIZE 15

#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_INDEX_OFFSET 0
```

```
#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_RESERVED0_15_SIZE 16

#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_RESERVED0_15_OFFSET 15

#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_SIZE 1

#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MATCH_OFFSET)

#define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_DATA_SIZE 31

#define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_DATA_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_SIZE 1

#define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_4B_DATA_RESULT_MATCH_OFFSET)

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_SIZE 31

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_WORD_SELECT 0

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA0_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_SIZE 1

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_WORD_SELECT 0

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_WORD_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_SIZE 32

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_OFFSET 32

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_WORD_SELECT 1

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_DATA1_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_8B_DATA_RESULT_WORD_COUNT 2

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_SIZE 31
```

```
#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_WORD_SELECT 0

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA0_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_SIZE 1

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_WORD_SELECT 0

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_WORD_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA1_SIZE 32

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA1_OFFSET 32

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA1_WORD_SELECT 1

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA1_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA2_SIZE 32

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA2_OFFSET 64

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA2_WORD_SELECT 2

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_DATA2_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_12B_DATA_RESULT_WORD_COUNT 3

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA0_SIZE 31

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA0_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA0_WORD_SELECT 0

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA0_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_MATCH_SIZE 1

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_MATCH_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_MATCH_WORD_SELECT 0

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_MATCH_WORD_OFFSET 31

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_MATCH_MASK (1 <<
EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_MATCH_WORD_OFFSET)
```

```
#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA1_SIZE 32

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA1_OFFSET 32

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA1_WORD_SELECT 1

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA1_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA2_SIZE 32

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA2_OFFSET 64

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA2_WORD_SELECT 2

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA2_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA3_SIZE 32

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA3_OFFSET 96

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA3_WORD_SELECT 3

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_DATA3_WORD_OFFSET 0

#define EZDP_LOOKUP_INT_TCAM_16B_DATA_RESULT_WORD_COUNT 4

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_NO_CONTEXT_MATCH_ERROR_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_NO_CONTEXT_MATCH_ERROR_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_NO_CONTEXT_MATCH_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_NO_CONTEXT_MATCH_ERROR_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_MAC_ERROR_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_MAC_ERROR_OFFSET 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_MAC_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_MAC_ERROR_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_DEVICE_ERROR_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_DEVICE_ERROR_OFFSET 2

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_DEVICE_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_DEVICE_ERROR_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_TIME_OUT_ERROR_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_TIME_OUT_ERROR_OFFSET 3

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_TIME_OUT_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_TIME_OUT_ERROR_OFFSET)
```

```
#define EZDP_LOOKUP_EXT_TCAM_RETVAL_ANY_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_ANY_MATCH_OFFSET 4

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_ANY_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_ANY_MATCH_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_MULTI_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_MULTI_MATCH_OFFSET 5

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_MULTI_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_MULTI_MATCH_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_TRUNCATED_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_TRUNCATED_OFFSET 6

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_TRUNCATED_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_LOOKUP_ERROR_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_LOOKUP_ERROR_OFFSET 7

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_LOOKUP_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_RETVAL_LOOKUP_ERROR_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_RESERVED_BIT8_31_SIZE 24

#define EZDP_LOOKUP_EXT_TCAM_RETVAL_RESERVED_BIT8_31_OFFSET 8

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_INDEX_SIZE 21

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_INDEX_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_DEVICE_ID_SIZE 2

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_DEVICE_ID_OFFSET 21

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_RESERVED23_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_RESERVED23_OFFSET 23

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_ANY_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_ANY_MATCH_OFFSET 24

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_ANY_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_ANY_MATCH_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_TRUNCATED_SIZE 1
```

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_TRUNCATED_OFFSET 25

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_TRUNCATED_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET 26

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_LOOKUP_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_TYPE_OFFSET 27

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_MATCH_OFFSET 30

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_MATCH_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_VALID_OFFSET 31

#define EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_VALID_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_RESULT_ELEMENT_VALID_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_SIZE 21

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_OFFSET 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_WORD_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_SIZE 2

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_OFFSET 21

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_SELECT
0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_OFFSET
21
```

```
#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_RESERVED23_24_SIZE 2

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_RESERVED23_24_OFFSET 2
3

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET)

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET 26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET 26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (1
<<
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TYPE_OFFSET 27

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TYPE_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_TYPE_WORD_OFFSET 27
```



```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_OFFSET 30

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET 30

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_VALID_OFFSET 31

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_VALID_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET 31

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_VALID_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET)

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE 32

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET 32

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT 4

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_SELECT 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_WORD_COUNT 2

#define EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_INDEX_SIZE 21

#define EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_INDEX_OFFSET 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_INDEX_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_INDEX_WORD_OFFSET 0
```

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_DEVICE_ID_SIZE 2

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_DEVICE_ID_OFFSET 21

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_SELECT
0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_OFFSET
21

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_RESERVED23_24_SIZE 2

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_RESERVED23_24_OFFSET 2
3

#define EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELEC
T 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSE
T 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSE
T)

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET
26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_SE
LECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_O
FFSET 26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (1
<<
```

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET)

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE 3

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET 27

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT 0

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET 27

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE 1

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET 30

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT 0

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET 30

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK (1 <<  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET)

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE 1

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET 31

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT 0

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET 31

#define EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK (1 <<  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET)

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE 64

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET 32

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT 8

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT 1

#define  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET 0

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_8B_DATA_RESULT_ELEMENT_WORD_COUNT 3

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_INDEX_SIZE 21

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_INDEX_OFFSET 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_INDEX_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_INDEX_WORD_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_DEVICE_ID_SIZE 2

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_DEVICE_ID_OFFSET 21

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_SELECT
0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_OFFSET
21

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_RESERVED23_24_SIZE 2

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_RESERVED23_24_OFFSET
23

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELE
CT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFS
ET 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFS
ET)

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1
```

```
#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET
26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_S
ELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_O
FFSET 26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (
1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_O
FFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TYPE_OFFSET 27

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TYPE_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_TYPE_WORD_OFFSET 27

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_MATCH_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_MATCH_OFFSET 30

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET 3
0

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_MATCH_MASK (1
<<
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_VALID_OFFSET 31

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_VALID_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET 31
```

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_VALID_MASK (1 <<  
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET)
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE 128
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET 32
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT 16
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_SELECT 1
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_OFFSET 0
```

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_16B_DATA_RESULT_ELEMENT_WORD_COUNT 5
```

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_INDEX_SIZE 21
```

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_INDEX_OFFSET 0
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_INDEX_WORD_SELECT 0
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_INDEX_WORD_OFFSET 0
```

```
#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_DEVICE_ID_SIZE 2
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_DEVICE_ID_OFFSET 21
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_SELECT  
0
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_OFFSET  
21
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_RESERVED23_24_SIZE 2
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_RESERVED23_24_OFFSET  
23
```

```
#define  
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1
```

```
#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET 25

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET)

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET 26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET 26

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (
1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_OFFSET 27

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_WORD_OFFSET 27

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_OFFSET 30

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT 0
```

```
#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET 3
0

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_MASK (1
<<
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_OFFSET 31

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET 31

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET)

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE 256

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET 32

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT 32

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_SEL
ECT 1

#define
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_OFF
SET 0

#define EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_WORD_COUNT 9

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE 24

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT 3

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_RESERVED24_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_RESERVED24_OFFSET 24

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 25
```



```
#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET 26

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (1
<< EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_TYPE_OFFSET 27

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_MATCH_OFFSET 30

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_MATCH_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_VALID_OFFSET 31

#define EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_VALID_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_4B_DATA_RESULT_ELEMENT_VALID_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE 56

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT 7

#define
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_RESERVED24_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_RESERVED24_OFFSET 56

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 57

#define
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELECT 1

#define
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET 25
```

```
#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET 58

#define
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_SELECT
1

#define
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET
26

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (1
<<
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TYPE_OFFSET 59

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TYPE_WORD_SELECT 1

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_TYPE_WORD_OFFSET 27

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_MATCH_OFFSET 62

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT 1

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET 30

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_VALID_OFFSET 63

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_VALID_WORD_SELECT 1

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET 31

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_VALID_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_8B_DATA_RESULT_ELEMENT_WORD_COUNT 2

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE 120
```

```
#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT 15

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_RESERVED24_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_RESERVED24_OFFSET 120

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 121

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELECT 3

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET 25

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET 122

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_SELECT
3

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET
26

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TYPE_OFFSET 123

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TYPE_WORD_SELECT 3

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_TYPE_WORD_OFFSET 27

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_MATCH_SIZE 1
```

```
#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_MATCH_OFFSET 126

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT 3

#define
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET 30

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_VALID_OFFSET 127

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_VALID_WORD_SELECT 3

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET 31

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_VALID_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_16B_DATA_RESULT_ELEMENT_WORD_COUNT 4

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE 248

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT 31

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_SELECT 0

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_OFFSET 0

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_RESERVED24_SIZE 1

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_RESERVED24_OFFSET 248

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET 249

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELECT 7

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET 25

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TRUNCATED_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE 1
```

```
#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET 250

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_SELECT
7

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET
26

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TYPE_SIZE 3

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TYPE_OFFSET 251

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TYPE_WORD_SELECT 7

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_TYPE_WORD_OFFSET 27

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_MATCH_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_MATCH_OFFSET 254

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT 7

#define
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET 30

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_MATCH_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_VALID_SIZE 1

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_VALID_OFFSET 255

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_VALID_WORD_SELECT 7

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET 31

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_VALID_MASK (1 <<
EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET)

#define EZDP_LOOKUP_EXT_TCAM_32B_DATA_RESULT_ELEMENT_WORD_COUNT 8

#define EZDP_LOOKUP_INT_TCAM_STANDARD_RESULT_MAX_NUM 4
```

---

## Typedef Documentation

typedef uint32\_t [ezdp\\_hashed\\_key\\_t](#)

typedef uint32\_t [ezdp\\_lookup\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_standard\\_result\\_t](#)

typedef uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_4B\\_data\\_result\\_t](#)

typedef uint32\_t [ezdp\\_lookup\\_ext\\_tcam\\_retval\\_t](#)

typedef uint32\_t [ezdp\\_lookup\\_ext\\_tcam\\_index\\_result\\_element\\_t](#)

typedef uint32\_t [ezdp\\_lookup\\_ext\\_tcam\\_4B\\_data\\_result\\_element\\_t](#)

typedef struct ezdp\_table\_struct\_desc [ezdp\\_table\\_struct\\_desc\\_t](#)

typedef struct ezdp\_hash\_struct\_desc [ezdp\\_hash\\_struct\\_desc\\_t](#)

typedef struct ezdp\_alg\_tcam\_struct\_desc [ezdp\\_alg\\_tcam\\_struct\\_desc\\_t](#)

typedef [ezdp\\_sum\\_addr\\_table\\_desc\\_t](#) [ezdp\\_search\\_base\\_addr\\_t](#)

typedef uint32\_t [ezdp\\_lookup\\_int\\_tcam\\_retval\\_t](#)

---

## Enumeration Type Documentation

enum [ezdp\\_ext\\_tcam\\_result\\_element\\_type](#)

ext tcam result element type possible values.

### Enumerator:

**EZDP\_INDEX** Index only.

**EZDP\_INDEX\_4B\_DATA** Index plus 4 Byte (32 bit) associated data.

**EZDP\_INDEX\_8B\_DATA** Index plus 8 Byte (64 bit) associated data.

**EZDP\_INDEX\_16B\_DATA** Index plus 16 Byte (128 bit) associated data.

**EZDP\_INDEX\_32B\_DATA** Index plus 32 Byte (256 bit) associated data.

**EZDP\_USER\_DEFINED\_ASSOC\_DATA1** User defined associated data 1.

**EZDP\_USER\_DEFINED\_ASSOC\_DATA2** User defined associated data 2.

**EZDP\_USER\_DEFINED\_ASSOC\_DATA3** User defined associated data 3.

enum [ezdp\\_search\\_hash\\_flags](#)

search hash flags.

### Enumerator:

***EZDP\_NONE*** Default.

***EZDP\_OPPORTUNISTIC*** Opportunistic (no memory error handling or waiting for resources).

***EZDP\_COMPRESS*** Compress hash slot.

***EZDP\_UNCONDITIONAL*** Unconditional (perform operation with no lookup and no lock).

enum [\*\*ezdp\\_scan\\_hash\\_slot\\_action\*\*](#)

scan hash slot callback function possible return values.

**Enumerator:**

***EZDP\_ACCEPT\_ENTRY*** Accept the entry.

***EZDP\_DELETE\_ENTRY*** Delete the entry.

***EZDP\_UPDATE\_ENTRY*** Update the entry.

---

## Variable Documentation

enum [\*\*ezdp\\_scan\\_hash\\_slot\\_action\*\*](#)(\* [\*\*ezdp\\_scan\\_entry\\_cb\*\*](#))(void \_\_cmem \*result\_ptr, struct ezdp\_hash\_struct\_desc \*hash\_desc, uintptr\_t user\_data)

## dpe/dp/include/ezdp\_search\_prm.h File Reference

### Functions

- static `__always_inline` void [ezdp\\_prm\\_lock\\_table\\_line](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*table\_struct\_desc, uint32\_t key)
- Lock a table entry. static `__always_inline` bool [ezdp\\_prm\\_trylock\\_table\\_line](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*table\_struct\_desc, uint32\_t key)
- Try to lock a table entry. static `__always_inline` void [ezdp\\_prm\\_unlock\\_table\\_line](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*table\_struct\_desc, uint32\_t key)
- Release a table entry lock. static `__always_inline` [ezdp\\_search\\_base\\_addr\\_t](#) [ezdp\\_prm\\_get\\_table\\_base\\_addr](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*table\_struct\_desc)
- Get search base address (used for lookup) from table struct descriptor. static `__always_inline` [ezdp\\_lookup\\_retval\\_t](#) [ezdp\\_prm\\_lookup\\_table\\_entry](#) ([ezdp\\_search\\_base\\_addr\\_t](#) table\_base\_addr, uint32\_t entry\_size, uint32\_t key, void \*\_\_cmem entry\_ptr)
- Lookup an entry in a table structure. static `__always_inline` void [ezdp\\_prm\\_update\\_table\\_entry](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t entry\_size, uint32\_t key, void \*\_\_cmem entry\_ptr, char \*\_\_cmem work\_area\_ptr, uint32\_t work\_area\_size)
- Add a new entry in a table structure. static `__always_inline` void [ezdp\\_prm\\_delete\\_table\\_entry](#) ([ezdp\\_table\\_struct\\_desc\\_t](#) \*struct\_desc, uint32\_t entry\_size, uint32\_t key, char \*\_\_cmem work\_area\_ptr, uint32\_t work\_area\_size)
- Delete an entry from a table structure. static `__always_inline` [ezdp\\_hashed\\_key\\_t](#) [ezdp\\_prm\\_hash\\_key32](#) (uint32\_t data, uint32\_t key\_offset, uint32\_t key\_size)
- Calculate hash value for keys of up to 32 bits. static `__always_inline` [ezdp\\_hashed\\_key\\_t](#) [ezdp\\_prm\\_hash\\_key64](#) (uint64\_t data, uint32\_t key\_offset, uint32\_t key\_size)
- Calculate hash value for keys of up to 64 bits. static `__always_inline` [ezdp\\_hashed\\_key\\_t](#) [ezdp\\_prm\\_hash\\_bulk\\_key](#) (uint8\_t \_\_cmem \*key\_ptr, uint32\_t key\_size)
- Calculate hash value for keys > 8 bytes. static `__always_inline` void [ezdp\\_prm\\_lock\\_hash\\_slot](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, [ezdp\\_hashed\\_key\\_t](#) hashed\_key)
- Lock a hash slot according to the hashed key. static `__always_inline` bool [ezdp\\_prm\\_trylock\\_hash\\_slot](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, [ezdp\\_hashed\\_key\\_t](#) hashed\_key)
- Try to lock a hash slot according to the hashed key. static `__always_inline` void [ezdp\\_prm\\_unlock\\_hash\\_slot](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, [ezdp\\_hashed\\_key\\_t](#) hashed\_key)
- Release a hash slot lock. static `__always_inline` [ezdp\\_search\\_base\\_addr\\_t](#) [ezdp\\_prm\\_get\\_hash\\_base\\_addr](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc)
- Get search base address (used for lookup) from hash struct descriptor. static `__always_inline` [ezdp\\_lookup\\_retval\\_t](#) [ezdp\\_prm\\_lookup\\_hash\\_entry](#) ([ezdp\\_search\\_base\\_addr\\_t](#) hash\_base\_addr, bool single\_cycle, uint32\_t key\_size, uint32\_t result\_size, uint32\_t entry\_size, uint32\_t hashed\_key, void \_\_cmem \*key\_ptr, void \_\_cmem \*entry\_ptr, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- Lookup an entry in a hash structure. static `__always_inline` [ezdp\\_lookup\\_retval\\_t](#) [ezdp\\_prm\\_locate\\_hash\\_entry](#) ([ezdp\\_search\\_base\\_addr\\_t](#) hash\_base\_addr, bool single\_cycle, uint32\_t key\_size, uint32\_t result\_size, uint32\_t entry\_size, [ezdp\\_hashed\\_key\\_t](#) hashed\_key, void \_\_cmem \*key\_ptr, void \_\_cmem \*entry\_ptr, [ezdp\\_hash\\_op\\_ctx\\_t](#) \_\_cmem \*op\_ctx)
- Lookup an entry location in a hash structure. static `__always_inline` uint32\_t [ezdp\\_prm\\_add\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, bool single\_cycle, uint32\_t key\_size, uint32\_t result\_size, uint32\_t entry\_size, [ezdp\\_hashed\\_key\\_t](#) hashed\_key, void \_\_cmem \*key\_ptr, void \_\_cmem \*result\_ptr, [ezdp\\_hash\\_op\\_ctx\\_t](#) \_\_cmem \*op\_ctx, void \_\_cmem \*entry\_ptr)
- Add a new entry in a hash structure. static `__always_inline` void [ezdp\\_prm\\_modify\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, bool single\_cycle, uint32\_t key\_size, uint32\_t result\_size, uint32\_t entry\_size, [ezdp\\_hash\\_op\\_ctx\\_t](#) \_\_cmem \*op\_ctx, void \_\_cmem \*entry\_ptr)
- Modify an existing entry in a hash structure. static `__always_inline` void [ezdp\\_prm\\_delete\\_hash\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*struct\_desc, bool single\_cycle, uint32\_t entry\_size, [ezdp\\_hash\\_op\\_ctx\\_t](#) \_\_cmem \*op\_ctx)
- Delete an existing entry from a hash structure. static `__always_inline` [ezdp\\_lookup\\_retval\\_t](#) [ezdp\\_prm\\_get\\_hash\\_first\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*hash\_desc, bool single\_cycle, uint32\_t entry\_size, uint32\_t hash\_index, void \_\_cmem \*entry\_ptr, [ezdp\\_hash\\_op\\_ctx\\_t](#) \_\_cmem \*op\_ctx)



- *Get first entry of a hash slot.* static \_\_always\_inline [ezdp\\_lookup\\_retval\\_t ezdp\\_prm\\_get\\_hash\\_next\\_entry](#) ([ezdp\\_hash\\_struct\\_desc\\_t](#) \*hash\_desc, uint32\_t entry\_size, void \_\_cmem \*entry\_ptr, ezdp\_hash\_op\_ctx\_t \_\_cmem \*op\_ctx)
- *Get next entry of a hash slot.* static \_\_always\_inline void [ezdp\\_prm\\_compress\\_hash\\_entry](#) (struct ezdp\_hash\_struct\_desc \*hash\_desc, bool single\_cycle, uint32\_t entry\_size, ezdp\_hash\_op\_ctx\_t \_\_cmem \*op\_ctx, void \_\_cmem \*entry\_ptr)
- *Compress a hash entry with other entries in the same hash slot, if possible.* static \_\_always\_inline [ezdp\\_search\\_base\\_addr\\_t ezdp\\_prm\\_get\\_ultra\\_ip\\_base\\_addr](#) (ezdp\_ultra\_ip\_struct\_desc\_t \*struct\_desc)
- *Get search base address (used for lookup) from UltraIP struct descriptor.* static \_\_always\_inline [ezdp\\_lookup\\_retval\\_t ezdp\\_prm\\_lookup\\_ultra\\_ip\\_entry](#) ([ezdp\\_search\\_base\\_addr\\_t](#) uip\_base\_addr, void \_\_cmem \*key\_ptr, uint32\_t key\_size, char \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)
- *Lookup an entry in an UltraIP structure.* static \_\_always\_inline [ezdp\\_lookup\\_retval\\_t ezdp\\_prm\\_lookup\\_alg\\_tcaml](#) (struct ezdp\_alg\_tcaml\_struct\_desc \*struct\_desc, uint8\_t \_\_cmem \*key\_ptr, uint32\_t key\_size, void \_\_cmem \*work\_area\_ptr, uint32\_t work\_area\_size)

Lookup an entry in an algorithmic TCAM structure.

---

## Function Documentation

**static \_\_always\_inline void ezdp\_prm\_lock\_table\_line** ([ezdp\\_table\\_struct\\_desc\\_t](#) \*  
*table\_struct\_desc*, uint32\_t *key*) [*static*]

Lock a table entry.

Calling thread shall acquire the lock if it is not held by another thread. Otherwise, the thread shall spin until the lock becomes available

### Parameters:

[in] *table\_struct\_desc* - table structure description  
[in] *key* - index into table

### Returns:

none

**static \_\_always\_inline bool ezdp\_prm\_trylock\_table\_line** ([ezdp\\_table\\_struct\\_desc\\_t](#) \*  
*table\_struct\_desc*, uint32\_t *key*) [*static*]

Try to lock a table entry.

Calling thread shall acquire the lock if it is not held by another thread.

### Parameters:

[in] *table\_struct\_desc* - table structure description  
[in] *key* - index into table

### Returns:

true: locked, false: did not succeed to lock

**static \_\_always\_inline void ezdp\_prm\_unlock\_table\_line** ([ezdp\\_table\\_struct\\_desc\\_t](#) \*  
*table\_struct\_desc*, uint32\_t *key*) [*static*]

Release a table entry lock.

### Parameters:

[in] *table\_struct\_desc* - table structure description  
[in] *key* - index into table

### Returns:

none

```
static __always_inline ezdp\_search\_base\_addr\_t ezdp_prm_get_table_base_addr
(ezdp\_table\_struct\_desc\_t * table\_struct\_desc) [static]
```

Get search base address (used for lookup) from table struct descriptor.

**Parameters:**

[in] [table\\_struct\\_desc](#) - table structure description

**Returns:**

[ezdp\\_prm\\_table\\_lookup\\_struct\\_desc\\_t](#)

```
static __always_inline ezdp\_lookup\_retval\_t ezdp_prm_lookup_table_entry
(ezdp\_search\_base\_addr\_t table\_base\_addr,  uint32_t entry\_size,  uint32_t key,   void
*__cmem entry\_ptr) [static]
```

Lookup an entry in a table structure.

The lookup result is written to CMEM.

**Parameters:**

[in] [table\\_base\\_addr](#) - lookup table structure description

[in] [entry\\_size](#) - size of the table entry (result)

[in] [key](#) - index into table

[out] [entry\\_ptr](#) - pointer (in CMEM) to write table entry (result)

**Note:**

Does not handle memory error

**Returns:**

[ezdp\\_lookup\\_retval\\_t](#) - lookup result as defined in [ezdp\\_lookup\\_retval](#)

```
static __always_inline void ezdp_prm_update_table_entry (ezdp\_table\_struct\_desc\_t *
struct\_desc,  uint32_t entry\_size,  uint32_t key,   void *__cmem entry\_ptr,  char *__cmem
work\_area\_ptr,  uint32_t work\_area\_size) [static]
```

Add a new entry in a table structure.

Override the existing entry.

**Parameters:**

[in] [struct\\_desc](#) - table structure description can be derived from [ezdp\\_get\\_table\\_struct\\_desc](#) API, based on struct ID

[in] [key](#) - key

[in] [entry\\_size](#) - size of the table entry (result)

[in] [entry\\_ptr](#) - pointer (in CMEM) to updated entry (result)

[in] [work\\_area\\_ptr](#) - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by `EZDP_TABLE_LOW_LEVEL_WORK_AREA_SIZE`

[in] [work\\_area\\_size](#) - size of work area pointer

**Returns:**

void

```
static __always_inline void ezdp_prm_delete_table_entry (ezdp\_table\_struct\_desc\_t *
struct\_desc,  uint32_t entry\_size,  uint32_t key,   char *__cmem work\_area\_ptr,  uint32_t
work\_area\_size) [static]
```

Delete an entry from a table structure.

**Parameters:**

- [in] *struct\_desc* - table structure description can be derived from `ezdp_get_table_struct_desc` API, based on struct ID
- [in] *key* - key
- [in] *entry\_size* - size of the table entry (result)
- [in] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by `EZDP_TABLE_LOW_LEVEL_WORK_AREA_SIZE`
- [in] *work\_area\_size* - size of work area pointer

**Returns:**

None

```
static __always_inline ezdp\_hashed\_key\_t ezdp_prm_hash_key32 (uint32_t data,    uint32_t
key_offset,    uint32_t key_size) [static]
```

Calculate hash value for keys of up to 32 bits.

**Parameters:**

- [in] *data* - contains the key to calculate hash from
- [in] *key\_offset* - offset in bytes to key in data  $0 \leq \text{key\_offset} \leq 3$
- [in] *key\_size* - size in bytes of the key to hash  $1 \leq \text{key\_size} \leq 4$

**Returns:**

the calculated hash of the key

```
static __always_inline ezdp\_hashed\_key\_t ezdp_prm_hash_key64 (uint64_t data,    uint32_t
key_offset,    uint32_t key_size) [static]
```

Calculate hash value for keys of up to 64 bits.

**Parameters:**

- [in] *data* - contains the key to calculate hash from
- [in] *key\_offset* - offset in bytes to key in data  $0 \leq \text{key\_offset} \leq 3$
- [in] *key\_size* - size in bytes of the key to hash  $1 \leq \text{key\_size} \leq 8$

**Returns:**

the calculated hash of the key

```
static __always_inline ezdp\_hashed\_key\_t ezdp_prm_hash_bulk_key (uint8_t __cmem * key_ptr,
uint32_t key_size) [static]
```

Calculate hash value for keys > 8 bytes.

**Parameters:**

- [in] *key\_ptr* - pointer to key in cmem
- [in] *key\_size* - size in bytes of the key to hash (values 9..64)

**Returns:**

the calculated hash of the key

```
static __always_inline void ezdp_prm_lock_hash_slot (ezdp\_hash\_struct\_desc\_t * struct_desc,
ezdp\_hashed\_key\_t hashed_key) [static]
```

Lock a hash slot according to the hashed key.

Calling thread shall acquire the lock if it is not held by another thread. Otherwise, the thread shall spin until the lock becomes available

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *hashed\_key* - the hash of the key

**Returns:**

none

```
static __always_inline bool ezdp_prm_trylock_hash_slot (ezdp\_hash\_struct\_desc\_t * struct_desc,
ezdp\_hashed\_key\_t hashed_key) [static]
```

Try to lock a hash slot according to the hashed key.

Calling thread shall acquire the lock if it is not held by another thread.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *hashed\_key* - the hash of the key

**Returns:**

true: locked, false: did not succeed to lock

```
static __always_inline void ezdp_prm_unlock_hash_slot (ezdp\_hash\_struct\_desc\_t * struct_desc,
ezdp\_hashed\_key\_t hashed_key) [static]
```

Release a hash slot lock.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *hashed\_key* - the hash of the key

**Returns:**

none

```
static __always_inline ezdp\_search\_base\_addr\_t ezdp_prm_get_hash_base_addr
(ezdp\_hash\_struct\_desc\_t * struct_desc) [static]
```

Get search base address (used for lookup) from hash struct descriptor.

**Parameters:**

[in] *struct\_desc* - hash structure description

**Returns:**

[ezdp\\_prm\\_hash\\_lookup\\_struct\\_desc\\_t](#)

```
static __always_inline ezdp\_lookup\_retval\_t ezdp_prm_lookup_hash_entry
(ezdp\_search\_base\_addr\_t hash_base_addr, bool single_cycle, uint32_t key_size, uint32_t
result_size, uint32_t entry_size, uint32_t hashed_key, void __cmem * key_ptr, void
__cmem * entry_ptr, char __cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Lookup an entry in a hash structure.

The lookup result is written to CMEM.

**Parameters:**

[in] *hash\_base\_addr* - lookup hash struct descriptor

[in] *single\_cycle* - single-cycle or non-single-cycle hash  
 [in] *key\_size* - size of the key  
 [in] *result\_size* - size of the result (for best performance should be a multiple of 4)  
 [in] *entry\_size* - size of the entry  
 [in] *hashed\_key* - the hash of the key  
 [in] *key\_ptr* - pointer to key (in CMEM)  
 [out] *entry\_ptr* - pointer (in CMEM) to return entry  
 [in] *work\_area\_ptr* - work area pointer (temporary memory on CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_HASH\_LOW\_LEVEL\_WORK\_AREA\_SIZE  
 [in] *work\_area\_size* - size of work area pointer

**Note:**

This function does not handle memory error. When the entry is not found (no match), *entry\_ptr* is not valid. For best performance *result\_size* should be a multiple of 4

**Returns:**

*ezdp\_lookup\_retval\_t* - lookup result as defined in [ezdp\\_lookup\\_retval](#)

```
static __always_inline ezdp\_lookup\_retval\_t ezdp_prm_locate_hash_entry
(ezdp\_search\_base\_addr\_t hash_base_addr, bool single_cycle, uint32_t key_size, uint32_t
result_size, uint32_t entry_size, ezdp\_hashed\_key\_t hashed_key, void __cmem * key_ptr,
void __cmem * entry_ptr, ezdp\_hash\_op\_ctx\_t __cmem * op_ctx) [static]
```

Lookup an entry location in a hash structure.

The lookup result is written to CMEM. The state of the lookup is stored for future update operations. In addition, the operation context is returned, to be used later to perform add/modify/delete.

**Parameters:**

[in] *hash\_base\_addr* - lookup hash struct descriptor  
 [in] *single\_cycle* - single-cycle or non-single-cycle hash  
 [in] *key\_size* - size of the key  
 [in] *result\_size* - size of the result (for best performance should be a multiple of 4)  
 [in] *entry\_size* - size of the entry  
 [in] *hashed\_key* - the hash of the key  
 [in] *key\_ptr* - pointer to key (in CMEM)  
 [out] *entry\_ptr* - pointer (in CMEM) to return entry  
 [in] *op\_ctx* - operation context status (in CMEM)

**Note:**

This function does not handle memory error and multi thread synchronization. When the entry is not found (no match), *entry\_ptr* is not valid. For best performance *result\_size* should be a multiple of 4

**Returns:**

*ezdp\_lookup\_retval\_t* - lookup result as defined in [ezdp\\_lookup\\_retval](#)

```
static __always_inline uint32_t ezdp_prm_add_hash_entry (ezdp\_hash\_struct\_desc\_t *
struct_desc, bool single_cycle, uint32_t key_size, uint32_t result_size, uint32_t
entry_size, ezdp\_hashed\_key\_t hashed_key, void __cmem * key_ptr, void __cmem *
result_ptr, ezdp\_hash\_op\_ctx\_t __cmem * op_ctx, void __cmem * entry_ptr) [static]
```

Add a new entry in a hash structure.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *single\_cycle* - single-cycle or non-single-cycle hash  
 [in] *key\_size* - size of the key  
 [in] *result\_size* - size of the result (for best performance should be a multiple of 4)  
 [in] *entry\_size* - size of the entry  
 [in] *hashed\_key* - the hash of the key

[in] *key\_ptr* - pointer to key (in CMEM)  
 [in] *result\_ptr* - pointer to updated result (in CMEM)  
 [in] *op\_ctx* - operation context status (in CMEM)  
 [in] *entry\_ptr* - temporary entry size buffer on CMEM

**Note:**

Not thread safe. Must be used with `ezdp_lock_hash()` before retrieving *op\_ctx*.

**Returns:**

0 (success), ENOMEM (hash is full) use [ezdp\\_get\\_err\\_msg\(\)](#) API to get the detail error message of the failure

```
static __always_inline void ezdp_prm_modify_hash_entry (ezdp\_hash\_struct\_desc\_t *  
struct_desc, bool single_cycle, uint32_t key_size, uint32_t result_size, uint32_t  
entry_size, ezdp_hash_op_ctx_t __cmem * op_ctx, void __cmem * entry_ptr) [static]
```

Modify an existing entry in a hash structure.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *single\_cycle* - single-cycle or non-single-cycle hash  
 [in] *key\_size* - size of the key  
 [in] *result\_size* - size of the result (for best performance should be a multiple of 4)  
 [in] *entry\_size* - size of the entry  
 [in] *op\_ctx* - operation context status (in CMEM)  
 [in] *entry\_ptr* - temporary entry size buffer on CMEM to be used by the function

**Note:**

Not thread safe. Must be used with `ezdp_lock_hash()` before retrieving *op\_ctx*.

**Returns:**

none

```
static __always_inline void ezdp_prm_delete_hash_entry (ezdp\_hash\_struct\_desc\_t *  
struct_desc, bool single_cycle, uint32_t entry_size, ezdp_hash_op_ctx_t __cmem * op_ctx)  
[static]
```

Delete an existing entry from a hash structure.

**Parameters:**

[in] *struct\_desc* - hash struct descriptor  
 [in] *single\_cycle* - single-cycle or non-single-cycle hash  
 [in] *entry\_size* - size of the entry  
 [in] *op\_ctx* - operation context status (in CMEM)

**Note:**

Not thread safe. Must be used with `ezdp_lock_hash()` before retrieving *op\_ctx*.

**Returns:**

None

```
static __always_inline ezdp\_lookup\_retval\_t ezdp_prm_get_hash_first_entry  
(ezdp\_hash\_struct\_desc\_t * hash_desc, bool single_cycle, uint32_t entry_size, uint32_t  
hash_index, void __cmem * entry_ptr, ezdp_hash_op_ctx_t __cmem * op_ctx) [static]
```

Get first entry of a hash slot.

**Parameters:**

[in] *hash\_desc* - hash descriptor for the search table  
[in] *single\_cycle* - single-cycle or non-single-cycle hash  
[in] *entry\_size* - size of the entry  
[in] *hash\_index* - hash index (line/slot) number to scan  
[out] *entry\_ptr* - pointer (in CMEM) to return entry  
[in] *op\_ctx* - operation context status (in CMEM)

**Note:**

When hash index is empty (first entry not found), *entry\_ptr* is not valid. Not thread safe. Must be used with `ezdp_lock_hash()` before retrieving *op\_ctx*.

**Returns:**

`ezdp_lookup_retval_t` - lookup result as defined in [ezdp\\_lookup\\_retval](#)

```
static __always_inline ezdp\_lookup\_retval\_t ezdp_prm_get_hash_next_entry
(ezdp\_hash\_struct\_desc\_t * hash_desc,  uint32\_t entry_size,  void __cmem * entry_ptr,
ezdp\_hash\_op\_ctx\_t __cmem * op_ctx) [static]
```

Get next entry of a hash slot.

**Parameters:**

[in] *hash\_desc* - hash descriptor for the search table  
[in] *entry\_size* - size of the entry  
[out] *entry\_ptr* - pointer (in CMEM) to return entry  
[in] *op\_ctx* - operation context status (in CMEM)

**Note:**

When next entry is not found, *entry\_ptr* is not valid. Not thread safe. Must be used with `ezdp_lock_hash()` before retrieving *op\_ctx*.

**Returns:**

`ezdp_lookup_retval_t` - lookup result as defined in [ezdp\\_lookup\\_retval](#)

```
static __always_inline void ezdp_prm_compress_hash_entry (struct ezdp\_hash\_struct\_desc *
hash_desc,  bool single_cycle,  uint32\_t entry_size,  ezdp\_hash\_op\_ctx\_t __cmem * op_ctx,
void __cmem * entry_ptr) [static]
```

Compress a hash entry with other entries in the same hash slot, if possible.

**Parameters:**

[in] *hash\_desc* - hash descriptor for the search table  
[in] *single\_cycle* - single-cycle or non-single-cycle hash  
[in] *entry\_size* - size of the entry  
[in] *op\_ctx* - operation context status (in CMEM)  
[in] *entry\_ptr* - pointer (in CMEM) to entry for internal use

**Note:**

Not thread safe. Must be used with `ezdp_lock_hash()` before retrieving *op\_ctx*.

**Returns:**

`void`

```
static __always_inline ezdp\_search\_base\_addr\_t ezdp_prm_get_ultra_ip_base_addr
(ezdp\_ultra\_ip\_struct\_desc\_t * struct_desc) [static]
```

Get search base address (used for lookup) from UltraIP struct descriptor.

**Parameters:**

[in] *struct\_desc* - ultra ip structure description

**Returns:**

*ezdp\_prm\_ultra\_ip\_lookup\_struct\_desc\_t*

```
static __always_inline ezdp\_lookup\_retval\_t ezdp_prm_lookup_ultra_ip_entry
(ezdp\_search\_base\_addr\_t uip_base_addr, void __cmem * key_ptr, uint32_t key_size, char
__cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Lookup an entry in an UltraIP structure.

**Parameters:**

[in] *uip\_base\_addr* - lookup table summarized address descriptor

[in] *key\_ptr* - pointer to key (in CMEM)

[in] *key\_size* - size of key

[in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_ULTRA\_IP\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Note:**

Does not handle memory error.

**Returns:**

*ezdp\_lookup\_retval\_t* - lookup result as defined in [ezdp\\_lookup\\_retval](#) [ezdp\\_lookup\\_retval.data](#) is 28bit data saved in found ultra ip entry

```
static __always_inline ezdp\_lookup\_retval\_t ezdp_prm_lookup_alg_tcam (struct
ezdp_alg_tcam_struct_desc * struct_desc, uint8_t __cmem * key_ptr, uint32_t key_size,
void __cmem * work_area_ptr, uint32_t work_area_size) [static]
```

Lookup an entry in an algorithmic TCAM structure.

**Parameters:**

[in] *struct\_desc* - algo tcam struct descriptor

[in] *key\_ptr* - pointer to the key in CMEM

[in] *key\_size* - size of the key

[in] *work\_area\_ptr* - pointer to work area (temporary memory in CMEM to be used by the function). The size of the temporary memory is determined by EZDP\_ALG\_TCAM\_WORK\_AREA\_SIZE

[in] *work\_area\_size* - size of work area pointer

**Note:**

Does not handle memory error.

**Returns:**

*ezdp\_lookup\_retval\_t* - lookup result as defined in [ezdp\\_lookup\\_retval](#) [ezdp\\_lookup\\_retval.data](#) is 28bit priority



## dpe/dp/include/ezdp\_security.h File Reference

### Functions

- static \_\_always\_inline void [ezdp\\_encrypt](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*plain\_data\_ptr, void \_\_cmem \*encr\_data\_ptr, uint32\_t size)
- *Encrypt a data segment.* static \_\_always\_inline void [ezdp\\_encrypt\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*plain\_data\_ptr, void \_\_cmem \*encr\_data\_ptr, uint32\_t size)
- *Non blocking version of [ezdp\\_encrypt\(\)](#).* static \_\_always\_inline void [ezdp\\_decrypt](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*encr\_data\_ptr, void \_\_cmem \*plain\_data\_ptr, uint32\_t size)
- *Decrypt a data segment.* static \_\_always\_inline void [ezdp\\_decrypt\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*encr\_data\_ptr, void \_\_cmem \*plain\_data\_ptr, uint32\_t size)
- *Non blocking version of [ezdp\\_decrypt\(\)](#).* static \_\_always\_inline void [ezdp\\_mac\\_calculation](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*data\_ptr, const bool init, const bool last, uint32\_t size)
- *Calculate the message authentication code (MAC) on a data segment.* static \_\_always\_inline void [ezdp\\_mac\\_calculation\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*data\_ptr, const bool init, const bool last, uint32\_t size)
- *Non blocking version of [ezdp\\_mac\\_calculation\(\)](#).* static \_\_always\_inline void [ezdp\\_start\\_hmac\\_calculation](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t size)
- *Start a hash-based message authentication code (MAC) calculation.* static \_\_always\_inline void [ezdp\\_start\\_hmac\\_calculation\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t size)
- *Non blocking version of [ezdp\\_start\\_hmac\\_calculation\(\)](#).* static \_\_always\_inline void [ezdp\\_end\\_hmac\\_calculation](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t size)
- *Complete a hash-based message authentication code (MAC) calculation.* static \_\_always\_inline void [ezdp\\_end\\_hmac\\_calculation\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t size)
- *Non blocking version of [ezdp\\_end\\_hmac\\_calculation\(\)](#).* static \_\_always\_inline void [ezdp\\_generate\\_security\\_initial\\_vector](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*data\_ptr, const bool init, uint32\_t size)
- *Generate security initial vector.* static \_\_always\_inline void [ezdp\\_generate\\_security\\_initial\\_vector\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*data\_ptr, const bool init, uint32\_t size)
- *Non blocking version of [ezdp\\_generate\\_security\\_initial\\_vector\(\)](#).* static \_\_always\_inline void [ezdp\\_end\\_gcm\\_mac\\_calculation](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t size)
- *Complete a GCM hash-based message authentication code (MAC) calculation.* static \_\_always\_inline void [ezdp\\_end\\_gcm\\_mac\\_calculation\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t size)
- *Non blocking version of [ezdp\\_end\\_gcm\\_mac\\_calculation\(\)](#).* static \_\_always\_inline void [ezdp\\_expand\\_security\\_key](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle)
- *Expands the key in the security context memory.* static \_\_always\_inline void [ezdp\\_expand\\_security\\_key\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle)
- *Non blocking version of [ezdp\\_expand\\_security\\_key\(\)](#).* static \_\_always\_inline void [ezdp\\_write\\_security\\_state](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*sec\_state\_ptr, uint32\_t sec\_state\_size)
- *Copy the security state data from CMEM to the security context memory.* static \_\_always\_inline void [ezdp\\_write\\_security\\_state\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*sec\_state\_ptr, uint32\_t sec\_state\_size)
- *Non blocking version of [ezdp\\_write\\_security\\_state\(\)](#).* static \_\_always\_inline void [ezdp\\_read\\_security\\_state](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*sec\_state\_ptr, uint32\_t sec\_state\_size)
- *Copy the security state data from the security context memory to CMEM.* static \_\_always\_inline void [ezdp\\_read\\_security\\_state\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*sec\_state\_ptr, uint32\_t sec\_state\_size)
- *Non blocking version of [ezdp\\_read\\_security\\_state\(\)](#).* static \_\_always\_inline uint32\_t [ezdp\\_security\\_state\\_size](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle)
- *Return the state size.* static \_\_always\_inline void [ezdp\\_write\\_security\\_key](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t key\_size)
- *Copy the security key from CMEM to the security context memory.* static \_\_always\_inline void [ezdp\\_write\\_security\\_key\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t key\_size)
- *Non blocking version of [ezdp\\_write\\_security\\_key\(\)](#).* static \_\_always\_inline void [ezdp\\_read\\_security\\_key](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t key\_size)

- Copy the security key from the security context memory to CMEM. static \_\_always\_inline void [ezdp\\_read\\_security\\_key\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*key\_ptr, uint32\_t key\_size)
- Non blocking version of [ezdp\\_read\\_security\\_key\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_security\\_key\\_size](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle)
- Return the key size. static \_\_always\_inline void [ezdp\\_write\\_security\\_mac](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*mac\_ptr, uint32\_t mac\_size)
- Copy the security MAC from CMEM to the security context memory. static \_\_always\_inline void [ezdp\\_write\\_security\\_mac\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*mac\_ptr, uint32\_t mac\_size)
- Non blocking version of [ezdp\\_write\\_security\\_mac\(\)](#). static \_\_always\_inline void [ezdp\\_read\\_security\\_mac](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*mac\_ptr, uint32\_t mac\_size)
- Copy the security MAC from the security context memory to CMEM. static \_\_always\_inline void [ezdp\\_read\\_security\\_mac\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*mac\_ptr, uint32\_t mac\_size)
- Non blocking version of [ezdp\\_read\\_security\\_mac\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_security\\_mac\\_size](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle)
- Return the MAC size. static \_\_always\_inline void [ezdp\\_write\\_security\\_initial\\_vector](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*iv\_ptr, uint32\_t iv\_size)
- Copy the security initial vector from CMEM to the security context memory. static \_\_always\_inline void [ezdp\\_write\\_security\\_initial\\_vector\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*iv\_ptr, uint32\_t iv\_size)
- Non blocking version of [ezdp\\_write\\_security\\_initial\\_vector\(\)](#). static \_\_always\_inline void [ezdp\\_read\\_security\\_initial\\_vector](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*iv\_ptr, uint32\_t iv\_size)
- Copy the security initial vector from the security context memory to CMEM. static \_\_always\_inline void [ezdp\\_read\\_security\\_initial\\_vector\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*iv\_ptr, uint32\_t iv\_size)
- Non blocking version of [ezdp\\_read\\_security\\_initial\\_vector\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_security\\_initial\\_vector\\_size](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle)
- Return the initial vector size. static \_\_always\_inline void [ezdp\\_write\\_security\\_context](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*ctx\_ptr, uint32\_t ctx\_size)
- Copy the security context from CMEM to the security context memory. static \_\_always\_inline void [ezdp\\_write\\_security\\_context\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*ctx\_ptr, uint32\_t ctx\_size)
- Non blocking version of [ezdp\\_write\\_security\\_context\(\)](#). static \_\_always\_inline void [ezdp\\_read\\_security\\_context](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*ctx\_ptr, uint32\_t ctx\_size)
- Copy the security context from the security context memory to CMEM. static \_\_always\_inline void [ezdp\\_read\\_security\\_context\\_async](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*ctx\_ptr, uint32\_t ctx\_size)
- Non blocking version of [ezdp\\_read\\_security\\_context\(\)](#). static \_\_always\_inline uint32\_t [ezdp\\_security\\_block\\_size](#) ([ezdp\\_security\\_handle\\_t](#) sec\_handle)

Return the algorithmic engine minimal block size.

## Function Documentation

**static \_\_always\_inline void ezdp\_encrypt** ([ezdp\\_security\\_handle\\_t](#) sec\_handle, void \_\_cmem \*plain\_data\_ptr, void \_\_cmem \*encr\_data\_ptr, uint32\_t size) [static]

Encrypt a data segment.

The encryption is performed based on the algorithm configured in the security handle. The encrypted data is written back to CMEM and the initial vector is updated in the security context memory. In addition, for authenticated encryption algorithms such as AES\_GCM and AES\_CCM, the message digest is also updated in the security context memory.

### Parameters:

- [in] *sec\_handle* - security handle
- [in] *plain\_data\_ptr*- pointer to source (plain) data in CMEM to be encrypted
- [out] *encr\_data\_ptr*- pointer to destination in CMEM to write encrypted data to

[in] *size* - number of bytes to encrypt

**Note:**

- The size must be a multiple of X as defined per algorithm.

**Returns:**

none

```
static __always_inline void ezdp_encrypt_async(ezdp\_security\_handle\_t sec_handle, void
__cmem * plain_data_ptr, void __cmem * encr_data_ptr, uint32_t size) [static]
```

Non blocking version of [ezdp\\_encrypt\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle  
 [in] *plain\_data\_ptr*- pointer to source (plain) data in CMEM to be encrypted  
 [out] *encr\_data\_ptr*- pointer to destination in CMEM to write encrypted data to  
 [in] *size* - number of bytes to encrypt

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the encrypted data is ready in CMEM.

**Returns:**

none

```
static __always_inline void ezdp_decrypt(ezdp\_security\_handle\_t sec_handle, void __cmem *
encr_data_ptr, void __cmem * plain_data_ptr, uint32_t size) [static]
```

Decrypt a data segment.

The decryption is performed based on the algorithm configured in the security handle. The decrypted data is written back to CMEM and the initial vector is updated in the security context memory. In addition, for authenticated encryption algorithms such as AES\_GCM and AES\_CCM, the message digest is also updated in the security context memory.

**Parameters:**

[in] *sec\_handle* - security handle  
 [in] *encr\_data\_ptr* - pointer to source (encrypted) data in CMEM to be decrypted  
 [out] *plain\_data\_ptr*- pointer to destination in CMEM to write decrypted data to  
 [in] *size* - number of bytes to decrypt

**Note:**

- The size must be a multiple of X as defined per algorithm.

**Returns:**

none

```
static __always_inline void ezdp_decrypt_async(ezdp\_security\_handle\_t sec_handle, void
__cmem * encr_data_ptr, void __cmem * plain_data_ptr, uint32_t size) [static]
```

Non blocking version of [ezdp\\_decrypt\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle  
 [in] *encr\_data\_ptr* - pointer to source (encrypted) data in CMEM to be decrypted  
 [out] *plain\_data\_ptr*- pointer to destination in CMEM to write decrypted data to  
 [in] *size* - number of bytes to decrypt

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the decrypted data is ready in CMEM.

**Returns:**

none

```
static __always_inline void ezdp_mac_calculation (ezdp\_security\_handle\_t sec_handle, void
__cmem * data_ptr, const bool init, const bool last, uint32_t size) [static]
```

Calculate the message authentication code (MAC) on a data segment.

The MAC calculation is done based on the algorithm configured in the security handle. The message digest is updated in the security context memory.

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *data\_ptr* - pointer to source data in CMEM
- [in] *init* - indicates first segment
- [in] *last* - indicates last segment
- [in] *size* - number of bytes to calculate MAC on

**Returns:**

none

```
static __always_inline void ezdp_mac_calculation_async (ezdp\_security\_handle\_t sec_handle,
void __cmem * data_ptr, const bool init, const bool last, uint32_t size) [static]
```

Non blocking version of [ezdp\\_mac\\_calculation\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *data\_ptr* - pointer to source data in CMEM
- [in] *init* - indicates first segment
- [in] *last* - indicates last segment
- [in] *size* - number of bytes to calculate MAC on

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was read from CMEM and the message digest is updated in the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_start_hmac_calculation (ezdp\_security\_handle\_t sec_handle,
void __cmem * key_ptr, uint32_t size) [static]
```

Start a hash-based message authentication code (MAC) calculation.

This operation pads the data (encryption key) if required, XORs it with 0x36 (aka i\_pad) and then calculates the message authentication code (MAC) on the result. The MAC calculation is performed based on the algorithm configured in the security handle. The message digest is updated in the security context memory.

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *key\_ptr* - pointer to the encryption key in CMEM
- [in] *size* - number of bytes in the encryption key

**Note:**

- Must be called with encryption key as data, as defined in HMAC standards.

**Returns:**

none

```
static __always_inline void ezdp_start_hmac_calculation_async(ezdp\_security\_handle\_t sec_handle, void __cmem * key_ptr, uint32_t size) [static]
```

Non blocking version of [ezdp\\_start\\_hmac\\_calculation\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *key\_ptr* - pointer to the encryption key in CMEM
- [in] *size* - number of bytes in the encryption key

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was read from CMEM and the message digest is updated in the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_end_hmac_calculation(ezdp\_security\_handle\_t sec_handle, void __cmem * key_ptr, uint32_t size) [static]
```

Complete a hash-based message authentication code (MAC) calculation.

This operation pads the data (encryption key) if required, XORs it with 0x5c (aka o\_pa) and then calculates the message authentication code (MAC) on the result concatenated with the message digest in the security context memory. The MAC calculation is performed based on the algorithm configured in the security handle. The message digest is updated in the security context memory.

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *key\_ptr* - pointer to the encryption key in CMEM
- [in] *size* - number of bytes in the encryption key

**Note:**

- Must be called with encryption key as data, as defined in HMAC standards.

**Returns:**

none

```
static __always_inline void ezdp_end_hmac_calculation_async(ezdp\_security\_handle\_t sec_handle, void __cmem * key_ptr, uint32_t size) [static]
```

Non blocking version of [ezdp\\_end\\_hmac\\_calculation\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *key\_ptr* - pointer to the encryption key in CMEM
- [in] *size* - number of bytes in the encryption key

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was read from CMEM and the message digest is updated in the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_generate_security_initial_vector (ezdp\_security\_handle\_t sec_handle, void __cmem * data_ptr, const bool init, uint32_t size) [static]
```

Generate security initial vector.

The calculation of the initial vector is done based on the algorithm configured in the security handle. The initial vector is updated in the security context memory. For GCM algorithm, this function generates JO, which is later used as the initial vector.

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *data\_ptr* - pointer to source data in CMEM
- [in] *init* - indicates first segment
- [in] *size* - number of bytes to calculate MAC on

**Note:**

Applicable to EZDP\_AES\_GCM\_128\_ALG EZDP\_AES\_GCM\_192\_ALG EZDP\_AES\_GCM\_256\_ALG algorithms only

**Returns:**

none

```
static __always_inline void ezdp_generate_security_initial_vector_async (ezdp\_security\_handle\_t sec_handle, void __cmem * data_ptr, const bool init, uint32_t size) [static]
```

Non blocking version of [ezdp\\_generate\\_security\\_initial\\_vector\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *data\_ptr* - pointer to source data in CMEM
- [in] *init* - indicates first segment
- [in] *size* - number of bytes to calculate MAC on

**Note:**

- Applicable to EZDP\_AES\_GCM\_128\_ALG EZDP\_AES\_GCM\_192\_ALG EZDP\_AES\_GCM\_256\_ALG algorithms only

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was read from CMEM and the initial vector is updated in the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_end_gcm_mac_calculation (ezdp\_security\_handle\_t sec_handle, void __cmem * key_ptr, uint32_t size) [static]
```

Complete a GCM hash-based message authentication code (MAC) calculation.

Authentication code (MAC) on the result concatenated with the message digest in the security context memory. The MAC calculation is performed based on the GCM algorithm. The message digest is updated in the security context memory.

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *key\_ptr* - pointer to the encryption key in CMEM
- [in] *size* - number of bytes in the encryption key

**Returns:**

none

```
static __always_inline void ezdp_end_gcm_mac_calculation_async (ezdp\_security\_handle\_t
sec_handle, void __cmem * key_ptr, uint32_t size) [static]
```

Non blocking version of [ezdp\\_end\\_gcm\\_mac\\_calculation\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle  
[in] *key\_ptr* - pointer to the encryption key in CMEM  
[in] *size* - number of bytes in the encryption key

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was read from CMEM and the message digest is updated in the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_expand_security_key (ezdp\_security\_handle\_t sec_handle)
[static]
```

Expands the key in the security context memory.

Reads the security key in the security context memory, expands it and stores it back to the security context memory. This operation should be performed once after loading a non-expanded key to the security context memory.

**Parameters:**

[in] *sec\_handle* - security handle

**Note:**

- Relevant for decryption algorithms.

**Returns:**

none

```
static __always_inline void ezdp_expand_security_key_async (ezdp\_security\_handle\_t
sec_handle) [static]
```

Non blocking version of [ezdp\\_expand\\_security\\_key\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data is ready in the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_write_security_state (ezdp\_security\_handle\_t sec_handle,
void __cmem * sec_state_ptr, uint32_t sec_state_size) [static]
```

Copy the security state data from CMEM to the security context memory.



**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *sec\_state\_ptr* - pointer to the security state data in CMEM
- [in] *sec\_state\_size* - the size of the security *sec\_state\_ptr*

**Note:**

- The size of the security state is according to the the algorithm type, see *ezdp\_sec\_state\_size* enum.

**Returns:**

none

```
static __always_inline void ezdp_write_security_state_async(ezdp\_security\_handle\_t sec_handle, void __cmem * sec_state_ptr, uint32_t sec_state_size) [static]
```

Non blocking version of [ezdp\\_write\\_security\\_state\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *sec\_state\_ptr* - pointer to the security state data in CMEM
- [in] *sec\_state\_size* - the size of the security *sec\_state\_ptr*

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was written into the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_read_security_state(ezdp\_security\_handle\_t sec_handle, void __cmem * sec_state_ptr, uint32_t sec_state_size) [static]
```

Copy the security state data from the security context memory to CMEM.

**Parameters:**

- [in] *sec\_handle* - security handle
- [out] *sec\_state\_ptr*- pointer in CMEM to write the security state data to
- [in] *sec\_state\_size* - the size of the security *sec\_state\_ptr*

**Note:**

- The size of the security state is according to the the algorithm type, see *ezdp\_sec\_state\_size* enum.

**Returns:**

none

```
static __always_inline void ezdp_read_security_state_async(ezdp\_security\_handle\_t sec_handle, void __cmem * sec_state_ptr, uint32_t sec_state_size) [static]
```

Non blocking version of [ezdp\\_read\\_security\\_state\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [out] *sec\_state\_ptr*- pointer in CMEM to write the security state data to
- [in] *sec\_state\_size* - the size of the security *sec\_state\_ptr*



**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data is ready in CMEM.

**Returns:**

none

```
static __always_inline uint32_t ezdp_security_state_size (ezdp\_security\_handle\_t sec_handle)
[static]
```

Return the state size.

The size of the state is according to alg type configured in security handle

**Parameters:**

[in] *sec\_handle* - security handle, only alg\_type is used

**Returns:**

0 - error, state size

```
static __always_inline void ezdp_write_security_key (ezdp\_security\_handle\_t sec_handle, void
__cmem * key_ptr, uint32_t key_size) [static]
```

Copy the security key from CMEM to the security context memory.

**Parameters:**

[in] *sec\_handle* - security handle

[in] *key\_ptr* - pointer to the security key in CMEM

[in] *key\_size* - the size of the key\_ptr

**Note:**

- The size of the security key is according to algorithm type, see [ezdp\\_sec\\_key\\_size](#) enum.

**Returns:**

none

```
static __always_inline void ezdp_write_security_key_async (ezdp\_security\_handle\_t sec_handle,
void __cmem * key_ptr, uint32_t key_size) [static]
```

Non blocking version of [ezdp\\_write\\_security\\_key\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle

[in] *key\_ptr* - pointer to the security key in CMEM

[in] *key\_size* - the size of the key\_ptr

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was written to the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_read_security_key (ezdp\_security\_handle\_t sec_handle, void
__cmem * key_ptr, uint32_t key_size) [static]
```

Copy the security key from the security context memory to CMEM.

**Parameters:**

[in] *sec\_handle* - security handle  
 [out] *key\_ptr* - pointer in CMEM to write the security key to  
 [in] *key\_size* - the size of the key\_ptr

**Note:**

- The size of the security key is derived from algorithm type, see `ezdp_sec_key_size` enum.

**Returns:**

none

```
static __always_inline void ezdp_read_security_key_async(ezdp\_security\_handle\_t sec_handle,
void __cmem * key_ptr,  uint32_t key_size) [static]
```

Non blocking version of [ezdp\\_read\\_security\\_key\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle  
 [out] *key\_ptr* - pointer in CMEM to write the security key to  
 [in] *key\_size* - the size of the key\_ptr

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data is ready in CMEM.

**Returns:**

none

```
static __always_inline uint32_t ezdp_security_key_size(ezdp\_security\_handle\_t sec_handle)
[static]
```

Return the key size.

The size of the key is according to alg type configured in security handle

**Parameters:**

[in] *sec\_handle* - security handle, only `alg_type` is used

**Returns:**

0 - error, key size

```
static __always_inline void ezdp_write_security_mac(ezdp\_security\_handle\_t sec_handle,  void
__cmem * mac_ptr,  uint32_t mac_size) [static]
```

Copy the security MAC from CMEM to the security context memory.

**Parameters:**

[in] *sec\_handle* - security handle  
 [in] *mac\_ptr* - pointer to the security MAC in CMEM  
 [in] *mac\_size* - size of the mac\_ptr

**Note:**

- The size of the security mac is according to algorithm type, see `ezdp_sec_mac_size` enum.

**Returns:**

none

```
static __always_inline void ezdp_write_security_mac_async(ezdp\_security\_handle\_t sec_handle,
void __cmem * mac_ptr,  uint32_t mac_size) [static]
```

Non blocking version of [ezdp\\_write\\_security\\_mac\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *mac\_ptr* - pointer to the security MAC in CMEM
- [in] *mac\_size* - size of the mac\_ptr

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was written to the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_read_security_mac (ezdp\_security\_handle\_t sec_handle, void
__cmem * mac_ptr, uint32_t mac_size) [static]
```

Copy the security MAC from the security context memory to CMEM.

**Parameters:**

- [in] *sec\_handle* - security handle
- [out] *mac\_ptr* - pointer in CMEM to write the security MAC to
- [in] *mac\_size* - size of the mac\_ptr

**Note:**

- The size of the security MAC is derived from algorithm type, see *ezdp\_sec\_mac\_size* enum.

**Returns:**

none

```
static __always_inline void ezdp_read_security_mac_async (ezdp\_security\_handle\_t sec_handle,
void __cmem * mac_ptr, uint32_t mac_size) [static]
```

Non blocking version of [ezdp\\_read\\_security\\_mac\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [out] *mac\_ptr* - pointer in CMEM to write the security MAC to
- [in] *mac\_size* - size of the mac\_ptr

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data is ready in CMEM.

**Returns:**

none

```
static __always_inline uint32_t ezdp_security_mac_size (ezdp\_security\_handle\_t sec_handle)
[static]
```

Return the MAC size.

The size of the MAC is according to alg type configured in security handle

**Parameters:**

- [in] *sec\_handle* - security handle, only *alg\_type* is used

**Returns:**

0 - error, MAC size

```
static __always_inline void ezdp_write_security_initial_vector (ezdp\_security\_handle\_t
sec_handle, void __cmem * iv_ptr, uint32_t iv_size) [static]
```

Copy the security initial vector from CMEM to the security context memory.

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *iv\_ptr* - pointer to the security initial vector in CMEM
- [in] *iv\_size* - size of the *iv\_ptr*

**Note:**

- The size of the security initial vector is according to algorithm type, see `ezdp_security_initial_vector_size` enum.

**Returns:**

none

```
static __always_inline void ezdp_write_security_initial_vector_async (ezdp\_security\_handle\_t
sec_handle, void __cmem * iv_ptr, uint32_t iv_size) [static]
```

Non blocking version of [ezdp\\_write\\_security\\_initial\\_vector\(\)](#).

**Parameters:**

- [in] *sec\_handle* - security handle
- [in] *iv\_ptr* - pointer to the security initial vector in CMEM
- [in] *iv\_size* - size of the *iv\_ptr*

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was written to the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_read_security_initial_vector (ezdp\_security\_handle\_t
sec_handle, void __cmem * iv_ptr, uint32_t iv_size) [static]
```

Copy the security initial vector from the security context memory to CMEM.

**Parameters:**

- [in] *sec\_handle* - security handle
- [out] *iv\_ptr* - pointer in CMEM to write the security initial vector to
- [in] *iv\_size* - size of the *iv\_ptr*

**Note:**

- The size of the security initial vector is derived from algorithm type, see `ezdp_sec_mac_size` enum.

**Returns:**

none

```
static __always_inline void ezdp_read_security_initial_vector_async (ezdp\_security\_handle\_t
sec_handle, void __cmem * iv_ptr, uint32_t iv_size) [static]
```

Non blocking version of [ezdp\\_read\\_security\\_initial\\_vector\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle  
 [out] *iv\_ptr* - pointer in CMEM to write the security initial vector to  
 [in] *iv\_size* - size of the *iv\_ptr*

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data is ready in CMEM.

**Returns:**

none

```
static __always_inline uint32_t ezdp_security_initial_vector_size (ezdp\_security\_handle\_t
sec_handle) [static]
```

Return the initial vector size.

The size of the initial vector is according to alg type configured in security handle

**Parameters:**

[in] *sec\_handle* - security handle, only alg\_type is used

**Returns:**

0 - error, initial vector size

```
static __always_inline void ezdp_write_security_context (ezdp\_security\_handle\_t sec_handle,
void __cmem * cntx_ptr,  uint32_t cntx_size) [static]
```

Copy the security context from CMEM to the security context memory.

**Parameters:**

[in] *sec\_handle* - security handle  
 [in] *cntx\_ptr* - pointer to the security context data in CMEM  
 [in] *cntx\_size* - size of the *cntx\_ptr*

**Note:**

- The size of the security context is 64 bytes

**Returns:**

none

```
static __always_inline void ezdp_write_security_context_async (ezdp\_security\_handle\_t
sec_handle,  void __cmem * cntx_ptr,  uint32_t cntx_size) [static]
```

Non blocking version of [ezdp\\_write\\_security\\_context\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle  
 [in] *cntx\_ptr* - pointer to context data on CMEM  
 [in] *cntx\_size* - size of the *cntx\_ptr*

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data was written to the security context memory.

**Returns:**

none

```
static __always_inline void ezdp_read_security_context(ezdp\_security\_handle\_t sec_handle,
void __cmem * cntx_ptr,  uint32_t cntx_size) [static]
```

Copy the security context from the security context memory to CMEM.

**Parameters:**

[in] *sec\_handle* - security handle  
[out] *cntx\_ptr* - pointer in CMEM to write the context data to  
[in] *cntx\_size* - size of the cntx\_ptr

**Note:**

- The size of the security context is 64 bytes

**Returns:**

none

```
static __always_inline void ezdp_read_security_context_async(ezdp\_security\_handle\_t
sec_handle,  void __cmem * cntx_ptr,  uint32_t cntx_size) [static]
```

Non blocking version of [ezdp\\_read\\_security\\_context\(\)](#).

**Parameters:**

[in] *sec\_handle* - security handle  
[out] *cntx\_ptr* - pointer CMEM to write the context data to  
[in] *cntx\_size* - size of the cntx\_ptr

**Note:**

- Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the data is ready in CMEM.

**Returns:**

none

```
static __always_inline uint32_t ezdp_security_block_size(ezdp\_security\_handle\_t sec_handle)
[static]
```

Return the algorithmic engine minimal block size.

The size of the block is according to alg type configured in security handle

**Parameters:**

[in] *sec\_handle* - security handle, only alg\_type is used

**Returns:**

0 - error, block size

## dpe/dp/include/ezdp\_security\_defs.h File Reference

### Data Structures

- struct [ezdp\\_security\\_handle](#)

### security handle configuration data structure Defines

- #define [EZDP\\_SECURITY\\_CONTEXT\\_SIZE](#) 80
- *The size of the security context.* #define [EZDP\\_SECURITY\\_CLUSTER\\_MAX\\_CONTEXTS](#) 128
- *The number of security contexts in cluster.* #define [EZDP\\_SECURITY\\_HANDLE\\_CONTEXT\\_ID\\_SIZE](#) 8
- #define [EZDP\\_SECURITY\\_HANDLE\\_CONTEXT\\_ID\\_OFFSET](#) 0
- #define [EZDP\\_SECURITY\\_HANDLE\\_RESERVED8\\_15\\_SIZE](#) 8
- #define [EZDP\\_SECURITY\\_HANDLE\\_RESERVED8\\_15\\_OFFSET](#) 8
- #define [EZDP\\_SECURITY\\_HANDLE\\_ALG\\_TYPE\\_SIZE](#) 8
- #define [EZDP\\_SECURITY\\_HANDLE\\_ALG\\_TYPE\\_OFFSET](#) 16
- #define [EZDP\\_SECURITY\\_HANDLE\\_RESERVED24\\_31\\_SIZE](#) 8
- #define [EZDP\\_SECURITY\\_HANDLE\\_RESERVED24\\_31\\_OFFSET](#) 24

### Typedefs

- typedef uint32\_t [ezdp\\_security\\_handle\\_t](#)

### Enumerations

- enum [ezdp\\_sec\\_alg](#) { [EZDP\\_DES\\_CBC\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_DES\_CBC, [EZDP\\_3DES2\\_CBC\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES2\_CBC, [EZDP\\_3DES3\\_CBC\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES3\_CBC, [EZDP\\_DES\\_CFB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_DES\_CFB, [EZDP\\_3DES2\\_CFB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES2\_CFB, [EZDP\\_3DES3\\_CFB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES3\_CFB, [EZDP\\_DES\\_OFB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_DES\_OFB, [EZDP\\_3DES2\\_OFB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES2\_OFB, [EZDP\\_3DES3\\_OFB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES3\_OFB, [EZDP\\_DES\\_CTR\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_DES\_CTR, [EZDP\\_3DES2\\_CTR\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES2\_CTR, [EZDP\\_3DES3\\_CTR\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES3\_CTR, [EZDP\\_DES\\_ECB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_DES\_ECB, [EZDP\\_3DES2\\_ECB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES2\_ECB, [EZDP\\_3DES3\\_ECB\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_3DES3\_ECB, [EZDP\\_AES\\_CBC\\_128\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CBC\_128, [EZDP\\_AES\\_CBC\\_192\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CBC\_192, [EZDP\\_AES\\_CBC\\_256\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CBC\_256, [EZDP\\_AES\\_CFB\\_128\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CFB\_128, [EZDP\\_AES\\_CFB\\_192\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CFB\_192, [EZDP\\_AES\\_CFB\\_256\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CFB\_256, [EZDP\\_AES\\_OFB\\_128\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_OFB\_128, [EZDP\\_AES\\_OFB\\_192\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_OFB\_192, [EZDP\\_AES\\_OFB\\_256\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_OFB\_256, [EZDP\\_AES\\_CTR\\_128\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CTR\_128, [EZDP\\_AES\\_CTR\\_192\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CTR\_192, [EZDP\\_AES\\_CTR\\_256\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CTR\_256, [EZDP\\_AES\\_ECB\\_128\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_ECB\_128, [EZDP\\_AES\\_ECB\\_192\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_ECB\_192, [EZDP\\_AES\\_ECB\\_256\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_ECB\_256, [EZDP\\_AES\\_CCM\\_128\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CCM\_128, [EZDP\\_AES\\_CCM\\_192\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CCM\_192, [EZDP\\_AES\\_CCM\\_256\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_CCM\_256, [EZDP\\_AES\\_GCM\\_128\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_GCM\_128, [EZDP\\_AES\\_GCM\\_192\\_ALG](#) = EZASM\_SECURITY\_ALGORITHM\_TYPE\_AES\_GCM\_192,

```

EZDP_AES_GCM_256_ALG = EZASM_SECURITY_ALGORITHM_TYPE_AES_GCM_256,
EZDP_MD5_ALG = EZASM_SECURITY_ALGORITHM_TYPE_MD5, EZDP_SHA1_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_SHA1, EZDP_GHASH_128_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_GHASH_128, EZDP_GHASH_192_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_GHASH_192, EZDP_GHASH_256_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_GHASH_256, EZDP_SHA2_224_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_SHA2_224, EZDP_SHA2_256_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_SHA2_256, EZDP_SHA2_384_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_SHA2_384, EZDP_SHA2_512_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_SHA2_512, EZDP_AES_CMAC_128_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_AES_CMAC_128, EZDP_AES_CMAC_192_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_AES_CMAC_192, EZDP_AES_CMAC_256_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_AES_CMAC_256, EZDP_AES_XCBC_MAC_128_ALG =
EZASM_SECURITY_ALGORITHM_TYPE_AES_XCBC_MAC_128 }

```

*security encryption types.*

- enum `ezdp_sec_key_size` { `EZDP_DES_XXX_KEY_SIZE` = 8, `EZDP_3DES2_XXX_KEY_SIZE` = 16, `EZDP_3DES3_XXX_KEY_SIZE` = 24, `EZDP_AES_XXX_128_KEY_SIZE` = 16, `EZDP_AES_XXX_192_KEY_SIZE` = 24, `EZDP_AES_XXX_256_KEY_SIZE` = 32, `EZDP_AES_XXX_MAC_128_KEY_SIZE` = 16, `EZDP_AES_XXX_MAC_192_KEY_SIZE` = 24, `EZDP_AES_XXX_MAC_256_KEY_SIZE` = 32, `EZDP_GHASH_128_KEY_SIZE` = 16, `EZDP_GHASH_192_KEY_SIZE` = 24, `EZDP_GHASH_256_KEY_SIZE` = 32 }

*security key size possible values.*

- enum `ezdp_sec_initial_vector_size` { `EZDP_DES_IV_SIZE` = 8, `EZDP_3DES_IV_SIZE` = 8, `EZDP_AES_IV_SIZE` = 16, `EZDP_XXX_MAC_IV_SIZE` = 16 }

*security key size possible values.*

- enum `ezdp_sec_state_size` { `EZDP_DES_STATE_SIZE` = 8, `EZDP_AES_STATE_SIZE` = 16, `EZDP_AES_CCM_STATE_SIZE` = 48, `EZDP_AES_GCM_STATE_SIZE` = 48, `EZDP_AES_CMAC_XXX_STATE_SIZE` = 32, `EZDP_AES_XCBC_MAC_STATE_SIZE` = 48, `EZDP_MD5_STATE_SIZE` = 32, `EZDP_SHA1_STATE_SIZE` = 36, `EZDP_GHASH_STATE_SIZE` = 32, `EZDP_GHASH_XXX_STATE_SIZE` = 32, `EZDP_SHA2_224_STATE_SIZE` = 48, `EZDP_SHA2_256_STATE_SIZE` = 48, `EZDP_SHA2_384_STATE_SIZE` = 80, `EZDP_SHA2_512_STATE_SIZE` = 80 }

*security algorithm type possible values.*

- enum `ezdp_sec_mac_size` { `EZDP_AES_CCM_MAC_SIZE` = 16, `EZDP_AES_GCM_MAC_SIZE` = 16, `EZDP_MD5_MAC_SIZE` = 16, `EZDP_SHA1_MAC_SIZE` = 20, `EZDP_SHA2_224_MAC_SIZE` = 28, `EZDP_SHA2_256_MAC_SIZE` = 32, `EZDP_SHA2_384_MAC_SIZE` = 48, `EZDP_SHA2_512_MAC_SIZE` = 64, `EZDP_AES_CMAC_XXX_MAC_SIZE` = 16, `EZDP_AES_XCBC_MAC_128_MAC_SIZE` = 16, `EZDP_GHASH_MAC_SIZE` = 16 }

*security algorithm type possible values.*

- enum `ezdp_sec_block_size` { `EZDP_DES_BLOCK_SIZE` = 8, `EZDP_3DES_BLOCK_SIZE` = 8, `EZDP_AES_BLOCK_SIZE` = 16, `EZDP_GHASH_BLOCK_SIZE` = 16, `EZDP_MD5_BLOCK_SIZE` = 64, `EZDP_SHA1_BLOCK_SIZE` = 64, `EZDP_SHA2_224_BLOCK_SIZE` = 64, `EZDP_SHA2_256_BLOCK_SIZE` = 64, `EZDP_SHA2_384_BLOCK_SIZE` = 128, `EZDP_SHA2_512_BLOCK_SIZE` = 128 }

*security block size possible values.*

---

## Define Documentation

### #define EZDP\_SECURITY\_CONTEXT\_SIZE 80

The size of the security context.



```
#define EZDP_SECURITY_CLUSTER_MAX_CONTEXTS 128
```

The number of security contexts in cluster.

```
#define EZDP_SECURITY_HANDLE_CONTEXT_ID_SIZE 8
```

```
#define EZDP_SECURITY_HANDLE_CONTEXT_ID_OFFSET 0
```

```
#define EZDP_SECURITY_HANDLE_RESERVED8_15_SIZE 8
```

```
#define EZDP_SECURITY_HANDLE_RESERVED8_15_OFFSET 8
```

```
#define EZDP_SECURITY_HANDLE_ALG_TYPE_SIZE 8
```

```
#define EZDP_SECURITY_HANDLE_ALG_TYPE_OFFSET 16
```

```
#define EZDP_SECURITY_HANDLE_RESERVED24_31_SIZE 8
```

```
#define EZDP_SECURITY_HANDLE_RESERVED24_31_OFFSET 24
```

## Typedef Documentation

```
typedef uint32_t ezdp\_security\_handle\_t
```

## Enumeration Type Documentation

```
enum ezdp\_sec\_alg
```

security encryption types.

### Enumerator:

**EZDP\_DES\_CBC\_ALG** Type: Encryption, Algorithm: DES, Mode: CBC, Key: single key of 64 bits.

**EZDP\_3DES2\_CBC\_ALG** Type: Encryption, Algorithm: 3DES, Mode: CBC, Key: 2 keys of 64 bits (K1=K3, K2).

**EZDP\_3DES3\_CBC\_ALG** Type: Encryption, Algorithm: 3DES, Mode: CBC, Key: 3 keys of 64 bits (K1, K2, K3).

**EZDP\_DES\_CFB\_ALG** Type: Encryption, Algorithm: DES, Mode: CFB, Key: single key of 64 bits.

**EZDP\_3DES2\_CFB\_ALG** Type: Encryption, Algorithm: 3DES, Mode: CFB, Key: 2 keys of 64 bits (K1=K3, K2).

**EZDP\_3DES3\_CFB\_ALG** Type: Encryption, Algorithm: 3DES, Mode: CFB, Key: 3 keys of 64 bits (K1, K2, K3).

**EZDP\_DES\_OFB\_ALG** Type: Encryption, Algorithm: DES, Mode: OFB, Key: single key of 64 bits.

**EZDP\_3DES2\_OFB\_ALG** Type: Encryption, Algorithm: 3DES, Mode: OFB, Key: 2 keys of 64 bits (K1=K3, K2).

**EZDP\_3DES3\_OFB\_ALG** Type: Encryption, Algorithm: 3DES, Mode: OFB, Key: 3 keys of 64 bits (K1, K2, K3).

**EZDP\_DES\_CTR\_ALG** Type: Encryption, Algorithm: DES, Mode: CTR, Key: single key of 64 bits.

**EZDP\_3DES2\_CTR\_ALG** Type: Encryption, Algorithm: 3DES, Mode: CTR, Key: 2 keys of 64 bits (K1=K3, K2).

**EZDP\_3DES3\_CTR\_ALG** Type: Encryption, Algorithm: 3DES, Mode: CTR, Key: 3 keys of 64 bits (K1, K2, K3).

**EZDP\_DES\_ECB\_ALG** Type: Encryption, Algorithm: DES, Mode: ECB, Key: single key of 64 bits.

**EZDP\_3DES2\_ECB\_ALG** Type: Encryption, Algorithm: 3DES, Mode: ECB, Key: 2 keys of 64 bits (K1=K3, K2).

**EZDP\_3DES3\_ECB\_ALG** Type: Encryption, Algorithm: 3DES, Mode: ECB, Key: 3 keys of 64 bits (K1, K2, K3).

**EZDP\_AES\_CBC\_128\_ALG** Type: Encryption, Algorithm: AES, Mode: CBC, Key: 128 bits.

**EZDP\_AES\_CBC\_192\_ALG** Type: Encryption, Algorithm: AES, Mode: CBC, Key: 192 bits.

**EZDP\_AES\_CBC\_256\_ALG** Type: Encryption, Algorithm: AES, Mode: CBC, Key: 192 bits.

**EZDP\_AES\_CFB\_128\_ALG** Type: Encryption, Algorithm: AES, Mode: CFB, Key: 128 bits.

**EZDP\_AES\_CFB\_192\_ALG** Type: Encryption, Algorithm: AES, Mode: CFB, Key: 192 bits.

**EZDP\_AES\_CFB\_256\_ALG** Type: Encryption, Algorithm: AES, Mode: CFB, Key: 192 bits.

**EZDP\_AES\_OFB\_128\_ALG** Type: Encryption, Algorithm: AES, Mode: OFB, Key: 128 bits.

**EZDP\_AES\_OFB\_192\_ALG** Type: Encryption, Algorithm: AES, Mode: OFB, Key: 192 bits.

**EZDP\_AES\_OFB\_256\_ALG** Type: Encryption, Algorithm: AES, Mode: OFB, Key: 192 bits.

**EZDP\_AES\_CTR\_128\_ALG** Type: Encryption, Algorithm: AES, Mode: CTR, Key: 128 bits.

**EZDP\_AES\_CTR\_192\_ALG** Type: Encryption, Algorithm: AES, Mode: CTR, Key: 192 bits.

**EZDP\_AES\_CTR\_256\_ALG** Type: Encryption, Algorithm: AES, Mode: CTR, Key: 192 bits.

**EZDP\_AES\_ECB\_128\_ALG** Type: Encryption, Algorithm: AES, Mode: ECB, Key: 128 bits.

**EZDP\_AES\_ECB\_192\_ALG** Type: Encryption, Algorithm: AES, Mode: ECB, Key: 192 bits.

**EZDP\_AES\_ECB\_256\_ALG** Type: Encryption, Algorithm: AES, Mode: ECB, Key: 192 bits.

**EZDP\_AES\_CCM\_128\_ALG** Type: Encryption and Authentication, Algorithm: AES, Mode: CCM, Key: 128 bits.

**EZDP\_AES\_CCM\_192\_ALG** Type: Encryption and Authentication, Algorithm: AES, Mode: CCM, Key: 192 bits.

**EZDP\_AES\_CCM\_256\_ALG** Type: Encryption and Authentication, Algorithm: AES, Mode: CCM, Key: 192 bits.

**EZDP\_AES\_GCM\_128\_ALG** Type: Encryption and Authentication, Algorithm: AES, Mode: GCM, Key: 128 bits.

**EZDP\_AES\_GCM\_192\_ALG** Type: Encryption and Authentication, Algorithm: AES, Mode: GCM, Key: 192 bits.

**EZDP\_AES\_GCM\_256\_ALG** Type: Encryption and Authentication, Algorithm: AES, Mode: GCM, Key: 192 bits.

**EZDP\_MD5\_ALG** Type: Authentication, Algorithm: MD5.

**EZDP\_SHA1\_ALG** Type: Authentication, Algorithm: SHA1.

**EZDP\_GHASH\_128\_ALG** Type: Authentication, Algorithm: GHASH Key: 128 bits.

**EZDP\_GHASH\_192\_ALG** Type: Authentication, Algorithm: GHASH Key: 192 bits.

**EZDP\_GHASH\_256\_ALG** Type: Authentication, Algorithm: GHASH Key: 256 bits.

**EZDP\_SHA2\_224\_ALG** Type: Authentication, Algorithm: SHA2-224.

**EZDP\_SHA2\_256\_ALG** Type: Authentication, Algorithm: SHA2-256.

**EZDP\_SHA2\_384\_ALG** Type: Authentication, Algorithm: SHA2-384.

**EZDP\_SHA2\_512\_ALG** Type: Authentication, Algorithm: SHA2-512.

**EZDP\_AES\_CMAC\_128\_ALG** Type: Authentication, Algorithm: AES, Mode: CMAC, Key: 128 bits.

**EZDP\_AES\_CMAC\_192\_ALG** Type: Authentication, Algorithm: AES, Mode: CMAC, Key: 192 bits.

**EZDP\_AES\_CMAC\_256\_ALG** Type: Authentication, Algorithm: AES, Mode: CMAC, Key: 192 bits.

**EZDP\_AES\_XCBC\_MAC\_128\_ALG** Type: Authentication, Algorithm: AES, Mode: XCBC MAC, Key: 128 bits.

#### enum [ezdp\\_sec\\_key\\_size](#)

security key size possible values.

##### Enumerator:

**EZDP\_DES\_XXX\_KEY\_SIZE** Type: Encryption, Algorithm: DES - key size 8 bytes (single key of 64 bits).

**EZDP\_3DES2\_XXX\_KEY\_SIZE** Type: Encryption, Algorithm: 3DES - key size 16 bytes (2 keys of 64 bits (K1=K3, K2)).

**EZDP\_3DES3\_XXX\_KEY\_SIZE** Type: Encryption, Algorithm: 3DES - key size 24 bytes (3 keys of 64 bits (K1, K2, K3)).

**EZDP\_AES\_XXX\_128\_KEY\_SIZE** Type: Encryption, Algorithm: AES - key size 16 bytes.

**EZDP\_AES\_XXX\_192\_KEY\_SIZE** Type: Encryption, Algorithm: AES - key size 24 bytes.

**EZDP\_AES\_XXX\_256\_KEY\_SIZE** Type: Encryption, Algorithm: AES - key size 32 bytes.

**EZDP\_AES\_XXX\_MAC\_128\_KEY\_SIZE** Type: Authentication, Algorithm: AES - key size 16 bytes.

**EZDP\_AES\_XXX\_MAC\_192\_KEY\_SIZE** Type: Authentication, Algorithm: AES, Mode - key size 24 bytes.

**EZDP\_AES\_XXX\_MAC\_256\_KEY\_SIZE** Type: Authentication, Algorithm: AES - key size 32 bytes.

**EZDP\_GHASH\_128\_KEY\_SIZE** Type: Encryption, Algorithm: GHASH - key size 16 bytes.

**EZDP\_GHASH\_192\_KEY\_SIZE** Type: Encryption, Algorithm: GHASH - key size 24 bytes.

**EZDP\_GHASH\_256\_KEY\_SIZE** Type: Encryption, Algorithm: GHASH - key size 32 bytes.

#### enum [ezdp\\_sec\\_initial\\_vector\\_size](#)

security key size possible values.

**Enumerator:**

**EZDP\_DES\_IV\_SIZE** Type: Encryption, Algorithm: DES - Initial vector size 8 bytes.

**EZDP\_3DES\_IV\_SIZE** Type: Encryption, Algorithm: 3DES - Initial vector size 8 bytes.

**EZDP\_AES\_IV\_SIZE** Type: Encryption, Algorithm: AES - Initial vector size 16 bytes.

**EZDP\_XXX\_MAC\_IV\_SIZE** Type: Authentication, Algorithm: AES - - Initial vector size 16 bytes.

**enum [ezdp\\_sec\\_state\\_size](#)**

security algorithm type possible values.

**Enumerator:**

**EZDP\_DES\_STATE\_SIZE** Type: Encryption, Algorithm: DES - state size 8 bytes.

**EZDP\_AES\_STATE\_SIZE** Type: Encryption, Algorithm: AES - state size 16 bytes.

**EZDP\_AES\_CCM\_STATE\_SIZE** Type: Authentication, Algorithm: AES\_CCM - state size 48 bytes.

**EZDP\_AES\_GCM\_STATE\_SIZE** Type: Authentication, Algorithm: AES\_GCM - state size 48 bytes.

**EZDP\_AES\_CMAC\_XXX\_STATE\_SIZE** Type: Authentication, Algorithm: AES\_CMAC\_XXX - state size 32 bytes.

**EZDP\_AES\_XCBC\_MAC\_STATE\_SIZE** Type: Authentication, Algorithm: AES\_XCBC\_MAC\_128 - state size 48 bytes.

**EZDP\_MD5\_STATE\_SIZE** Type: Authentication, Algorithm: MD5 - state size 32 bytes.

**EZDP\_SHA1\_STATE\_SIZE** Type: Authentication, Algorithm: SHA1 - state size 36 bytes.

**EZDP\_GHASH\_STATE\_SIZE** Type: Authentication, Algorithm: GHASH - state size 32 bytes.

**EZDP\_GHASH\_XXX\_STATE\_SIZE** Type: Authentication, Algorithm: GHASH - state size 32 bytes.

**EZDP\_SHA2\_224\_STATE\_SIZE** Type: Authentication, Algorithm: SHA2-224 - state size 48 bytes.

**EZDP\_SHA2\_256\_STATE\_SIZE** Type: Authentication, Algorithm: SHA2-256 - state size 48 bytes.

**EZDP\_SHA2\_384\_STATE\_SIZE** Type: Authentication, Algorithm: SHA2-384 - state size 80 bytes.

**EZDP\_SHA2\_512\_STATE\_SIZE** Type: Authentication, Algorithm: SHA2-512 - state size 80 bytes.

**enum [ezdp\\_sec\\_mac\\_size](#)**

security algorithm type possible values.

**Enumerator:**

**EZDP\_AES\_CCM\_MAC\_SIZE** Type: Authentication, Algorithm: AES\_CCM - MAC size 16 bytes.

**EZDP\_AES\_GCM\_MAC\_SIZE** Type: Authentication, Algorithm: AES\_GCM - MAC size 16 bytes.

**EZDP\_MD5\_MAC\_SIZE** Type: Authentication, Algorithm: MD5 - MAC size 16 bytes.

**EZDP\_SHA1\_MAC\_SIZE** Type: Authentication, Algorithm: SHA1 - MAC size 20 bytes.

**EZDP\_SHA2\_224\_MAC\_SIZE** Type: Authentication, Algorithm: SHA2-224 - MAC size 32 bytes.

**EZDP\_SHA2\_256\_MAC\_SIZE** Type: Authentication, Algorithm: SHA2-256 - MAC size 32 bytes.

**EZDP\_SHA2\_384\_MAC\_SIZE** Type: Authentication, Algorithm: SHA2-384 - MAC size 48 bytes.

**EZDP\_SHA2\_512\_MAC\_SIZE** Type: Authentication, Algorithm: SHA2-512 - MAC size 64 bytes.

**EZDP\_AES\_CMAC\_XXX\_MAC\_SIZE** Type: Authentication, Algorithm: AES\_CMAC\_XXX - MAC size 16 bytes.

**EZDP\_AES\_XCBC\_MAC\_128\_MAC\_SIZE** Type: Authentication, Algorithm: AES\_XCBC\_MAC\_128 - MAC size 16 bytes.

**EZDP\_GHASH\_MAC\_SIZE** Type: Authentication, Algorithm: GHASH - MAC size 16 bytes.

#### enum [ezdp\\_sec\\_block\\_size](#)

security block size possible values.

#### Enumerator:

**EZDP\_DES\_BLOCK\_SIZE** Type: Encryption, Algorithm: DES - Block size 8 bytes.

**EZDP\_3DES\_BLOCK\_SIZE** Type: Encryption, Algorithm: 3DES - Block size 8 bytes.

**EZDP\_AES\_BLOCK\_SIZE** Type: Encryption, Algorithm: AES - Block size 16 bytes.

**EZDP\_GHASH\_BLOCK\_SIZE** Type: Encryption, Algorithm: GHASH - Block size 16 bytes.

**EZDP\_MD5\_BLOCK\_SIZE** Type: Authentication, Algorithm: MD5 - Block size 64 bytes.

**EZDP\_SHA1\_BLOCK\_SIZE** Type: Authentication, Algorithm: SHA1 - Block size 64 bytes.

**EZDP\_SHA2\_224\_BLOCK\_SIZE** Type: Authentication, Algorithm: SHA2-224 - Block size 64 bytes.

**EZDP\_SHA2\_256\_BLOCK\_SIZE** Type: Authentication, Algorithm: SHA2-256 - Block size 64 bytes.

**EZDP\_SHA2\_384\_BLOCK\_SIZE** Type: Authentication, Algorithm: SHA2-384 - Block size 128 bytes.

**EZDP\_SHA2\_512\_BLOCK\_SIZE** Type: Authentication, Algorithm: SHA2-512 - Block size 128 bytes.

## dpe/dp/include/ezdp\_string.h File Reference

### Functions

- static \_\_always\_inline void \* [ezdp\\_mem\\_copy](#) (void \*dst, const void \*src, uint32\_t size)
  - *Copy a block of memory.* static \_\_always\_inline void \* [ezdp\\_mem\\_set](#) (void \*dst, int val, uint32\_t size)
  - *Set a block of memory to the specified value.* static uint32\_t [ezdp\\_mem\\_cmp](#) (uint8\_t \_\_cmem \*ptr1, uint8\_t \_\_cmem \*ptr2, uint32\_t size)
  - *Compare two blocks of memory in CMEM.* static \_\_always\_inline uint32\_t [ezdp\\_mem\\_cmp\\_byte\\_skip](#) (uint8\_t \_\_cmem \*ptr1, uint8\_t \_\_cmem \*ptr2, uint32\_t size, uint32\_t ptr1\_offset)
- Compare two blocks of memory in CMEM, skipping intermediate bytes.*
- 

### Function Documentation

**static \_\_always\_inline void\* ezdp\_mem\_copy (void \* dst,   const void \* src,   uint32\_t size)**  
[static]

Copy a block of memory.

#### Parameters:

[in] *dst* - copy destination  
[in] *src* - copy source  
[in] *size* - size in bytes to copy

#### Note:

May not work properly if dst and src overlap.

#### Returns:

pointer to dst

**static \_\_always\_inline void\* ezdp\_mem\_set (void \* dst,   int val,   uint32\_t size)** [static]

Set a block of memory to the specified value.

#### Parameters:

[in] *dst* - copy destination  
[in] *val* - set value  
[in] *size* - size in bytes

#### Returns:

pointer to dst

**static uint32\_t ezdp\_mem\_cmp (uint8\_t \_\_cmem \* ptr1,   uint8\_t \_\_cmem \* ptr2,   uint32\_t size)**  
[inline, static]

Compare two blocks of memory in CMEM.

#### Parameters:

[in] *ptr1* - pointer to first block of memory in CMEM  
[in] *ptr2* - pointer to second block of memory in CMEM  
[in] *size* - number of bytes to compare

#### Returns:

uint32\_t - 0 if memory is identical

```
static __always_inline uint32_t ezdp_mem_cmp_byte_skip(uint8_t __cmem * ptr1, uint8_t __cmem * ptr2, uint32_t size, uint32_t ptr1_offset) [static]
```

Compare two blocks of memory in CMEM, skipping intermediate bytes.

Ignores a single byte every `byte_skip_cycle_size` bytes. This can be used, for example, to compare data that is ECC protected in-band using a single byte of protection data for each 16 or 32 bytes.

**Parameters:**

- [in] *ptr1* - pointer to first block of memory in CMEM
- [in] *ptr2* - pointer to second block of memory in CMEM
- [in] *size* - number of bytes to compare
- [in] *ptr1\_offset* - offset in *ptr1* to begin from

**Returns:**

uint32\_t - 0 if memory is identical

## dpe/dp/include/ezdp\_time.h File Reference

### Functions

- static \_\_always\_inline uint32\_t [ezdp\\_get\\_system\\_tick](#) (uint64\_t \_\_cmem \*core\_cycles)
- *Get system tick (in core cycles).* static \_\_always\_inline void [ezdp\\_get\\_system\\_tick\\_async](#) (uint64\_t \_\_cmem \*core\_cycles)
- *Non blocking version of [ezdp\\_get\\_system\\_tick\(\)](#).* static \_\_always\_inline uint32\_t [ezdp\\_get\\_real\\_time\\_clock](#) (struct [ezdp\\_rtc](#) \_\_cmem \*real\_time\_clock)
- *Get real time clock.* static \_\_always\_inline void [ezdp\\_get\\_real\\_time\\_clock\\_async](#) (struct [ezdp\\_rtc](#) \_\_cmem \*real\_time\_clock)

*Non blocking version of [ezdp\\_get\\_real\\_time\\_clock\(\)](#).*

---

### Function Documentation

**static \_\_always\_inline uint32\_t ezdp\_get\_system\_tick (uint64\_t \_\_cmem \* core\_cycles) [static]**

Get system tick (in core cycles).

#### Parameters:

[out] *core\_cycles* - pointer to CMEM to write the current system tick into

#### Returns:

uint32\_t - 32 LSB of the system tick

**static \_\_always\_inline void ezdp\_get\_system\_tick\_async (uint64\_t \_\_cmem \* core\_cycles) [static]**

Non blocking version of [ezdp\\_get\\_system\\_tick\(\)](#).

#### Parameters:

[out] *core\_cycles* - pointer to CMEM to write the current system tick into

#### Note:

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the result is ready in CMEM.

#### Returns:

none

**static \_\_always\_inline uint32\_t ezdp\_get\_real\_time\_clock (struct [ezdp\\_rtc](#) \_\_cmem \* real\_time\_clock) [static]**

Get real time clock.

#### Parameters:

[out] *real\_time\_clock* - pointer to CMEM to write the real time clock into

#### Returns:

uint32\_t - 32 LSB of the real time clock

**static \_\_always\_inline void ezdp\_get\_real\_time\_clock\_async (struct [ezdp\\_rtc](#) \_\_cmem \* real\_time\_clock) [static]**



Non blocking version of [ezdp\\_get\\_real\\_time\\_clock\(\)](#).

**Parameters:**

[out] *real\_time\_clock* - pointer to CMEM to write the real time clock into

**Note:**

Call [ezdp\\_sync\(\)](#) to wait for the operation to complete, confirming that the result is ready in CMEM.

**Returns:**

none

## dpe/dp/include/ezdp\_time\_defs.h File Reference

### Data Structures

- struct [ezdp\\_rtc](#)

### [ezdp\\_rtc](#) struct for ezdp Defines

- #define [EZDP\\_RTC\\_SEC\\_SIZE](#) 32
  - #define [EZDP\\_RTC\\_SEC\\_OFFSET](#) 0
  - #define [EZDP\\_RTC\\_SEC\\_WORD\\_SELECT](#) 0
  - #define [EZDP\\_RTC\\_SEC\\_WORD\\_OFFSET](#) 0
  - #define [EZDP\\_RTC\\_NSEC\\_SIZE](#) 32
  - #define [EZDP\\_RTC\\_NSEC\\_OFFSET](#) 32
  - #define [EZDP\\_RTC\\_NSEC\\_WORD\\_SELECT](#) 1
  - #define [EZDP\\_RTC\\_NSEC\\_WORD\\_OFFSET](#) 0
  - #define [EZDP\\_RTC\\_WORD\\_COUNT](#) 2
- 

### Define Documentation

**#define EZDP\_RTC\_SEC\_SIZE 32**

**#define EZDP\_RTC\_SEC\_OFFSET 0**

**#define EZDP\_RTC\_SEC\_WORD\_SELECT 0**

**#define EZDP\_RTC\_SEC\_WORD\_OFFSET 0**

**#define EZDP\_RTC\_NSEC\_SIZE 32**

**#define EZDP\_RTC\_NSEC\_OFFSET 32**

**#define EZDP\_RTC\_NSEC\_WORD\_SELECT 1**

**#define EZDP\_RTC\_NSEC\_WORD\_OFFSET 0**

**#define EZDP\_RTC\_WORD\_COUNT 2**

## dpe/dp/include/ezdp\_version.h File Reference

### Data Structures

- struct [ezdp\\_version](#)

### version info data structure Defines

- #define [EZDP\\_VERSION\\_MAX\\_STRING\\_LENGTH](#) 1024
  - #define [EZDP\\_VERSION\\_STR\(s\) #s](#)
  - #define [EZDP\\_VERSION\\_XSTR\(s\) EZDP\\_VERSION\\_STR\(s\)](#)
  - #define [\\_SCM\\_REV\\_](#) local
  - #define [EZDP\\_VERSION\\_DEF\\_PROJECT\\_NAME](#) "NPS-400"
  - #define [EZDP\\_VERSION\\_DEF\\_MAJOR\\_VER](#) 1
  - #define [EZDP\\_VERSION\\_DEF\\_MINOR\\_VER](#) 9
  - #define [EZDP\\_VERSION\\_DEF\\_VERSION\\_CHAR](#) 't'
  - #define [EZDP\\_VERSION\\_DEF\\_VERSION\\_STRING](#) ""
  - #define [EZDP\\_VERSION\\_DEF\\_PATCH\\_MAJOR\\_VER](#) 0
  - #define [EZDP\\_VERSION\\_DEF\\_PATCH\\_MINOR\\_VER](#) 0
  - #define [EZDP\\_VERSION\\_DEF\\_PATCH\\_MICRO\\_VER](#) 0
  - #define [ezdp\\_version\\_get\\_string\(version\\_info\\_, version\\_string\\_\)](#)
- 

### Define Documentation

**#define EZDP\_VERSION\_MAX\_STRING\_LENGTH 1024**

**#define EZDP\_VERSION\_STR(s) #s**

**#define EZDP\_VERSION\_XSTR(s) EZDP\_VERSION\_STR(s)**

**#define \_SCM\_REV\_ local**

**#define EZDP\_VERSION\_DEF\_PROJECT\_NAME "NPS-400"**

**#define EZDP\_VERSION\_DEF\_MAJOR\_VER 1**

**#define EZDP\_VERSION\_DEF\_MINOR\_VER 9**

**#define EZDP\_VERSION\_DEF\_VERSION\_CHAR 't'**

**#define EZDP\_VERSION\_DEF\_VERSION\_STRING ""**

**#define EZDP\_VERSION\_DEF\_PATCH\_MAJOR\_VER 0**

**#define EZDP\_VERSION\_DEF\_PATCH\_MINOR\_VER 0**

**#define EZDP\_VERSION\_DEF\_PATCH\_MICRO\_VER 0**

**#define ezdp\_version\_get\_string(version\_info\_, version\_string\_)**

# Index

- \_\_aligned\_cmemb\_ext\_addr  
ezdp\_defs.h, 346
- \_\_alter\_cmemb\_shared\_var  
ezdp\_defs.h, 346
- \_\_alter\_cmemb\_var  
ezdp\_defs.h, 346
- \_\_cmemb  
ezdp\_defs.h, 346
- \_\_cmemb\_shared\_var  
ezdp\_defs.h, 346
- \_\_cmemb\_var  
ezdp\_defs.h, 346
- \_\_ememb\_var  
ezdp\_defs.h, 346
- \_\_imemb\_1\_cluster\_func  
ezdp\_defs.h, 346
- \_\_imemb\_1\_cluster\_var  
ezdp\_defs.h, 346
- \_\_imemb\_16\_cluster\_func  
ezdp\_defs.h, 346
- \_\_imemb\_16\_cluster\_var  
ezdp\_defs.h, 346
- \_\_imemb\_2\_cluster\_func  
ezdp\_defs.h, 346
- \_\_imemb\_2\_cluster\_var  
ezdp\_defs.h, 346
- \_\_imemb\_4\_cluster\_func  
ezdp\_defs.h, 346
- \_\_imemb\_4\_cluster\_var  
ezdp\_defs.h, 346
- \_\_imemb\_all\_cluster\_func  
ezdp\_defs.h, 347
- \_\_imemb\_all\_cluster\_var  
ezdp\_defs.h, 346
- \_\_imemb\_half\_cluster\_func  
ezdp\_defs.h, 346
- \_\_imemb\_half\_cluster\_var  
ezdp\_defs.h, 346
- \_\_imemb\_private\_var  
ezdp\_defs.h, 346
- \_\_no\_inline  
ezdp\_defs.h, 346
- \_\_packed  
ezdp\_defs.h, 346
- \_\_packed\_struct  
ezdp\_defs.h, 346
- \_\_pad0\_\_  
ezdp\_1588\_header, 8  
ezdp\_1step\_1588\_header, 9  
ezdp\_2step\_1588\_header, 11  
ezdp\_app\_schlr\_status, 13  
ezdp\_bitwise\_ctr\_cfg, 14  
ezdp\_buffer\_desc, 15  
ezdp\_congestion\_status, 17  
ezdp\_ctr\_msg, 19  
ezdp\_decode\_eth\_type\_retval, 21  
ezdp\_decode\_ip\_protocol\_retval, 26  
ezdp\_decode\_ipv4\_control, 30  
ezdp\_decode\_ipv4\_errors, 31  
ezdp\_decode\_ipv4\_result, 33  
ezdp\_decode\_ipv4\_retval, 35  
ezdp\_decode\_ipv6\_control, 37  
ezdp\_decode\_ipv6\_errors, 39  
ezdp\_decode\_ipv6\_result, 41  
ezdp\_decode\_ipv6\_retval, 43  
ezdp\_decode\_mac\_control, 45  
ezdp\_decode\_mac\_errors, 48  
ezdp\_decode\_mac\_protocol\_type, 50  
ezdp\_decode\_mac\_result, 53  
ezdp\_decode\_mac\_retval, 55  
ezdp\_decode\_mpls\_label\_result, 57  
ezdp\_decode\_mpls\_label\_retval, 59  
ezdp\_decode\_mpls\_result, 61  
ezdp\_decode\_mpls\_retval, 64  
ezdp\_decode\_tcp\_errors, 67  
ezdp\_decode\_tcp\_retval, 68  
ezdp\_dual\_ctr\_cfg, 74  
ezdp\_dual\_ctr\_result, 76  
ezdp\_ext\_addr, 77  
ezdp\_flow\_control\_status, 80  
ezdp\_frame\_desc, 82  
ezdp\_group\_schlr\_status, 85  
ezdp\_hier\_tb\_ctr\_cfg, 86  
ezdp\_hier\_tb\_result, 88  
ezdp\_hier\_tb\_ug\_app\_bits, 90  
ezdp\_hier\_tb\_update, 93  
ezdp\_input\_queue\_status, 94  
ezdp\_job\_container\_cmd\_desc, 96  
ezdp\_job\_container\_desc, 98  
ezdp\_job\_discard\_cmd\_info, 101  
ezdp\_job\_queue\_cmd\_info, 102  
ezdp\_job\_rx\_confirmation\_info, 103  
ezdp\_job\_rx\_info, 105  
ezdp\_job\_rx\_interface\_info, 107  
ezdp\_job\_rx\_loopback\_info, 110  
ezdp\_job\_rx\_timer\_info, 111  
ezdp\_job\_transmit\_cmd\_info, 113  
ezdp\_job\_tx\_info, 115  
ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element, 124  
ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element, 126  
ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element, 128  
ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element, 130  
ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 132  
ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 134  
ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element, 136

ezdp\_lookup\_ext\_tcaml\_index\_8B\_data\_result\_element, 138  
 ezdp\_lookup\_ext\_tcaml\_index\_result\_element, 140  
 ezdp\_lookup\_ext\_tcaml\_retval, 141  
 ezdp\_lookup\_int\_tcaml\_standard\_result, 149  
 ezdp\_output\_queue\_status, 155  
 ezdp\_pci\_addr, 156  
 ezdp\_pci\_info, 158  
 ezdp\_pci\_msg, 159  
 ezdp\_pci\_msg\_ctrl, 160  
 ezdp\_pci\_msg\_payload\_ats, 161  
 ezdp\_pci\_msg\_payload\_elbi, 162  
 ezdp\_pci\_msg\_payload\_msix, 163  
 ezdp\_posted\_ctr\_msg, 164  
 ezdp\_security\_handle, 168  
 ezdp\_single\_ctr\_cfg, 169  
 ezdp\_sum\_addr\_table\_desc, 173  
 ezdp\_tb\_ctr\_cfg, 175  
 ezdp\_tb\_ctr\_result, 177  
 ezdp\_watchdog\_accumulative\_window\_cfg, 181  
 ezdp\_watchdog\_ctr\_cfg, 183  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 ezdp\_watchdog\_sliding\_window\_cfg, 188  
 \_pad1\_  
 ezdp\_1588\_header, 8  
 ezdp\_1step\_1588\_header, 10  
 ezdp\_2step\_1588\_header, 11  
 ezdp\_bitwise\_ctr\_cfg, 14  
 ezdp\_congestion\_status, 17  
 ezdp\_ctr\_msg, 19  
 ezdp\_decode\_ipv4\_result, 34  
 ezdp\_decode\_ipv6\_control, 37  
 ezdp\_decode\_ipv6\_result, 42  
 ezdp\_decode\_ipv6\_retval, 44  
 ezdp\_decode\_mac\_result, 53  
 ezdp\_decode\_mac\_retval, 55  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 ezdp\_decode\_tcp\_retval, 68  
 ezdp\_dual\_ctr\_cfg, 74  
 ezdp\_ext\_addr, 77  
 ezdp\_frame\_desc, 82  
 ezdp\_hier\_tb\_ctr\_cfg, 87  
 ezdp\_hier\_tb\_result, 89  
 ezdp\_job\_container\_desc, 98  
 ezdp\_job\_discard\_cmd\_info, 101  
 ezdp\_job\_rx\_info, 105  
 ezdp\_job\_rx\_interface\_info, 108  
 ezdp\_job\_rx\_loopback\_info, 110  
 ezdp\_job\_rx\_timer\_info, 111  
 ezdp\_job\_tx\_info, 115  
 ezdp\_pci\_addr, 156  
 ezdp\_pci\_msg\_ctrl, 160  
 ezdp\_pci\_msg\_payload\_msix, 163  
 ezdp\_posted\_ctr\_msg, 164  
 ezdp\_security\_handle, 168  
 ezdp\_single\_ctr\_cfg, 169  
 ezdp\_tb\_ctr\_cfg, 176  
 ezdp\_tb\_ctr\_result, 177  
 ezdp\_watchdog\_accumulative\_window\_cfg, 181  
 ezdp\_watchdog\_ctr\_cfg, 183  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 ezdp\_watchdog\_sliding\_window\_cfg, 188  
 \_pad2\_  
 ezdp\_1step\_1588\_header, 10  
 ezdp\_2step\_1588\_header, 11  
 ezdp\_bitwise\_ctr\_cfg, 14  
 ezdp\_congestion\_status, 18  
 ezdp\_ctr\_msg, 19  
 ezdp\_decode\_ipv6\_result, 42  
 ezdp\_decode\_mac\_result, 54  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 ezdp\_decode\_tcp\_retval, 68  
 ezdp\_dual\_ctr\_cfg, 74  
 ezdp\_ext\_addr, 77  
 ezdp\_frame\_desc, 82  
 ezdp\_hier\_tb\_ctr\_cfg, 87  
 ezdp\_hier\_tb\_result, 89  
 ezdp\_job\_rx\_info, 105  
 ezdp\_job\_rx\_interface\_info, 108  
 ezdp\_job\_tx\_info, 116  
 ezdp\_pci\_addr, 156  
 ezdp\_pci\_msg\_payload\_msix, 163  
 ezdp\_posted\_ctr\_msg, 164  
 ezdp\_single\_ctr\_cfg, 169  
 ezdp\_tb\_ctr\_cfg, 176  
 ezdp\_tb\_ctr\_result, 178  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 ezdp\_watchdog\_ctr\_cfg, 183  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 ezdp\_watchdog\_sliding\_window\_cfg, 188  
 \_pad3\_  
 ezdp\_2step\_1588\_header, 11  
 ezdp\_bitwise\_ctr\_cfg, 14  
 ezdp\_congestion\_status, 18  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 ezdp\_dual\_ctr\_cfg, 75  
 ezdp\_frame\_desc, 84  
 ezdp\_job\_rx\_interface\_info, 108  
 ezdp\_job\_tx\_info, 116  
 ezdp\_single\_ctr\_cfg, 170  
 ezdp\_tb\_ctr\_cfg, 176  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 ezdp\_watchdog\_ctr\_cfg, 183  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 ezdp\_watchdog\_sliding\_window\_cfg, 189  
 \_pad4\_  
 ezdp\_2step\_1588\_header, 12  
 ezdp\_job\_tx\_info, 117  
 ezdp\_single\_ctr\_cfg, 170  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 ezdp\_watchdog\_sliding\_window\_cfg, 189  
 \_pad5\_  
 ezdp\_2step\_1588\_header, 12  
 ezdp\_job\_tx\_info, 117  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 ezdp\_watchdog\_ctr\_check\_result, 186

ezdp\_watchdog\_sliding\_window\_cfg, 189  
 \_\_pad6\_\_  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 ezdp\_watchdog\_sliding\_window\_cfg, 189  
 \_\_pad7\_\_  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 \_\_pad8\_\_  
 ezdp\_watchdog\_ctr\_check\_result, 186  
 \_\_unused  
 ezdp\_defs.h, 346  
 \_SCM\_REV\_  
 ezdp\_version.h, 589  
 accumulative\_events  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 accumulative\_window  
 ezdp\_watchdog\_ctr\_cfg, 183  
 address  
 ezdp\_ext\_addr, 77  
 ezdp\_pci\_addr, 157  
 ezdp\_pci\_msg\_payload\_elbi, 162  
 address\_msb  
 ezdp\_ext\_addr, 77  
 ezdp\_pci\_addr, 157  
 ah\_prot  
 ezdp\_decode\_ip\_protocol\_retval, 26  
 alert  
 ezdp\_watchdog\_ctr\_check\_result, 185  
 any\_match  
 ezdp\_lookup\_ext\_tcam\_index\_result\_element, 140  
 ezdp\_lookup\_ext\_tcam\_retval, 142  
 app\_bits  
 ezdp\_hier\_tb\_ctr\_cfg, 87  
 ezdp\_hier\_tb\_result, 89  
 ezdp\_hier\_tb\_ug\_app\_bits, 90  
 ezdp\_hier\_tb\_update, 93  
 arp  
 ezdp\_decode\_eth\_type\_retval, 22  
 ezdp\_decode\_mac\_protocol\_type, 51  
 assoc\_12B\_data  
 ezdp\_lookup\_int\_tcam\_result, 147  
 assoc\_16B\_data  
 ezdp\_lookup\_int\_tcam\_result, 147  
 assoc\_4B\_data  
 ezdp\_lookup\_int\_tcam\_result, 147  
 assoc\_8B\_data  
 ezdp\_lookup\_int\_tcam\_result, 147  
 assoc\_data  
 ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element,  
 124  
 ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element,  
 126  
 ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element,  
 128  
 ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element,  
 130  
 ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_ele  
 ment, 132  
 ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_ele  
 ment, 134  
 ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_eleme  
 nt, 136  
 ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_eleme  
 nt, 138  
 ezdp\_lookup\_int\_tcam\_retval, 148  
 ats\_payload  
 ezdp\_pci\_msg, 159  
 bar\_num  
 ezdp\_pci\_msg\_ctrl, 160  
 base\_addr  
 ezdp\_mem\_pool\_config, 151  
 ezdp\_ring\_cfg, 166  
 base\_index  
 ezdp\_sum\_addr\_table\_desc, 174  
 buf\_budget\_id  
 ezdp\_2step\_1588\_header, 12  
 ezdp\_frame\_desc, 83  
 buf\_data\_addr  
 ezdp\_driver\_desc, 69  
 buf\_desc  
 ezdp\_2step\_1588\_header, 12  
 ezdp\_frame\_desc, 83  
 ezdp\_linked\_buffers\_desc\_line, 121  
 buf\_info  
 ezdp\_linked\_buffers\_desc\_line, 121  
 build\_number  
 ezdp\_version, 180  
 busy  
 ezdp\_app\_schlr\_status, 13  
 byte  
 ezdp\_dual\_ctr, 73  
 byte\_report\_exceeded  
 ezdp\_dual\_ctr\_cfg, 74  
 byte\_value\_lsb  
 ezdp\_dual\_ctr\_result, 76  
 byte\_value\_msb  
 ezdp\_dual\_ctr\_result, 76  
 byte\_value\_size  
 ezdp\_dual\_ctr\_cfg, 75  
 cache\_size  
 ezdp\_mem\_section\_info, 152  
 checksum  
 ezdp\_1step\_1588\_header, 10  
 checksum\_error  
 ezdp\_decode\_ipv4\_errors, 32  
 checksum\_offset  
 ezdp\_1step\_1588\_header, 10  
 class\_of\_service  
 ezdp\_2step\_1588\_header, 12  
 ezdp\_frame\_desc, 82  
 clear  
 ezdp\_posted\_ctr\_msg, 164  
 clr\_ctr  
 ezdp\_hier\_tb\_update, 92  
 color\_aware  
 ezdp\_tb\_ctr\_cfg, 175  
 color\_state\_g  
 ezdp\_hier\_tb\_ug\_app\_bits, 90  
 color\_state\_y  
 ezdp\_hier\_tb\_ug\_app\_bits, 90

commit\_profile\_id  
     ezdp\_tb\_ctr\_cfg, 176  
 cond\_set\_active\_state  
     ezdp\_hier\_tb\_update, 92  
 confirmation\_info  
     ezdp\_job\_rx\_info, 104  
 congestion  
     ezdp\_output\_queue\_status, 155  
 context\_id  
     ezdp\_security\_handle, 168  
 control  
     ezdp\_decode\_ipv4\_result, 33  
     ezdp\_decode\_ipv4\_retval, 35  
     ezdp\_decode\_ipv6\_result, 42  
     ezdp\_decode\_ipv6\_retval, 44  
     ezdp\_decode\_mac\_result, 54  
     ezdp\_decode\_mac\_retval, 56  
 control\_addr  
     ezdp\_ring\_cfg, 166  
 correction  
     ezdp\_1step\_1588\_header, 10  
 correction\_odd\_start  
     ezdp\_1step\_1588\_header, 9  
 correction\_offset  
     ezdp\_1step\_1588\_header, 10  
 counters  
     ezdp\_watchdog\_sliding\_window\_cfg, 189  
 coupling\_flag  
     ezdp\_tb\_ctr\_cfg, 175  
 crc\_checked\_flag  
     ezdp\_job\_rx\_interface\_info, 108  
 crc\_ok\_flag  
     ezdp\_job\_rx\_interface\_info, 108  
 creation\_date  
     ezdp\_version, 180  
 creation\_time  
     ezdp\_version, 180  
 ctr\_sum\_fail\_threshold  
     ezdp\_hier\_tb\_ctr\_cfg, 87  
 ctr\_sum\_updt\_threshold  
     ezdp\_hier\_tb\_ctr\_cfg, 87  
 ctr0  
     ezdp\_hier\_tb\_result, 89  
 ctr0\_fail\_threshold  
     ezdp\_hier\_tb\_ctr\_cfg, 86  
 ctr0\_updt\_threshold  
     ezdp\_hier\_tb\_ctr\_cfg, 87  
 ctr1  
     ezdp\_hier\_tb\_result, 88  
 ctr1\_fail\_threshold  
     ezdp\_hier\_tb\_ctr\_cfg, 87  
 ctr1\_updt\_threshold  
     ezdp\_hier\_tb\_ctr\_cfg, 87  
 ctrl  
     ezdp\_pci\_msg, 159  
 curr\_events  
     ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 da\_sa\_hash  
     ezdp\_decode\_mac\_result, 54  
 data  
     ezdp\_bitwise\_ctr\_cfg, 14  
     ezdp\_driver\_desc\_flags, 70  
     ezdp\_lookup\_int\_tcam\_4B\_data\_result, 145  
     ezdp\_lookup\_retval, 150  
     ezdp\_pci\_msg\_payload\_elbi, 162  
 data\_buf\_count  
     ezdp\_frame\_desc, 83  
 data\_lsb  
     ezdp\_pci\_msg\_payload\_atl, 161  
 data\_msb  
     ezdp\_pci\_msg\_payload\_atl, 161  
 data\_offset  
     ezdp\_decode\_tcp\_retval, 68  
 data\_offset\_lt\_5  
     ezdp\_decode\_tcp\_errors, 67  
 data0  
     ezdp\_lookup\_int\_tcam\_12B\_data\_result, 143  
     ezdp\_lookup\_int\_tcam\_16B\_data\_result, 144  
     ezdp\_lookup\_int\_tcam\_8B\_data\_result, 146  
 data1  
     ezdp\_lookup\_int\_tcam\_12B\_data\_result, 143  
     ezdp\_lookup\_int\_tcam\_16B\_data\_result, 144  
     ezdp\_lookup\_int\_tcam\_8B\_data\_result, 146  
 data2  
     ezdp\_lookup\_int\_tcam\_12B\_data\_result, 143  
     ezdp\_lookup\_int\_tcam\_16B\_data\_result, 144  
 data3  
     ezdp\_lookup\_int\_tcam\_16B\_data\_result, 144  
 decode\_error  
     ezdp\_decode\_ipv4\_errors, 31  
     ezdp\_decode\_ipv6\_errors, 40  
     ezdp\_decode\_mac\_errors, 48  
     ezdp\_decode\_mpls\_result, 62  
     ezdp\_decode\_mpls\_retval, 65  
     ezdp\_decode\_tcp\_errors, 67  
 def\_ip\_prot\_0  
     ezdp\_decode\_ip\_protocol\_retval, 27  
 def\_ip\_prot\_1  
     ezdp\_decode\_ip\_protocol\_retval, 27  
 def\_ip\_prot\_2  
     ezdp\_decode\_ip\_protocol\_retval, 27  
 def\_ip\_prot\_3  
     ezdp\_decode\_ip\_protocol\_retval, 27  
 dest\_queue  
     ezdp\_job\_tx\_info, 115  
 device\_error  
     ezdp\_lookup\_ext\_tcam\_retval, 142  
 device\_id  
     ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 132  
     ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 134  
     ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element, 136  
     ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_element, 138  
     ezdp\_lookup\_ext\_tcam\_index\_result\_element, 140  
 dip\_is\_multicast  
     ezdp\_decode\_ipv6\_control, 37  
 dip\_is\_one

ezdp\_decode\_ipv6\_errors, 40  
 dip\_is\_wellknown\_multicast  
 ezdp\_decode\_ipv6\_control, 37  
 dip\_is\_zero  
 ezdp\_decode\_ipv6\_errors, 40  
 discard\_info  
 ezdp\_job\_container\_cmd\_desc, 97  
 dispatched\_job  
 ezdp\_app\_schlr\_status, 13  
 ezdp\_group\_schlr\_status, 85  
 ezdp\_input\_queue\_status, 94  
 dmac\_is\_zero  
 ezdp\_decode\_mac\_errors, 48  
 dpe/dp/include/ezdp.h, 190  
 dpe/dp/include/ezdp\_atomic.h, 195  
 dpe/dp/include/ezdp\_counter.h, 228  
 dpe/dp/include/ezdp\_counter\_defs.h, 265  
 dpe/dp/include/ezdp\_decode.h, 295  
 dpe/dp/include/ezdp\_decode\_defs.h, 300  
 dpe/dp/include/ezdp\_defs.h, 345  
 dpe/dp/include/ezdp\_dma.h, 348  
 dpe/dp/include/ezdp\_frame.h, 359  
 dpe/dp/include/ezdp\_frame\_defs.h, 378  
 dpe/dp/include/ezdp\_job.h, 391  
 dpe/dp/include/ezdp\_job\_defs.h, 409  
 dpe/dp/include/ezdp\_lock.h, 435  
 dpe/dp/include/ezdp\_lock\_defs.h, 440  
 dpe/dp/include/ezdp\_math.h, 441  
 dpe/dp/include/ezdp\_memory.h, 459  
 dpe/dp/include/ezdp\_memory\_defs.h, 462  
 dpe/dp/include/ezdp\_pci.h, 472  
 dpe/dp/include/ezdp\_pci\_defs.h, 484  
 dpe/dp/include/ezdp\_pool.h, 493  
 dpe/dp/include/ezdp\_pool\_defs.h, 497  
 dpe/dp/include/ezdp\_processor.h, 498  
 dpe/dp/include/ezdp\_queue.h, 501  
 dpe/dp/include/ezdp\_queue\_defs.h, 505  
 dpe/dp/include/ezdp\_search.h, 506  
 dpe/dp/include/ezdp\_search\_defs.h, 518  
 dpe/dp/include/ezdp\_search\_prm.h, 554  
 dpe/dp/include/ezdp\_security.h, 563  
 dpe/dp/include/ezdp\_security\_defs.h, 577  
 dpe/dp/include/ezdp\_string.h, 584  
 dpe/dp/include/ezdp\_time.h, 586  
 dpe/dp/include/ezdp\_time\_defs.h, 588  
 dpe/dp/include/ezdp\_version.h, 589  
 dual\_ctr\_value  
 ezdp\_ctr\_msg, 20  
 ecc  
 ezdp\_frame\_desc, 81  
 ezdp\_linked\_buffers\_desc\_line, 121  
 eigth\_mode\_ret\_bits  
 ezdp\_hier\_tb\_ug\_app\_bits, 90  
 elbi\_payload  
 ezdp\_pci\_msg, 159  
 element\_index  
 ezdp\_sum\_addr, 172  
 emem\_buf\_guarantee  
 ezdp\_congestion\_status, 18  
 emem\_data\_size

ezdp\_mem\_section\_info, 153  
 empty\_commit\_bucket  
 ezdp\_tb\_ctr\_result, 178  
 empty\_commit\_bucket\_ug  
 ezdp\_tb\_ctr\_result, 177  
 empty\_excess\_bucket  
 ezdp\_tb\_ctr\_result, 177  
 empty\_excess\_bucket\_ug  
 ezdp\_tb\_ctr\_result, 177  
 enable  
 ezdp\_app\_schlr\_status, 13  
 ezdp\_flow\_control\_status, 80  
 enable\_exceed\_message  
 ezdp\_dual\_ctr\_cfg, 75  
 ezdp\_single\_ctr\_cfg, 169  
 end\_of\_stack  
 ezdp\_decode\_mpls\_label\_result, 58  
 ezdp\_decode\_mpls\_label\_retval, 60  
 endpoint  
 ezdp\_pci\_info, 158  
 error  
 ezdp\_driver\_desc\_flags, 70  
 error\_codes  
 ezdp\_decode\_ipv4\_result, 33  
 ezdp\_decode\_ipv4\_retval, 35  
 ezdp\_decode\_ipv6\_result, 41  
 ezdp\_decode\_ipv6\_retval, 43  
 ezdp\_decode\_mac\_result, 54  
 ezdp\_decode\_mac\_retval, 56  
 ezdp\_decode\_tcp\_retval, 68  
 esp\_prot  
 ezdp\_decode\_ip\_protocol\_retval, 27  
 eth\_8100  
 ezdp\_decode\_eth\_type\_retval, 22  
 eth\_88a8  
 ezdp\_decode\_eth\_type\_retval, 22  
 event  
 ezdp\_dual\_ctr, 73  
 event\_id  
 ezdp\_job\_rx\_timer\_info, 111  
 event\_report\_exceeded  
 ezdp\_dual\_ctr\_cfg, 75  
 event\_value  
 ezdp\_dual\_ctr\_result, 76  
 exception\_bit  
 ezdp\_decode\_mpls\_label\_result, 57  
 ezdp\_decode\_mpls\_label\_retval, 59  
 excess\_profile\_id  
 ezdp\_tb\_ctr\_cfg, 175  
 explicit\_packet\_switch\_id  
 ezdp\_job\_tx\_info, 115  
 ezdp.h  
 EZDP\_CMEM\_DATA, 192  
 ezdp\_data\_mem\_space, 192  
 EZDP\_EMEM\_DATA, 193  
 ezdp\_get\_err\_msg, 194  
 ezdp\_get\_mem\_section\_info, 194  
 ezdp\_get\_version, 194  
 EZDP\_IMEM\_1\_CLUSTER\_DATA, 192  
 EZDP\_IMEM\_16\_CLUSTER\_DATA, 192



EZDP\_IMEM\_2\_CLUSTER\_DATA, 192  
 EZDP\_IMEM\_4\_CLUSTER\_DATA, 192  
 EZDP\_IMEM\_ALL\_CLUSTER\_DATA, 193  
 EZDP\_IMEM\_HALF\_CLUSTER\_DATA, 192  
 EZDP\_IMEM\_PRIVATE\_DATA, 192  
 ezdp\_init\_global, 193  
 ezdp\_init\_local, 193  
 EZDP\_MAIN\_FUNC, 192  
 EZDP\_MEM\_CFG\_EMEM\_DATA\_CACHABLE, 192  
 EZDP\_MEM\_CFG\_IMEM\_1\_CLUSTER\_DATA\_CACHABLE, 191  
 EZDP\_MEM\_CFG\_IMEM\_16\_CLUSTER\_DATA\_CACHABLE, 191  
 EZDP\_MEM\_CFG\_IMEM\_2\_CLUSTER\_DATA\_CACHABLE, 191  
 EZDP\_MEM\_CFG\_IMEM\_4\_CLUSTER\_DATA\_CACHABLE, 191  
 EZDP\_MEM\_CFG\_IMEM\_ALL\_CLUSTER\_DATA\_CACHABLE, 192  
 EZDP\_MEM\_CFG\_IMEM\_HALF\_CLUSTER\_DATA\_CACHABLE, 191  
 EZDP\_MEM\_CFG\_IMEM\_PRIVATE\_DATA\_CACHABLE, 191  
 EZDP\_MEM\_CFG\_USE\_ALTER\_CMEM, 191  
 EZDP\_MEM\_CFG\_USE\_ALTER\_SHARED\_CMEM, 191  
 EZDP\_MEM\_CTOR\_FUNC, 192  
 ezdp\_mem\_section\_info\_str, 194  
 ezdp\_run, 194  
 EZDP\_SHARED\_CMEM\_DATA, 192  
 ezdp\_sync\_cp, 193  
 EZDP\_1\_BITS  
     ezdp\_counter\_defs.h, 294  
 EZDP\_1\_CLUSTER\_CODE  
     ezdp\_memory\_defs.h, 469  
 EZDP\_1\_CLUSTER\_DATA  
     ezdp\_memory\_defs.h, 469  
 ezdp\_1588\_header, 8  
     \_\_pad0\_\_, 8  
     \_\_pad1\_\_, 8  
     one\_step, 8  
     two\_step, 8  
     u, 8  
 ezdp\_1588\_type  
     ezdp\_frame\_defs.h, 389  
 EZDP\_16\_BITS  
     ezdp\_counter\_defs.h, 294  
 EZDP\_16\_CLUSTER\_CODE  
     ezdp\_memory\_defs.h, 470  
 EZDP\_16\_CLUSTER\_DATA  
     ezdp\_memory\_defs.h, 470  
 EZDP\_16BITS\_REPORT  
     ezdp\_job\_defs.h, 434  
 EZDP\_1STEP  
     ezdp\_frame\_defs.h, 389  
 ezdp\_1step\_1588\_header, 9  
     \_\_pad0\_\_, 9  
     \_\_pad1\_\_, 10  
     \_\_pad2\_\_, 10  
     checksum, 10  
     checksum\_offset, 10  
     correction, 10  
     correction\_odd\_start, 9  
     correction\_offset, 10  
     inject\_checksum\_flag, 10  
     raw\_data, 9  
     wrap\_around\_condition, 9  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFFSET  
     ezdp\_frame\_defs.h, 386  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFFSET\_OFFSET  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFFSET\_SIZE  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFFSET\_WORD\_OFFSET  
     ezdp\_frame\_defs.h, 388  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFFSET\_WORD\_SELECT  
     ezdp\_frame\_defs.h, 388  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_SIZE  
     ezdp\_frame\_defs.h, 386  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_WORD\_OFFSET  
     ezdp\_frame\_defs.h, 386  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_WORD\_SELECT  
     ezdp\_frame\_defs.h, 386  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_ODD\_START\_MASK  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_ODD\_START\_OFFSET  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_ODD\_START\_SIZE  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_ODD\_START\_WORD\_OFFSET  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_ODD\_START\_WORD\_SELECT  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_OFFSET  
     ezdp\_frame\_defs.h, 388  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_OFFSET\_OFFSET  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_OFFSET\_SIZE  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_OFFSET\_WORD\_OFFSET  
     ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_OFFSET\_WORD\_SELECT

ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_SIZE  
 ezdp\_frame\_defs.h, 388  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 388  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 388  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_MASK  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_OFFSET  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_SIZE  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED16\_23\_OFFSET  
 ezdp\_frame\_defs.h, 386  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED16\_23\_SIZE  
 ezdp\_frame\_defs.h, 386  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED24\_OFFSET  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED24\_SIZE  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED28\_31\_OFFSET  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED28\_31\_SIZE  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_WORD\_COUNT  
 ezdp\_frame\_defs.h, 388  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_MASK  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_OFFSET  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_SIZE  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 387  
 EZDP\_2\_BITS  
 ezdp\_counter\_defs.h, 294  
 EZDP\_2\_CLUSTER\_CODE  
 ezdp\_memory\_defs.h, 469  
 EZDP\_2\_CLUSTER\_DATA  
 ezdp\_memory\_defs.h, 469  
 EZDP\_2STEP  
 ezdp\_frame\_defs.h, 389  
 ezdp\_2step\_1588\_header, 11  
 \_\_pad0\_\_, 11  
 \_\_pad1\_\_, 11  
 \_\_pad2\_\_, 11  
 \_\_pad3\_\_, 11  
 \_\_pad4\_\_, 12  
 \_\_pad5\_\_, 12  
 buf\_budget\_id, 12  
 buf\_desc, 12  
 class\_of\_service, 12  
 free\_bytes, 12  
 header\_offset, 12  
 raw\_data, 11  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_OFFSET  
 ezdp\_frame\_defs.h, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_SIZE  
 ezdp\_frame\_defs.h, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_OFFSET  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_SIZE  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_OFFSET  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_SIZE  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_FREE\_BYTES\_OFFSET  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_FREE\_BYTES\_SIZE  
 ezdp\_frame\_defs.h, 386  
 EZDP\_2STEP\_1588\_HEADER\_FREE\_BYTES\_WORD\_OFFSET

ezdp_frame_defs.h, 386	ezdp_security_defs.h, 579
EZDP_2STEP_1588_HEADER_FREE_BYTES_WORD_SELECT	EZDP_3DES2_CFB_ALG
ezdp_frame_defs.h, 386	ezdp_security_defs.h, 579
EZDP_2STEP_1588_HEADER_HEADER_OFFSET_OFFSET	EZDP_3DES2_CTR_ALG
ezdp_frame_defs.h, 386	ezdp_security_defs.h, 580
EZDP_2STEP_1588_HEADER_HEADER_OFFSET_SIZE	EZDP_3DES2_ECB_ALG
ezdp_frame_defs.h, 386	ezdp_security_defs.h, 580
EZDP_2STEP_1588_HEADER_HEADER_OFFSET_WORD_OFFSET	EZDP_3DES2_OFB_ALG
ezdp_frame_defs.h, 386	ezdp_security_defs.h, 580
EZDP_2STEP_1588_HEADER_HEADER_OFFSET_WORD_SELECT	EZDP_3DES2_XXX_KEY_SIZE
ezdp_frame_defs.h, 386	ezdp_security_defs.h, 581
EZDP_2STEP_1588_HEADER_RESERVED0_23_OFFSET	EZDP_3DES3_CBC_ALG
ezdp_frame_defs.h, 385	ezdp_security_defs.h, 579
EZDP_2STEP_1588_HEADER_RESERVED0_23_SIZE	EZDP_3DES3_CFB_ALG
ezdp_frame_defs.h, 385	ezdp_security_defs.h, 579
EZDP_2STEP_1588_HEADER_RESERVED24_31_OFFSET	EZDP_3DES3_CTR_ALG
ezdp_frame_defs.h, 385	ezdp_security_defs.h, 580
EZDP_2STEP_1588_HEADER_RESERVED24_31_SIZE	EZDP_3DES3_ECB_ALG
ezdp_frame_defs.h, 385	ezdp_security_defs.h, 580
EZDP_2STEP_1588_HEADER_RESERVED24_OFFSET	EZDP_3DES3_OFB_ALG
ezdp_frame_defs.h, 385	ezdp_security_defs.h, 580
EZDP_2STEP_1588_HEADER_RESERVED24_SIZE	EZDP_3DES3_XXX_KEY_SIZE
ezdp_frame_defs.h, 385	ezdp_security_defs.h, 581
EZDP_2STEP_1588_HEADER_RESERVED32_63_OFFSET	EZDP_4_BITS
ezdp_frame_defs.h, 385	ezdp_counter_defs.h, 294
EZDP_2STEP_1588_HEADER_RESERVED32_63_SIZE	EZDP_4_CLUSTER_CODE
ezdp_frame_defs.h, 385	ezdp_memory_defs.h, 469
EZDP_2STEP_1588_HEADER_RESERVED74_75_OFFSET	EZDP_4_CLUSTER_DATA
ezdp_frame_defs.h, 386	ezdp_memory_defs.h, 469
EZDP_2STEP_1588_HEADER_RESERVED74_75_SIZE	EZDP_8_BITS
ezdp_frame_defs.h, 385	ezdp_counter_defs.h, 294
EZDP_2STEP_1588_HEADER_RESERVED76_77_OFFSET	EZDP_8BITS_REPORT
ezdp_frame_defs.h, 385	ezdp_job_defs.h, 434
EZDP_2STEP_1588_HEADER_RESERVED76_77_SIZE	EZDP_ACCEPT_ENTRY
ezdp_frame_defs.h, 386	ezdp_search_defs.h, 553
EZDP_2STEP_1588_HEADER_WORD_COUNT	ezdp_add
ezdp_frame_defs.h, 386	ezdp_math.h, 444
EZDP_32BITS_REPORT	ezdp_add_checksum
ezdp_job_defs.h, 434	ezdp_math.h, 457
EZDP_3DES_BLOCK_SIZE	ezdp_add_hash_entry
ezdp_security_defs.h, 583	ezdp_search.h, 511
EZDP_3DES_IV_SIZE	ezdp_add_posted_ctr
ezdp_security_defs.h, 582	ezdp_counter.h, 261
EZDP_3DES2_CBC_ALG	ezdp_add_posted_ctr_async
	ezdp_counter.h, 261
	ezdp_add_table_entry
	ezdp_search.h, 508
	EZDP_AES_BLOCK_SIZE
	ezdp_security_defs.h, 583
	EZDP_AES_CBC_128_ALG
	ezdp_security_defs.h, 580
	EZDP_AES_CBC_192_ALG
	ezdp_security_defs.h, 580
	EZDP_AES_CBC_256_ALG
	ezdp_security_defs.h, 580
	EZDP_AES_CCM_128_ALG
	ezdp_security_defs.h, 580
	EZDP_AES_CCM_192_ALG
	ezdp_security_defs.h, 580
	EZDP_AES_CCM_256_ALG

ezdp_security_defs.h, 580	ezdp_security_defs.h, 581
EZDP_AES_CCM_MAC_SIZE	EZDP_AES_XXX_192_KEY_SIZE
ezdp_security_defs.h, 582	ezdp_security_defs.h, 581
EZDP_AES_CCM_STATE_SIZE	EZDP_AES_XXX_256_KEY_SIZE
ezdp_security_defs.h, 582	ezdp_security_defs.h, 581
EZDP_AES_CFB_128_ALG	EZDP_AES_XXX_MAC_128_KEY_SIZE
ezdp_security_defs.h, 580	ezdp_security_defs.h, 581
EZDP_AES_CFB_192_ALG	EZDP_AES_XXX_MAC_192_KEY_SIZE
ezdp_security_defs.h, 580	ezdp_security_defs.h, 581
EZDP_AES_CFB_256_ALG	EZDP_AES_XXX_MAC_256_KEY_SIZE
ezdp_security_defs.h, 580	ezdp_security_defs.h, 581
EZDP_AES_CMAC_128_ALG	EZDP_ALG_TCAM_MAX_KEY_SIZE
ezdp_security_defs.h, 581	ezdp_search_defs.h, 530
EZDP_AES_CMAC_192_ALG	ezdp_alg_tcam_struct_desc_t
ezdp_security_defs.h, 581	ezdp_search_defs.h, 552
EZDP_AES_CMAC_256_ALG	EZDP_ALG_TCAM_WORK_AREA_SIZE
ezdp_security_defs.h, 581	ezdp_search_defs.h, 532
EZDP_AES_CMAC_XXX_MAC_SIZE	EZDP_ALL_CLUSTER_CODE
ezdp_security_defs.h, 583	ezdp_memory_defs.h, 470
EZDP_AES_CMAC_XXX_STATE_SIZE	EZDP_ALL_CLUSTER_DATA
ezdp_security_defs.h, 582	ezdp_memory_defs.h, 470
EZDP_AES_CTR_128_ALG	EZDP_ALL_CLUSTER_DATA_EXT_MEM
ezdp_security_defs.h, 580	ezdp_memory_defs.h, 470
EZDP_AES_CTR_192_ALG	EZDP_ALL_CLUSTER_IO
ezdp_security_defs.h, 580	ezdp_memory_defs.h, 470
EZDP_AES_CTR_256_ALG	ezdp_alloc_buf
ezdp_security_defs.h, 580	ezdp_frame.h, 361
EZDP_AES_ECB_128_ALG	ezdp_alloc_index
ezdp_security_defs.h, 580	ezdp_pool.h, 493
EZDP_AES_ECB_192_ALG	ezdp_alloc_job_id
ezdp_security_defs.h, 580	ezdp_job.h, 393
EZDP_AES_ECB_256_ALG	ezdp_alloc_job_id_async
ezdp_security_defs.h, 580	ezdp_job.h, 394
EZDP_AES_GCM_128_ALG	ezdp_alloc_mc_buf
ezdp_security_defs.h, 580	ezdp_frame.h, 372
EZDP_AES_GCM_192_ALG	ezdp_alloc_multi_buf
ezdp_security_defs.h, 580	ezdp_frame.h, 362
EZDP_AES_GCM_256_ALG	ezdp_alloc_multi_buf_async
ezdp_security_defs.h, 580	ezdp_frame.h, 362
EZDP_AES_GCM_MAC_SIZE	ezdp_alloc_multi_index
ezdp_security_defs.h, 583	ezdp_pool.h, 494
EZDP_AES_GCM_STATE_SIZE	ezdp_alloc_multi_index_async
ezdp_security_defs.h, 582	ezdp_pool.h, 494
EZDP_AES_IV_SIZE	ezdp_alloc_multi_job_id
ezdp_security_defs.h, 582	ezdp_job.h, 394
EZDP_AES_OFB_128_ALG	ezdp_alloc_multi_job_id_async
ezdp_security_defs.h, 580	ezdp_job.h, 394
EZDP_AES_OFB_192_ALG	ezdp_alloc_obj
ezdp_security_defs.h, 580	ezdp_pool.h, 495
EZDP_AES_OFB_256_ALG	ezdp_alloc_qlock_slot
ezdp_security_defs.h, 580	ezdp_lock.h, 437
EZDP_AES_STATE_SIZE	EZDP_ALLOW_REORDER
ezdp_security_defs.h, 582	ezdp_job_defs.h, 433
EZDP_AES_XCBC_MAC_128_ALG	ezdp_and
ezdp_security_defs.h, 581	ezdp_math.h, 444
EZDP_AES_XCBC_MAC_128_MAC_SIZE	ezdp_app_schlr_status, 13
ezdp_security_defs.h, 583	__pad0__, 13
EZDP_AES_XCBC_MAC_STATE_SIZE	busy, 13
ezdp_security_defs.h, 582	dispatched_job, 13
EZDP_AES_XXX_128_KEY_SIZE	enable, 13

raw\_data, 13  
 EZDP\_APP\_SCHLR\_STATUS\_BUSY\_MASK  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_BUSY\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_BUSY\_SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_DISPATCHED\_JOB\_  
   OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_DISPATCHED\_JOB\_  
   SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_ENABLE\_MASK  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_ENABLE\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_ENABLE\_SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_RESERVED13\_OFFS  
   ET  
   ezdp\_job\_defs.h, 429  
 EZDP\_APP\_SCHLR\_STATUS\_RESERVED13\_SIZE  
   ezdp\_job\_defs.h, 429  
 ezdp\_app\_schlr\_status\_t  
   ezdp\_job\_defs.h, 430  
 ezdp\_append\_buf  
   ezdp\_frame.h, 377  
 ezdp\_atomic.h  
   ezdp\_atomic\_add16\_ext\_addr, 207  
   ezdp\_atomic\_add16\_ext\_addr\_async, 208  
   ezdp\_atomic\_add32\_ext\_addr, 208  
   ezdp\_atomic\_add32\_ext\_addr\_async, 208  
   ezdp\_atomic\_add32\_sum\_addr, 209  
   ezdp\_atomic\_add32\_sum\_addr\_async, 209  
   ezdp\_atomic\_add64\_sum\_addr, 210  
   ezdp\_atomic\_add64\_sum\_addr\_async, 210  
   ezdp\_atomic\_add8\_ext\_addr, 207  
   ezdp\_atomic\_add8\_ext\_addr\_async, 207  
   ezdp\_atomic\_and16\_ext\_addr, 217  
   ezdp\_atomic\_and16\_ext\_addr\_async, 217  
   ezdp\_atomic\_and32\_ext\_addr, 218  
   ezdp\_atomic\_and32\_ext\_addr\_async, 218  
   ezdp\_atomic\_and32\_sum\_addr, 219  
   ezdp\_atomic\_and32\_sum\_addr\_async, 219  
   ezdp\_atomic\_and8\_ext\_addr, 216  
   ezdp\_atomic\_and8\_ext\_addr\_async, 217  
   ezdp\_atomic\_cmpxchg16\_ext\_addr, 204  
   ezdp\_atomic\_cmpxchg32\_ext\_addr, 204  
   ezdp\_atomic\_cmpxchg32\_sum\_addr, 204  
   ezdp\_atomic\_cmpxchg8\_ext\_addr, 203  
   ezdp\_atomic\_dual\_add32\_ext\_addr, 211  
   ezdp\_atomic\_dual\_add32\_ext\_addr\_async, 211  
   ezdp\_atomic\_dual\_add32\_sum\_addr, 211  
   ezdp\_atomic\_dual\_add32\_sum\_addr\_async, 212  
   ezdp\_atomic\_dual\_add64\_sum\_addr, 212  
   ezdp\_atomic\_dual\_add64\_sum\_addr\_async, 212  
   ezdp\_atomic\_or16\_ext\_addr, 221  
   ezdp\_atomic\_or16\_ext\_addr\_async, 221  
   ezdp\_atomic\_or16\_sum\_addr, 222

ezdp\_atomic\_or16\_sum\_addr\_async, 222  
 ezdp\_atomic\_or32\_ext\_addr, 222  
 ezdp\_atomic\_or32\_ext\_addr\_async, 223  
 ezdp\_atomic\_or32\_sum\_addr, 223  
 ezdp\_atomic\_or32\_sum\_addr\_async, 224  
 ezdp\_atomic\_or8\_ext\_addr, 220  
 ezdp\_atomic\_or8\_ext\_addr\_async, 220  
 ezdp\_atomic\_or8\_sum\_addr, 221  
 ezdp\_atomic\_or8\_sum\_addr\_async, 221  
 ezdp\_atomic\_read\_and\_add16\_ext\_addr, 208  
 ezdp\_atomic\_read\_and\_add32\_ext\_addr, 209  
 ezdp\_atomic\_read\_and\_add32\_sum\_addr, 210  
 ezdp\_atomic\_read\_and\_add64\_sum\_addr, 210  
 ezdp\_atomic\_read\_and\_add8\_ext\_addr, 207  
 ezdp\_atomic\_read\_and\_and16\_ext\_addr, 218  
 ezdp\_atomic\_read\_and\_and32\_ext\_addr, 219  
 ezdp\_atomic\_read\_and\_and32\_sum\_addr, 219  
 ezdp\_atomic\_read\_and\_and8\_ext\_addr, 217  
 ezdp\_atomic\_read\_and\_clear16\_ext\_addr, 206  
 ezdp\_atomic\_read\_and\_clear32\_ext\_addr, 206  
 ezdp\_atomic\_read\_and\_clear32\_sum\_addr, 206  
 ezdp\_atomic\_read\_and\_clear64\_sum\_addr, 206  
 ezdp\_atomic\_read\_and\_clear8\_ext\_addr, 205  
 ezdp\_atomic\_read\_and\_dec16\_ext\_addr, 215  
 ezdp\_atomic\_read\_and\_dec32\_ext\_addr, 215  
 ezdp\_atomic\_read\_and\_dec32\_sum\_addr, 215  
 ezdp\_atomic\_read\_and\_dec64\_sum\_addr, 216  
 ezdp\_atomic\_read\_and\_dec8\_ext\_addr, 214  
 ezdp\_atomic\_read\_and\_dual\_add32\_sum\_addr, 212  
 ezdp\_atomic\_read\_and\_dual\_add64\_sum\_addr, 213  
 ezdp\_atomic\_read\_and\_inc16\_ext\_addr, 213  
 ezdp\_atomic\_read\_and\_inc32\_cond\_ext\_addr, 216  
 ezdp\_atomic\_read\_and\_inc32\_cond\_sum\_addr, 216  
 ezdp\_atomic\_read\_and\_inc32\_ext\_addr, 214  
 ezdp\_atomic\_read\_and\_inc32\_sum\_addr, 214  
 ezdp\_atomic\_read\_and\_inc64\_sum\_addr, 214  
 ezdp\_atomic\_read\_and\_inc8\_ext\_addr, 213  
 ezdp\_atomic\_read\_and\_or16\_ext\_addr, 222  
 ezdp\_atomic\_read\_and\_or32\_ext\_addr, 223  
 ezdp\_atomic\_read\_and\_or32\_sum\_addr, 224  
 ezdp\_atomic\_read\_and\_or8\_ext\_addr, 220  
 ezdp\_atomic\_read\_and\_tst16\_ext\_addr, 205  
 ezdp\_atomic\_read\_and\_tst32\_ext\_addr, 205  
 ezdp\_atomic\_read\_and\_tst32\_sum\_addr, 205  
 ezdp\_atomic\_read\_and\_tst8\_ext\_addr, 204  
 ezdp\_atomic\_read\_and\_xor16\_ext\_addr, 226  
 ezdp\_atomic\_read\_and\_xor32\_ext\_addr, 226  
 ezdp\_atomic\_read\_and\_xor32\_sum\_addr, 227  
 ezdp\_atomic\_read\_and\_xor8\_ext\_addr, 225  
 ezdp\_atomic\_read16\_ext\_addr, 199  
 ezdp\_atomic\_read32\_ext\_addr, 199  
 ezdp\_atomic\_read32\_sum\_addr, 199  
 ezdp\_atomic\_read32\_sum\_addr\_async, 200  
 ezdp\_atomic\_read64\_sum\_addr, 200  
 ezdp\_atomic\_read64\_sum\_addr\_async, 200  
 ezdp\_atomic\_read8\_ext\_addr, 199  
 ezdp\_atomic\_write16\_ext\_addr, 201  
 ezdp\_atomic\_write16\_ext\_addr\_async, 201  
 ezdp\_atomic\_write32\_ext\_addr, 201  
 ezdp\_atomic\_write32\_ext\_addr\_async, 202

ezdp\_atomic\_write32\_sum\_addr, 202  
 ezdp\_atomic\_write32\_sum\_addr\_async, 202  
 ezdp\_atomic\_write64\_sum\_addr, 202  
 ezdp\_atomic\_write64\_sum\_addr\_async, 203  
 ezdp\_atomic\_write8\_ext\_addr, 200  
 ezdp\_atomic\_write8\_ext\_addr\_async, 201  
 ezdp\_atomic\_xchg32\_ext\_addr, 203  
 ezdp\_atomic\_xchg32\_sum\_addr, 203  
 ezdp\_atomic\_xor16\_ext\_addr, 225  
 ezdp\_atomic\_xor16\_ext\_addr\_async, 225  
 ezdp\_atomic\_xor32\_ext\_addr, 226  
 ezdp\_atomic\_xor32\_ext\_addr\_async, 226  
 ezdp\_atomic\_xor32\_sum\_addr, 227  
 ezdp\_atomic\_xor32\_sum\_addr\_async, 227  
 ezdp\_atomic\_xor8\_ext\_addr, 224  
 ezdp\_atomic\_xor8\_ext\_addr\_async, 224  
 ezdp\_atomic\_add16\_ext\_addr  
   ezdp\_atomic.h, 207  
 ezdp\_atomic\_add16\_ext\_addr\_async  
   ezdp\_atomic.h, 208  
 ezdp\_atomic\_add32\_ext\_addr  
   ezdp\_atomic.h, 208  
 ezdp\_atomic\_add32\_ext\_addr\_async  
   ezdp\_atomic.h, 208  
 ezdp\_atomic\_add32\_sum\_addr  
   ezdp\_atomic.h, 209  
 ezdp\_atomic\_add32\_sum\_addr\_async  
   ezdp\_atomic.h, 209  
 ezdp\_atomic\_add64\_sum\_addr  
   ezdp\_atomic.h, 210  
 ezdp\_atomic\_add64\_sum\_addr\_async  
   ezdp\_atomic.h, 210  
 ezdp\_atomic\_add8\_ext\_addr  
   ezdp\_atomic.h, 207  
 ezdp\_atomic\_add8\_ext\_addr\_async  
   ezdp\_atomic.h, 207  
 ezdp\_atomic\_and16\_ext\_addr  
   ezdp\_atomic.h, 217  
 ezdp\_atomic\_and16\_ext\_addr\_async  
   ezdp\_atomic.h, 217  
 ezdp\_atomic\_and32\_ext\_addr  
   ezdp\_atomic.h, 218  
 ezdp\_atomic\_and32\_ext\_addr\_async  
   ezdp\_atomic.h, 218  
 ezdp\_atomic\_and32\_sum\_addr  
   ezdp\_atomic.h, 219  
 ezdp\_atomic\_and32\_sum\_addr\_async  
   ezdp\_atomic.h, 219  
 ezdp\_atomic\_and8\_ext\_addr  
   ezdp\_atomic.h, 216  
 ezdp\_atomic\_and8\_ext\_addr\_async  
   ezdp\_atomic.h, 217  
 ezdp\_atomic\_cmpxchg16\_ext\_addr  
   ezdp\_atomic.h, 204  
 ezdp\_atomic\_cmpxchg32\_ext\_addr  
   ezdp\_atomic.h, 204  
 ezdp\_atomic\_cmpxchg32\_sum\_addr  
   ezdp\_atomic.h, 204  
 ezdp\_atomic\_cmpxchg8\_ext\_addr  
   ezdp\_atomic.h, 203

ezdp\_atomic\_dual\_add32\_ext\_addr  
   ezdp\_atomic.h, 211  
 ezdp\_atomic\_dual\_add32\_ext\_addr\_async  
   ezdp\_atomic.h, 211  
 ezdp\_atomic\_dual\_add32\_sum\_addr  
   ezdp\_atomic.h, 211  
 ezdp\_atomic\_dual\_add32\_sum\_addr\_async  
   ezdp\_atomic.h, 212  
 ezdp\_atomic\_dual\_add64\_sum\_addr  
   ezdp\_atomic.h, 212  
 ezdp\_atomic\_dual\_add64\_sum\_addr\_async  
   ezdp\_atomic.h, 212  
 ezdp\_atomic\_or16\_ext\_addr  
   ezdp\_atomic.h, 221  
 ezdp\_atomic\_or16\_ext\_addr\_async  
   ezdp\_atomic.h, 221  
 ezdp\_atomic\_or16\_sum\_addr  
   ezdp\_atomic.h, 222  
 ezdp\_atomic\_or16\_sum\_addr\_async  
   ezdp\_atomic.h, 222  
 ezdp\_atomic\_or32\_ext\_addr  
   ezdp\_atomic.h, 222  
 ezdp\_atomic\_or32\_ext\_addr\_async  
   ezdp\_atomic.h, 223  
 ezdp\_atomic\_or32\_sum\_addr  
   ezdp\_atomic.h, 223  
 ezdp\_atomic\_or32\_sum\_addr\_async  
   ezdp\_atomic.h, 224  
 ezdp\_atomic\_or8\_ext\_addr  
   ezdp\_atomic.h, 220  
 ezdp\_atomic\_or8\_ext\_addr\_async  
   ezdp\_atomic.h, 220  
 ezdp\_atomic\_or8\_sum\_addr  
   ezdp\_atomic.h, 221  
 ezdp\_atomic\_or8\_sum\_addr\_async  
   ezdp\_atomic.h, 221  
 ezdp\_atomic\_read\_and\_add16\_ext\_addr  
   ezdp\_atomic.h, 208  
 ezdp\_atomic\_read\_and\_add32\_ext\_addr  
   ezdp\_atomic.h, 209  
 ezdp\_atomic\_read\_and\_add32\_sum\_addr  
   ezdp\_atomic.h, 210  
 ezdp\_atomic\_read\_and\_add64\_sum\_addr  
   ezdp\_atomic.h, 210  
 ezdp\_atomic\_read\_and\_add8\_ext\_addr  
   ezdp\_atomic.h, 207  
 ezdp\_atomic\_read\_and\_and16\_ext\_addr  
   ezdp\_atomic.h, 218  
 ezdp\_atomic\_read\_and\_and32\_ext\_addr  
   ezdp\_atomic.h, 219  
 ezdp\_atomic\_read\_and\_and32\_sum\_addr  
   ezdp\_atomic.h, 219  
 ezdp\_atomic\_read\_and\_and8\_ext\_addr  
   ezdp\_atomic.h, 217  
 ezdp\_atomic\_read\_and\_clear16\_ext\_addr  
   ezdp\_atomic.h, 206  
 ezdp\_atomic\_read\_and\_clear32\_ext\_addr  
   ezdp\_atomic.h, 206  
 ezdp\_atomic\_read\_and\_clear32\_sum\_addr  
   ezdp\_atomic.h, 206

ezdp_atomic_read_and_clear64_sum_addr	ezdp_atomic.h, 206	ezdp_atomic_read16_ext_addr	ezdp_atomic.h, 199
ezdp_atomic_read_and_clear8_ext_addr	ezdp_atomic.h, 205	ezdp_atomic_read32_ext_addr	ezdp_atomic.h, 199
ezdp_atomic_read_and_dec_mc_buf_counter	ezdp_frame.h, 374	ezdp_atomic_read32_sum_addr	ezdp_atomic.h, 199
ezdp_atomic_read_and_dec16_ext_addr	ezdp_atomic.h, 215	ezdp_atomic_read32_sum_addr_async	ezdp_atomic.h, 200
ezdp_atomic_read_and_dec32_ext_addr	ezdp_atomic.h, 215	ezdp_atomic_read64_sum_addr	ezdp_atomic.h, 200
ezdp_atomic_read_and_dec32_sum_addr	ezdp_atomic.h, 215	ezdp_atomic_read64_sum_addr_async	ezdp_atomic.h, 200
ezdp_atomic_read_and_dec64_sum_addr	ezdp_atomic.h, 216	ezdp_atomic_read8_ext_addr	ezdp_atomic.h, 199
ezdp_atomic_read_and_dec8_ext_addr	ezdp_atomic.h, 214	ezdp_atomic_write16_ext_addr	ezdp_atomic.h, 201
ezdp_atomic_read_and_dual_add32_sum_addr	ezdp_atomic.h, 212	ezdp_atomic_write16_ext_addr_async	ezdp_atomic.h, 201
ezdp_atomic_read_and_dual_add64_sum_addr	ezdp_atomic.h, 213	ezdp_atomic_write32_ext_addr	ezdp_atomic.h, 201
ezdp_atomic_read_and_inc_mc_buf_counter	ezdp_frame.h, 373	ezdp_atomic_write32_ext_addr_async	ezdp_atomic.h, 202
ezdp_atomic_read_and_inc16_ext_addr	ezdp_atomic.h, 213	ezdp_atomic_write32_sum_addr	ezdp_atomic.h, 202
ezdp_atomic_read_and_inc32_cond_ext_addr	ezdp_atomic.h, 216	ezdp_atomic_write32_sum_addr_async	ezdp_atomic.h, 202
ezdp_atomic_read_and_inc32_cond_sum_addr	ezdp_atomic.h, 216	ezdp_atomic_write64_sum_addr	ezdp_atomic.h, 202
ezdp_atomic_read_and_inc32_ext_addr	ezdp_atomic.h, 214	ezdp_atomic_write64_sum_addr_async	ezdp_atomic.h, 203
ezdp_atomic_read_and_inc32_sum_addr	ezdp_atomic.h, 214	ezdp_atomic_write8_ext_addr	ezdp_atomic.h, 200
ezdp_atomic_read_and_inc64_sum_addr	ezdp_atomic.h, 214	ezdp_atomic_write8_ext_addr_async	ezdp_atomic.h, 201
ezdp_atomic_read_and_inc8_ext_addr	ezdp_atomic.h, 213	ezdp_atomic_xchg32_ext_addr	ezdp_atomic.h, 203
ezdp_atomic_read_and_or16_ext_addr	ezdp_atomic.h, 222	ezdp_atomic_xchg32_sum_addr	ezdp_atomic.h, 203
ezdp_atomic_read_and_or32_ext_addr	ezdp_atomic.h, 223	ezdp_atomic_xor16_ext_addr	ezdp_atomic.h, 225
ezdp_atomic_read_and_or32_sum_addr	ezdp_atomic.h, 224	ezdp_atomic_xor16_ext_addr_async	ezdp_atomic.h, 225
ezdp_atomic_read_and_or8_ext_addr	ezdp_atomic.h, 220	ezdp_atomic_xor32_ext_addr	ezdp_atomic.h, 226
ezdp_atomic_read_and_tst16_ext_addr	ezdp_atomic.h, 205	ezdp_atomic_xor32_ext_addr_async	ezdp_atomic.h, 226
ezdp_atomic_read_and_tst32_ext_addr	ezdp_atomic.h, 205	ezdp_atomic_xor32_sum_addr	ezdp_atomic.h, 227
ezdp_atomic_read_and_tst32_sum_addr	ezdp_atomic.h, 205	ezdp_atomic_xor32_sum_addr_async	ezdp_atomic.h, 227
ezdp_atomic_read_and_tst8_ext_addr	ezdp_atomic.h, 204	ezdp_atomic_xor8_ext_addr	ezdp_atomic.h, 224
ezdp_atomic_read_and_xor16_ext_addr	ezdp_atomic.h, 226	ezdp_atomic_xor8_ext_addr_async	ezdp_atomic.h, 224
ezdp_atomic_read_and_xor32_ext_addr	ezdp_atomic.h, 226	ezdp_bit_mode	ezdp_math.h, 443
ezdp_atomic_read_and_xor32_sum_addr	ezdp_atomic.h, 227	EZDP_BIT_MODE_FALSE	ezdp_math.h, 443
ezdp_atomic_read_and_xor8_ext_addr	ezdp_atomic.h, 225	EZDP_BIT_MODE_INVERSE	ezdp_math.h, 443

EZDP_BIT_MODE_TRUE	ezdp_frame_defs.h, 382
ezdp_math.h, 443	EZDP_BUFFER_DESC_ID_SIZE
EZDP_BIT_MODE_VALUE	ezdp_frame_defs.h, 382
ezdp_math.h, 443	EZDP_BUFFER_DESC_MEM_TYPE_MASK
ezdp_bitwise_ctr_cfg, 14	ezdp_frame_defs.h, 382
__pad0__, 14	EZDP_BUFFER_DESC_MEM_TYPE_OFFSET
__pad1__, 14	ezdp_frame_defs.h, 382
__pad2__, 14	EZDP_BUFFER_DESC_MEM_TYPE_SIZE
__pad3__, 14	ezdp_frame_defs.h, 382
data, 14	EZDP_BUFFER_DESC_RESERVED28_29_OFFSET
raw_data, 14	ezdp_frame_defs.h, 382
EZDP_BITWISE_CTR_CFG_DATA_OFFSET	EZDP_BUFFER_DESC_RESERVED28_29_SIZE
ezdp_counter_defs.h, 284	ezdp_frame_defs.h, 382
EZDP_BITWISE_CTR_CFG_DATA_SIZE	ezdp_buffer_desc_t
ezdp_counter_defs.h, 284	ezdp_frame_defs.h, 388
EZDP_BITWISE_CTR_CFG_DATA_WORD_OFFSET	EZDP_BUFFER_DESC_VALID_DATA_BUF_MASK
T	K
ezdp_counter_defs.h, 284	ezdp_frame_defs.h, 382
EZDP_BITWISE_CTR_CFG_DATA_WORD_SELECTOR	EZDP_BUFFER_DESC_VALID_DATA_BUF_OFFSET
T	ET
ezdp_counter_defs.h, 284	ezdp_frame_defs.h, 382
EZDP_BITWISE_CTR_CFG_ECC_OFFSET	EZDP_BUFFER_DESC_VALID_DATA_BUF_SIZE
ezdp_counter_defs.h, 284	ezdp_frame_defs.h, 382
EZDP_BITWISE_CTR_CFG_ECC_SIZE	ezdp_buffer_info, 16
ezdp_counter_defs.h, 284	free_bytes, 16
EZDP_BITWISE_CTR_CFG_RESERVED0_18_OFFSET	ezdp_buffer_mem_type
ET	ezdp_frame_defs.h, 388
ezdp_counter_defs.h, 284	ezdp_bulk_hash
EZDP_BITWISE_CTR_CFG_RESERVED0_18_SIZE	ezdp_math.h, 457
ezdp_counter_defs.h, 284	ezdp_calc_checksum
EZDP_BITWISE_CTR_CFG_RESERVED32_63_OFFSET	ezdp_memory.h, 459
SET	ezdp_calc_checksum_ext_addr
ezdp_counter_defs.h, 284	ezdp_memory.h, 459
EZDP_BITWISE_CTR_CFG_RESERVED32_63_SIZE	ezdp_calc_cpu_id
E	ezdp_processor.h, 499
ezdp_counter_defs.h, 284	ezdp_calc_crc16
EZDP_BITWISE_CTR_CFG_SUB_TYPE_OFFSET	ezdp_math.h, 457
ezdp_counter_defs.h, 284	ezdp_calc_crc32
EZDP_BITWISE_CTR_CFG_SUB_TYPE_SIZE	ezdp_math.h, 457
ezdp_counter_defs.h, 284	ezdp_calc_frame_data_checksum
EZDP_BITWISE_CTR_CFG_WORD_COUNT	ezdp_frame.h, 374
ezdp_counter_defs.h, 284	ezdp_calc_header_offset
ezdp_bitwise_size	ezdp_frame.h, 375
ezdp_counter_defs.h, 294	ezdp_calc_sum_addr
EZDP_BROADCAST	ezdp_memory.h, 460
ezdp_frame_defs.h, 389	ezdp_calc_sum_addr_offset
ezdp_budget_type	ezdp_memory.h, 461
ezdp_job_defs.h, 431	ezdp_calc_tm_queue_depth_handle
ezdp_buf_alloc_failed	ezdp_job.h, 407
ezdp_frame.h, 363	EZDP_CAN_DROP
ezdp_buf_data_len	ezdp_job_defs.h, 432
ezdp_frame.h, 374	ezdp_cancel_job_request
EZDP_BUFFER_DATA_SIZE	ezdp_job.h, 398
ezdp_frame_defs.h, 382	ezdp_change_state_hier_tb_ctr
ezdp_buffer_desc, 15	ezdp_counter.h, 256
__pad0__, 15	EZDP_CHANNEL_NODE
id, 15	ezdp_job_defs.h, 431
raw_data, 15	ezdp_check_notice
valid_data_buf, 15	ezdp_job.h, 404
EZDP_BUFFER_DESC_ID_OFFSET	ezdp_check_tb_ctr



ezdp_counter.h, 251	ezdp_job_defs.h, 426
ezdp_check_tb_ctr_async	EZDP_CONGESTION_STATUS_EMEM_BUF_CON
ezdp_counter.h, 251	GESTION_LEVEL_SIZE
ezdp_check_watchdog_ctr	ezdp_job_defs.h, 426
ezdp_counter.h, 258	EZDP_CONGESTION_STATUS_EMEM_BUF_GUA
ezdp_check_watchdog_ctr_async	RANTEE_MASK
ezdp_counter.h, 258	ezdp_job_defs.h, 426
ezdp_clear_bit	EZDP_CONGESTION_STATUS_EMEM_BUF_GUA
ezdp_math.h, 448	RANTEE_OFFSET
ezdp_clear_bits_bitwise_ctr	ezdp_job_defs.h, 426
ezdp_counter.h, 247	EZDP_CONGESTION_STATUS_EMEM_BUF_GUA
ezdp_clear_bits_bitwise_ctr_async	RANTEE_SIZE
ezdp_counter.h, 248	ezdp_job_defs.h, 426
ezdp_clear_notice	EZDP_CONGESTION_STATUS_IMEM_BUF_CON
ezdp_job.h, 404	GESTION_LEVEL_OFFSET
ezdp_clone_frame_data	ezdp_job_defs.h, 426
ezdp_frame.h, 365	EZDP_CONGESTION_STATUS_IMEM_BUF_CON
ezdp_clone_frame_data_async	GESTION_LEVEL_SIZE
ezdp_frame.h, 365	ezdp_job_defs.h, 426
ezdp_clone_frame_lbd	EZDP_CONGESTION_STATUS_IMEM_BUF_GUA
ezdp_frame.h, 369	RANTEE_MASK
ezdp_clone_frame_lbd_async	ezdp_job_defs.h, 426
ezdp_frame.h, 369	EZDP_CONGESTION_STATUS_IMEM_BUF_GUA
EZDP_CMEM_DATA	RANTEE_OFFSET
ezdp.h, 192	ezdp_job_defs.h, 426
ezdp_combine_4_bits	EZDP_CONGESTION_STATUS_IMEM_BUF_GUA
ezdp_math.h, 453	RANTEE_SIZE
ezdp_combine_merge_4_bits	ezdp_job_defs.h, 426
ezdp_math.h, 454	EZDP_CONGESTION_STATUS_JOB_CONGESTIO
EZDP_COMPRESS	N_LEVEL_OFFSET
ezdp_search_defs.h, 553	ezdp_job_defs.h, 427
ezdp_concat_frames_working_area_t	EZDP_CONGESTION_STATUS_JOB_CONGESTIO
ezdp_frame_defs.h, 388	N_LEVEL_SIZE
ezdp_cond_dec_single_ctr	ezdp_job_defs.h, 426
ezdp_counter.h, 237	EZDP_CONGESTION_STATUS_JOB_GUARANTE
ezdp_cond_dec_single_ctr_async	E_MASK
ezdp_counter.h, 237	ezdp_job_defs.h, 427
ezdp_congestion_level	EZDP_CONGESTION_STATUS_JOB_GUARANTE
ezdp_job_defs.h, 431	E_OFFSET
EZDP_CONGESTION_LEVEL_0	ezdp_job_defs.h, 427
ezdp_job_defs.h, 432	EZDP_CONGESTION_STATUS_JOB_GUARANTE
EZDP_CONGESTION_LEVEL_1	E_SIZE
ezdp_job_defs.h, 432	ezdp_job_defs.h, 427
EZDP_CONGESTION_LEVEL_2	EZDP_CONGESTION_STATUS_PORT_CONGESTI
ezdp_job_defs.h, 433	ON_LEVEL_OFFSET
EZDP_CONGESTION_LEVEL_3	ezdp_job_defs.h, 427
ezdp_job_defs.h, 433	EZDP_CONGESTION_STATUS_PORT_CONGESTI
EZDP_CONGESTION_LEVEL_4	ON_LEVEL_SIZE
ezdp_job_defs.h, 433	ezdp_job_defs.h, 427
ezdp_congestion_status, 17	EZDP_CONGESTION_STATUS_RESERVED11_OF
__pad0__, 17	FSET
__pad1__, 17	ezdp_job_defs.h, 427
__pad2__, 18	EZDP_CONGESTION_STATUS_RESERVED11_SIZ
__pad3__, 18	E
emem_buf_guarantee, 18	ezdp_job_defs.h, 427
imem_buf_guarantee, 18	EZDP_CONGESTION_STATUS_RESERVED14_15_
job_guarantee, 17	OFFSET
raw_data, 17	ezdp_job_defs.h, 427
EZDP_CONGESTION_STATUS_EMEM_BUF_CON	EZDP_CONGESTION_STATUS_RESERVED14_15_
GESTION_LEVEL_OFFSET	SIZE

ezdp\_job\_defs.h, 427  
 EZDP\_CONGESTION\_STATUS\_RESERVED3\_OFF  
 SET  
 ezdp\_job\_defs.h, 426  
 EZDP\_CONGESTION\_STATUS\_RESERVED3\_SIZE  
 ezdp\_job\_defs.h, 426  
 EZDP\_CONGESTION\_STATUS\_RESERVED7\_OFF  
 SET  
 ezdp\_job\_defs.h, 426  
 EZDP\_CONGESTION\_STATUS\_RESERVED7\_SIZE  
 ezdp\_job\_defs.h, 426  
 ezdp\_congestion\_status\_t  
 ezdp\_job\_defs.h, 430  
 ezdp\_container\_info  
 ezdp\_job.h, 403  
 ezdp\_container\_job\_count  
 ezdp\_job.h, 403  
 ezdp\_convert\_std2ext\_working\_area\_t  
 ezdp\_frame\_defs.h, 388  
 ezdp\_copy\_data\_by\_ext\_addr  
 ezdp\_dma.h, 349  
 ezdp\_copy\_data\_by\_ext\_addr\_async  
 ezdp\_dma.h, 349  
 ezdp\_copy\_frame\_data  
 ezdp\_frame.h, 364  
 ezdp\_copy\_frame\_data\_async  
 ezdp\_frame.h, 365  
 ezdp\_copy\_frame\_data\_from\_ext\_addr  
 ezdp\_frame.h, 366  
 ezdp\_copy\_frame\_data\_from\_ext\_addr\_async  
 ezdp\_frame.h, 367  
 ezdp\_copy\_frame\_data\_from\_pci  
 ezdp\_pci.h, 475  
 ezdp\_copy\_frame\_data\_from\_pci\_async  
 ezdp\_pci.h, 476  
 ezdp\_copy\_frame\_data\_to\_ext\_addr  
 ezdp\_frame.h, 366  
 ezdp\_copy\_frame\_data\_to\_ext\_addr\_async  
 ezdp\_frame.h, 366  
 ezdp\_copy\_frame\_data\_to\_pci  
 ezdp\_pci.h, 475  
 ezdp\_copy\_frame\_data\_to\_pci\_async  
 ezdp\_pci.h, 475  
 ezdp\_copy\_frame\_lbd  
 ezdp\_frame.h, 368  
 ezdp\_copy\_frame\_lbd\_async  
 ezdp\_frame.h, 369  
 ezdp\_copy\_frame\_lbd\_from\_ext\_addr  
 ezdp\_frame.h, 370  
 ezdp\_copy\_frame\_lbd\_from\_ext\_addr\_async  
 ezdp\_frame.h, 371  
 ezdp\_copy\_frame\_lbd\_to\_ext\_addr  
 ezdp\_frame.h, 370  
 ezdp\_copy\_frame\_lbd\_to\_ext\_addr\_async  
 ezdp\_frame.h, 370  
 ezdp\_copy\_pci\_data\_from\_ext\_addr  
 ezdp\_pci.h, 478  
 ezdp\_copy\_pci\_data\_from\_ext\_addr\_async  
 ezdp\_pci.h, 479  
 ezdp\_copy\_pci\_data\_to\_ext\_addr  
 ezdp\_pci.h, 478  
 ezdp\_copy\_pci\_data\_to\_ext\_addr\_async  
 ezdp\_pci.h, 478  
 ezdp\_count\_bits  
 ezdp\_math.h, 447  
 ezdp\_counter.h  
 ezdp\_add\_posted\_ctr, 261  
 ezdp\_add\_posted\_ctr\_async, 261  
 ezdp\_change\_state\_hier\_tb\_ctr, 256  
 ezdp\_check\_tb\_ctr, 251  
 ezdp\_check\_tb\_ctr\_async, 251  
 ezdp\_check\_watchdog\_ctr, 258  
 ezdp\_check\_watchdog\_ctr\_async, 258  
 ezdp\_clear\_bits\_bitwise\_ctr, 247  
 ezdp\_clear\_bits\_bitwise\_ctr\_async, 248  
 ezdp\_cond\_dec\_single\_ctr, 237  
 ezdp\_cond\_dec\_single\_ctr\_async, 237  
 ezdp\_dec\_bits\_bitwise\_ctr, 245  
 ezdp\_dec\_bits\_bitwise\_ctr\_async, 245  
 ezdp\_dec\_dual\_ctr, 240  
 ezdp\_dec\_dual\_ctr\_async, 240  
 ezdp\_dec\_single\_ctr, 235  
 ezdp\_dec\_single\_ctr\_async, 235  
 ezdp\_dec\_tb\_ctr, 252  
 ezdp\_dec\_tb\_ctr\_async, 253  
 ezdp\_dual\_add\_posted\_ctr, 261  
 ezdp\_dual\_add\_posted\_ctr\_async, 261  
 ezdp\_dual\_report\_and\_clear\_posted\_ctr, 262  
 ezdp\_dual\_report\_posted\_ctr, 262  
 ezdp\_dual\_reset\_posted\_ctr, 263  
 ezdp\_dual\_reset\_posted\_ctr\_async, 263  
 ezdp\_dual\_write\_posted\_ctr, 260  
 ezdp\_dual\_write\_posted\_ctr\_async, 260  
 ezdp\_inc\_bits\_bitwise\_ctr, 244  
 ezdp\_inc\_bits\_bitwise\_ctr\_async, 244  
 ezdp\_inc\_dual\_ctr, 239  
 ezdp\_inc\_dual\_ctr\_async, 239  
 ezdp\_inc\_hier\_tb\_ctr, 255  
 ezdp\_inc\_hier\_tb\_ctr\_async, 255  
 ezdp\_inc\_single\_ctr, 234  
 ezdp\_inc\_single\_ctr\_async, 235  
 ezdp\_inc\_tb\_ctr, 251  
 ezdp\_inc\_tb\_ctr\_async, 252  
 ezdp\_inc\_watchdog\_ctr, 258  
 ezdp\_inc\_watchdog\_ctr\_async, 258  
 ezdp\_init\_ctr\_msg\_queue\_desc, 259  
 ezdp\_init\_posted\_ctr\_msg\_queue\_desc, 263  
 ezdp\_prefetch\_bitwise\_ctr, 249  
 ezdp\_prefetch\_bitwise\_ctr\_async, 249  
 ezdp\_prefetch\_dual\_ctr, 241  
 ezdp\_prefetch\_dual\_ctr\_async, 241  
 ezdp\_prefetch\_single\_ctr, 237  
 ezdp\_prefetch\_single\_ctr\_async, 238  
 ezdp\_prefetch\_tb\_ctr, 253  
 ezdp\_prefetch\_tb\_ctr\_async, 254  
 ezdp\_prefetch\_watchdog\_ctr, 259  
 ezdp\_prefetch\_watchdog\_ctr\_async, 259  
 ezdp\_read\_and\_clear\_bits\_bitwise\_ctr, 248  
 ezdp\_read\_and\_cond\_dec\_single\_ctr, 237  
 ezdp\_read\_and\_cond\_write\_bits\_bitwise\_ctr, 248

ezdp\_read\_and\_dec\_bits\_bitwise\_ctr, 246  
 ezdp\_read\_and\_dec\_dual\_ctr, 240  
 ezdp\_read\_and\_dec\_single\_ctr, 236  
 ezdp\_read\_and\_dec\_tb\_ctr, 253  
 ezdp\_read\_and\_inc\_bits\_bitwise\_ctr, 245  
 ezdp\_read\_and\_inc\_dual\_ctr, 239  
 ezdp\_read\_and\_inc\_hier\_tb\_ctr, 255  
 ezdp\_read\_and\_inc\_single\_ctr, 235  
 ezdp\_read\_and\_inc\_tb\_ctr, 252  
 ezdp\_read\_and\_reset\_bitwise\_ctr, 246  
 ezdp\_read\_and\_reset\_dual\_ctr, 241  
 ezdp\_read\_and\_reset\_single\_ctr, 236  
 ezdp\_read\_and\_set\_bits\_bitwise\_ctr, 247  
 ezdp\_read\_and\_update\_hier\_tb\_ctr, 256  
 ezdp\_read\_bits\_bitwise\_ctr, 244  
 ezdp\_read\_bitwise\_ctr, 244  
 ezdp\_read\_bitwise\_ctr\_cfg, 242  
 ezdp\_read\_bitwise\_ctr\_cfg\_async, 242  
 ezdp\_read\_ctr\_msg, 259  
 ezdp\_read\_dual\_ctr, 239  
 ezdp\_read\_dual\_ctr\_cfg, 238  
 ezdp\_read\_hier\_tb\_ctr\_cfg, 254  
 ezdp\_read\_posted\_ctr\_msg, 264  
 ezdp\_read\_single\_ctr, 234  
 ezdp\_read\_single\_ctr\_cfg, 233  
 ezdp\_read\_single\_ctr\_cfg\_async, 233  
 ezdp\_read\_tb\_ctr, 250  
 ezdp\_read\_tb\_ctr\_async, 251  
 ezdp\_read\_tb\_ctr\_cfg, 250  
 ezdp\_read\_watchdog\_ctr\_cfg, 257  
 ezdp\_report\_and\_clear\_posted\_ctr, 262  
 ezdp\_report\_posted\_ctr, 262  
 ezdp\_reset\_bitwise\_ctr, 246  
 ezdp\_reset\_bitwise\_ctr\_async, 246  
 ezdp\_reset\_dual\_ctr, 240  
 ezdp\_reset\_dual\_ctr\_async, 241  
 ezdp\_reset\_posted\_ctr, 262  
 ezdp\_reset\_posted\_ctr\_async, 263  
 ezdp\_reset\_single\_ctr, 236  
 ezdp\_reset\_single\_ctr\_async, 236  
 ezdp\_set\_bits\_bitwise\_ctr, 247  
 ezdp\_set\_bits\_bitwise\_ctr\_async, 247  
 ezdp\_start\_watchdog\_ctr, 257  
 ezdp\_start\_watchdog\_ctr\_async, 257  
 ezdp\_update\_hier\_tb\_ctr, 255  
 ezdp\_update\_hier\_tb\_ctr\_async, 256  
 ezdp\_update\_tb\_ctr, 250  
 ezdp\_update\_tb\_ctr\_async, 250  
 ezdp\_write\_bits\_bitwise\_ctr, 243  
 ezdp\_write\_bits\_bitwise\_ctr\_async, 243  
 ezdp\_write\_bitwise\_ctr\_cfg, 242  
 ezdp\_write\_bitwise\_ctr\_cfg\_async, 242  
 ezdp\_write\_dual\_ctr\_cfg, 238  
 ezdp\_write\_dual\_ctr\_cfg\_async, 238  
 ezdp\_write\_hier\_tb\_ctr\_cfg, 254  
 ezdp\_write\_hier\_tb\_ctr\_cfg\_async, 254  
 ezdp\_write\_posted\_ctr, 260  
 ezdp\_write\_posted\_ctr\_async, 260  
 ezdp\_write\_single\_ctr, 233  
 ezdp\_write\_single\_ctr\_async, 234

ezdp\_write\_single\_ctr\_cfg, 232  
 ezdp\_write\_single\_ctr\_cfg\_async, 233  
 ezdp\_write\_tb\_ctr\_cfg, 249  
 ezdp\_write\_tb\_ctr\_cfg\_async, 249  
 ezdp\_write\_watchdog\_ctr\_cfg, 256  
 ezdp\_write\_watchdog\_ctr\_cfg\_async, 257  
 ezdp\_xchg\_bits\_bitwise\_ctr, 243  
 ezdp\_xchg\_single\_ctr, 234  
 ezdp\_counter\_defs.h  
 EZDP\_1\_BITS, 294  
 EZDP\_16\_BITS, 294  
 EZDP\_2\_BITS, 294  
 EZDP\_4\_BITS, 294  
 EZDP\_8\_BITS, 294  
 EZDP\_BITWISE\_CTR\_CFG\_DATA\_OFFSET, 284  
 EZDP\_BITWISE\_CTR\_CFG\_DATA\_SIZE, 284  
 EZDP\_BITWISE\_CTR\_CFG\_DATA\_WORD\_OFFSET, 284  
 EZDP\_BITWISE\_CTR\_CFG\_DATA\_WORD\_SELECT, 284  
 EZDP\_BITWISE\_CTR\_CFG\_ECC\_OFFSET, 284  
 EZDP\_BITWISE\_CTR\_CFG\_ECC\_SIZE, 284  
 EZDP\_BITWISE\_CTR\_CFG\_RESERVED0\_18\_OFFSET, 284  
 EZDP\_BITWISE\_CTR\_CFG\_RESERVED0\_18\_SIZE, 284  
 EZDP\_BITWISE\_CTR\_CFG\_RESERVED32\_63\_OFFSET, 284  
 EZDP\_BITWISE\_CTR\_CFG\_RESERVED32\_63\_SIZE, 284  
 EZDP\_BITWISE\_CTR\_CFG\_SUB\_TYPE\_OFFSET, 284  
 EZDP\_BITWISE\_CTR\_CFG\_SUB\_TYPE\_SIZE, 284  
 EZDP\_BITWISE\_CTR\_CFG\_WORD\_COUNT, 284  
 ezdp\_bitwise\_size, 294  
 EZDP\_COUNTER\_VERSION\_MAJOR, 275  
 EZDP\_COUNTER\_VERSION\_MINOR, 275  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_OFFSET, 289  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_SIZE, 289  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_WORD\_OFFSET, 289  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_WORD\_SELECT, 289  
 EZDP\_CTR\_MSG\_ECC\_OFFSET, 290  
 EZDP\_CTR\_MSG\_ECC\_SIZE, 290  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_OFFSET, 289  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_SIZE, 289  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_WORD\_OFFSET, 289  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_WORD\_SELECT, 289  
 EZDP\_CTR\_MSG\_OVERFLOW\_MASK, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_OFFSET, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_SIZE, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_WORD\_OFFSET, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_WORD\_SELECT, 290

EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDI  
 TION\_MASK, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDI  
 TION\_OFFSET, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDI  
 TION\_SIZE, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDI  
 TION\_WORD\_OFFSET, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDI  
 TION\_WORD\_SELECT, 290  
 ezdp\_ctr\_msg\_queue\_desc\_t, 292  
 EZDP\_CTR\_MSG\_QUEUE\_WORK\_AREA\_SIZE,  
 274  
 EZDP\_CTR\_MSG\_RESERVED6\_OFFSET, 290  
 EZDP\_CTR\_MSG\_RESERVED6\_SIZE, 290  
 EZDP\_CTR\_MSG\_RESERVED8\_23\_OFFSET, 290  
 EZDP\_CTR\_MSG\_RESERVED8\_23\_SIZE, 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_OFFSET, 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_SIZE, 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_WORD\_OFFSET,  
 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_WORD\_SELECT  
 , 290  
 ezdp\_ctr\_msg\_type, 293  
 EZDP\_CTR\_MSG\_WORD\_COUNT, 290  
 EZDP\_CTR\_MSG\_WORK\_AREA\_SIZE, 292  
 ezdp\_ctr\_type, 293  
 EZDP\_DUAL\_CTR, 293  
 EZDP\_DUAL\_CTR\_BYTE\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_BYTE\_SIZE, 276  
 EZDP\_DUAL\_CTR\_BYTE\_WORD\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_BYTE\_WORD\_SELECT, 276  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_REPORT\_EXCE  
 EDED\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_REPORT\_EXCE  
 EDED\_SIZE, 277  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_REPORT\_EXCE  
 EDED\_WORD\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_REPORT\_EXCE  
 EDED\_WORD\_SELECT, 277  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_VALUE\_SIZE\_O  
 FFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_VALUE\_SIZE\_S  
 IZE, 277  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_VALUE\_SIZE\_  
 WORD\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_BYTE\_VALUE\_SIZE\_  
 WORD\_SELECT, 277  
 EZDP\_DUAL\_CTR\_CFG\_CLR\_ON\_GC\_OFFSET,  
 277  
 EZDP\_DUAL\_CTR\_CFG\_CLR\_ON\_GC\_SIZE,  
 277  
 EZDP\_DUAL\_CTR\_CFG\_ECC\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_ECC\_SIZE, 277  
 EZDP\_DUAL\_CTR\_CFG\_ENABLE\_EXCEED\_M  
 ESSAGE\_MASK, 277  
 EZDP\_DUAL\_CTR\_CFG\_ENABLE\_EXCEED\_M  
 ESSAGE\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_ENABLE\_EXCEED\_M  
 ESSAGE\_SIZE, 277  
 EZDP\_DUAL\_CTR\_CFG\_ENABLE\_EXCEED\_M  
 ESSAGE\_WORD\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_ENABLE\_EXCEED\_M  
 ESSAGE\_WORD\_SELECT, 277  
 EZDP\_DUAL\_CTR\_CFG\_EVENT\_REPORT\_EXC  
 EEDED\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_EVENT\_REPORT\_EXC  
 EEDED\_SIZE, 277  
 EZDP\_DUAL\_CTR\_CFG\_EVENT\_REPORT\_EXC  
 EEDED\_WORD\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_EVENT\_REPORT\_EXC  
 EEDED\_WORD\_SELECT, 277  
 EZDP\_DUAL\_CTR\_CFG\_RESERVED0\_OFFSET,  
 276  
 EZDP\_DUAL\_CTR\_CFG\_RESERVED0\_SIZE, 276  
 EZDP\_DUAL\_CTR\_CFG\_RESERVED19\_23\_OFF  
 SET, 277  
 EZDP\_DUAL\_CTR\_CFG\_RESERVED19\_23\_SIZE  
 , 277  
 EZDP\_DUAL\_CTR\_CFG\_VALUE\_OFFSET, 277  
 EZDP\_DUAL\_CTR\_CFG\_VALUE\_SIZE, 277  
 EZDP\_DUAL\_CTR\_CFG\_VALUE\_WORD\_OFFS  
 ET, 278  
 EZDP\_DUAL\_CTR\_CFG\_VALUE\_WORD\_SELE  
 CT, 277  
 EZDP\_DUAL\_CTR\_CFG\_WORD\_COUNT, 278  
 EZDP\_DUAL\_CTR\_EVENT\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_EVENT\_SIZE, 276  
 EZDP\_DUAL\_CTR\_EVENT\_WORD\_OFFSET,  
 276  
 EZDP\_DUAL\_CTR\_EVENT\_WORD\_SELECT,  
 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_LS  
 B\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_LS  
 B\_SIZE, 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_LS  
 B\_WORD\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_LS  
 B\_WORD\_SELECT, 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_MS  
 B\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_MS  
 B\_SIZE, 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_MS  
 B\_WORD\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_RESULT\_BYTE\_VALUE\_MS  
 B\_WORD\_SELECT, 276  
 EZDP\_DUAL\_CTR\_RESULT\_EVENT\_VALUE\_O  
 FFSET, 276  
 EZDP\_DUAL\_CTR\_RESULT\_EVENT\_VALUE\_S  
 IZE, 276  
 EZDP\_DUAL\_CTR\_RESULT\_EVENT\_VALUE\_  
 WORD\_OFFSET, 276  
 EZDP\_DUAL\_CTR\_RESULT\_EVENT\_VALUE\_  
 WORD\_SELECT, 276  
 EZDP\_DUAL\_CTR\_RESULT\_RESERVED31\_OFF  
 SET, 276  
 EZDP\_DUAL\_CTR\_RESULT\_RESERVED31\_SIZ  
 E, 276

EZDP\_DUAL\_CTR\_RESULT\_WORD\_COUNT, 276

EZDP\_DUAL\_CTR\_WORD\_COUNT, 276

ezdp\_get\_color\_hier\_tb\_ctr\_working\_area\_t, 292

EZDP\_GREEN\_TRAFFIC, 293

EZDP\_HIER\_TB\_CALC\_COLOR, 294

EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_SIZE, 281

EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_WORD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_WORD\_SELECT, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_FAIL\_T\_HRESHOLD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_FAIL\_T\_HRESHOLD\_SIZE, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_FAIL\_T\_HRESHOLD\_WORD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_FAIL\_T\_HRESHOLD\_WORD\_SELECT, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_UPDT\_T\_HRESHOLD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_UPDT\_T\_HRESHOLD\_SIZE, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_UPDT\_T\_HRESHOLD\_WORD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_UPDT\_T\_HRESHOLD\_WORD\_SELECT, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_FAIL\_THRES\_HOLD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_FAIL\_THRES\_HOLD\_SIZE, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_FAIL\_THRES\_HOLD\_WORD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_FAIL\_THRES\_HOLD\_WORD\_SELECT, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_UPDT\_THRE\_SHOLD\_OFFSET, 282

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_UPDT\_THRE\_SHOLD\_SIZE, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_UPDT\_THRE\_SHOLD\_WORD\_OFFSET, 282

EZDP\_HIER\_TB\_CTR\_CFG\_CTR0\_UPDT\_THRE\_SHOLD\_WORD\_SELECT, 282

EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_FAIL\_THRES\_HOLD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_FAIL\_THRES\_HOLD\_SIZE, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_FAIL\_THRES\_HOLD\_WORD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_FAIL\_THRES\_HOLD\_WORD\_SELECT, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_UPDT\_THRE\_SHOLD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_UPDT\_THRE\_SHOLD\_SIZE, 280

EZDP\_HIER\_TB\_CTR\_CFG\_CTR1\_UPDT\_THRE\_SHOLD\_WORD\_OFFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_CTRL1\_UPDT\_THRE  
SHOLD\_WORD\_SELECT, 281

EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED0\_1\_OFF  
SET, 280

EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED0\_1\_SIZ  
E, 280

EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED22\_26\_O  
FFSET, 281

EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED22\_26\_S  
IZE, 281

EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED63\_OFF  
SET, 282

EZDP\_HIER\_TB\_CTR\_CFG\_RESERVED63\_SIZE,  
282

EZDP\_HIER\_TB\_CTR\_CFG\_TIMESTAMP\_THRE  
SHOLD\_OFFSET, 282

EZDP\_HIER\_TB\_CTR\_CFG\_TIMESTAMP\_THRE  
SHOLD\_SIZE, 282

EZDP\_HIER\_TB\_CTR\_CFG\_TIMESTAMP\_THRE  
SHOLD\_WORD\_OFFSET, 282

EZDP\_HIER\_TB\_CTR\_CFG\_TIMESTAMP\_THRE  
SHOLD\_WORD\_SELECT, 282

EZDP\_HIER\_TB\_CTR\_CFG\_WORD\_COUNT, 282

EZDP\_HIER\_TB\_RESULT\_APP\_BITS\_OFFSET,  
283

EZDP\_HIER\_TB\_RESULT\_APP\_BITS\_SIZE, 283

EZDP\_HIER\_TB\_RESULT\_APP\_BITS\_WORD\_O  
FFSET, 283

EZDP\_HIER\_TB\_RESULT\_APP\_BITS\_WORD\_S  
ELECT, 283

EZDP\_HIER\_TB\_RESULT\_CTRL0\_OFFSET, 283

EZDP\_HIER\_TB\_RESULT\_CTRL0\_SIZE, 283

EZDP\_HIER\_TB\_RESULT\_CTRL0\_WORD\_OFFSE  
T, 283

EZDP\_HIER\_TB\_RESULT\_CTRL0\_WORD\_SELEC  
T, 283

EZDP\_HIER\_TB\_RESULT\_CTRL1\_OFFSET, 282

EZDP\_HIER\_TB\_RESULT\_CTRL1\_SIZE, 282

EZDP\_HIER\_TB\_RESULT\_CTRL1\_WORD\_OFFSE  
T, 282

EZDP\_HIER\_TB\_RESULT\_CTRL1\_WORD\_SELEC  
T, 282

EZDP\_HIER\_TB\_RESULT\_FAIL\_MASK, 283

EZDP\_HIER\_TB\_RESULT\_FAIL\_OFFSET, 283

EZDP\_HIER\_TB\_RESULT\_FAIL\_SIZE, 282

EZDP\_HIER\_TB\_RESULT\_FAIL\_WORD\_OFFSE  
T, 283

EZDP\_HIER\_TB\_RESULT\_FAIL\_WORD\_SELEC  
T, 283

EZDP\_HIER\_TB\_RESULT\_RESERVED0\_9\_OFFS  
ET, 282

EZDP\_HIER\_TB\_RESULT\_RESERVED0\_9\_SIZE,  
282

EZDP\_HIER\_TB\_RESULT\_RESERVED56\_63\_OF  
FSET, 283

EZDP\_HIER\_TB\_RESULT\_RESERVED56\_63\_SI  
ZE, 283

EZDP\_HIER\_TB\_RESULT\_RESERVED82\_95\_OF  
FSET, 283

EZDP\_HIER\_TB\_RESULT\_RESERVED82\_95\_S  
 ZE, 283  
 EZDP\_HIER\_TB\_RESULT\_STATE\_OFFSET, 282  
 EZDP\_HIER\_TB\_RESULT\_STATE\_SIZE, 282  
 EZDP\_HIER\_TB\_RESULT\_STATE\_WORD\_OFFS  
 ET, 282  
 EZDP\_HIER\_TB\_RESULT\_STATE\_WORD\_SELE  
 CT, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_MA  
 SK, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_OFF  
 SET, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_SIZ  
 E, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_WO  
 RD\_OFFSET, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_WO  
 RD\_SELECT, 282  
 EZDP\_HIER\_TB\_RESULT\_WORD\_COUNT, 283  
 ezdp\_hier\_tb\_state, 293  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_APP\_BITS\_OFF  
 SET, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_APP\_BITS\_SIZ  
 E, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STAT  
 E\_G\_OFFSET, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STAT  
 E\_G\_SIZE, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STAT  
 E\_Y\_OFFSET, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STAT  
 E\_Y\_SIZE, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_EIGHTH\_MODE\_  
 RET\_BITS\_OFFSET, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_EIGHTH\_MODE\_  
 RET\_BITS\_SIZE, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_RESERVED24\_  
 31\_OFFSET, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_RESERVED24\_  
 31\_SIZE, 280  
 ezdp\_hier\_tb\_ug\_app\_bits\_t, 292  
 EZDP\_HIER\_TB\_UPDATE\_APP\_BITS\_OFFSET,  
 283  
 EZDP\_HIER\_TB\_UPDATE\_APP\_BITS\_SIZE, 283  
 EZDP\_HIER\_TB\_UPDATE\_BES, 292  
 EZDP\_HIER\_TB\_UPDATE\_CLR\_CTR\_MASK,  
 284  
 EZDP\_HIER\_TB\_UPDATE\_CLR\_CTR\_OFFSET,  
 284  
 EZDP\_HIER\_TB\_UPDATE\_CLR\_CTR\_SIZE, 284  
 EZDP\_HIER\_TB\_UPDATE\_COND\_SET\_ACTIVE  
 \_STATE\_MASK, 283  
 EZDP\_HIER\_TB\_UPDATE\_COND\_SET\_ACTIVE  
 \_STATE\_OFFSET, 283  
 EZDP\_HIER\_TB\_UPDATE\_COND\_SET\_ACTIVE  
 \_STATE\_SIZE, 283  
 EZDP\_HIER\_TB\_UPDATE\_RESERVED24\_27\_OF  
 FSET, 283  
 EZDP\_HIER\_TB\_UPDATE\_RESERVED24\_27\_SI  
 ZE, 283  
 EZDP\_HIER\_TB\_UPDATE\_SET\_ACTIVE\_STAT  
 E\_MASK, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_ACTIVE\_STAT  
 E\_OFFSET, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_ACTIVE\_STAT  
 E\_SIZE, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_APP\_BITS\_MAS  
 K, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_APP\_BITS\_OFF  
 SET, 283  
 EZDP\_HIER\_TB\_UPDATE\_SET\_APP\_BITS\_SIZE  
 , 283  
 ezdp\_hier\_tb\_update\_t, 292  
 EZDP\_INACTIVE, 293  
 ezdp\_msg\_posted\_ctr\_type, 294  
 EZDP\_NULL\_CTR\_MSG, 293  
 EZDP\_NULL\_POSTED\_CTR\_MSG, 294  
 EZDP\_PERIODIC\_CTR\_MSG, 293  
 EZDP\_PERIODIC\_POSTED\_CTR\_MSG, 294  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_MASK, 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_OFFSET,  
 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_SIZE, 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_WORD\_OF  
 FSET, 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_WORD\_SEL  
 ECT, 291  
 EZDP\_POSTED\_CTR\_MSG\_ECC\_OFFSET, 291  
 EZDP\_POSTED\_CTR\_MSG\_ECC\_SIZE, 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_MASK, 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_OFFSET,  
 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_SIZE, 290  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_WORD\_OFF  
 SET, 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_WORD\_SEL  
 ECT, 291  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_OFFSE  
 T, 290  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_SIZE,  
 290  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_WORD  
 \_OFFSET, 290  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_WORD  
 \_SELECT, 290  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR  
 \_CONDITION\_MASK, 291  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR  
 \_CONDITION\_OFFSET, 291  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR  
 \_CONDITION\_SIZE, 291  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR  
 \_CONDITION\_WORD\_OFFSET, 291  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR  
 \_CONDITION\_WORD\_SELECT, 291  
 ezdp\_posted\_ctr\_msg\_queue\_desc\_t, 292  
 EZDP\_POSTED\_CTR\_MSG\_RESERVED5\_6\_OFF  
 SET, 291  
 EZDP\_POSTED\_CTR\_MSG\_RESERVED5\_6\_SIZ  
 E, 291

EZDP\_POSTED\_CTR\_MSG\_RESERVED8\_23\_OF  
FSET, 291  
 EZDP\_POSTED\_CTR\_MSG\_RESERVED8\_23\_SI  
ZE, 291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_OFFSE  
T, 291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_SIZE,  
291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_WORD  
\_OFFSET, 291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_WORD  
\_SELECT, 291  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_OFFSET,  
292  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_SIZE, 291  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_WORD\_OF  
FSET, 292  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_WORD\_SE  
LECT, 292  
 EZDP\_POSTED\_CTR\_MSG\_WORD\_COUNT, 292  
 EZDP\_POSTED\_CTR\_MSG\_WORK\_AREA\_SIZE,  
292  
 EZDP\_RED\_TRAFFIC, 293  
 EZDP\_REPORT\_POSTED\_CTR\_MSG, 294  
 EZDP\_SINGLE\_BUCKET, 293  
 EZDP\_SINGLE\_CTR, 293  
 EZDP\_SINGLE\_CTR\_CFG\_ECC\_OFFSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ECC\_SIZE, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_  
MESSAGE\_MASK, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_  
MESSAGE\_OFFSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_  
MESSAGE\_SIZE, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_  
MESSAGE\_WORD\_OFFSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_  
MESSAGE\_WORD\_SELECT, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDE  
D\_OFFSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDE  
D\_SIZE, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDE  
D\_WORD\_OFFSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDE  
D\_WORD\_SELECT, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED0\_10\_OFF  
SET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED0\_10\_SIZ  
E, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED32\_63\_OF  
FSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED32\_63\_SI  
ZE, 275  
 EZDP\_SINGLE\_CTR\_CFG\_SUB\_TYPE\_OFFSET,  
275  
 EZDP\_SINGLE\_CTR\_CFG\_SUB\_TYPE\_SIZE, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_OFFSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_SIZE, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_WORD\_OFF  
SET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_WORD\_SEL  
ECT, 275  
 EZDP\_SINGLE\_CTR\_CFG\_WORD\_COUNT, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ZERO\_OFFSET, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ZERO\_SIZE, 275  
 EZDP\_SR\_TCM, 293  
 ezdp\_tb\_algo, 293  
 ezdp\_tb\_color, 292  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_OFF  
SET, 279  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_SIZE,  
279  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_WOR  
D\_OFFSET, 279  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_WOR  
D\_SELECT, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_MASK,  
279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_OFFSET  
, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_SIZE,  
279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_WORD\_  
OFFSET, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_WORD\_  
SELECT, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_O  
FFSET, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_SI  
ZE, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_W  
ORD\_OFFSET, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_W  
ORD\_SELECT, 279  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_MASK,  
280  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_OFFSE  
T, 280  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_SIZE,  
279  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_WORD  
\_OFFSET, 280  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_WORD  
\_SELECT, 280  
 EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_OF  
FSET, 279  
 EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_SIZ  
E, 279  
 EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_W  
ORD\_OFFSET, 279  
 EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_W  
ORD\_SELECT, 279  
 EZDP\_TB\_CTR\_CFG\_RESERVED26\_31\_OFFSET  
, 280  
 EZDP\_TB\_CTR\_CFG\_RESERVED26\_31\_SIZE,  
280  
 EZDP\_TB\_CTR\_CFG\_RESERVED32\_63\_OFFSET  
, 280

EZDP_TB_CTR_CFG_RESERVED32_63_SIZE, 280	EZDP_TB_CTR_RESULT_RESERVED0_31_SIZE, 278
EZDP_TB_CTR_CFG_RESERVED64_95_OFFSET, 280	EZDP_TB_CTR_RESULT_RESERVED34_57_OFFSET, 278
EZDP_TB_CTR_CFG_RESERVED64_95_SIZE, 280	EZDP_TB_CTR_RESULT_RESERVED34_57_SIZE, 278
EZDP_TB_CTR_CFG_RESERVED96_127_OFFSET, 280	EZDP_TB_CTR_RESULT_RESERVED60_63_OFFSET, 279
EZDP_TB_CTR_CFG_RESERVED96_127_SIZE, 280	EZDP_TB_CTR_RESULT_RESERVED60_63_SIZE, 279
EZDP_TB_CTR_CFG_WORD_COUNT, 280	EZDP_TB_CTR_RESULT_WORD_COUNT, 279
EZDP_TB_CTR_RESULT_COLOR_OFFSET, 278	EZDP_THRESHOLD_CTR_MSG, 293
EZDP_TB_CTR_RESULT_COLOR_SIZE, 278	EZDP_THRESHOLD_POSTED_CTR_MSG, 294
EZDP_TB_CTR_RESULT_COLOR_WORD_OFFSET, 278	EZDP_TR_TCM, 293
EZDP_TB_CTR_RESULT_COLOR_WORD_SELECT, 278	EZDP_TR_TCM_MEF, 293
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_MASK, 278	EZDP_UG_FAST_PATH, 293
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_OFFSET, 278	EZDP_UG_SLOW_PATH, 293
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_SIZE, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_OFFSET, 284
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_UG_MASK, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_SIZE, 284
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_UG_OFFSET, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_WORD_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_UG_SIZE, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ACCUMULATIVE_EVENTS_WORD_SELECT, 284
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_UG_WORD_OFFSET, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ALERT_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_UG_WORD_SELECT, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_ALERT_SIZE, 285
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_WORD_OFFSET, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_COMMIT_BUFFER_WORD_SELECT, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_SIZE, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_MASK, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_WORD_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_OFFSET, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_CURR_EVENTS_WORD_SELECT, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_SIZE, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_INIT_BIT_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_UG_MASK, 279	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_INIT_BIT_SIZE, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_UG_OFFSET, 279	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_UG_SIZE, 279	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_SIZE, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_UG_WORD_OFFSET, 279	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_WORD_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_UG_WORD_SELECT, 279	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_LAST_EVENTS_WORD_SELECT, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_WORD_OFFSET, 278	EZDP_WATCHDOG_ACCUMULATIVE_WINDOW_CFG_MAX_THRESHOLD_ALERT_OFFSET, 285
EZDP_TB_CTR_RESULT_EMPTY_EXCESS_BUFFER_WORD_SELECT, 278	
EZDP_TB_CTR_RESULT_RESERVED0_31_OFFSET, 278	



EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MAX\_THRESHOLD\_ALERT\_SIZE, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MIN\_THRESHOLD\_ALERT\_OFFSET, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MIN\_THRESHOLD\_ALERT\_SIZE, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PARITY\_OFFSET, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PARITY\_SIZE, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_OFFSET, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_SIZE, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_WORD\_OFFSET, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_WORD\_SELECT, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED5\_28\_OFFSET, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED5\_28\_SIZE, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED63\_OFFSET, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED63\_SIZE, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_MASK, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_OFFSET, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_SIZE, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_WORD\_OFFSET, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_WORD\_SELECT, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_WORD\_COUNT, 286  
 EZDP\_WATCHDOG\_CTR\_CFG\_ECC\_OFFSET, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_ECC\_SIZE, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED0\_18\_OFFSET, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED0\_18\_SIZE, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED32\_63\_OFFSET, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED32\_63\_SIZE, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_SUB\_TYPE\_OFFSET, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_SUB\_TYPE\_SIZE, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_WORD\_COUNT, 287  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ACCUMULATIVE\_EVENTS\_OFFSET, 287  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ACCUMULATIVE\_EVENTS\_SIZE, 287  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_MASK, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_SIZE, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_WORD\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_WORD\_SELECT, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_CURRENT\_EVENTS\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_CURRENT\_EVENTS\_SIZE, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_INIT\_BIT\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_INIT\_BIT\_SIZE, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_LAST\_EVENTS\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_LAST\_EVENTS\_SIZE, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MAX\_THRESHOLD\_ALERT\_MASK, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MAX\_THRESHOLD\_ALERT\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MAX\_THRESHOLD\_ALERT\_SIZE, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MAX\_THRESHOLD\_ALERT\_WORD\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MAX\_THRESHOLD\_ALERT\_WORD\_SELECT, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MIN\_THRESHOLD\_ALERT\_MASK, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MIN\_THRESHOLD\_ALERT\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MIN\_THRESHOLD\_ALERT\_SIZE, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MIN\_THRESHOLD\_ALERT\_WORD\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_MIN\_THRESHOLD\_ALERT\_WORD\_SELECT, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_PROFILE\_ID\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_PROFILE\_ID\_SIZE, 289  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_RESERVED4\_28\_OFFSET, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_RESERVED4\_28\_SIZE, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_RESERVED62\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_RESERVED62\_SIZE, 289

EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_VA  
 LID\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_VA  
 LID\_SIZE, 289  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_WI  
 NDOW\_RELATED\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_WI  
 NDOW\_RELATED\_SIZE, 289  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_W  
 ORD\_COUNT, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_LS  
 B\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_LS  
 B\_SIZE, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_LS  
 B\_WORD\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_LS  
 B\_WORD\_SELECT, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_MS  
 B\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_MS  
 B\_SIZE, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_MS  
 B\_WORD\_OFFSET, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_MS  
 B\_WORD\_SELECT, 289  
 EZDP\_WATCHDOG\_CTR\_START\_RESULT\_WO  
 RD\_COUNT, 289  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 ALERT\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 ALERT\_SIZE, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 COUNTERS\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 COUNTERS\_SIZE, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 COUNTERS\_WORD\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 COUNTERS\_WORD\_SELECT, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 MAX\_THRESHOLD\_ALERT\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 MAX\_THRESHOLD\_ALERT\_SIZE, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 MIN\_THRESHOLD\_ALERT\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 MIN\_THRESHOLD\_ALERT\_SIZE, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 PARITY\_OFFSET, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 PARITY\_SIZE, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 PROFILE\_ID\_OFFSET, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 PROFILE\_ID\_SIZE, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 PROFILE\_ID\_WORD\_OFFSET, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 PROFILE\_ID\_WORD\_SELECT, 287

EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 RESERVED5\_28\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 RESERVED5\_28\_SIZE, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 RESERVED56\_OFFSET, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 RESERVED56\_SIZE, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 RESERVED63\_OFFSET, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 RESERVED63\_SIZE, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_MASK, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_OFFSET, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_SIZE, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_WINDOWS\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_WINDOWS\_SIZE, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_WINDOWS\_WORD\_OFFSET, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_WINDOWS\_WORD\_SELECT, 286  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_WORD\_OFFSET, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 VALID\_WORD\_SELECT, 287  
 EZDP\_WATCHDOG\_SLIDING\_WINDOW\_CFG\_  
 WORD\_COUNT, 287  
 EZDP\_YELLOW\_TRAFFIC, 293  
 EZDP\_COUNTER\_VERSION\_MAJOR  
 ezdp\_counter\_defs.h, 275  
 EZDP\_COUNTER\_VERSION\_MINOR  
 ezdp\_counter\_defs.h, 275  
 EZDP\_CRITICAL\_LEVEL  
 ezdp\_job\_defs.h, 431  
 ezdp\_ctr\_msg, 19  
 \_\_pad0\_\_, 19  
 \_\_pad1\_\_, 19  
 \_\_pad2\_\_, 19  
 dual\_ctr\_value, 20  
 overflow, 19  
 overrun\_error\_condition, 19  
 raw\_data, 19  
 single\_ctr\_value, 20  
 sum\_addr, 20  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_OFFSET  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_SIZE  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_WORD\_OFFS  
 ET  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_COUNTER\_TYPE\_WORD\_SELE  
 CT  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_ECC\_OFFSET

ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_ECC\_SIZE  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_OFFSET  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_SIZE  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_MSG\_TYPE\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 289  
 EZDP\_CTR\_MSG\_OVERFLOW\_MASK  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_SIZE  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERFLOW\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_MASK  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_SIZE  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 290  
 ezdp\_ctr\_msg\_queue\_desc\_t  
 ezdp\_counter\_defs.h, 292  
 EZDP\_CTR\_MSG\_QUEUE\_WORK\_AREA\_SIZE  
 ezdp\_counter\_defs.h, 274  
 EZDP\_CTR\_MSG\_RESERVED6\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_RESERVED6\_SIZE  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_RESERVED8\_23\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_RESERVED8\_23\_SIZE  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_SIZE  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 290  
 EZDP\_CTR\_MSG\_SUM\_ADDR\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 290  
 ezdp\_ctr\_msg\_type  
 ezdp\_counter\_defs.h, 293  
 EZDP\_CTR\_MSG\_WORD\_COUNT  
 ezdp\_counter\_defs.h, 290

EZDP\_CTR\_MSG\_WORK\_AREA\_SIZE  
 ezdp\_counter\_defs.h, 292  
 ezdp\_ctr\_type  
 ezdp\_counter\_defs.h, 293  
 ezdp\_data\_mem\_space  
 ezdp.h, 192  
 ezdp\_dec\_bits\_bitwise\_ctr  
 ezdp\_counter.h, 245  
 ezdp\_dec\_bits\_bitwise\_ctr\_async  
 ezdp\_counter.h, 245  
 ezdp\_dec\_dual\_ctr  
 ezdp\_counter.h, 240  
 ezdp\_dec\_dual\_ctr\_async  
 ezdp\_counter.h, 240  
 ezdp\_dec\_single\_ctr  
 ezdp\_counter.h, 235  
 ezdp\_dec\_single\_ctr\_async  
 ezdp\_counter.h, 235  
 ezdp\_dec\_tb\_ctr  
 ezdp\_counter.h, 252  
 ezdp\_dec\_tb\_ctr\_async  
 ezdp\_counter.h, 253  
 ezdp\_dec\_tm\_imem\_buf\_ctr  
 ezdp\_frame.h, 375  
 ezdp\_dec\_tm\_imem\_buf\_ctr\_async  
 ezdp\_frame.h, 375  
 ezdp\_decode.h  
 ezdp\_decode\_eth\_type, 299  
 ezdp\_decode\_ip\_protocol, 298  
 ezdp\_decode\_ipv4, 296  
 ezdp\_decode\_ipv4\_async, 296  
 ezdp\_decode\_ipv6, 296  
 ezdp\_decode\_ipv6\_async, 297  
 ezdp\_decode\_mac, 295  
 ezdp\_decode\_mac\_async, 295  
 ezdp\_decode\_mpls, 297  
 ezdp\_decode\_mpls\_async, 297  
 ezdp\_decode\_mpls\_label, 298  
 ezdp\_decode\_mpls\_label\_async, 298  
 ezdp\_decode\_tcp, 298  
 ezdp\_decode\_defs.h  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_MASK, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_OFFSET, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_SIZE, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_MASK, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_OFFSET, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_SIZE, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8\_MASK, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8\_OFFSET, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8\_SIZE, 329

EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_MASK, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_OFFSET, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_SIZE, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_MASK, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_OFFSET, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_SIZE, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_MASK, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_OFFSET, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_SIZE, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MULTICAST\_MASK, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MULTICAST\_OFFSET, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MULTICAST\_SIZE, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNICAST\_MASK, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNICAST\_OFFSET, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNICAST\_SIZE, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_MASK, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_OFFSET, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_SIZE, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DISCOVERY\_MASK, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DISCOVERY\_OFFSET, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DISCOVERY\_SIZE, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_MASK, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_OFFSET, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_SIZE, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_RESERVED13\_31\_OFFSET, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_RESERVED13\_31\_SIZE, 330  
 ezdp\_decode\_eth\_type\_retval\_t, 344  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_MASK, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_OFFSET, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_SIZE, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_MASK, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_OFFSET, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_SIZE, 330  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_MASK, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_OFFSET, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_SIZE, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_GMP\_MASK, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_GMP\_OFFSET, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_GMP\_SIZE, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_MASK, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_OFFSET, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_SIZE, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_MASK, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_OFFSET, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_SIZE, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_MASK, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_OFFSET, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_SIZE, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_MASK, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_OFFSET, 319  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_SIZE, 319  
 ezdp\_decode\_ip\_next\_protocol\_t, 344  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_MASK, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_OFFSET, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_SIZE, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_MASK, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_OFFSET, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_SIZE, 318  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_AH\_PROT\_MASK, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_AH\_PROT\_OFFSET, 328

EZDP_DECODE_IP_PROTOCOL_RETVAL_AH_PROT_SIZE, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_MPLS_SIZE, 327
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_0_MASK, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_OTHER_MASK, 328
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_0_OFFSET, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_OTHER_OFFSET, 328
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_0_SIZE, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_OTHER_SIZE, 328
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_1_MASK, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_RESERVED14_31_OFFSET, 329
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_1_OFFSET, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_RESERVED14_31_SIZE, 328
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_1_SIZE, 328	ezdp_decode_ip_protocol_retval_t, 344
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_2_MASK, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_TCP_MASK, 327
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_2_OFFSET, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_TCP_OFFSET, 327
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_2_SIZE, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_TCP_SIZE, 327
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_3_MASK, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_UDP_MASK, 327
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_3_OFFSET, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_UDP_OFFSET, 327
EZDP_DECODE_IP_PROTOCOL_RETVAL_DEF_IP_PROT_3_SIZE, 328	EZDP_DECODE_IP_PROTOCOL_RETVAL_UDP_SIZE, 327
EZDP_DECODE_IP_PROTOCOL_RETVAL_ESP_PROT_MASK, 328	EZDP_DECODE_IPV4_CONTROL_ICMP_MASK, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_ESP_PROT_OFFSET, 328	EZDP_DECODE_IPV4_CONTROL_ICMP_OFFSET, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_ESP_PROT_SIZE, 328	EZDP_DECODE_IPV4_CONTROL_ICMP_SIZE, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_GRE_MASK, 327	EZDP_DECODE_IPV4_CONTROL_IGMP_MASK, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_GRE_OFFSET, 327	EZDP_DECODE_IPV4_CONTROL_IGMP_OFFSET, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_GRE_SIZE, 327	EZDP_DECODE_IPV4_CONTROL_IGMP_SIZE, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_ICMP_IGMP_MASK, 328	EZDP_DECODE_IPV4_CONTROL_INTERNETWORK_MULTICAST_RANGE_MASK, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_ICMP_IGMP_OFFSET, 327	EZDP_DECODE_IPV4_CONTROL_INTERNETWORK_MULTICAST_RANGE_OFFSET, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_ICMP_IGMP_SIZE, 327	EZDP_DECODE_IPV4_CONTROL_INTERNETWORK_MULTICAST_RANGE_SIZE, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV4_MASK, 327	EZDP_DECODE_IPV4_CONTROL_LINK_LOCAL_MULTICAST_RANGE_MASK, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV4_OFFSET, 327	EZDP_DECODE_IPV4_CONTROL_LINK_LOCAL_MULTICAST_RANGE_OFFSET, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV4_SIZE, 327	EZDP_DECODE_IPV4_CONTROL_LINK_LOCAL_MULTICAST_RANGE_SIZE, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV6_MASK, 327	EZDP_DECODE_IPV4_CONTROL_RESERVED_2_OFFSET, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV6_OFFSET, 327	EZDP_DECODE_IPV4_CONTROL_RESERVED_2_SIZE, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_IPV6_SIZE, 327	ezdp_decode_ipv4_control_t, 344
EZDP_DECODE_IP_PROTOCOL_RETVAL_MPLS_MASK, 327	EZDP_DECODE_IPV4_CONTROL_USER_CONFIG0_MASK, 316
EZDP_DECODE_IP_PROTOCOL_RETVAL_MPLS_OFFSET, 327	EZDP_DECODE_IPV4_CONTROL_USER_CONFIG0_OFFSET, 316
	EZDP_DECODE_IPV4_CONTROL_USER_CONFIG0_SIZE, 316

EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_MASK, 316  
 EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_OFFSET, 316  
 EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_SIZE, 316  
 EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_MASK, 317  
 EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_SIZE, 316  
 EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_MASK, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_SIZE, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_MASK, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_OFFSET, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_SIZE, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_MASK, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_SIZE, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_MASK, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_SIZE, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_MASK, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_SIZE, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_RESERVED9\_15\_OFFSET, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_RESERVED9\_15\_SIZE, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_MASK, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_OFFSET, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_SIZE, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_MASK, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_SIZE, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_MASK, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_SIZE, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_MASK, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_OFFSET, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_SIZE, 317  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_OFFSET, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_SIZE, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_WORD\_OFFSET, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_WORD\_SELECT, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_ERROR\_CODE\_OFFSET, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_ERROR\_CODE\_SIZE, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_ERROR\_CODE\_WORD\_OFFSET, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_ERROR\_CODE\_WORD\_SELECT, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_FIRST\_FRAGMENT\_MASK, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_FIRST\_FRAGMENT\_OFFSET, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_FIRST\_FRAGMENT\_SIZE, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_FIRST\_FRAGMENT\_WORD\_OFFSET, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_FIRST\_FRAGMENT\_WORD\_SELECT, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_NEXT\_PROTOCOL\_OFFSET, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_NEXT\_PROTOCOL\_SIZE, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_NEXT\_PROTOCOL\_WORD\_OFFSET, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_NEXT\_PROTOCOL\_WORD\_SELECT, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_OPTION\_EXIST\_MASK, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_OPTION\_EXIST\_OFFSET, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_OPTION\_EXIST\_SIZE, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_OPTION\_EXIST\_WORD\_OFFSET, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_OPTION\_EXIST\_WORD\_SELECT, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_RESERVED\_2\_6\_OFFSET, 331  
 EZDP\_DECODE\_IPV4\_RESULT\_RESERVED\_2\_6\_SIZE, 331

EZDP_DECODE_IPV4_RESULT_RESERVED_56_63_OFFSET, 332	EZDP_DECODE_IPV6_CONTROL_DIP_IS_MULTICAST_SIZE, 319
EZDP_DECODE_IPV4_RESULT_RESERVED_56_63_SIZE, 332	EZDP_DECODE_IPV6_CONTROL_DIP_IS_WELLKNOWN_MULTICAST_MASK, 320
EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_OFFSET, 332	EZDP_DECODE_IPV6_CONTROL_DIP_IS_WELLKNOWN_MULTICAST_OFFSET, 320
EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_SIZE, 332	EZDP_DECODE_IPV6_CONTROL_DIP_IS_WELLKNOWN_MULTICAST_SIZE, 320
EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_WORD_OFFSET, 332	EZDP_DECODE_IPV6_CONTROL_INTERNETWORK_MULTICAST_RANGE_MASK, 319
EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_WORD_SELECT, 332	EZDP_DECODE_IPV6_CONTROL_INTERNETWORK_MULTICAST_RANGE_OFFSET, 319
EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_MASK, 331	EZDP_DECODE_IPV6_CONTROL_INTERNETWORK_MULTICAST_RANGE_SIZE, 319
EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_OFFSET, 331	EZDP_DECODE_IPV6_CONTROL_LINK_LOCAL_MULTICAST_RANGE_MASK, 319
EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_SIZE, 331	EZDP_DECODE_IPV6_CONTROL_LINK_LOCAL_MULTICAST_RANGE_OFFSET, 319
EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_WORD_OFFSET, 331	EZDP_DECODE_IPV6_CONTROL_LINK_LOCAL_MULTICAST_RANGE_SIZE, 319
EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_WORD_SELECT, 331	EZDP_DECODE_IPV6_CONTROL_RESERVED_3_OFFSET, 319
EZDP_DECODE_IPV4_RESULT_WORD_COUNT, 332	EZDP_DECODE_IPV6_CONTROL_RESERVED_3_SIZE, 319
EZDP_DECODE_IPV4_RETVAL_CONTROL_OFFSET, 331	EZDP_DECODE_IPV6_CONTROL_RESERVED_7_OFFSET, 320
EZDP_DECODE_IPV4_RETVAL_CONTROL_SIZE, 331	EZDP_DECODE_IPV6_CONTROL_RESERVED_7_SIZE, 320
EZDP_DECODE_IPV4_RETVAL_ERROR_CODE_S_OFFSET, 331	EZDP_DECODE_IPV6_CONTROL_SOLICITED_NODE_MULTICAST_RANGE_MASK, 319
EZDP_DECODE_IPV4_RETVAL_ERROR_CODE_S_SIZE, 331	EZDP_DECODE_IPV6_CONTROL_SOLICITED_NODE_MULTICAST_RANGE_OFFSET, 319
EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_MASK, 331	EZDP_DECODE_IPV6_CONTROL_SOLICITED_NODE_MULTICAST_RANGE_SIZE, 319
EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_OFFSET, 331	ezdp_decode_ipv6_control_t, 344
EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_SIZE, 331	EZDP_DECODE_IPV6_ERRORS_DECODE_ERROR_MASK, 321
EZDP_DECODE_IPV4_RETVAL_OPTION_EXISTS_MASK, 330	EZDP_DECODE_IPV6_ERRORS_DECODE_ERROR_OFFSET, 321
EZDP_DECODE_IPV4_RETVAL_OPTION_EXISTS_OFFSET, 330	EZDP_DECODE_IPV6_ERRORS_DECODE_ERROR_SIZE, 321
EZDP_DECODE_IPV4_RETVAL_OPTION_EXISTS_SIZE, 330	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_MASK, 320
EZDP_DECODE_IPV4_RETVAL_RESERVED_2_6_OFFSET, 331	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_OFFSET, 320
EZDP_DECODE_IPV4_RETVAL_RESERVED_2_6_SIZE, 331	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_SIZE, 320
ezdp_decode_ipv4_retval_t, 344	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_MASK, 320
EZDP_DECODE_IPV4_RETVAL_USER_CONFIG_SIP_MASK, 331	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_OFFSET, 320
EZDP_DECODE_IPV4_RETVAL_USER_CONFIG_SIP_OFFSET, 331	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_SIZE, 320
EZDP_DECODE_IPV4_RETVAL_USER_CONFIG_SIP_SIZE, 330	EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_MASK, 320
EZDP_DECODE_IPV6_CONTROL_DIP_IS_MULTICAST_MASK, 319	EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_OFFSET, 320
EZDP_DECODE_IPV6_CONTROL_DIP_IS_MULTICAST_OFFSET, 319	EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_SIZE, 320

EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_GT\_FRAME\_LENGTH\_MASK, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_GT\_FRAME\_LENGTH\_OFFSET, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_GT\_FRAME\_LENGTH\_SIZE, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_MISSING\_MASK, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_MISSING\_OFFSET, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_PAYLOAD\_MISSING\_SIZE, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_RESERVED10\_15\_OFFSET, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_RESERVED10\_15\_SIZE, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_EQUAL\_DIP\_MASK, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_EQUAL\_DIP\_OFFSET, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_EQUAL\_DIP\_SIZE, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_MULTICAST\_MASK, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_MULTICAST\_OFFSET, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_MULTICAST\_SIZE, 321  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ONE\_MASK, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ONE\_OFFSET, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ONE\_SIZE, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ZERO\_MASK, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ZERO\_OFFSET, 320  
 EZDP\_DECODE\_IPV6\_ERRORS\_SIP\_IS\_ZERO\_SIZE, 320  
 ezdp\_decode\_ipv6\_errors\_t, 344  
 EZDP\_DECODE\_IPV6\_RESULT\_CONTROL\_OFFSET, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_CONTROL\_SIZE, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_CONTROL\_WORD\_OFFSET, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_CONTROL\_WORD\_SELECT, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_ERROR\_CODE\_S\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_ERROR\_CODE\_S\_SIZE, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_ERROR\_CODE\_S\_WORD\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_ERROR\_CODE\_S\_WORD\_SELECT, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_GLOBAL\_ADDRESSES\_MASK, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_GLOBAL\_ADDRESSES\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_GLOBAL\_ADDRESSES\_SIZE, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_GLOBAL\_ADDRESSES\_WORD\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_GLOBAL\_ADDRESSES\_WORD\_SELECT, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_LINK\_LOCAL\_ADDRESS\_MASK, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_LINK\_LOCAL\_ADDRESS\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_LINK\_LOCAL\_ADDRESS\_SIZE, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_LINK\_LOCAL\_ADDRESS\_WORD\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_LINK\_LOCAL\_ADDRESS\_WORD\_SELECT, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_NEXT\_PROTOCOL\_OFFSET, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_NEXT\_PROTOCOL\_SIZE, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_NEXT\_PROTOCOL\_WORD\_OFFSET, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_NEXT\_PROTOCOL\_WORD\_SELECT, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_OPTIONS\_EXIST\_MASK, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_OPTIONS\_EXIST\_OFFSET, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_OPTIONS\_EXIST\_SIZE, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_OPTIONS\_EXIST\_WORD\_OFFSET, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_OPTIONS\_EXIST\_WORD\_SELECT, 333  
 EZDP\_DECODE\_IPV6\_RESULT\_RESERVED\_15\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_RESERVED\_15\_SIZE, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_RESERVED\_56\_63\_OFFSET, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_RESERVED\_56\_63\_SIZE, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_RESERVED9\_11\_OFFSET, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_RESERVED9\_11\_SIZE, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_SIP\_DIP\_HASH\_OFFSET, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_SIP\_DIP\_HASH\_SIZE, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_SIP\_DIP\_HASH\_WORD\_OFFSET, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_SIP\_DIP\_HASH\_WORD\_SELECT, 335  
 EZDP\_DECODE\_IPV6\_RESULT\_SITE\_LOCAL\_ADDRESS\_MASK, 334  
 EZDP\_DECODE\_IPV6\_RESULT\_SITE\_LOCAL\_ADDRESS\_OFFSET, 334



EZDP\_DECODE\_IPV6\_RESULT\_SITE\_LOCAL\_ADDRESS\_SIZE, 334

EZDP\_DECODE\_IPV6\_RESULT\_SITE\_LOCAL\_ADDRESS\_WORD\_OFFSET, 334

EZDP\_DECODE\_IPV6\_RESULT\_SITE\_LOCAL\_ADDRESS\_WORD\_SELECT, 334

EZDP\_DECODE\_IPV6\_RESULT\_WORD\_COUNT, 335

EZDP\_DECODE\_IPV6\_RETVAL\_CONTROL\_OF\_FSET, 332

EZDP\_DECODE\_IPV6\_RETVAL\_CONTROL\_SIZE, 332

EZDP\_DECODE\_IPV6\_RETVAL\_ERROR\_CODE\_S\_OFFSET, 333

EZDP\_DECODE\_IPV6\_RETVAL\_ERROR\_CODE\_S\_SIZE, 333

EZDP\_DECODE\_IPV6\_RETVAL\_GLOBAL\_ADDRESSES\_MASK, 333

EZDP\_DECODE\_IPV6\_RETVAL\_GLOBAL\_ADDRESSES\_OFFSET, 333

EZDP\_DECODE\_IPV6\_RETVAL\_GLOBAL\_ADDRESSES\_SIZE, 333

EZDP\_DECODE\_IPV6\_RETVAL\_LINK\_LOCAL\_ADDRESS\_MASK, 333

EZDP\_DECODE\_IPV6\_RETVAL\_LINK\_LOCAL\_ADDRESS\_OFFSET, 333

EZDP\_DECODE\_IPV6\_RETVAL\_LINK\_LOCAL\_ADDRESS\_SIZE, 333

EZDP\_DECODE\_IPV6\_RETVAL\_OPTIONS\_EXIST\_MASK, 333

EZDP\_DECODE\_IPV6\_RETVAL\_OPTIONS\_EXIST\_OFFSET, 332

EZDP\_DECODE\_IPV6\_RETVAL\_OPTIONS\_EXIST\_SIZE, 332

EZDP\_DECODE\_IPV6\_RETVAL\_RESERVED\_15\_OFFSET, 333

EZDP\_DECODE\_IPV6\_RETVAL\_RESERVED\_15\_SIZE, 333

EZDP\_DECODE\_IPV6\_RETVAL\_RESERVED9\_11\_OFFSET, 333

EZDP\_DECODE\_IPV6\_RETVAL\_RESERVED9\_11\_SIZE, 333

EZDP\_DECODE\_IPV6\_RETVAL\_SITE\_LOCAL\_ADDRESS\_MASK, 333

EZDP\_DECODE\_IPV6\_RETVAL\_SITE\_LOCAL\_ADDRESS\_OFFSET, 333

EZDP\_DECODE\_IPV6\_RETVAL\_SITE\_LOCAL\_ADDRESS\_SIZE, 333

ezdp\_decode\_ipv6\_retval\_t, 344

EZDP\_DECODE\_MAC\_CONTROL\_IPV4\_MULTICAST\_MASK, 322

EZDP\_DECODE\_MAC\_CONTROL\_IPV4\_MULTICAST\_OFFSET, 322

EZDP\_DECODE\_MAC\_CONTROL\_IPV4\_MULTICAST\_SIZE, 322

EZDP\_DECODE\_MAC\_CONTROL\_IPV6\_MULTICAST\_MASK, 322

EZDP\_DECODE\_MAC\_CONTROL\_IPV6\_MULTICAST\_OFFSET, 322

EZDP\_DECODE\_MAC\_CONTROL\_IPV6\_MULTI  
CAST\_SIZE, 322

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_0X\_MASK, 321

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_0X\_OFFSET, 321

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_0X\_SIZE, 321

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_1X\_MASK, 321

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_1X\_OFFSET, 321

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_1X\_SIZE, 321

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_2X\_MASK, 322

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_2X\_OFFSET, 322

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_LSB\_2X\_SIZE, 322

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_OTHER\_MASK, 322

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_OTHER\_OFFSET, 322

EZDP\_DECODE\_MAC\_CONTROL\_MAC\_CONT  
ROL\_OTHER\_SIZE, 322

EZDP\_DECODE\_MAC\_CONTROL\_MY\_MAC\_M  
ASK, 321

EZDP\_DECODE\_MAC\_CONTROL\_MY\_MAC\_O  
FFSET, 321

EZDP\_DECODE\_MAC\_CONTROL\_MY\_MAC\_SI  
ZE, 321

EZDP\_DECODE\_MAC\_CONTROL\_RESERVED1  
3\_15\_OFFSET, 323

EZDP\_DECODE\_MAC\_CONTROL\_RESERVED1  
3\_15\_SIZE, 323

EZDP\_DECODE\_MAC\_CONTROL\_SMAC\_EQU  
ALS\_DMAL\_MASK, 323

EZDP\_DECODE\_MAC\_CONTROL\_SMAC\_EQU  
ALS\_DMAL\_OFFSET, 323

EZDP\_DECODE\_MAC\_CONTROL\_SMAC\_EQU  
ALS\_DMAL\_SIZE, 323

ezdp\_decode\_mac\_control\_t, 344

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG0\_MASK, 322

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG0\_OFFSET, 322

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG0\_SIZE, 322

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG1\_MASK, 322

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG1\_OFFSET, 322

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG1\_SIZE, 322

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG2\_MASK, 323

EZDP\_DECODE\_MAC\_CONTROL\_USER\_CONF  
IG2\_OFFSET, 322

EZDP_DECODE_MAC_CONTROL_USER_CONF IG2_SIZE, 322	EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV4 _SIZE, 324
EZDP_DECODE_MAC_CONTROL_USER_CONF IG3_MASK, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV6 _MASK, 325
EZDP_DECODE_MAC_CONTROL_USER_CONF IG3_OFFSET, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV6 _OFFSET, 324
EZDP_DECODE_MAC_CONTROL_USER_CONF IG3_SIZE, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV6 _SIZE, 324
EZDP_DECODE_MAC_CONTROL_VRRP_MAC_ MASK, 322	EZDP_DECODE_MAC_PROTOCOL_TYPE_LEN GTH_MASK, 325
EZDP_DECODE_MAC_CONTROL_VRRP_MAC_ OFFSET, 322	EZDP_DECODE_MAC_PROTOCOL_TYPE_LEN GTH_OFFSET, 325
EZDP_DECODE_MAC_CONTROL_VRRP_MAC_ SIZE, 322	EZDP_DECODE_MAC_PROTOCOL_TYPE_LEN GTH_SIZE, 325
EZDP_DECODE_MAC_ERRORS_DECODE_ERR OR_MASK, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_MPL S_MULTICAST_MASK, 324
EZDP_DECODE_MAC_ERRORS_DECODE_ERR OR_OFFSET, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_MPL S_MULTICAST_OFFSET, 324
EZDP_DECODE_MAC_ERRORS_DECODE_ERR OR_SIZE, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_MPL S_MULTICAST_SIZE, 324
EZDP_DECODE_MAC_ERRORS_DMACE_IS_ZER O_MASK, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_MPL S_UNICAST_MASK, 324
EZDP_DECODE_MAC_ERRORS_DMACE_IS_ZER O_OFFSET, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_MPL S_UNICAST_OFFSET, 324
EZDP_DECODE_MAC_ERRORS_DMACE_IS_ZER O_SIZE, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_MPL S_UNICAST_SIZE, 324
EZDP_DECODE_MAC_ERRORS_IP_VERSION_ MISMATCH_IN_PPPOE_MASK, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPO E_DISCOVERY_MASK, 325
EZDP_DECODE_MAC_ERRORS_IP_VERSION_ MISMATCH_IN_PPPOE_OFFSET, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPO E_DISCOVERY_OFFSET, 325
EZDP_DECODE_MAC_ERRORS_IP_VERSION_ MISMATCH_IN_PPPOE_SIZE, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPO E_DISCOVERY_SIZE, 325
EZDP_DECODE_MAC_ERRORS_RESERVED5_7 _OFFSET, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPO E_SESSION_MASK, 325
EZDP_DECODE_MAC_ERRORS_RESERVED5_7 _SIZE, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPO E_SESSION_OFFSET, 325
EZDP_DECODE_MAC_ERRORS_SMACE_IS_NOT _UNICAST_MASK, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_PPPO E_SESSION_SIZE, 325
EZDP_DECODE_MAC_ERRORS_SMACE_IS_NOT _UNICAST_OFFSET, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_RES ERVED_15_OFFSET, 326
EZDP_DECODE_MAC_ERRORS_SMACE_IS_NOT _UNICAST_SIZE, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_RES ERVED_15_SIZE, 326
EZDP_DECODE_MAC_ERRORS_SMACE_IS_ZER O_MASK, 323	ezdp_decode_mac_protocol_type_t, 344
EZDP_DECODE_MAC_ERRORS_SMACE_IS_ZER O_OFFSET, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN0_MASK, 324
EZDP_DECODE_MAC_ERRORS_SMACE_IS_ZER O_SIZE, 323	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN0_OFFSET, 324
ezdp_decode_mac_errors_t, 344	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN0_SIZE, 324
EZDP_DECODE_MAC_PROTOCOL_TYPE_ARP _MASK, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN1_MASK, 324
EZDP_DECODE_MAC_PROTOCOL_TYPE_ARP _OFFSET, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN1_OFFSET, 324
EZDP_DECODE_MAC_PROTOCOL_TYPE_ARP _SIZE, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN1_SIZE, 324
EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV4 _MASK, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN2_MASK, 326
EZDP_DECODE_MAC_PROTOCOL_TYPE_IPV4 _OFFSET, 324	EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN2_OFFSET, 326

EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG_VLAN2_SIZE, 325	EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPO E_MASK, 336
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG0_MASK, 325	EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPO E_OFFSET, 336
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG0_OFFSET, 325	EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPO E_SIZE, 336
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG0_SIZE, 325	EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPO E_WORD_OFFSET, 336
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG1_MASK, 325	EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPO E_WORD_SELECT, 336
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG1_OFFSET, 325	EZDP_DECODE_MAC_RESULT_LAST_TAG_PR OTOCOL_TYPE_OFFSET, 337
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG1_SIZE, 325	EZDP_DECODE_MAC_RESULT_LAST_TAG_PR OTOCOL_TYPE_SIZE, 337
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG2_MASK, 325	EZDP_DECODE_MAC_RESULT_LAST_TAG_PR OTOCOL_TYPE_WORD_OFFSET, 337
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG2_OFFSET, 325	EZDP_DECODE_MAC_RESULT_LAST_TAG_PR OTOCOL_TYPE_WORD_SELECT, 337
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG2_SIZE, 325	EZDP_DECODE_MAC_RESULT_LAYER2_SIZE _OFFSET, 337
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG3_MASK, 325	EZDP_DECODE_MAC_RESULT_LAYER2_SIZE _SIZE, 337
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG3_OFFSET, 325	EZDP_DECODE_MAC_RESULT_LAYER2_SIZE _WORD_OFFSET, 337
EZDP_DECODE_MAC_PROTOCOL_TYPE_USE R_CONFIG3_SIZE, 325	EZDP_DECODE_MAC_RESULT_LAYER2_SIZE _WORD_SELECT, 337
EZDP_DECODE_MAC_RESULT_CONTROL_OF FSET, 336	EZDP_DECODE_MAC_RESULT_NUMBER_OF_ TAGS_OFFSET, 336
EZDP_DECODE_MAC_RESULT_CONTROL_SIZ E, 336	EZDP_DECODE_MAC_RESULT_NUMBER_OF_ TAGS_SIZE, 336
EZDP_DECODE_MAC_RESULT_CONTROL_WO RD_OFFSET, 336	EZDP_DECODE_MAC_RESULT_NUMBER_OF_ TAGS_WORD_OFFSET, 336
EZDP_DECODE_MAC_RESULT_CONTROL_WO RD_SELECT, 336	EZDP_DECODE_MAC_RESULT_NUMBER_OF_ TAGS_WORD_SELECT, 336
EZDP_DECODE_MAC_RESULT_DA_SA_HASH _OFFSET, 337	EZDP_DECODE_MAC_RESULT_RESERVED_31 _OFFSET, 336
EZDP_DECODE_MAC_RESULT_DA_SA_HASH _SIZE, 337	EZDP_DECODE_MAC_RESULT_RESERVED_31 _SIZE, 336
EZDP_DECODE_MAC_RESULT_DA_SA_HASH _WORD_OFFSET, 337	EZDP_DECODE_MAC_RESULT_RESERVED120 _127_OFFSET, 337
EZDP_DECODE_MAC_RESULT_DA_SA_HASH _WORD_SELECT, 337	EZDP_DECODE_MAC_RESULT_RESERVED120 _127_SIZE, 337
EZDP_DECODE_MAC_RESULT_ERROR_CODE S_OFFSET, 336	EZDP_DECODE_MAC_RESULT_RESERVED26_ 27_OFFSET, 336
EZDP_DECODE_MAC_RESULT_ERROR_CODE S_SIZE, 336	EZDP_DECODE_MAC_RESULT_RESERVED26_ 27_SIZE, 336
EZDP_DECODE_MAC_RESULT_ERROR_CODE S_WORD_OFFSET, 336	EZDP_DECODE_MAC_RESULT_TAG1_PROTO COL_TYPE_OFFSET, 337
EZDP_DECODE_MAC_RESULT_ERROR_CODE S_WORD_SELECT, 336	EZDP_DECODE_MAC_RESULT_TAG1_PROTO COL_TYPE_SIZE, 337
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPO E_MASK, 336	EZDP_DECODE_MAC_RESULT_TAG1_PROTO COL_TYPE_WORD_OFFSET, 337
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPO E_OFFSET, 336	EZDP_DECODE_MAC_RESULT_TAG1_PROTO COL_TYPE_WORD_SELECT, 337
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPO E_SIZE, 336	EZDP_DECODE_MAC_RESULT_TAG2_PROTO COL_TYPE_OFFSET, 337
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPO E_WORD_OFFSET, 336	EZDP_DECODE_MAC_RESULT_TAG2_PROTO COL_TYPE_SIZE, 337
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPO E_WORD_SELECT, 336	EZDP_DECODE_MAC_RESULT_TAG2_PROTO COL_TYPE_WORD_OFFSET, 337

EZDP_DECODE_MAC_RESULT_TAG2_PROTO COL_TYPE_WORD_SELECT, 337	EZDP_DECODE_MPLS_LABEL_RESULT_RESE RVED_LABEL_OFFSET, 342
EZDP_DECODE_MAC_RESULT_TAG3_PROTO COL_TYPE_OFFSET, 337	EZDP_DECODE_MPLS_LABEL_RESULT_RESE RVED_LABEL_SIZE, 342
EZDP_DECODE_MAC_RESULT_TAG3_PROTO COL_TYPE_SIZE, 337	EZDP_DECODE_MPLS_LABEL_RESULT_RESE RVED10_31_OFFSET, 343
EZDP_DECODE_MAC_RESULT_TAG3_PROTO COL_TYPE_WORD_OFFSET, 337	EZDP_DECODE_MPLS_LABEL_RESULT_RESE RVED10_31_SIZE, 343
EZDP_DECODE_MAC_RESULT_TAG3_PROTO COL_TYPE_WORD_SELECT, 337	EZDP_DECODE_MPLS_LABEL_RESULT_STOP _BIT_MASK, 343
EZDP_DECODE_MAC_RESULT_WORD_COUN T, 338	EZDP_DECODE_MPLS_LABEL_RESULT_STOP _BIT_OFFSET, 343
EZDP_DECODE_MAC_RETVAL_CONTROL_OF FSET, 335	EZDP_DECODE_MPLS_LABEL_RESULT_STOP _BIT_SIZE, 343
EZDP_DECODE_MAC_RETVAL_CONTROL_SIZ E, 335	ezdp_decode_mpls_label_result_t, 344
EZDP_DECODE_MAC_RETVAL_ERROR_CODE S_OFFSET, 335	EZDP_DECODE_MPLS_LABEL_RESULT_TTL_I S_ONE_MASK, 343
EZDP_DECODE_MAC_RETVAL_ERROR_CODE S_SIZE, 335	EZDP_DECODE_MPLS_LABEL_RESULT_TTL_I S_ONE_OFFSET, 343
EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPO E_MASK, 335	EZDP_DECODE_MPLS_LABEL_RESULT_TTL_I S_ONE_SIZE, 342
EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPO E_OFFSET, 335	EZDP_DECODE_MPLS_LABEL_RESULT_TTL_I S_ZERO_MASK, 342
EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPO E_SIZE, 335	EZDP_DECODE_MPLS_LABEL_RESULT_TTL_I S_ZERO_OFFSET, 342
EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPO E_MASK, 335	EZDP_DECODE_MPLS_LABEL_RESULT_TTL_I S_ZERO_SIZE, 342
EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPO E_OFFSET, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG0_MASK, 343
EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPO E_SIZE, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG0_OFFSET, 343
EZDP_DECODE_MAC_RETVAL_NUMBER_OF_ TAGS_OFFSET, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG0_SIZE, 343
EZDP_DECODE_MAC_RETVAL_NUMBER_OF_ TAGS_SIZE, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG1_MASK, 343
EZDP_DECODE_MAC_RETVAL_RESERVED_31 _OFFSET, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG1_OFFSET, 343
EZDP_DECODE_MAC_RETVAL_RESERVED_31 _SIZE, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG1_SIZE, 343
EZDP_DECODE_MAC_RETVAL_RESERVED26_ 27_OFFSET, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG2_MASK, 343
EZDP_DECODE_MAC_RETVAL_RESERVED26_ 27_SIZE, 335	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG2_OFFSET, 343
ezdp_decode_mac_retval_t, 344	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG2_SIZE, 343
EZDP_DECODE_MPLS_LABEL_RESULT_END_ OF_STACK_MASK, 342	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG3_MASK, 343
EZDP_DECODE_MPLS_LABEL_RESULT_END_ OF_STACK_OFFSET, 342	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG3_OFFSET, 343
EZDP_DECODE_MPLS_LABEL_RESULT_END_ OF_STACK_SIZE, 342	EZDP_DECODE_MPLS_LABEL_RESULT_USER _CONFIG3_SIZE, 343
EZDP_DECODE_MPLS_LABEL_RESULT_EXCE PTION_BIT_MASK, 343	EZDP_DECODE_MPLS_LABEL_RESULT_END_ OF_STACK_MASK, 341
EZDP_DECODE_MPLS_LABEL_RESULT_EXCE PTION_BIT_OFFSET, 343	EZDP_DECODE_MPLS_LABEL_RESULT_END_ OF_STACK_OFFSET, 341
EZDP_DECODE_MPLS_LABEL_RESULT_EXCE PTION_BIT_SIZE, 343	EZDP_DECODE_MPLS_LABEL_RESULT_END_ OF_STACK_SIZE, 341
EZDP_DECODE_MPLS_LABEL_RESULT_RESE RVED_LABEL_MASK, 342	EZDP_DECODE_MPLS_LABEL_RESULT_EXCE PTION_BIT_MASK, 342

EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_EXCEPTION\_BIT\_OFFSET, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_EXCEPTION\_BIT\_SIZE, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED\_LABEL\_MASK, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED\_LABEL\_OFFSET, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED\_LABEL\_SIZE, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED10\_31\_OFFSET, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_RESERVED10\_31\_SIZE, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_STOP\_BIT\_MASK, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_STOP\_BIT\_OFFSET, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_STOP\_BIT\_SIZE, 342  
 ezdp\_decode\_mpls\_label\_retval\_t, 344  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ONE\_MASK, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ONE\_OFFSET, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ONE\_SIZE, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ZERO\_MASK, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ZERO\_OFFSET, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS\_ZERO\_SIZE, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG0\_MASK, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG0\_OFFSET, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG0\_SIZE, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG1\_MASK, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG1\_OFFSET, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG1\_SIZE, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG2\_MASK, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG2\_OFFSET, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG2\_SIZE, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG3\_MASK, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG3\_OFFSET, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_CONFIG3\_SIZE, 342  
 EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERROR\_MASK, 339

EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERROR\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERROR\_SIZE, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ONE\_MASK, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ONE\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ONE\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ZERO\_MASK, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ZERO\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS\_ZERO\_SIZE, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ONE\_MASK, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ONE\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ONE\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ZERO\_MASK, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ZERO\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS\_ZERO\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ONE\_MASK, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ONE\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ONE\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ZERO\_MASK, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ZERO\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS\_ZERO\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ONE\_MASK, 341  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ONE\_OFFSET, 341  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ONE\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ZERO\_MASK, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ZERO\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS\_ZERO\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LAST\_ENTRY\_IN\_STACK\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_LAST\_ENTRY\_IN\_STACK\_SIZE, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_RESERVED1\_7\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_RESERVED1\_7\_SIZE, 339

EZDP\_DECODE\_MPLS\_RESULT\_RESERVED10  
     \_15\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_RESERVED10  
     \_15\_SIZE, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_RESERVED20  
     \_23\_OFFSET, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_RESERVED20  
     \_23\_SIZE, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_RESERVED28  
     \_31\_OFFSET, 341  
 EZDP\_DECODE\_MPLS\_RESULT\_RESERVED28  
     \_31\_SIZE, 341  
 ezdp\_decode\_mpls\_result\_t, 344  
 EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ER  
     ROR\_MASK, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ER  
     ROR\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ER  
     ROR\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL  
     \_IS\_ONE\_MASK, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL  
     \_IS\_ONE\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL  
     \_IS\_ONE\_SIZE, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL  
     \_IS\_ZERO\_MASK, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL  
     \_IS\_ZERO\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL  
     \_IS\_ZERO\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL  
     \_IS\_ONE\_MASK, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL  
     \_IS\_ONE\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL  
     \_IS\_ONE\_SIZE, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL  
     \_IS\_ZERO\_MASK, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL  
     \_IS\_ZERO\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL  
     \_IS\_ZERO\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL  
     \_IS\_ONE\_MASK, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL  
     \_IS\_ONE\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL  
     \_IS\_ONE\_SIZE, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL  
     \_IS\_ZERO\_MASK, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL  
     \_IS\_ZERO\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL3\_TTL  
     \_IS\_ZERO\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL  
     \_IS\_ONE\_MASK, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL  
     \_IS\_ONE\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL  
     \_IS\_ZERO\_MASK, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL  
     \_IS\_ZERO\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LABEL4\_TTL  
     \_IS\_ZERO\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LAST\_ENTRY  
     \_IN\_STACK\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_LAST\_ENTRY  
     \_IN\_STACK\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED1\_  
     7\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED1\_  
     7\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED10  
     \_15\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED10  
     \_15\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED20  
     \_23\_OFFSET, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED20  
     \_23\_SIZE, 338  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED28  
     \_31\_OFFSET, 339  
 EZDP\_DECODE\_MPLS\_RETVAL\_RESERVED28  
     \_31\_SIZE, 339  
 ezdp\_decode\_mpls\_retval\_t, 344  
 EZDP\_DECODE\_TCP\_ERRORS\_DATA\_OFFSET  
     \_LT\_5\_MASK, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_DATA\_OFFSET  
     \_LT\_5\_OFFSET, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_DATA\_OFFSET  
     \_LT\_5\_SIZE, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_DECODE\_ERRO  
     R\_MASK, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_DECODE\_ERRO  
     R\_OFFSET, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_DECODE\_ERRO  
     R\_SIZE, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_RESERVED3\_7\_  
     OFFSET, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_RESERVED3\_7\_  
     SIZE, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_SYN\_AND\_FIN  
     \_EQ\_1\_MASK, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_SYN\_AND\_FIN  
     \_EQ\_1\_OFFSET, 326  
 EZDP\_DECODE\_TCP\_ERRORS\_SYN\_AND\_FIN  
     \_EQ\_1\_SIZE, 326  
 ezdp\_decode\_tcp\_errors\_t, 344  
 EZDP\_DECODE\_TCP\_RETVAL\_DATA\_OFFSET  
     \_OFFSET, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_DATA\_OFFSET  
     \_SIZE, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_ERROR\_CODES  
     \_OFFSET, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_ERROR\_CODES  
     \_SIZE, 326

EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIS  
     T\_MASK, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIS  
     T\_OFFSET, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIS  
     T\_SIZE, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED22\_2  
     3\_OFFSET, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED22\_2  
     3\_SIZE, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED24\_3  
     1\_OFFSET, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED24\_3  
     1\_SIZE, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED9\_15  
     \_OFFSET, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED9\_15  
     \_SIZE, 326  
 ezdp\_decode\_tcp\_retval\_t, 344  
 EZDP\_DECODE\_VERSION\_MAJOR, 316  
 EZDP\_DECODE\_VERSION\_MINOR, 316  
 ezdp\_decode\_eth\_type  
     ezdp\_decode.h, 299  
 ezdp\_decode\_eth\_type\_retval, 21  
     \_\_pad0\_\_, 21  
     arp, 22  
     eth\_8100, 22  
     eth\_88a8, 22  
     ipv4, 22  
     ipv6, 22  
     length, 22  
     mpls\_multicast, 22  
     mpls\_unicast, 22  
     other, 21  
     pppoe\_discovery, 22  
     pppoe\_session, 22  
     raw\_data, 21  
     user\_def0, 22  
     user\_def1, 22  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_MAS  
     K  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_OFFS  
     ET  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ARP\_SIZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_  
     MASK  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_  
     OFFSET  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_8100\_  
     SIZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8  
     \_MASK  
         ezdp\_decode\_defs.h, 329

EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8  
     \_OFFSET  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_ETH\_88A8  
     \_SIZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_MAS  
     K  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_OFFS  
     ET  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV4\_SIZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_MAS  
     K  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_OFFS  
     ET  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_IPV6\_SIZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_  
     MASK  
         ezdp\_decode\_defs.h, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_  
     OFFSET  
         ezdp\_decode\_defs.h, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_LENGTH\_S  
     IZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MU  
     LTICAST\_MASK  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MU  
     LTICAST\_OFFSET  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_MU  
     LTICAST\_SIZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNI  
     CAST\_MASK  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNI  
     CAST\_OFFSET  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_MPLS\_UNI  
     CAST\_SIZE  
         ezdp\_decode\_defs.h, 329  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_M  
     ASK  
         ezdp\_decode\_defs.h, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_O  
     FFSET  
         ezdp\_decode\_defs.h, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_OTHER\_SI  
     ZE  
         ezdp\_decode\_defs.h, 330  
 EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DI  
     SCOVERY\_MASK

ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DISCOVERY\_OFFSET  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_DISCOVERY\_SIZE  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_MASK  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_OFFSET  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_PPPOE\_SESSION\_SIZE  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_RESERVE\_D13\_31\_OFFSET  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_RESERVE\_D13\_31\_SIZE  
ezdp\_decode\_defs.h, 330  
ezdp\_decode\_eth\_type\_retval\_t  
ezdp\_decode\_defs.h, 344  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_MASK  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_OFFSET  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF0\_SIZE  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_MASK  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_OFFSET  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_ETH\_TYPE\_RETVAL\_USER\_DEF1\_SIZE  
ezdp\_decode\_defs.h, 330  
ezdp\_decode\_ip\_next\_protocol, 24  
gre, 24  
icmp\_igmp, 24  
ipv4, 24  
ipv6, 24  
mpls, 25  
other, 24  
raw\_data, 24  
tcp, 25  
udp, 25  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_MASK  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_OFFSET  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_GRE\_SIZE  
ezdp\_decode\_defs.h, 318

EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_IGMP\_MASK  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_IGMP\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_ICMP\_IGMP\_SIZE  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_MASK  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_OFFSET  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV4\_SIZE  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_MASK  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_IPV6\_SIZE  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_MASK  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_OFFSET  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_MPLS\_SIZE  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_MASK  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_OTHER\_SIZE  
ezdp\_decode\_defs.h, 319  
ezdp\_decode\_ip\_next\_protocol\_t  
ezdp\_decode\_defs.h, 344  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_MASK  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_OFFSET  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_TCP\_SIZE  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_MASK  
ezdp\_decode\_defs.h, 318  
EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_OFFSET  
SET



ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IP\_NEXT\_PROTOCOL\_UDP\_SIZE  
 ezdp\_decode\_defs.h, 318  
 ezdp\_decode\_ip\_protocol  
 ezdp\_decode.h, 298  
 ezdp\_decode\_ip\_protocol\_retval, 26  
 \_\_pad0\_\_, 26  
 ah\_prot, 26  
 def\_ip\_prot\_0, 27  
 def\_ip\_prot\_1, 27  
 def\_ip\_prot\_2, 27  
 def\_ip\_prot\_3, 27  
 esp\_prot, 27  
 gre, 27  
 icmp\_igmp, 27  
 ipv4, 27  
 ipv6, 27  
 mpls, 27  
 other, 26  
 raw\_data, 26  
 tcp, 27  
 udp, 27  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_AH\_PROTOCOL\_MASK  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_AH\_PROTOCOL\_OFFSET  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_AH\_PROTOCOL\_SIZE  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_0\_MASK  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_0\_OFFSET  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_0\_SIZE  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_1\_MASK  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_1\_OFFSET  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_1\_SIZE  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_2\_MASK  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_2\_OFFSET  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_2\_SIZE  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_3\_MASK  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_3\_OFFSET  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_DEF\_IP\_PROTOCOL\_3\_SIZE  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_GRE\_MASK  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_GRE\_OFFSET  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_GRE\_SIZE  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_ICMP\_IGMP\_MASK  
 ezdp\_decode\_defs.h, 328  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_ICMP\_IGMP\_OFFSET  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_ICMP\_IGMP\_SIZE  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV4\_MASK  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV4\_OFFSET  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV4\_SIZE  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV6\_MASK  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV6\_OFFSET  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_IPV6\_SIZE  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_MPLS\_MASK  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_MPLS\_OFFSET  
 ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_MPLS\_SIZE

ezdp\_decode\_defs.h, 327  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_OTHER\_MASK  
ezdp\_decode\_defs.h, 328  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_OTHER\_OFFSET  
ezdp\_decode\_defs.h, 328  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_OTHER\_SIZE  
ezdp\_decode\_defs.h, 328  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_RESERVED14\_31\_OFFSET  
ezdp\_decode\_defs.h, 329  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_RESERVED14\_31\_SIZE  
ezdp\_decode\_defs.h, 328  
ezdp\_decode\_ip\_protocol\_retval\_t  
ezdp\_decode\_defs.h, 344  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_TCP\_MASK  
ezdp\_decode\_defs.h, 327  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_TCP\_OFFSET  
ezdp\_decode\_defs.h, 327  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_TCP\_SIZE  
ezdp\_decode\_defs.h, 327  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_UDP\_MASK  
ezdp\_decode\_defs.h, 327  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_UDP\_OFFSET  
ezdp\_decode\_defs.h, 327  
EZDP\_DECODE\_IP\_PROTOCOL\_RETVAL\_UDP\_SIZE  
ezdp\_decode\_defs.h, 327  
ezdp\_decode\_ipv4  
ezdp\_decode.h, 296  
ezdp\_decode\_ipv4\_async  
ezdp\_decode.h, 296  
ezdp\_decode\_ipv4\_control, 29  
\_\_pad0\_\_, 30  
icmp, 30  
igmp, 29  
internetwork\_multicast\_range, 30  
link\_local\_multicast\_range, 30  
raw\_data, 29  
user\_config0, 29  
user\_config1, 29  
user\_config2, 29  
EZDP\_DECODE\_IPV4\_CONTROL\_ICMP\_MASK  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_ICMP\_OFFSET  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_ICMP\_SIZE  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_IGMP\_MASK  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_IGMP\_OFFSET  
ezdp\_decode\_defs.h, 316

EZDP\_DECODE\_IPV4\_CONTROL\_IGMP\_SIZE  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_MASK  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_OFFSET  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_SIZE  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_MASK  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_OFFSET  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_SIZE  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_RESERVED2\_OFFSET  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_RESERVED2\_SIZE  
ezdp\_decode\_defs.h, 316  
ezdp\_decode\_ipv4\_control\_t  
ezdp\_decode\_defs.h, 344  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG0\_MASK  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG0\_OFFSET  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG0\_SIZE  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_MASK  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_OFFSET  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG1\_SIZE  
ezdp\_decode\_defs.h, 316  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_MASK  
ezdp\_decode\_defs.h, 317  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_OFFSET  
ezdp\_decode\_defs.h, 317  
EZDP\_DECODE\_IPV4\_CONTROL\_USER\_CONFIG2\_SIZE  
ezdp\_decode\_defs.h, 316  
ezdp\_decode\_ipv4\_errors, 31  
\_\_pad0\_\_, 31  
checksum\_error, 32  
decode\_error, 31  
header\_length\_gt\_frame\_length, 32

header\_length\_lt\_5, 32  
 not\_ipv4\_version, 32  
 raw\_data, 31  
 sip\_equal\_dip, 31  
 sip\_is\_multicast, 32  
 sip\_is\_zero, 32  
 total\_length\_gt\_frame\_length, 32  
 EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_MASK  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_OFFSET  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_CHECKSUM\_ERROR\_SIZE  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_MASK  
     ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_OFFSET  
     ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_DECODE\_ERROR\_SIZE  
     ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_MASK  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_OFFSET  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_GT\_FRAME\_LENGTH\_SIZE  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_MASK  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_OFFSET  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_HEADER\_LENGTH\_LT\_5\_SIZE  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_MASK  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_OFFSET  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_NOT\_IPV4\_VERSION\_SIZE  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_RESERVED9\_15\_OFFSET  
     ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_RESERVED9\_15\_SIZE  
     ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_MASK  
     ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_OFFSET  
     ezdp\_decode\_defs.h, 318  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_EQUAL\_DIP\_SIZE  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_MASK  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_OFFSET  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_MULTICAST\_SIZE  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_MASK  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_OFFSET  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_SIP\_IS\_ZERO\_SIZE  
     ezdp\_decode\_defs.h, 317  
 ezdp\_decode\_ipv4\_errors\_t  
     ezdp\_decode\_defs.h, 344  
 EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_MASK  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_OFFSET  
     ezdp\_decode\_defs.h, 317  
 EZDP\_DECODE\_IPV4\_ERRORS\_TOTAL\_LENGTH\_GT\_FRAME\_LENGTH\_SIZE  
     ezdp\_decode\_defs.h, 317  
 ezdp\_decode\_ipv4\_result, 33  
     \_\_pad0\_\_, 33  
     \_\_pad1\_\_, 34  
     control, 33  
     error\_codes, 33  
     first\_fragment, 33  
     next\_protocol, 34  
     option\_exist, 34  
     raw\_data, 33  
     sip\_dip\_hash, 34  
     user\_config\_sip, 34  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_OFFSET  
     ezdp\_decode\_defs.h, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_SIZE  
     ezdp\_decode\_defs.h, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_WORD\_OFFSET  
     ezdp\_decode\_defs.h, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_CONTROL\_WORD\_SELECT  
     ezdp\_decode\_defs.h, 332  
 EZDP\_DECODE\_IPV4\_RESULT\_ERROR\_CODES\_OFFSET

ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV4_RESULT_ERROR_CODES_SIZE	EZDP_DECODE_IPV4_RESULT_RESERVED_56_63_SIZE
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV4_RESULT_ERROR_CODES_WORD_OFFSET	EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_OFFSET
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV4_RESULT_ERROR_CODES_WORD_SELECT	EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_SIZE
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_MASK	EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_WORD_OFFSET
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_OFFSET	EZDP_DECODE_IPV4_RESULT_SIP_DIP_HASH_WORD_SELECT
ezdp_decode_defs.h, 331	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_SIZE	EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_MASK
ezdp_decode_defs.h, 331	ezdp_decode_defs.h, 331
EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_WORD_OFFSET	EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_OFFSET
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 331
EZDP_DECODE_IPV4_RESULT_FIRST_FRAGMENT_WORD_SELECT	EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_SIZE
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 331
EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_OFFSET	EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_WORD_OFFSET
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 331
EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_SIZE	EZDP_DECODE_IPV4_RESULT_USER_CONFIG_SIP_WORD_SELECT
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 331
EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_WORD_OFFSET	EZDP_DECODE_IPV4_RESULT_WORD_COUNT
ezdp_decode_defs.h, 332	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV4_RESULT_NEXT_PROTOCOL_WORD_SELECT	ezdp_decode_ipv4_retval, 35
ezdp_decode_defs.h, 332	__pad0__, 35
EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_MASK	control, 35
ezdp_decode_defs.h, 331	error_codes, 35
EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_OFFSET	first_fragment, 35
ezdp_decode_defs.h, 331	option_exist, 36
EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_SIZE	raw_data, 35
ezdp_decode_defs.h, 331	user_config_sip, 35
EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_WORD_OFFSET	EZDP_DECODE_IPV4_RETVAL_CONTROL_OFFSET
ezdp_decode_defs.h, 331	ET
EZDP_DECODE_IPV4_RESULT_OPTION_EXIST_WORD_SELECT	ezdp_decode_defs.h, 331
ezdp_decode_defs.h, 331	EZDP_DECODE_IPV4_RETVAL_CONTROL_SIZE
EZDP_DECODE_IPV4_RESULT_RESERVED_2_6_OFFSET	ezdp_decode_defs.h, 331
ezdp_decode_defs.h, 331	EZDP_DECODE_IPV4_RETVAL_ERROR_CODES_OFFSET
EZDP_DECODE_IPV4_RESULT_RESERVED_2_6_SIZE	ezdp_decode_defs.h, 331
ezdp_decode_defs.h, 331	EZDP_DECODE_IPV4_RETVAL_ERROR_CODES_SIZE
EZDP_DECODE_IPV4_RESULT_RESERVED_56_63_OFFSET	ezdp_decode_defs.h, 331
ezdp_decode_defs.h, 331	EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_MASK
	ezdp_decode_defs.h, 331
	EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_OFFSET
	ezdp_decode_defs.h, 331
	EZDP_DECODE_IPV4_RETVAL_FIRST_FRAGMENT_SIZE
	NT_SIZE

ezdp\_decode\_defs.h, 331  
EZDP\_DECODE\_IPV4\_RETVAL\_OPTION\_EXIST\_MASK  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_IPV4\_RETVAL\_OPTION\_EXIST\_OFFSET  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_IPV4\_RETVAL\_OPTION\_EXIST\_SIZE  
ezdp\_decode\_defs.h, 330  
EZDP\_DECODE\_IPV4\_RETVAL\_RESERVED\_2\_6\_OFFSET  
ezdp\_decode\_defs.h, 331  
EZDP\_DECODE\_IPV4\_RETVAL\_RESERVED\_2\_6\_SIZE  
ezdp\_decode\_defs.h, 331  
ezdp\_decode\_ipv4\_retval\_t  
ezdp\_decode\_defs.h, 344  
EZDP\_DECODE\_IPV4\_RETVAL\_USER\_CONFIG\_S\_IP\_MASK  
ezdp\_decode\_defs.h, 331  
EZDP\_DECODE\_IPV4\_RETVAL\_USER\_CONFIG\_S\_IP\_OFFSET  
ezdp\_decode\_defs.h, 331  
EZDP\_DECODE\_IPV4\_RETVAL\_USER\_CONFIG\_S\_IP\_SIZE  
ezdp\_decode\_defs.h, 330  
ezdp\_decode\_ipv6  
ezdp\_decode.h, 296  
ezdp\_decode\_ipv6\_async  
ezdp\_decode.h, 297  
ezdp\_decode\_ipv6\_control, 37  
\_\_pad0\_\_, 37  
\_\_pad1\_\_, 37  
dip\_is\_multicast, 37  
dip\_is\_wellknown\_multicast, 37  
internetwork\_multicast\_range, 38  
link\_local\_multicast\_range, 38  
raw\_data, 37  
solicited\_node\_multicast\_range, 38  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_MULTICAST\_MASK  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_MULTICAST\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_MULTICAST\_SIZE  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_WELLKNOWN\_MULTICAST\_MASK  
ezdp\_decode\_defs.h, 320  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_WELLKNOWN\_MULTICAST\_OFFSET  
ezdp\_decode\_defs.h, 320  
EZDP\_DECODE\_IPV6\_CONTROL\_DIP\_IS\_WELLKNOWN\_MULTICAST\_SIZE  
ezdp\_decode\_defs.h, 320  
EZDP\_DECODE\_IPV6\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_MASK

ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_INTERNETWORK\_MULTICAST\_RANGE\_SIZE  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_MASK  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_LINK\_LOCAL\_MULTICAST\_RANGE\_SIZE  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_RESERVED\_3\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_RESERVED\_3\_SIZE  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_RESERVED7\_8\_OFFSET  
ezdp\_decode\_defs.h, 320  
EZDP\_DECODE\_IPV6\_CONTROL\_RESERVED7\_8\_SIZE  
ezdp\_decode\_defs.h, 320  
EZDP\_DECODE\_IPV6\_CONTROL\_SOLICITED\_NO\_DE\_MULTICAST\_RANGE\_MASK  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_SOLICITED\_NO\_DE\_MULTICAST\_RANGE\_OFFSET  
ezdp\_decode\_defs.h, 319  
EZDP\_DECODE\_IPV6\_CONTROL\_SOLICITED\_NO\_DE\_MULTICAST\_RANGE\_SIZE  
ezdp\_decode\_defs.h, 319  
ezdp\_decode\_ipv6\_control\_t  
ezdp\_decode\_defs.h, 344  
ezdp\_decode\_ipv6\_errors, 39  
\_\_pad0\_\_, 39  
decode\_error, 40  
dip\_is\_one, 40  
dip\_is\_zero, 40  
not\_ipv6\_version, 40  
payload\_gt\_frame\_length, 40  
payload\_missing, 39  
raw\_data, 39  
sip\_equal\_dip, 40  
sip\_is\_multicast, 39  
sip\_is\_one, 40  
sip\_is\_zero, 40  
EZDP\_DECODE\_IPV6\_ERRORS\_DECODE\_ERROR\_MASK  
ezdp\_decode\_defs.h, 321  
EZDP\_DECODE\_IPV6\_ERRORS\_DECODE\_ERROR\_OFFSET  
ezdp\_decode\_defs.h, 321  
EZDP\_DECODE\_IPV6\_ERRORS\_DECODE\_ERROR\_SIZE

ezdp_decode_defs.h, 321	ezdp_decode_defs.h, 321
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_MASK	EZDP_DECODE_IPV6_ERRORS_DIP_IS_MULTICAST_MASK
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 321
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_OFFSET	EZDP_DECODE_IPV6_ERRORS_DIP_IS_MULTICAST_OFFSET
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 321
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ONE_SIZE	EZDP_DECODE_IPV6_ERRORS_DIP_IS_MULTICAST_SIZE
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 321
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_MASK	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_OFFSET
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 320
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_OFFSET	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_SIZE
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 320
EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_SIZE	EZDP_DECODE_IPV6_ERRORS_DIP_IS_ZERO_SIZE
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 320
EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_MASK	EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_OFFSET
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 320
EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_OFFSET	EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_SIZE
ezdp_decode_defs.h, 320	ezdp_decode_defs.h, 320
EZDP_DECODE_IPV6_ERRORS_NOT_IPV6_VERSION_SIZE	ezdp_decode_defs.h, 320
ezdp_decode_defs.h, 320	ezdp_decode_ipv6_errors_t
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_GT_FRAME_LENGTH_MASK	ezdp_decode_defs.h, 344
ezdp_decode_defs.h, 320	ezdp_decode_ipv6_result, 41
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_GT_FRAME_LENGTH_OFFSET	__pad0__, 41
ezdp_decode_defs.h, 320	__pad1__, 42
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_GT_FRAME_LENGTH_SIZE	__pad2__, 42
ezdp_decode_defs.h, 320	control, 42
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_MISSING_MASK	error_codes, 41
ezdp_decode_defs.h, 321	global_addresses, 41
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_MISSING_OFFSET	link_local_address, 42
ezdp_decode_defs.h, 321	next_protocol, 42
EZDP_DECODE_IPV6_ERRORS_PAYLOAD_MISSING_SIZE	options_exist, 42
ezdp_decode_defs.h, 321	raw_data, 41
EZDP_DECODE_IPV6_ERRORS_RESERVED10_15_OFFSET	sip_dip_hash, 42
ezdp_decode_defs.h, 321	site_local_address, 41
EZDP_DECODE_IPV6_ERRORS_RESERVED10_15_SIZE	EZDP_DECODE_IPV6_RESULT_CONTROL_OFFSET
ezdp_decode_defs.h, 321	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_ERRORS_SIP_EQUAL_DIP_MASK	EZDP_DECODE_IPV6_RESULT_CONTROL_SIZE
ezdp_decode_defs.h, 321	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_ERRORS_SIP_EQUAL_DIP_OFFSET	EZDP_DECODE_IPV6_RESULT_CONTROL_WORD_OFFSET
ezdp_decode_defs.h, 321	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_ERRORS_SIP_EQUAL_DIP_SIZE	EZDP_DECODE_IPV6_RESULT_CONTROL_WORD_SELECT
	ezdp_decode_defs.h, 333
	EZDP_DECODE_IPV6_RESULT_ERROR_CODES_OFFSET
	ezdp_decode_defs.h, 334
	EZDP_DECODE_IPV6_RESULT_ERROR_CODES_SIZE
	ezdp_decode_defs.h, 334

EZDP_DECODE_IPV6_RESULT_ERROR_CODES_WORD_OFFSET	EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_WORD_SELECT
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RESULT_ERROR_CODES_WORD_SELECT	EZDP_DECODE_IPV6_RESULT_RESERVED_15_OFFSET
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_MASK	EZDP_DECODE_IPV6_RESULT_RESERVED_15_SIZE
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_OFFSET	EZDP_DECODE_IPV6_RESULT_RESERVED_56_63_OFFSET
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 335
EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_SIZE	EZDP_DECODE_IPV6_RESULT_RESERVED_56_63_SIZE
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 335
EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_OFFSET	EZDP_DECODE_IPV6_RESULT_RESERVED9_11_OFFSET
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_GLOBAL_ADDRESSES_WORD_SELECT	EZDP_DECODE_IPV6_RESULT_RESERVED9_11_SIZE
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_MASK	EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_OFFSET
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 335
EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_OFFSET	EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_SIZE
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_SIZE	EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_WORD_OFFSET
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 335
EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_WORD_OFFSET	EZDP_DECODE_IPV6_RESULT_SIP_DIP_HASH_WORD_SELECT
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 335
EZDP_DECODE_IPV6_RESULT_LINK_LOCAL_ADDRESS_WORD_SELECT	EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_MASK
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_OFFSET	EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_OFFSET
ezdp_decode_defs.h, 335	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_SIZE	EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_SIZE
ezdp_decode_defs.h, 335	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_WORD_OFFSET	EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_OFFSET
ezdp_decode_defs.h, 335	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_NEXT_PROTOCOL_WORD_SELECT	EZDP_DECODE_IPV6_RESULT_SITE_LOCAL_ADDRESS_WORD_SELECT
ezdp_decode_defs.h, 335	ezdp_decode_defs.h, 334
EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_MASK	EZDP_DECODE_IPV6_RESULT_WORD_COUNT
ezdp_decode_defs.h, 334	ezdp_decode_defs.h, 335
EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_OFFSET	ezdp_decode_ipv6_retval, 43
ezdp_decode_defs.h, 333	__pad0__, 43
EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_SIZE	__pad1__, 44
ezdp_decode_defs.h, 333	control, 44
EZDP_DECODE_IPV6_RESULT_OPTIONS_EXIST_WORD_OFFSET	error_codes, 43
ezdp_decode_defs.h, 333	global_addresses, 43
	link_local_address, 43
	options_exist, 44
	raw_data, 43
	site_local_address, 43

EZDP_DECODE_IPV6_RETVAL_CONTROL_OFFSET	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV6_RETVAL_CONTROL_SIZE	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV6_RETVAL_ERROR_CODES_OFFSET	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_ERROR_CODES_SIZE	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_GLOBAL_ADDRESSES_MASK	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_GLOBAL_ADDRESSES_OFFSET	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_GLOBAL_ADDRESSES_SIZE	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_LINK_LOCAL_ADDRESS_MASK	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_LINK_LOCAL_ADDRESS_OFFSET	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_LINK_LOCAL_ADDRESS_SIZE	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_OPTIONS_EXISTS_MASK	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_OPTIONS_EXISTS_OFFSET	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV6_RETVAL_OPTIONS_EXISTS_SIZE	ezdp_decode_defs.h, 332
EZDP_DECODE_IPV6_RETVAL_RESERVED_15_OFFSET	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_RESERVED_15_SIZE	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_RESERVED9_11_OFFSET	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_RESERVED9_11_SIZE	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_SITE_LOCAL_ADDRESS_MASK	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_SITE_LOCAL_ADDRESS_OFFSET	ezdp_decode_defs.h, 333
EZDP_DECODE_IPV6_RETVAL_SITE_LOCAL_ADDRESS_SIZE	ezdp_decode_defs.h, 333
ezdp_decode_ipv6_retval_t	ezdp_decode_defs.h, 344
ezdp_decode_mac	ezdp_decode.h, 295
ezdp_decode_mac_async	ezdp_decode.h, 295
ezdp_decode_mac_control	__pad0__, 45
ipv4_multicast	46
ipv6_multicast	46
mac_control_lsb_0x	46
mac_control_lsb_1x	46
mac_control_lsb_2x	46
mac_control_other	46
my_mac	47
raw_data	45
smac_equals_dmac	46
user_config0	46
user_config1	46
user_config2	46
user_config3	46
vrrp_mac	46
EZDP_DECODE_MAC_CONTROL_IPV4_MULTICAST_MASK	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_IPV4_MULTICAST_OFFSET	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_IPV4_MULTICAST_SIZE	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_IPV6_MULTICAST_MASK	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_IPV6_MULTICAST_OFFSET	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_IPV6_MULTICAST_SIZE	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_L_LSB_0X_MASK	ezdp_decode_defs.h, 321
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_L_LSB_0X_OFFSET	ezdp_decode_defs.h, 321
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_L_LSB_0X_SIZE	ezdp_decode_defs.h, 321
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_L_LSB_1X_MASK	ezdp_decode_defs.h, 321
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_L_LSB_1X_OFFSET	ezdp_decode_defs.h, 321
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_L_LSB_1X_SIZE	ezdp_decode_defs.h, 321
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_L_LSB_2X_MASK	ezdp_decode_defs.h, 322



EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_2X_OFFSET	EZDP_DECODE_MAC_CONTROL_USER_CONFIG_2_MASK
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_LSB_2X_SIZE	EZDP_DECODE_MAC_CONTROL_USER_CONFIG_2_OFFSET
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_OTHER_MASK	EZDP_DECODE_MAC_CONTROL_USER_CONFIG_2_SIZE
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_OTHER_OFFSET	EZDP_DECODE_MAC_CONTROL_USER_CONFIG_3_MASK
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_MAC_CONTROL_OTHER_SIZE	EZDP_DECODE_MAC_CONTROL_USER_CONFIG_3_OFFSET
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_MY_MAC_MASK	EZDP_DECODE_MAC_CONTROL_USER_CONFIG_3_SIZE
ezdp_decode_defs.h, 321	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_MY_MAC_OFFSET	EZDP_DECODE_MAC_CONTROL_VRRP_MAC_MASK
ezdp_decode_defs.h, 321	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_MY_MAC_SIZE	EZDP_DECODE_MAC_CONTROL_VRRP_MAC_OFFSET
ezdp_decode_defs.h, 321	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_RESERVED13_15_OFFSET	EZDP_DECODE_MAC_CONTROL_VRRP_MAC_SIZE
ezdp_decode_defs.h, 323	ezdp_decode_defs.h, 322
EZDP_DECODE_MAC_CONTROL_RESERVED13_15_SIZE	ezdp_decode_mac_errors, 48
ezdp_decode_defs.h, 323	__pad0__, 48
EZDP_DECODE_MAC_CONTROL_SMAC_EQUALS_DMAC_MASK	decode_error, 48
ezdp_decode_defs.h, 323	dmac_is_zero, 48
EZDP_DECODE_MAC_CONTROL_SMAC_EQUALS_DMAC_OFFSET	ip_version_mismatch_in_pppoe, 48
ezdp_decode_defs.h, 323	raw_data, 48
EZDP_DECODE_MAC_CONTROL_SMAC_EQUALS_DMAC_SIZE	smac_is_not_unicast, 49
ezdp_decode_defs.h, 323	smac_is_zero, 48
ezdp_decode_mac_control_t	EZDP_DECODE_MAC_ERRORS_DECODE_ERROR_MASK
ezdp_decode_defs.h, 344	ezdp_decode_defs.h, 324
EZDP_DECODE_MAC_CONTROL_USER_CONFIG_0_MASK	EZDP_DECODE_MAC_ERRORS_DECODE_ERROR_OFFSET
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_USER_CONFIG_0_OFFSET	EZDP_DECODE_MAC_ERRORS_DECODE_ERROR_SIZE
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_USER_CONFIG_0_SIZE	EZDP_DECODE_MAC_ERRORS_DMAC_IS_ZERO_MASK
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_USER_CONFIG_1_MASK	EZDP_DECODE_MAC_ERRORS_DMAC_IS_ZERO_OFFSET
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_USER_CONFIG_1_OFFSET	EZDP_DECODE_MAC_ERRORS_DMAC_IS_ZERO_SIZE
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
EZDP_DECODE_MAC_CONTROL_USER_CONFIG_1_SIZE	EZDP_DECODE_MAC_ERRORS_IP_VERSION_MISMATCH_IN_PPPOE_MASK
ezdp_decode_defs.h, 322	ezdp_decode_defs.h, 323
	EZDP_DECODE_MAC_ERRORS_IP_VERSION_MISMATCH_IN_PPPOE_OFFSET
	ezdp_decode_defs.h, 323

EZDP\_DECODE\_MAC\_ERRORS\_IP\_VERSION\_MI  
 SMATCH\_IN\_PPPOE\_SIZE  
 ezdp\_decode\_defs.h, 323  
 EZDP\_DECODE\_MAC\_ERRORS\_RESERVED5\_7\_  
 OFFSET  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_ERRORS\_RESERVED5\_7\_S  
 IZE  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_NOT\_  
 UNICAST\_MASK  
 ezdp\_decode\_defs.h, 323  
 EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_NOT\_  
 UNICAST\_OFFSET  
 ezdp\_decode\_defs.h, 323  
 EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_NOT\_  
 UNICAST\_SIZE  
 ezdp\_decode\_defs.h, 323  
 EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_ZERO  
 \_MASK  
 ezdp\_decode\_defs.h, 323  
 EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_ZERO  
 \_OFFSET  
 ezdp\_decode\_defs.h, 323  
 EZDP\_DECODE\_MAC\_ERRORS\_SMAC\_IS\_ZERO  
 \_SIZE  
 ezdp\_decode\_defs.h, 323  
 ezdp\_decode\_mac\_errors\_t  
 ezdp\_decode\_defs.h, 344  
 ezdp\_decode\_mac\_protocol\_type, 50  
 \_\_pad0\_\_, 50  
 arp, 51  
 ipv4, 52  
 ipv6, 51  
 length, 51  
 mpls\_multicast, 51  
 mpls\_unicast, 51  
 pppoe\_discovery, 51  
 pppoe\_session, 51  
 raw\_data, 50  
 user\_config\_vlan0, 52  
 user\_config\_vlan1, 52  
 user\_config\_vlan2, 51  
 user\_config0, 51  
 user\_config1, 51  
 user\_config2, 51  
 user\_config3, 51  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_ARP\_M  
 ASK  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_ARP\_O  
 FFSET  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_ARP\_SI  
 ZE  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV4\_  
 MASK  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV4\_O  
 FFSET  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV4\_S  
 IZE  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV6\_  
 MASK  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV6\_O  
 FFSET  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_IPV6\_S  
 IZE  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE LENGT  
 H\_MASK  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE LENGT  
 H\_OFFSET  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE LENGT  
 H\_SIZE  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_  
 MULTICAST\_MASK  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_  
 MULTICAST\_OFFSET  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_  
 MULTICAST\_SIZE  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_  
 UNICAST\_MASK  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_  
 UNICAST\_OFFSET  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_MPLS\_  
 UNICAST\_SIZE  
 ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE  
 \_DISCOVERY\_MASK  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE  
 \_DISCOVERY\_OFFSET  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE  
 \_DISCOVERY\_SIZE  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE  
 \_SESSION\_MASK  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE  
 \_SESSION\_OFFSET  
 ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_PPPOE  
 \_SESSION\_SIZE  
 ezdp\_decode\_defs.h, 325

EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_RESER  
VED\_15\_OFFSET  
ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_RESER  
VED\_15\_SIZE  
ezdp\_decode\_defs.h, 326  
 ezdp\_decode\_mac\_protocol\_type\_t  
ezdp\_decode\_defs.h, 344  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN0\_MASK  
ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN0\_OFFSET  
ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN0\_SIZE  
ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN1\_MASK  
ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN1\_OFFSET  
ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN1\_SIZE  
ezdp\_decode\_defs.h, 324  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN2\_MASK  
ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN2\_OFFSET  
ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG\_VLAN2\_SIZE  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG0\_MASK  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG0\_OFFSET  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG0\_SIZE  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG1\_MASK  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG1\_OFFSET  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG1\_SIZE  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG2\_MASK  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG2\_OFFSET  
ezdp\_decode\_defs.h, 325

EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG2\_SIZE  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG3\_MASK  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG3\_OFFSET  
ezdp\_decode\_defs.h, 325  
 EZDP\_DECODE\_MAC\_PROTOCOL\_TYPE\_USER\_  
CONFIG3\_SIZE  
ezdp\_decode\_defs.h, 325  
 ezdp\_decode\_mac\_result, 53  
 \_\_pad0\_\_, 53  
 \_\_pad1\_\_, 53  
 \_\_pad2\_\_, 54  
 control, 54  
 da\_sa\_hash, 54  
 error\_codes, 54  
 ipv4\_in\_pppoe, 54  
 ipv6\_in\_pppoe, 53  
 last\_tag\_protocol\_type, 54  
 layer2\_size, 54  
 number\_of\_tags, 53  
 raw\_data, 53  
 tag1\_protocol\_type, 54  
 tag2\_protocol\_type, 54  
 tag3\_protocol\_type, 54  
 EZDP\_DECODE\_MAC\_RESULT\_CONTROL\_OFFS  
ET  
ezdp\_decode\_defs.h, 336  
 EZDP\_DECODE\_MAC\_RESULT\_CONTROL\_SIZE  
ezdp\_decode\_defs.h, 336  
 EZDP\_DECODE\_MAC\_RESULT\_CONTROL\_WOR  
D\_OFFSET  
ezdp\_decode\_defs.h, 336  
 EZDP\_DECODE\_MAC\_RESULT\_CONTROL\_WOR  
D\_SELECT  
ezdp\_decode\_defs.h, 336  
 EZDP\_DECODE\_MAC\_RESULT\_DA\_SA\_HASH\_O  
FFSET  
ezdp\_decode\_defs.h, 337  
 EZDP\_DECODE\_MAC\_RESULT\_DA\_SA\_HASH\_SI  
ZE  
ezdp\_decode\_defs.h, 337  
 EZDP\_DECODE\_MAC\_RESULT\_DA\_SA\_HASH\_W  
ORD\_OFFSET  
ezdp\_decode\_defs.h, 337  
 EZDP\_DECODE\_MAC\_RESULT\_DA\_SA\_HASH\_W  
ORD\_SELECT  
ezdp\_decode\_defs.h, 337  
 EZDP\_DECODE\_MAC\_RESULT\_ERROR\_CODES\_  
OFFSET  
ezdp\_decode\_defs.h, 336  
 EZDP\_DECODE\_MAC\_RESULT\_ERROR\_CODES\_  
SIZE  
ezdp\_decode\_defs.h, 336  
 EZDP\_DECODE\_MAC\_RESULT\_ERROR\_CODES\_  
WORD\_OFFSET  
ezdp\_decode\_defs.h, 336

EZDP_DECODE_MAC_RESULT_ERROR_CODES_WORD_SELECT ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_SIZE ezdp_decode_defs.h, 336
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_MASK ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_WORD_OFFSET ezdp_decode_defs.h, 336
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_OFFSET ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_WORD_SELECT ezdp_decode_defs.h, 336
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_SIZE ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_RESERVED_31_OFFSET ezdp_decode_defs.h, 336
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_WORD_OFFSET ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_RESERVED_31_SIZE ezdp_decode_defs.h, 336
EZDP_DECODE_MAC_RESULT_IPV4_IN_PPPOE_WORD_SELECT ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_RESERVED120_127_OFFSET ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_MASK ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_RESERVED120_127_SIZE ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_OFFSET ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_RESERVED26_27_OFFSET ezdp_decode_defs.h, 336
EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_SIZE ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_RESERVED26_27_SIZE ezdp_decode_defs.h, 336
EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_WORD_OFFSET ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_OFFSET ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_IPV6_IN_PPPOE_WORD_SELECT ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_SIZE ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_OFFSET ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_WORD_OFFSET ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_SIZE ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG1_PROTOCOL_TYPE_WORD_SELECT ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_WORD_OFFSET ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_OFFSET ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAST_TAG_PROTOCOL_TYPE_WORD_SELECT ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_SIZE ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_OFFSET ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_WORD_OFFSET ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_SIZE ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG2_PROTOCOL_TYPE_WORD_SELECT ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_WORD_OFFSET ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_OFFSET ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_LAYER2_SIZE_WORD_SELECT ezdp_decode_defs.h, 337	EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_SIZE ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_NUMBER_OF_TAGS_OFFSET ezdp_decode_defs.h, 336	EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_WORD_OFFSET ezdp_decode_defs.h, 337

EZDP_DECODE_MAC_RESULT_TAG3_PROTOCOL_TYPE_WORD_SELECT	ezdp_decode_defs.h, 337
EZDP_DECODE_MAC_RESULT_WORD_COUNT	ezdp_decode_defs.h, 338
ezdp_decode_mac_retval, 55	
__pad0__, 55	
__pad1__, 55	
control, 56	
error_codes, 56	
ipv4_in_pppoe, 55	
ipv6_in_pppoe, 55	
number_of_tags, 55	
raw_data, 55	
EZDP_DECODE_MAC_RETVAL_CONTROL_OFFSET	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_CONTROL_SIZE	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_ERROR_CODES_OFFSET	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_ERROR_CODES_SIZE	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_MASK	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_OFFSET	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_IPV4_IN_PPPOE_SIZE	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_MASK	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_OFFSET	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_IPV6_IN_PPPOE_SIZE	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_NUMBER_OF_TAGS_OFFSET	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_NUMBER_OF_TAGS_SIZE	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_RESERVED31_OFFSET	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_RESERVED31_SIZE	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_RESERVED26_27_OFFSET	ezdp_decode_defs.h, 335
EZDP_DECODE_MAC_RETVAL_RESERVED26_27_SIZE	ezdp_decode_defs.h, 335
ezdp_decode_defs.h, 335	
ezdp_decode_mac_retval_t	ezdp_decode_defs.h, 344
ezdp_decode_mpls	ezdp_decode.h, 297
ezdp_decode_mpls_async	ezdp_decode.h, 297
ezdp_decode_mpls_label	ezdp_decode.h, 298
ezdp_decode_mpls_label_async	ezdp_decode.h, 298
ezdp_decode_mpls_label_result, 57	
__pad0__, 57	
end_of_stack, 58	
exception_bit, 57	
raw_data, 57	
reserved_label, 58	
stop_bit, 57	
ttl_is_one, 58	
ttl_is_zero, 58	
user_config0, 58	
user_config1, 58	
user_config2, 58	
user_config3, 58	
EZDP_DECODE_MPLS_LABEL_RESULT_END_OF_STACK_MASK	ezdp_decode_defs.h, 342
EZDP_DECODE_MPLS_LABEL_RESULT_END_OF_STACK_OFFSET	ezdp_decode_defs.h, 342
EZDP_DECODE_MPLS_LABEL_RESULT_END_OF_STACK_SIZE	ezdp_decode_defs.h, 342
EZDP_DECODE_MPLS_LABEL_RESULT_EXCEPTION_BIT_MASK	ezdp_decode_defs.h, 343
EZDP_DECODE_MPLS_LABEL_RESULT_EXCEPTION_BIT_OFFSET	ezdp_decode_defs.h, 343
EZDP_DECODE_MPLS_LABEL_RESULT_EXCEPTION_BIT_SIZE	ezdp_decode_defs.h, 343
EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED_LABEL_MASK	ezdp_decode_defs.h, 342
EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED_LABEL_OFFSET	ezdp_decode_defs.h, 342
EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED_LABEL_SIZE	ezdp_decode_defs.h, 342
EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED10_31_OFFSET	ezdp_decode_defs.h, 343
EZDP_DECODE_MPLS_LABEL_RESULT_RESERVED10_31_SIZE	ezdp_decode_defs.h, 343
EZDP_DECODE_MPLS_LABEL_RESULT_STOP_BIT_MASK	ezdp_decode_defs.h, 343

<p>EZDP_DECODE_MPLS_LABEL_RESULT_STOP_BIT_OFFSET ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_STOP_BIT_SIZE ezdp_decode_defs.h, 343</p> <p>ezdp_decode_mpls_label_result_t ezdp_decode_defs.h, 344</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ONE_MASK ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ONE_OFFSET ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ONE_SIZE ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ZERO_MASK ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ZERO_OFFSET ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_TTL_IS_ZERO_SIZE ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG0_MASK ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG0_OFFSET ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG0_SIZE ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG1_MASK ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG1_OFFSET ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG1_SIZE ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG2_MASK ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG2_OFFSET ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG2_SIZE ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG3_MASK ezdp_decode_defs.h, 343</p> <p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG3_OFFSET ezdp_decode_defs.h, 343</p>	<p>EZDP_DECODE_MPLS_LABEL_RESULT_USER_CONFIG3_SIZE ezdp_decode_defs.h, 343</p> <p>ezdp_decode_mpls_label_retval, 59</p> <p>__pad0__, 59</p> <p>end_of_stack, 60</p> <p>exception_bit, 59</p> <p>raw_data, 59</p> <p>reserved_label, 60</p> <p>stop_bit, 59</p> <p>ttl_is_one, 60</p> <p>ttl_is_zero, 60</p> <p>user_config0, 60</p> <p>user_config1, 60</p> <p>user_config2, 60</p> <p>user_config3, 60</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_END_OF_STACK_MASK ezdp_decode_defs.h, 341</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_END_OF_STACK_OFFSET ezdp_decode_defs.h, 341</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_END_OF_STACK_SIZE ezdp_decode_defs.h, 341</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_EXCEPTION_BIT_MASK ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_EXCEPTION_BIT_OFFSET ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_EXCEPTION_BIT_SIZE ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED_LABEL_MASK ezdp_decode_defs.h, 341</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED_LABEL_OFFSET ezdp_decode_defs.h, 341</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED_LABEL_SIZE ezdp_decode_defs.h, 341</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED10_31_OFFSET ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_RESERVED10_31_SIZE ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_STOP_BIT_MASK ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_STOP_BIT_OFFSET ezdp_decode_defs.h, 342</p> <p>EZDP_DECODE_MPLS_LABEL_RETVAL_STOP_BIT_SIZE ezdp_decode_defs.h, 342</p> <p>ezdp_decode_mpls_label_retval_t ezdp_decode_defs.h, 344</p>
--	---

EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS  
     \_ONE\_MASK  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS  
     \_ONE\_OFFSET  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS  
     \_ONE\_SIZE  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS  
     \_ZERO\_MASK  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS  
     \_ZERO\_OFFSET  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_TTL\_IS  
     \_ZERO\_SIZE  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG0\_MASK  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG0\_OFFSET  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG0\_SIZE  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG1\_MASK  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG1\_OFFSET  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG1\_SIZE  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG2\_MASK  
     ezdp\_decode\_defs.h, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG2\_OFFSET  
     ezdp\_decode\_defs.h, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG2\_SIZE  
     ezdp\_decode\_defs.h, 341  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG3\_MASK  
     ezdp\_decode\_defs.h, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG3\_OFFSET  
     ezdp\_decode\_defs.h, 342  
 EZDP\_DECODE\_MPLS\_LABEL\_RETVAL\_USER\_  
     CONFIG3\_SIZE  
     ezdp\_decode\_defs.h, 342  
 ezdp\_decode\_mpls\_result, 61  
     \_\_pad0\_\_, 61  
     \_\_pad1\_\_, 62  
     \_\_pad2\_\_, 62  
     \_\_pad3\_\_, 62  
 decode\_error, 62

label1\_ttl\_is\_one, 62  
 label1\_ttl\_is\_zero, 62  
 label2\_ttl\_is\_one, 62  
 label2\_ttl\_is\_zero, 62  
 label3\_ttl\_is\_one, 62  
 label3\_ttl\_is\_zero, 62  
 label4\_ttl\_is\_one, 61  
 label4\_ttl\_is\_zero, 62  
 last\_entry\_in\_stack, 62  
 raw\_data, 61  
 EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERRO  
     R\_MASK  
     ezdp\_decode\_defs.h, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERRO  
     R\_OFFSET  
     ezdp\_decode\_defs.h, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_DECODE\_ERRO  
     R\_SIZE  
     ezdp\_decode\_defs.h, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS  
     \_ONE\_MASK  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS  
     \_ONE\_OFFSET  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS  
     \_ONE\_SIZE  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS  
     \_ZERO\_MASK  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS  
     \_ZERO\_OFFSET  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL1\_TTL\_IS  
     \_ZERO\_SIZE  
     ezdp\_decode\_defs.h, 339  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS  
     \_ONE\_MASK  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS  
     \_ONE\_OFFSET  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS  
     \_ONE\_SIZE  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS  
     \_ZERO\_MASK  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS  
     \_ZERO\_OFFSET  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL2\_TTL\_IS  
     \_ZERO\_SIZE  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS  
     \_ONE\_MASK  
     ezdp\_decode\_defs.h, 340  
 EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS  
     \_ONE\_OFFSET

ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS  
ONE\_SIZE  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS  
ZERO\_MASK  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS  
ZERO\_OFFSET  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL3\_TTL\_IS  
ZERO\_SIZE  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS  
ONE\_MASK  
ezdp\_decode\_defs.h, 341  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS  
ONE\_OFFSET  
ezdp\_decode\_defs.h, 341  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS  
ONE\_SIZE  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS  
ZERO\_MASK  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS  
ZERO\_OFFSET  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LABEL4\_TTL\_IS  
ZERO\_SIZE  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_LAST\_ENTRY\_I  
N\_STACK\_OFFSET  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RESULT\_LAST\_ENTRY\_I  
N\_STACK\_SIZE  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED1\_7\_  
OFFSET  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED1\_7\_  
SIZE  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED10\_1  
5\_OFFSET  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED10\_1  
5\_SIZE  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED20\_2  
3\_OFFSET  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED20\_2  
3\_SIZE  
ezdp\_decode\_defs.h, 340  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED28\_3  
1\_OFFSET  
ezdp\_decode\_defs.h, 341  
EZDP\_DECODE\_MPLS\_RESULT\_RESERVED28\_3  
1\_SIZE

ezdp\_decode\_defs.h, 341  
ezdp\_decode\_mpls\_result\_t  
ezdp\_decode\_defs.h, 344  
ezdp\_decode\_mpls\_retval, 64  
\_\_pad0\_\_, 64  
\_\_pad1\_\_, 65  
\_\_pad2\_\_, 65  
\_\_pad3\_\_, 65  
decode\_error, 65  
label1\_ttl\_is\_one, 65  
label1\_ttl\_is\_zero, 65  
label2\_ttl\_is\_one, 65  
label2\_ttl\_is\_zero, 65  
label3\_ttl\_is\_one, 65  
label3\_ttl\_is\_zero, 65  
label4\_ttl\_is\_one, 64  
label4\_ttl\_is\_zero, 65  
last\_entry\_in\_stack, 65  
raw\_data, 64  
EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ERRO  
R\_MASK  
ezdp\_decode\_defs.h, 338  
EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ERRO  
R\_OFFSET  
ezdp\_decode\_defs.h, 338  
EZDP\_DECODE\_MPLS\_RETVAL\_DECODE\_ERRO  
R\_SIZE  
ezdp\_decode\_defs.h, 338  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_I  
S\_ONE\_MASK  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_I  
S\_ONE\_OFFSET  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_I  
S\_ONE\_SIZE  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_I  
S\_ZERO\_MASK  
ezdp\_decode\_defs.h, 338  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_I  
S\_ZERO\_OFFSET  
ezdp\_decode\_defs.h, 338  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL1\_TTL\_I  
S\_ZERO\_SIZE  
ezdp\_decode\_defs.h, 338  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_I  
S\_ONE\_MASK  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_I  
S\_ONE\_OFFSET  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_I  
S\_ONE\_SIZE  
ezdp\_decode\_defs.h, 339  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_I  
S\_ZERO\_MASK  
ezdp\_decode\_defs.h, 338  
EZDP\_DECODE\_MPLS\_RETVAL\_LABEL2\_TTL\_I  
S\_ZERO\_OFFSET



ezdp_decode_defs.h, 338	ezdp_decode_defs.h, 338
EZDP_DECODE_MPLS_RETVAL_LABEL2_TTL_I	EZDP_DECODE_MPLS_RETVAL_RESERVED20_2
S_ZERO_SIZE	3_SIZE
ezdp_decode_defs.h, 338	ezdp_decode_defs.h, 338
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_I	EZDP_DECODE_MPLS_RETVAL_RESERVED28_3
S_ONE_MASK	1_OFFSET
ezdp_decode_defs.h, 339	ezdp_decode_defs.h, 339
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_I	EZDP_DECODE_MPLS_RETVAL_RESERVED28_3
S_ONE_OFFSET	1_SIZE
ezdp_decode_defs.h, 339	ezdp_decode_defs.h, 339
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_I	ezdp_decode_mpls_retval_t
S_ONE_SIZE	ezdp_decode_defs.h, 344
ezdp_decode_defs.h, 339	ezdp_decode_tcp
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_I	ezdp_decode.h, 298
S_ZERO_MASK	ezdp_decode_tcp_errors, 67
ezdp_decode_defs.h, 338	__pad0__, 67
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_I	data_offset_lt_5, 67
S_ZERO_OFFSET	decode_error, 67
ezdp_decode_defs.h, 338	raw_data, 67
EZDP_DECODE_MPLS_RETVAL_LABEL3_TTL_I	syn_and_fin_eq_1, 67
S_ZERO_SIZE	EZDP_DECODE_TCP_ERRORS_DATA_OFFSET_L
ezdp_decode_defs.h, 338	T_5_MASK
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_I	ezdp_decode_defs.h, 326
S_ONE_MASK	EZDP_DECODE_TCP_ERRORS_DATA_OFFSET_L
ezdp_decode_defs.h, 339	T_5_OFFSET
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_I	ezdp_decode_defs.h, 326
S_ONE_OFFSET	EZDP_DECODE_TCP_ERRORS_DATA_OFFSET_L
ezdp_decode_defs.h, 339	T_5_SIZE
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_I	ezdp_decode_defs.h, 326
S_ONE_SIZE	EZDP_DECODE_TCP_ERRORS_DECODE_ERROR
ezdp_decode_defs.h, 339	_MASK
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_I	ezdp_decode_defs.h, 326
S_ZERO_MASK	EZDP_DECODE_TCP_ERRORS_DECODE_ERROR
ezdp_decode_defs.h, 338	_OFFSET
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_I	ezdp_decode_defs.h, 326
S_ZERO_OFFSET	EZDP_DECODE_TCP_ERRORS_DECODE_ERROR
ezdp_decode_defs.h, 338	_SIZE
EZDP_DECODE_MPLS_RETVAL_LABEL4_TTL_I	ezdp_decode_defs.h, 326
S_ZERO_SIZE	EZDP_DECODE_TCP_ERRORS_RESERVED3_7_O
ezdp_decode_defs.h, 338	FFSET
EZDP_DECODE_MPLS_RETVAL_LAST_ENTRY_I	ezdp_decode_defs.h, 326
N_STACK_OFFSET	EZDP_DECODE_TCP_ERRORS_RESERVED3_7_SI
ezdp_decode_defs.h, 338	ZE
EZDP_DECODE_MPLS_RETVAL_LAST_ENTRY_I	ezdp_decode_defs.h, 326
N_STACK_SIZE	EZDP_DECODE_TCP_ERRORS_SYN_AND_FIN_E
ezdp_decode_defs.h, 338	Q_1_MASK
EZDP_DECODE_MPLS_RETVAL_RESERVED1_7_	ezdp_decode_defs.h, 326
OFFSET	EZDP_DECODE_TCP_ERRORS_SYN_AND_FIN_E
ezdp_decode_defs.h, 338	Q_1_OFFSET
EZDP_DECODE_MPLS_RETVAL_RESERVED1_7_	ezdp_decode_defs.h, 326
SIZE	EZDP_DECODE_TCP_ERRORS_SYN_AND_FIN_E
ezdp_decode_defs.h, 338	Q_1_SIZE
EZDP_DECODE_MPLS_RETVAL_RESERVED10_1	ezdp_decode_defs.h, 326
5_OFFSET	ezdp_decode_tcp_errors_t
ezdp_decode_defs.h, 338	ezdp_decode_defs.h, 344
EZDP_DECODE_MPLS_RETVAL_RESERVED10_1	ezdp_decode_tcp_retval, 68
5_SIZE	__pad0__, 68
ezdp_decode_defs.h, 338	__pad1__, 68
EZDP_DECODE_MPLS_RETVAL_RESERVED20_2	__pad2__, 68
3_OFFSET	data_offset, 68

error\_codes, 68  
 options\_exist, 68  
 raw\_data, 68  
 EZDP\_DECODE\_TCP\_RETVAL\_DATA\_OFFSET\_OFFSET  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_DATA\_OFFSET\_SIZE  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_ERROR\_CODES\_OFFSET  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_ERROR\_CODES\_SIZE  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIST\_MASK  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIST\_OFFSET  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_OPTIONS\_EXIST\_SIZE  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED22\_23\_OFFSET  
     ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED22\_23\_SIZE  
     ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED24\_31\_OFFSET  
     ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED24\_31\_SIZE  
     ezdp\_decode\_defs.h, 327  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED9\_15\_OFFSET  
     ezdp\_decode\_defs.h, 326  
 EZDP\_DECODE\_TCP\_RETVAL\_RESERVED9\_15\_SIZE  
     ezdp\_decode\_defs.h, 326  
 ezdp\_decode\_tcp\_retval\_t  
     ezdp\_decode\_defs.h, 344  
 EZDP\_DECODE\_VERSION\_MAJOR  
     ezdp\_decode\_defs.h, 316  
 EZDP\_DECODE\_VERSION\_MINOR  
     ezdp\_decode\_defs.h, 316  
 ezdp\_decrypt  
     ezdp\_security.h, 565  
 ezdp\_decrypt\_async  
     ezdp\_security.h, 565  
 ezdp\_defs.h  
     \_\_aligned\_cmemb\_ext\_addr, 346  
     \_\_alter\_cmemb\_shared\_var, 346  
     \_\_alter\_cmemb\_var, 346  
     \_\_cmemb, 346  
     \_\_cmemb\_shared\_var, 346  
     \_\_cmemb\_var, 346  
     \_\_ememb\_var, 346  
     \_\_imemb\_1\_cluster\_func, 346  
     \_\_imemb\_1\_cluster\_var, 346  
     \_\_imemb\_16\_cluster\_func, 346  
     \_\_imemb\_16\_cluster\_var, 346  
     \_\_imemb\_2\_cluster\_func, 346  
     \_\_imemb\_2\_cluster\_var, 346  
     \_\_imemb\_4\_cluster\_func, 346  
     \_\_imemb\_4\_cluster\_var, 346  
     \_\_imemb\_all\_cluster\_func, 347  
     \_\_imemb\_all\_cluster\_var, 346  
     \_\_imemb\_half\_cluster\_func, 346  
     \_\_imemb\_half\_cluster\_var, 346  
     \_\_imemb\_private\_var, 346  
     \_\_no\_inline, 346  
     \_\_packed, 346  
     \_\_packed\_struct, 346  
     \_\_unused, 346  
     likely, 346  
     unlikely, 346  
 EZDP\_DELETE\_ENTRY  
     ezdp\_search\_defs.h, 553  
 ezdp\_delete\_hash\_entry  
     ezdp\_search.h, 512  
 ezdp\_delete\_table\_entry  
     ezdp\_search.h, 509  
 ezdp\_dequeue\_list  
     ezdp\_queue.h, 503  
 ezdp\_dequeue\_qlock  
     ezdp\_lock.h, 438  
 ezdp\_dequeue\_ring  
     ezdp\_queue.h, 502  
 EZDP\_DES\_BLOCK\_SIZE  
     ezdp\_security\_defs.h, 583  
 EZDP\_DES\_CBC\_ALG  
     ezdp\_security\_defs.h, 579  
 EZDP\_DES\_CFB\_ALG  
     ezdp\_security\_defs.h, 579  
 EZDP\_DES\_CTR\_ALG  
     ezdp\_security\_defs.h, 580  
 EZDP\_DES\_ECB\_ALG  
     ezdp\_security\_defs.h, 580  
 EZDP\_DES\_IV\_SIZE  
     ezdp\_security\_defs.h, 582  
 EZDP\_DES\_OFB\_ALG  
     ezdp\_security\_defs.h, 579  
 EZDP\_DES\_STATE\_SIZE  
     ezdp\_security\_defs.h, 582  
 EZDP\_DES\_XXX\_KEY\_SIZE  
     ezdp\_security\_defs.h, 581  
 ezdp\_destroy\_list  
     ezdp\_queue.h, 504  
 ezdp\_destroy\_qlock  
     ezdp\_lock.h, 436  
 EZDP\_DISCARD  
     ezdp\_job\_defs.h, 433  
 ezdp\_discard\_job  
     ezdp\_job.h, 402  
 ezdp\_discard\_job\_id  
     ezdp\_job.h, 401  
 ezdp\_discard\_job\_id\_async

ezdp\_job.h, 402  
ezdp\_div  
ezdp\_math.h, 447  
ezdp\_dma.h  
ezdp\_copy\_data\_by\_ext\_addr, 349  
ezdp\_copy\_data\_by\_ext\_addr\_async, 349  
ezdp\_load\_16\_byte\_data\_from\_ext\_addr, 350  
ezdp\_load\_16\_byte\_data\_from\_ext\_addr\_async, 351  
ezdp\_load\_16\_byte\_data\_from\_sum\_addr, 354  
ezdp\_load\_16\_byte\_data\_from\_sum\_addr\_async, 354  
ezdp\_load\_32\_byte\_data\_from\_ext\_addr, 351  
ezdp\_load\_32\_byte\_data\_from\_ext\_addr\_async, 351  
ezdp\_load\_32\_byte\_data\_from\_sum\_addr, 355  
ezdp\_load\_32\_byte\_data\_from\_sum\_addr\_async, 355  
ezdp\_load\_data\_from\_ext\_addr, 350  
ezdp\_load\_data\_from\_ext\_addr\_async, 350  
ezdp\_load\_data\_from\_sum\_addr, 353  
ezdp\_load\_data\_from\_sum\_addr\_async, 354  
ezdp\_store\_16\_byte\_data\_to\_ext\_addr, 352  
ezdp\_store\_16\_byte\_data\_to\_ext\_addr\_async, 352  
ezdp\_store\_16\_byte\_data\_to\_sum\_addr, 356  
ezdp\_store\_16\_byte\_data\_to\_sum\_addr\_async, 357  
ezdp\_store\_32\_byte\_data\_to\_ext\_addr, 353  
ezdp\_store\_32\_byte\_data\_to\_ext\_addr\_async, 353  
ezdp\_store\_32\_byte\_data\_to\_sum\_addr, 357  
ezdp\_store\_32\_byte\_data\_to\_sum\_addr\_async, 357  
ezdp\_store\_data\_to\_ext\_addr, 351  
ezdp\_store\_data\_to\_ext\_addr\_async, 352  
ezdp\_store\_data\_to\_sum\_addr, 356  
ezdp\_store\_data\_to\_sum\_addr\_async, 356  
ezdp\_dma\_flags  
ezdp\_memory\_defs.h, 470  
EZDP\_DONT\_DROP  
ezdp\_job\_defs.h, 432  
ezdp\_driver\_desc, 69  
buf\_data\_addr, 69  
flags, 69  
len, 69  
raw\_data, 69  
sub\_type, 69  
total, 69  
EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_OFFSET  
T  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_SIZE  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_WORD\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_WORD\_SELECT  
ezdp\_pci\_defs.h, 490  
ezdp\_driver\_desc\_flags, 70  
data, 70  
error, 70  
owner, 70  
raw\_data, 70  
type, 70

EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_MASK  
ezdp\_pci\_defs.h, 489  
EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_OFFSET  
ezdp\_pci\_defs.h, 489  
EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_SIZE  
ezdp\_pci\_defs.h, 489  
EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_MASK  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_SIZE  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_MASK  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_SIZE  
ezdp\_pci\_defs.h, 489  
EZDP\_DRIVER\_DESC\_FLAGS\_SIZE  
ezdp\_pci\_defs.h, 490  
ezdp\_driver\_desc\_flags\_t  
ezdp\_pci\_defs.h, 491  
EZDP\_DRIVER\_DESC\_FLAGS\_TYPE\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_TYPE\_SIZE  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_WORD\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_FLAGS\_WORD\_SELECT  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_LEN\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_LEN\_SIZE  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_LEN\_WORD\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_LEN\_WORD\_SELECT  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_SUB\_TYPE\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_SUB\_TYPE\_SIZE  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_SUB\_TYPE\_WORD\_OFFSET  
T  
ezdp\_pci\_defs.h, 491  
EZDP\_DRIVER\_DESC\_SUB\_TYPE\_WORD\_SELECT  
T  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_TOTAL\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_TOTAL\_SIZE  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_TOTAL\_WORD\_OFFSET  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_TOTAL\_WORD\_SELECT  
ezdp\_pci\_defs.h, 490  
EZDP\_DRIVER\_DESC\_WORD\_COUNT  
ezdp\_pci\_defs.h, 491

```

ezdp_dual_add_posted_ctr
    ezdp_counter.h, 261
ezdp_dual_add_posted_ctr_async
    ezdp_counter.h, 261
ezdp_dual_add32_result, 71
    original_value1, 71
    original_value2, 71
    raw_data, 71
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E1_OFFSET
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E1_SIZE
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E1_WORD_OFFSET
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E1_WORD_SELECT
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E2_OFFSET
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E2_SIZE
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E2_WORD_OFFSET
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_ORIGINAL_VALU
E2_WORD_SELECT
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD32_RESULT_WORD_COUNT
    ezdp_memory_defs.h, 468
ezdp_dual_add64_result, 72
    original_value1, 72
    original_value2, 72
    raw_data, 72
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E1_OFFSET
    ezdp_memory_defs.h, 469
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E1_SIZE
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E1_WORD_OFFSET
    ezdp_memory_defs.h, 469
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E1_WORD_SELECT
    ezdp_memory_defs.h, 469
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E2_OFFSET
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E2_SIZE
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E2_WORD_OFFSET
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD64_RESULT_ORIGINAL_VALU
E2_WORD_SELECT
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD64_RESULT_WORD_COUNT
    ezdp_memory_defs.h, 468
EZDP_DUAL_ADD64_RESULT_WORD_SELECT
    ezdp_memory_defs.h, 468
EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEE
DED_OFFSET
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEE
DED_SIZE
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEE
DED_WORD_OFFSET
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_BYTE_REPORT_EXCEE
DED_WORD_SELECT
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_OF
FSET
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_SIZ
E
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_WO
RD_OFFSET
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_BYTE_VALUE_SIZE_WO
RD_SELECT
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_CLR_ON_GC_OFFSET
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_CLR_ON_GC_SIZE
    ezdp_counter_defs.h, 277
EZDP_DUAL_CTR_CFG_ECC_OFFSET
    ezdp_counter_defs.h, 277

```

EZDP_DUAL_CTR_CFG_ECC_SIZE	event_value, 76
ezdp_counter_defs.h, 277	raw_data, 76
EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MES	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_
SAGE_MASK	OFFSET
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MES	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_
SAGE_OFFSET	SIZE
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MES	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_
SAGE_SIZE	WORD_OFFSET
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MES	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_LSB_
SAGE_WORD_OFFSET	WORD_SELECT
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_ENABLE_EXCEED_MES	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_
SAGE_WORD_SELECT	OFFSET
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEE	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_
DED_OFFSET	SIZE
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEE	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_
DED_SIZE	WORD_OFFSET
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEE	EZDP_DUAL_CTR_RESULT_BYTE_VALUE_MSB_
DED_WORD_OFFSET	WORD_SELECT
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_EVENT_REPORT_EXCEE	EZDP_DUAL_CTR_RESULT_EVENT_VALUE_OFF
DED_WORD_SELECT	SET
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_RESERVED0_OFFSET	EZDP_DUAL_CTR_RESULT_EVENT_VALUE_SIZ
ezdp_counter_defs.h, 276	E
EZDP_DUAL_CTR_CFG_RESERVED0_SIZE	ezdp_counter_defs.h, 276
ezdp_counter_defs.h, 276	EZDP_DUAL_CTR_RESULT_EVENT_VALUE_WO
EZDP_DUAL_CTR_CFG_RESERVED19_23_OFFSE	RD_OFFSET
T	ezdp_counter_defs.h, 276
ezdp_counter_defs.h, 277	EZDP_DUAL_CTR_RESULT_EVENT_VALUE_WO
EZDP_DUAL_CTR_CFG_RESERVED19_23_SIZE	RD_SELECT
ezdp_counter_defs.h, 277	ezdp_counter_defs.h, 276
EZDP_DUAL_CTR_CFG_VALUE_OFFSET	EZDP_DUAL_CTR_RESULT_RESERVED31_OFFS
ezdp_counter_defs.h, 277	ET
EZDP_DUAL_CTR_CFG_VALUE_SIZE	ezdp_counter_defs.h, 276
ezdp_counter_defs.h, 277	EZDP_DUAL_CTR_RESULT_RESERVED31_SIZE
EZDP_DUAL_CTR_CFG_VALUE_WORD_OFFSET	ezdp_counter_defs.h, 276
ezdp_counter_defs.h, 278	EZDP_DUAL_CTR_RESULT_WORD_COUNT
EZDP_DUAL_CTR_CFG_VALUE_WORD_SELECT	ezdp_counter_defs.h, 276
ezdp_counter_defs.h, 277	EZDP_DUAL_CTR_WORD_COUNT
EZDP_DUAL_CTR_CFG_WORD_COUNT	ezdp_counter_defs.h, 276
ezdp_counter_defs.h, 278	ezdp_dual_report_and_clear_posted_ctr
EZDP_DUAL_CTR_EVENT_OFFSET	ezdp_counter.h, 262
ezdp_counter_defs.h, 276	ezdp_dual_report_posted_ctr
EZDP_DUAL_CTR_EVENT_SIZE	ezdp_counter.h, 262
ezdp_counter_defs.h, 276	ezdp_dual_reset_posted_ctr
EZDP_DUAL_CTR_EVENT_WORD_OFFSET	ezdp_counter.h, 263
ezdp_counter_defs.h, 276	ezdp_dual_reset_posted_ctr_async
EZDP_DUAL_CTR_EVENT_WORD_SELECT	ezdp_counter.h, 263
ezdp_counter_defs.h, 276	ezdp_dual_write_posted_ctr
ezdp_dual_ctr_result, 76	ezdp_counter.h, 260
__pad0__, 76	ezdp_dual_write_posted_ctr_async
byte_value_lsb, 76	ezdp_counter.h, 260
byte_value_msb, 76	EZDP_EMEM_DATA

ezdp.h, 193  
 ezdp\_encrypt  
   ezdp\_security.h, 564  
 ezdp\_encrypt\_async  
   ezdp\_security.h, 565  
 ezdp\_end\_gcm\_mac\_calculation  
   ezdp\_security.h, 568  
 ezdp\_end\_gcm\_mac\_calculation\_async  
   ezdp\_security.h, 569  
 ezdp\_end\_hmac\_calculation  
   ezdp\_security.h, 567  
 ezdp\_end\_hmac\_calculation\_async  
   ezdp\_security.h, 567  
 ezdp\_enqueue\_list  
   ezdp\_queue.h, 503  
 ezdp\_enqueue\_qlock  
   ezdp\_lock.h, 438  
 ezdp\_enqueue\_ring  
   ezdp\_queue.h, 502  
 ezdp\_expand\_security\_key  
   ezdp\_security.h, 569  
 ezdp\_expand\_security\_key\_async  
   ezdp\_security.h, 569  
 EZDP\_EXPLICIT\_PSID  
   ezdp\_job\_defs.h, 432  
 ezdp\_ext\_addr, 77  
   \_\_pad0\_\_, 77  
   \_\_pad1\_\_, 77  
   \_\_pad2\_\_, 77  
   address, 77  
   address\_msb, 77  
   msid, 77  
   raw\_data, 77  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_OFFSET  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_SIZE  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_WORD\_OFFSET  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_WORD\_SELECT  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_MASK  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_OFFSET  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_SIZE  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_WORD\_OFFSET  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_WORD\_SELECT  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_RESERVED14\_15\_OFFSET  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_RESERVED14\_15\_SIZE  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_RESERVED16\_31\_OFFSET  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_RESERVED16\_31\_SIZE  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_RESERVED4\_7\_OFFSET  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_ADDR\_RESERVED4\_7\_SIZE  
   ezdp\_memory\_defs.h, 465  
 ezdp\_ext\_addr\_to\_sum\_addr  
   ezdp\_memory.h, 460  
 EZDP\_EXT\_ADDR\_WORD\_COUNT  
   ezdp\_memory\_defs.h, 465  
 EZDP\_EXT\_FRAME  
   ezdp\_frame\_defs.h, 389  
 ezdp\_ext\_linked\_buffers\_desc, 79  
   line, 79  
 EZDP\_EXT\_MEM  
   ezdp\_frame\_defs.h, 388  
 EZDP\_EXT\_MEM\_BUF\_BUDGET  
   ezdp\_job\_defs.h, 431  
 ezdp\_ext\_tcam\_result\_element\_type  
   ezdp\_search\_defs.h, 552  
 EZDP\_EXTENDED\_LBD  
   ezdp\_frame\_defs.h, 390  
 EZDP\_EXTERNAL\_MS  
   ezdp\_memory\_defs.h, 470  
 ezdp\_extract\_frame\_tail\_working\_area\_t  
   ezdp\_frame\_defs.h, 388  
 ezdp\_find\_first\_one  
   ezdp\_math.h, 448  
 ezdp\_find\_first\_zero  
   ezdp\_math.h, 449  
 EZDP\_FIXED\_BASE  
   ezdp\_job\_defs.h, 431  
 ezdp\_flow\_control\_congestion\_level  
   ezdp\_job\_defs.h, 432  
 ezdp\_flow\_control\_node  
   ezdp\_job\_defs.h, 430  
 ezdp\_flow\_control\_status, 80  
   \_\_pad0\_\_, 80  
   enable, 80  
   raw\_data, 80  
 EZDP\_FLOW\_CONTROL\_STATUS\_CONGESTION  
   \_LEVEL\_OFFSET  
   ezdp\_job\_defs.h, 427  
 EZDP\_FLOW\_CONTROL\_STATUS\_CONGESTION  
   \_LEVEL\_SIZE

ezdp\_job\_defs.h, 427  
 EZDP\_FLOW\_CONTROL\_STATUS\_ENABLE\_MAS  
 K  
 ezdp\_job\_defs.h, 427  
 EZDP\_FLOW\_CONTROL\_STATUS\_ENABLE\_OFF  
 SET  
 ezdp\_job\_defs.h, 427  
 EZDP\_FLOW\_CONTROL\_STATUS\_ENABLE\_SIZE  
 ezdp\_job\_defs.h, 427  
 EZDP\_FLOW\_CONTROL\_STATUS\_RESERVED4\_7  
 \_OFFSET  
 ezdp\_job\_defs.h, 427  
 EZDP\_FLOW\_CONTROL\_STATUS\_RESERVED4\_7  
 \_SIZE  
 ezdp\_job\_defs.h, 427  
 ezdp\_flow\_control\_status\_t  
 ezdp\_job\_defs.h, 430  
 ezdp\_frame.h  
 ezdp\_alloc\_buf, 361  
 ezdp\_alloc\_mc\_buf, 372  
 ezdp\_alloc\_multi\_buf, 362  
 ezdp\_alloc\_multi\_buf\_async, 362  
 ezdp\_append\_buf, 377  
 ezdp\_atomic\_read\_and\_dec\_mc\_buf\_counter, 374  
 ezdp\_atomic\_read\_and\_inc\_mc\_buf\_counter, 373  
 ezdp\_buf\_alloc\_failed, 363  
 ezdp\_buf\_data\_len, 374  
 ezdp\_calc\_frame\_data\_checksum, 374  
 ezdp\_calc\_header\_offset, 375  
 ezdp\_clone\_frame\_data, 365  
 ezdp\_clone\_frame\_data\_async, 365  
 ezdp\_clone\_frame\_lbd, 369  
 ezdp\_clone\_frame\_lbd\_async, 369  
 ezdp\_copy\_frame\_data, 364  
 ezdp\_copy\_frame\_data\_async, 365  
 ezdp\_copy\_frame\_data\_from\_ext\_addr, 366  
 ezdp\_copy\_frame\_data\_from\_ext\_addr\_async, 367  
 ezdp\_copy\_frame\_data\_to\_ext\_addr, 366  
 ezdp\_copy\_frame\_data\_to\_ext\_addr\_async, 366  
 ezdp\_copy\_frame\_lbd, 368  
 ezdp\_copy\_frame\_lbd\_async, 369  
 ezdp\_copy\_frame\_lbd\_from\_ext\_addr, 370  
 ezdp\_copy\_frame\_lbd\_from\_ext\_addr\_async, 371  
 ezdp\_copy\_frame\_lbd\_to\_ext\_addr, 370  
 ezdp\_copy\_frame\_lbd\_to\_ext\_addr\_async, 370  
 ezdp\_dec\_tm\_imem\_buf\_ctr, 375  
 ezdp\_dec\_tm\_imem\_buf\_ctr\_async, 375  
 ezdp\_free\_buf, 361  
 ezdp\_free\_buf\_async, 362  
 ezdp\_free\_mc\_buf, 372  
 ezdp\_free\_multi\_buf, 363  
 ezdp\_free\_multi\_buf\_async, 363  
 ezdp\_get\_first\_buf, 376  
 ezdp\_get\_next\_buf, 376  
 ezdp\_inc\_tm\_imem\_buf\_ctr, 375  
 ezdp\_inc\_tm\_imem\_buf\_ctr\_async, 375  
 ezdp\_init\_frame, 377  
 ezdp\_lbd\_len, 374  
 ezdp\_load\_frame\_data, 367  
 ezdp\_load\_frame\_data\_async, 367

ezdp\_load\_frame\_lbd, 371  
 ezdp\_load\_frame\_lbd\_async, 371  
 ezdp\_read\_free\_buf, 363  
 ezdp\_read\_mc\_buf\_counter, 373  
 ezdp\_read\_tm\_imem\_buf\_ctr, 376  
 ezdp\_rebudget\_buf, 364  
 ezdp\_rebudget\_buf\_async, 364  
 ezdp\_store\_frame\_data, 368  
 ezdp\_store\_frame\_data\_async, 368  
 ezdp\_store\_frame\_lbd, 372  
 ezdp\_store\_frame\_lbd\_async, 372  
 ezdp\_sync\_frame, 377  
 ezdp\_write\_mc\_buf\_counter, 373  
 ezdp\_write\_mc\_buf\_counter\_async, 373  
 ezdp\_frame\_buf\_iterator\_state\_t  
 ezdp\_frame\_defs.h, 388  
 ezdp\_frame\_defs.h  
 ezdp\_1588\_type, 389  
 EZDP\_1STEP, 389  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFF  
 SET, 386  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFF  
 SET\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFF  
 SET\_SIZE, 387  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFF  
 SET\_WORD\_OFFSET, 388  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_OFF  
 SET\_WORD\_SELECT, 388  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_SIZ  
 E, 386  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_WO  
 RD\_OFFSET, 386  
 EZDP\_1STEP\_1588\_HEADER\_CHECKSUM\_WO  
 RD\_SELECT, 386  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 DD\_START\_MASK, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 DD\_START\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 DD\_START\_SIZE, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 DD\_START\_WORD\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 DD\_START\_WORD\_SELECT, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 FFSET, 388  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 FFSET\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 FFSET\_SIZE, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 FFSET\_WORD\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_O  
 FFSET\_WORD\_SELECT, 387  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_SI  
 ZE, 388  
 EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_W  
 ORD\_OFFSET, 388

EZDP\_1STEP\_1588\_HEADER\_CORRECTION\_WORD\_SELECT, 388  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_MASK, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_SIZE, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_WORD\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_INJECT\_CHECKSUM\_FLAG\_WORD\_SELECT, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED16\_23\_OFFSET, 386  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED16\_23\_SIZE, 386  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED24\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED24\_SIZE, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED28\_31\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_RESERVED28\_31\_SIZE, 387  
 EZDP\_1STEP\_1588\_HEADER\_WORD\_COUNT, 388  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_MASK, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_SIZE, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_WORD\_OFFSET, 387  
 EZDP\_1STEP\_1588\_HEADER\_WRAP\_AROUND\_CONDITION\_WORD\_SELECT, 387  
 EZDP\_2STEP, 389  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_OFFSET, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_SIZE, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_WORD\_OFFSET, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_BUDGET\_ID\_WORD\_SELECT, 385  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_SIZE, 386  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_WORD\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_BUF\_DESC\_WORD\_SELECT, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_SIZE, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_WORD\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_CLASS\_OF\_SERVICE\_WORD\_SELECT, 386  
 EZDP\_2STEP\_1588\_HEADER\_FREE\_BYTES\_OF\_FSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_FREE\_BYTES\_SIZE, 386  
 EZDP\_2STEP\_1588\_HEADER\_FREE\_BYTES\_WORD\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_FREE\_BYTES\_WORD\_SELECT, 386  
 EZDP\_2STEP\_1588\_HEADER\_HEADER\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_HEADER\_OFFSET\_SIZE, 386  
 EZDP\_2STEP\_1588\_HEADER\_HEADER\_OFFSET\_WORD\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_HEADER\_OFFSET\_WORD\_SELECT, 386  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED0\_23\_OFFSET, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED0\_23\_SIZE, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED24\_31\_OFFSET, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED24\_31\_SIZE, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED24\_OFFSET, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED24\_OF\_FSET, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED24\_SIZE, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED32\_63\_OFFSET, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED32\_63\_SIZE, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED74\_75\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED74\_75\_SIZE, 385  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED76\_77\_OFFSET, 386  
 EZDP\_2STEP\_1588\_HEADER\_RESERVED76\_77\_SIZE, 386  
 EZDP\_2STEP\_1588\_HEADER\_WORD\_COUNT, 386  
 EZDP\_BROADCAST, 389  
 EZDP\_BUFFER\_DATA\_SIZE, 382  
 EZDP\_BUFFER\_DESC\_ID\_OFFSET, 382  
 EZDP\_BUFFER\_DESC\_ID\_SIZE, 382  
 EZDP\_BUFFER\_DESC\_MEM\_TYPE\_MASK, 382  
 EZDP\_BUFFER\_DESC\_MEM\_TYPE\_OFFSET, 382  
 EZDP\_BUFFER\_DESC\_MEM\_TYPE\_SIZE, 382  
 EZDP\_BUFFER\_DESC\_RESERVED28\_29\_OFFSET, 382  
 EZDP\_BUFFER\_DESC\_RESERVED28\_29\_SIZE, 382  
 ezdp\_buffer\_desc\_t, 388  
 EZDP\_BUFFER\_DESC\_VALID\_DATA\_BUFFER\_MASK, 382



EZDP\_BUFFER\_DESC\_VALID\_DATA\_BUF\_OFF  
SET, 382  
 EZDP\_BUFFER\_DESC\_VALID\_DATA\_BUF\_SIZ  
E, 382  
 ezdp\_buffer\_mem\_type, 388  
 ezdp\_concat\_frames\_working\_area\_t, 388  
 ezdp\_convert\_std2ext\_working\_area\_t, 388  
 EZDP\_EXT\_FRAME, 389  
 EZDP\_EXT\_MEM, 388  
 EZDP\_EXTENDED\_LBD, 390  
 ezdp\_extract\_frame\_tail\_working\_area\_t, 388  
 ezdp\_frame\_buf\_iterator\_state\_t, 388  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_OFFS  
ET, 382  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_SIZE,  
382  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_WOR  
D\_OFFSET, 382  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_WOR  
D\_SELECT, 382  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_OFFSET, 384  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_SIZE, 384  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_WORD\_OFFS  
ET, 384  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_WORD\_SELE  
CT, 384  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_OF  
FSET, 382  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_SI  
ZE, 382  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_W  
ORD\_OFFSET, 382  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_W  
ORD\_SELECT, 382  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_OFF  
SET, 384  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_SIZ  
E, 384  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_WO  
RD\_OFFSET, 384  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_WO  
RD\_SELECT, 384  
 EZDP\_FRAME\_DESC\_ECC\_OFFSET, 384  
 EZDP\_FRAME\_DESC\_ECC\_SIZE, 384  
 EZDP\_FRAME\_DESC\_ECC\_WORD\_OFFSET,  
384  
 EZDP\_FRAME\_DESC\_ECC\_WORD\_SELECT,  
384  
 EZDP\_FRAME\_DESC\_FRAME\_LENGTH\_OFFSE  
T, 384  
 EZDP\_FRAME\_DESC\_FRAME\_LENGTH\_SIZE,  
384  
 EZDP\_FRAME\_DESC\_FRAME\_LENGTH\_WORD  
\_OFFSET, 384  
 EZDP\_FRAME\_DESC\_FRAME\_LENGTH\_WORD  
\_SELECT, 384  
 EZDP\_FRAME\_DESC\_FREE\_BYTES\_OFFSET,  
385  
 EZDP\_FRAME\_DESC\_FREE\_BYTES\_SIZE, 385  
 EZDP\_FRAME\_DESC\_FREE\_BYTES\_WORD\_OF  
FSET, 385  
 EZDP\_FRAME\_DESC\_FREE\_BYTES\_WORD\_SE  
LECT, 385  
 EZDP\_FRAME\_DESC\_GROSS\_CHECKSUM\_FL  
AG\_MASK, 383  
 EZDP\_FRAME\_DESC\_GROSS\_CHECKSUM\_FL  
AG\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_GROSS\_CHECKSUM\_FL  
AG\_SIZE, 383  
 EZDP\_FRAME\_DESC\_GROSS\_CHECKSUM\_FL  
AG\_WORD\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_GROSS\_CHECKSUM\_FL  
AG\_WORD\_SELECT, 383  
 EZDP\_FRAME\_DESC\_HEADER\_OFFSET\_OFFS  
ET, 384  
 EZDP\_FRAME\_DESC\_HEADER\_OFFSET\_SIZE,  
384  
 EZDP\_FRAME\_DESC\_HEADER\_OFFSET\_WOR  
D\_OFFSET, 384  
 EZDP\_FRAME\_DESC\_HEADER\_OFFSET\_WOR  
D\_SELECT, 384  
 EZDP\_FRAME\_DESC\_JOB\_BUDGET\_ID\_OFFSE  
T, 384  
 EZDP\_FRAME\_DESC\_JOB\_BUDGET\_ID\_SIZE,  
384  
 EZDP\_FRAME\_DESC\_JOB\_BUDGET\_ID\_WORD  
\_OFFSET, 384  
 EZDP\_FRAME\_DESC\_JOB\_BUDGET\_ID\_WORD  
\_SELECT, 384  
 EZDP\_FRAME\_DESC\_LOGICAL\_ID\_OFFSET,  
385  
 EZDP\_FRAME\_DESC\_LOGICAL\_ID\_SIZE, 385  
 EZDP\_FRAME\_DESC\_LOGICAL\_ID\_WORD\_OF  
FSET, 385  
 EZDP\_FRAME\_DESC\_LOGICAL\_ID\_WORD\_SE  
LECT, 385  
 EZDP\_FRAME\_DESC\_MULTICAST\_CONTROL\_  
OFFSET, 385  
 EZDP\_FRAME\_DESC\_MULTICAST\_CONTROL\_  
SIZE, 385  
 EZDP\_FRAME\_DESC\_MULTICAST\_CONTROL\_  
WORD\_OFFSET, 385  
 EZDP\_FRAME\_DESC\_MULTICAST\_CONTROL\_  
WORD\_SELECT, 385  
 EZDP\_FRAME\_DESC\_RESERVED0\_1\_OFFSET,  
383  
 EZDP\_FRAME\_DESC\_RESERVED0\_1\_SIZE, 382  
 EZDP\_FRAME\_DESC\_RESERVED10\_11\_OFFSE  
T, 382  
 EZDP\_FRAME\_DESC\_RESERVED10\_11\_SIZE,  
382  
 EZDP\_FRAME\_DESC\_RESERVED106\_110\_OFF  
SET, 384  
 EZDP\_FRAME\_DESC\_RESERVED106\_110\_SIZE,  
384  
 EZDP\_FRAME\_DESC\_RESERVED14\_15\_OFFSE  
T, 382  
 EZDP\_FRAME\_DESC\_RESERVED14\_15\_SIZE,  
382

EZDP\_FRAME\_DESC\_TIMESTAMP\_FLAG\_MASK, 383  
 EZDP\_FRAME\_DESC\_TIMESTAMP\_FLAG\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TIMESTAMP\_FLAG\_SIZE, 383  
 EZDP\_FRAME\_DESC\_TIMESTAMP\_FLAG\_WORD\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TIMESTAMP\_FLAG\_WORD\_SELECT, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATION\_FLAG\_MASK, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATION\_FLAG\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATION\_FLAG\_SIZE, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATION\_FLAG\_WORD\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATION\_FLAG\_WORD\_SELECT, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUFFER\_FLAG\_MASK, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUFFER\_FLAG\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUFFER\_FLAG\_SIZE, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUFFER\_FLAG\_WORD\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUFFER\_FLAG\_WORD\_SELECT, 383  
 EZDP\_FRAME\_DESC\_TYPE\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TYPE\_SIZE, 383  
 EZDP\_FRAME\_DESC\_TYPE\_WORD\_OFFSET, 383  
 EZDP\_FRAME\_DESC\_TYPE\_WORD\_SELECT, 383  
 EZDP\_FRAME\_DESC\_WORD\_COUNT, 385  
 ezdp\_frame\_type, 388  
 EZDP\_INT\_MEM, 388  
 EZDP\_LARGE\_LBD, 390  
 EZDP\_LINKED\_BUFFER\_DESC\_LINE\_NUMBER\_OF\_BUFFERS\_DESC, 388  
 ezdp\_linked\_buffers\_desc\_size, 389  
 EZDP\_MEM\_FRAME\_DATA\_BUFFER\_SIZE, 382  
 EZDP\_MULTICAST, 389  
 ezdp\_multicast\_control, 389  
 EZDP\_NULL\_FRAME, 389  
 EZDP\_REPLICA, 389  
 EZDP\_SMALL\_LBD, 389  
 EZDP\_STD\_FRAME, 389  
 ezdp\_trim\_frame\_head\_working\_area\_t, 388  
 EZDP\_UNICAST, 389  
 ezdp\_frame\_desc, 81  
 \_\_pad0\_\_, 82  
 \_\_pad1\_\_, 82  
 \_\_pad2\_\_, 82  
 \_\_pad3\_\_, 84  
 buf\_budget\_id, 83  
 buf\_desc, 83  
 class\_of\_service, 82  
 data\_buf\_count, 83  
 ecc, 81  
 frame\_length, 83  
 free\_bytes, 83  
 gross\_checksum\_flag, 82  
 header\_offset, 83  
 job\_budget\_id, 84  
 logical\_id, 83  
 raw\_data, 81  
 timestamp\_flag, 82  
 transmit\_confirmation\_flag, 82  
 transmit\_keep\_buf\_flag, 82  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_OFFSET  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_SIZE  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_BUF\_BUDGET\_ID\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_OFFSET  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_SIZE  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_BUF\_DESC\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_OFFSET  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_SIZE  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_CLASS\_OF\_SERVICE\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 382  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_OFFSET  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_SIZE  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_DATA\_BUF\_COUNT\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_ECC\_OFFSET  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_ECC\_SIZE  
 ezdp\_frame\_defs.h, 384  
 EZDP\_FRAME\_DESC\_ECC\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 384

EZDP_FRAME_DESC_ECC_WORD_SELECT	ezdp_frame_defs.h, 384	ezdp_frame_defs.h, 385
EZDP_FRAME_DESC_FRAME_LENGTH_OFFSET	ezdp_frame_defs.h, 384	EZDP_FRAME_DESC_LOGICAL_ID_WORD_OFFSET
EZDP_FRAME_DESC_FRAME_LENGTH_SIZE	ezdp_frame_defs.h, 384	ET
EZDP_FRAME_DESC_FRAME_LENGTH_WORD_OFFSET	ezdp_frame_defs.h, 384	ezdp_frame_defs.h, 385
EZDP_FRAME_DESC_FRAME_LENGTH_WORD_SELECT	ezdp_frame_defs.h, 384	EZDP_FRAME_DESC_LOGICAL_ID_WORD_SELECT
EZDP_FRAME_DESC_FREE_BYTES_OFFSET	ezdp_frame_defs.h, 385	CT
EZDP_FRAME_DESC_FREE_BYTES_SIZE	ezdp_frame_defs.h, 385	ezdp_frame_defs.h, 385
EZDP_FRAME_DESC_FREE_BYTES_WORD_OFFSET	ezdp_frame_defs.h, 385	EZDP_FRAME_DESC_MULTICAST_CONTROL_OFFSET
EZDP_FRAME_DESC_FREE_BYTES_WORD_SELECT	ezdp_frame_defs.h, 385	ezdp_frame_defs.h, 385
EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_MASK	ezdp_frame_defs.h, 383	EZDP_FRAME_DESC_MULTICAST_CONTROL_SIZE
EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_OFFSET	ezdp_frame_defs.h, 383	ezdp_frame_defs.h, 385
EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_SIZE	ezdp_frame_defs.h, 383	EZDP_FRAME_DESC_MULTICAST_CONTROL_WORD_OFFSET
EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_WORD_OFFSET	ezdp_frame_defs.h, 383	ezdp_frame_defs.h, 385
EZDP_FRAME_DESC_GROSS_CHECKSUM_FLAG_WORD_SELECT	ezdp_frame_defs.h, 383	EZDP_FRAME_DESC_MULTICAST_CONTROL_WORD_SELECT
EZDP_FRAME_DESC_HEADER_OFFSET_OFFSET	ezdp_frame_defs.h, 384	ezdp_frame_defs.h, 385
EZDP_FRAME_DESC_HEADER_OFFSET_SIZE	ezdp_frame_defs.h, 384	EZDP_FRAME_DESC_RESERVED0_1_OFFSET
EZDP_FRAME_DESC_HEADER_OFFSET_WORD_OFFSET	ezdp_frame_defs.h, 384	ezdp_frame_defs.h, 383
EZDP_FRAME_DESC_HEADER_OFFSET_WORD_SELECT	ezdp_frame_defs.h, 384	EZDP_FRAME_DESC_RESERVED0_1_SIZE
EZDP_FRAME_DESC_JOB_BUDGET_ID_OFFSET	ezdp_frame_defs.h, 384	ezdp_frame_defs.h, 382
EZDP_FRAME_DESC_JOB_BUDGET_ID_SIZE	ezdp_frame_defs.h, 384	EZDP_FRAME_DESC_RESERVED10_11_OFFSET
EZDP_FRAME_DESC_JOB_BUDGET_ID_WORD_OFFSET	ezdp_frame_defs.h, 384	ezdp_frame_defs.h, 382
EZDP_FRAME_DESC_JOB_BUDGET_ID_WORD_SELECT	ezdp_frame_defs.h, 384	EZDP_FRAME_DESC_RESERVED10_11_SIZE
EZDP_FRAME_DESC_LOGICAL_ID_OFFSET	ezdp_frame_defs.h, 385	ezdp_frame_defs.h, 382
EZDP_FRAME_DESC_LOGICAL_ID_SIZE		EZDP_FRAME_DESC_RESERVED106_110_OFFSET
		ezdp_frame_defs.h, 384
		EZDP_FRAME_DESC_RESERVED106_110_SIZE
		ezdp_frame_defs.h, 384
		EZDP_FRAME_DESC_RESERVED14_15_OFFSET
		ezdp_frame_defs.h, 382
		EZDP_FRAME_DESC_RESERVED14_15_SIZE
		ezdp_frame_defs.h, 382
		EZDP_FRAME_DESC_TIMESTAMP_FLAG_MASK
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TIMESTAMP_FLAG_OFFSET
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TIMESTAMP_FLAG_SIZE
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TIMESTAMP_FLAG_WORD_OFFSET
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TIMESTAMP_FLAG_WORD_SELECT
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_MASK
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_OFFSET
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_SIZE
		ezdp_frame_defs.h, 383
		EZDP_FRAME_DESC_TRANSMIT_CONFIRMATION_FLAG_WORD_OFFSET

ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_CONFIRMATI  
 ON\_FLAG\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUF\_FL  
 AG\_MASK  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUF\_FL  
 AG\_OFFSET  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUF\_FL  
 AG\_SIZE  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUF\_FL  
 AG\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TRANSMIT\_KEEP\_BUF\_FL  
 AG\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TYPE\_OFFSET  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TYPE\_SIZE  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TYPE\_WORD\_OFFSET  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_TYPE\_WORD\_SELECT  
 ezdp\_frame\_defs.h, 383  
 EZDP\_FRAME\_DESC\_WORD\_COUNT  
 ezdp\_frame\_defs.h, 385  
 ezdp\_frame\_type  
 ezdp\_frame\_defs.h, 388  
 EZDP\_FREE  
 ezdp\_job\_defs.h, 433  
 ezdp\_free\_buf  
 ezdp\_frame.h, 361  
 ezdp\_free\_buf\_async  
 ezdp\_frame.h, 362  
 ezdp\_free\_index  
 ezdp\_pool.h, 493  
 ezdp\_free\_index\_async  
 ezdp\_pool.h, 494  
 ezdp\_free\_job\_id  
 ezdp\_job.h, 394  
 ezdp\_free\_job\_id\_async  
 ezdp\_job.h, 395  
 ezdp\_free\_mc\_buf  
 ezdp\_frame.h, 372  
 ezdp\_free\_multi\_buf  
 ezdp\_frame.h, 363  
 ezdp\_free\_multi\_buf\_async  
 ezdp\_frame.h, 363  
 ezdp\_free\_multi\_index  
 ezdp\_pool.h, 494  
 ezdp\_free\_multi\_index\_async  
 ezdp\_pool.h, 495  
 ezdp\_free\_obj  
 ezdp\_pool.h, 496  
 ezdp\_free\_qlock\_slot  
 ezdp\_lock.h, 437  
 ezdp\_fxor16  
 ezdp\_math.h, 446  
 ezdp\_fxor8  
 ezdp\_math.h, 445  
 ezdp\_generate\_security\_initial\_vector  
 ezdp\_security.h, 568  
 ezdp\_generate\_security\_initial\_vector\_async  
 ezdp\_security.h, 568  
 ezdp\_get\_2\_bitfields  
 ezdp\_math.h, 450  
 ezdp\_get\_2\_bits  
 ezdp\_math.h, 451  
 ezdp\_get\_3\_bits  
 ezdp\_math.h, 452  
 ezdp\_get\_4\_bits  
 ezdp\_math.h, 452  
 ezdp\_get\_4\_bytes  
 ezdp\_math.h, 455  
 ezdp\_get\_bit  
 ezdp\_math.h, 450  
 ezdp\_get\_bitfield  
 ezdp\_math.h, 449  
 ezdp\_get\_cluster\_id  
 ezdp\_processor.h, 499  
 ezdp\_get\_color\_hier\_tb\_ctr\_working\_area\_t  
 ezdp\_counter\_defs.h, 292  
 ezdp\_get\_core\_id  
 ezdp\_processor.h, 499  
 ezdp\_get\_cpu\_id  
 ezdp\_processor.h, 498  
 ezdp\_get\_err\_msg  
 ezdp.h, 194  
 ezdp\_get\_first\_buf  
 ezdp\_frame.h, 376  
 ezdp\_get\_hash\_entry\_key  
 ezdp\_search.h, 513  
 ezdp\_get\_mem\_section\_info  
 ezdp.h, 194  
 ezdp\_get\_next\_buf  
 ezdp\_frame.h, 376  
 ezdp\_get\_obj  
 ezdp\_pool.h, 496  
 ezdp\_get\_pci\_ctrl\_reg  
 ezdp\_pci.h, 482  
 ezdp\_get\_pci\_msg  
 ezdp\_pci.h, 473  
 ezdp\_get\_pci\_msgq\_read\_index  
 ezdp\_pci.h, 474  
 ezdp\_get\_pci\_msgq\_write\_index  
 ezdp\_pci.h, 474  
 ezdp\_get\_real\_time\_clock  
 ezdp\_time.h, 586  
 ezdp\_get\_real\_time\_clock\_async  
 ezdp\_time.h, 586  
 ezdp\_get\_system\_tick  
 ezdp\_time.h, 586  
 ezdp\_get\_system\_tick\_async  
 ezdp\_time.h, 586  
 ezdp\_get\_thread\_id  
 ezdp\_processor.h, 498  
 ezdp\_get\_tm\_queue\_depth

ezdp\_job.h, 408  
 ezdp\_get\_version  
 ezdp.h, 194  
 EZDP\_GHASH\_128\_ALG  
   ezdp\_security\_defs.h, 581  
 EZDP\_GHASH\_128\_KEY\_SIZE  
   ezdp\_security\_defs.h, 581  
 EZDP\_GHASH\_192\_ALG  
   ezdp\_security\_defs.h, 581  
 EZDP\_GHASH\_192\_KEY\_SIZE  
   ezdp\_security\_defs.h, 581  
 EZDP\_GHASH\_256\_ALG  
   ezdp\_security\_defs.h, 581  
 EZDP\_GHASH\_256\_KEY\_SIZE  
   ezdp\_security\_defs.h, 581  
 EZDP\_GHASH\_BLOCK\_SIZE  
   ezdp\_security\_defs.h, 583  
 EZDP\_GHASH\_MAC\_SIZE  
   ezdp\_security\_defs.h, 583  
 EZDP\_GHASH\_STATE\_SIZE  
   ezdp\_security\_defs.h, 582  
 EZDP\_GHASH\_XXX\_STATE\_SIZE  
   ezdp\_security\_defs.h, 582  
 EZDP\_GLOBAL\_NODE  
   ezdp\_job\_defs.h, 431  
 EZDP\_GREEN\_TRAFFIC  
   ezdp\_counter\_defs.h, 293  
 EZDP\_GROUP\_NODE  
   ezdp\_job\_defs.h, 431  
 ezdp\_group\_schlr\_status, 85  
   \_\_pad0\_\_, 85  
   dispatched\_job, 85  
   raw\_data, 85  
 EZDP\_GROUP\_SCHLR\_STATUS\_DISPATCHED\_J  
   OB\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_GROUP\_SCHLR\_STATUS\_DISPATCHED\_J  
   OB\_SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_GROUP\_SCHLR\_STATUS\_RESERVED13\_1  
   5\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_GROUP\_SCHLR\_STATUS\_RESERVED13\_1  
   5\_SIZE  
   ezdp\_job\_defs.h, 429  
 ezdp\_group\_schlr\_status\_t  
   ezdp\_job\_defs.h, 430  
 EZDP\_HALF\_CLUSTER\_CODE  
   ezdp\_memory\_defs.h, 469  
 EZDP\_HALF\_CLUSTER\_DATA  
   ezdp\_memory\_defs.h, 469  
 ezdp\_handle\_notice  
   ezdp\_job.h, 405  
 EZDP\_HANDLE\_NOTICE  
   ezdp\_job\_defs.h, 433  
 ezdp\_hash  
   ezdp\_math.h, 456  
 ezdp\_hash\_base\_matrix  
   ezdp\_math.h, 443  
 EZDP\_HASH\_BASE\_MATRIX\_HASH\_BASE\_MAT  
   RIX\_0  
   ezdp\_math.h, 443  
 EZDP\_HASH\_BASE\_MATRIX\_HASH\_BASE\_MAT  
   RIX\_1  
   ezdp\_math.h, 443  
 EZDP\_HASH\_HIGH\_LEVEL\_WORK\_AREA\_SIZE  
   ezdp\_search\_defs.h, 532  
 EZDP\_HASH\_LOW\_LEVEL\_WORK\_AREA\_SIZE  
   ezdp\_search\_defs.h, 532  
 ezdp\_hash\_permutation  
   ezdp\_math.h, 444  
 EZDP\_HASH\_PERMUTATION\_0  
   ezdp\_math.h, 444  
 EZDP\_HASH\_PERMUTATION\_1  
   ezdp\_math.h, 444  
 EZDP\_HASH\_PERMUTATION\_2  
   ezdp\_math.h, 444  
 EZDP\_HASH\_PERMUTATION\_3  
   ezdp\_math.h, 444  
 ezdp\_hash\_struct\_desc\_t  
   ezdp\_search\_defs.h, 552  
 ezdp\_hash32  
   ezdp\_math.h, 456  
 ezdp\_hash64  
   ezdp\_math.h, 456  
 ezdp\_hashed\_key\_t  
   ezdp\_search\_defs.h, 552  
 EZDP\_HIER\_TB\_CALC\_COLOR  
   ezdp\_counter\_defs.h, 294  
 ezdp\_hier\_tb\_ctr\_cfg, 86  
   \_\_pad0\_\_, 86  
   \_\_pad1\_\_, 87  
   \_\_pad2\_\_, 87  
   app\_bits, 87  
   ctr\_sum\_fail\_threshold, 87  
   ctr\_sum\_updt\_threshold, 87  
   ctr0\_fail\_threshold, 86  
   ctr0\_updt\_threshold, 87  
   ctr1\_fail\_threshold, 87  
   ctr1\_updt\_threshold, 87  
   raw\_data, 86  
   timestamp\_threshold, 87  
 EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_OFFSET  
   ezdp\_counter\_defs.h, 281  
 EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_SIZE  
   ezdp\_counter\_defs.h, 281  
 EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_WORD\_OF  
   FSET  
   ezdp\_counter\_defs.h, 281  
 EZDP\_HIER\_TB\_CTR\_CFG\_APP\_BITS\_WORD\_SE  
   LECT  
   ezdp\_counter\_defs.h, 281  
 EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_FAIL\_THR  
   ESHOLD\_OFFSET  
   ezdp\_counter\_defs.h, 281  
 EZDP\_HIER\_TB\_CTR\_CFG\_CTR\_SUM\_FAIL\_THR  
   ESHOLD\_SIZE  
   ezdp\_counter\_defs.h, 281

EZDP_HIER_TB_CTR_CFG_CTR_SUM_FAIL_THR ESHOLD_WORD_OFFSET ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESH OLD_WORD_OFFSET ezdp_counter_defs.h, 281
EZDP_HIER_TB_CTR_CFG_CTR_SUM_FAIL_THR ESHOLD_WORD_SELECT ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESH OLD_WORD_SELECT ezdp_counter_defs.h, 281
EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_TH RESHOLD_OFFSET ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_RESERVED0_1_OFFS ET ezdp_counter_defs.h, 280
EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_TH RESHOLD_SIZE ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_RESERVED0_1_SIZE ezdp_counter_defs.h, 280
EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_TH RESHOLD_WORD_OFFSET ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_RESERVED22_26_OFF SET ezdp_counter_defs.h, 281
EZDP_HIER_TB_CTR_CFG_CTR_SUM_UPDT_TH RESHOLD_WORD_SELECT ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_RESERVED22_26_SIZ E ezdp_counter_defs.h, 281
EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESH OLD_OFFSET ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_RESERVED63_OFFSE T ezdp_counter_defs.h, 282
EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESH OLD_SIZE ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_RESERVED63_SIZE ezdp_counter_defs.h, 282
EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESH OLD_WORD_OFFSET ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRES HOLD_OFFSET ezdp_counter_defs.h, 282
EZDP_HIER_TB_CTR_CFG_CTR0_FAIL_THRESH OLD_WORD_SELECT ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRES HOLD_SIZE ezdp_counter_defs.h, 282
EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESH OLD_OFFSET ezdp_counter_defs.h, 282	EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRES HOLD_WORD_OFFSET ezdp_counter_defs.h, 282
EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESH OLD_SIZE ezdp_counter_defs.h, 281	EZDP_HIER_TB_CTR_CFG_TIMESTAMP_THRES HOLD_WORD_SELECT ezdp_counter_defs.h, 282
EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESH OLD_WORD_OFFSET ezdp_counter_defs.h, 282	EZDP_HIER_TB_CTR_CFG_WORD_COUNT ezdp_counter_defs.h, 282
EZDP_HIER_TB_CTR_CFG_CTR0_UPDT_THRESH OLD_WORD_SELECT ezdp_counter_defs.h, 282	ezdp_hier_tb_result, 88
EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESH OLD_OFFSET ezdp_counter_defs.h, 281	__pad0__, 88
EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESH OLD_SIZE ezdp_counter_defs.h, 281	__pad1__, 89
EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESH OLD_WORD_OFFSET ezdp_counter_defs.h, 281	__pad2__, 89
EZDP_HIER_TB_CTR_CFG_CTR1_FAIL_THRESH OLD_WORD_SELECT ezdp_counter_defs.h, 281	app_bits, 89
EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESH OLD_OFFSET ezdp_counter_defs.h, 281	ctr0, 89
EZDP_HIER_TB_CTR_CFG_CTR1_UPDT_THRESH OLD_SIZE ezdp_counter_defs.h, 280	ctr1, 88
	fail, 88
	raw_data, 88
	update_task, 88
	EZDP_HIER_TB_RESULT_APP_BITS_OFFSET ezdp_counter_defs.h, 283
	EZDP_HIER_TB_RESULT_APP_BITS_SIZE ezdp_counter_defs.h, 283
	EZDP_HIER_TB_RESULT_APP_BITS_WORD_OFF SET ezdp_counter_defs.h, 283
	EZDP_HIER_TB_RESULT_APP_BITS_WORD_SEL ECT ezdp_counter_defs.h, 283
	EZDP_HIER_TB_RESULT_CTR0_OFFSET ezdp_counter_defs.h, 283
	EZDP_HIER_TB_RESULT_CTR0_SIZE ezdp_counter_defs.h, 283

EZDP\_HIER\_TB\_RESULT\_CTRL0\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_CTRL0\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_CTRL1\_OFFSET  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_CTRL1\_SIZE  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_CTRL1\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_CTRL1\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_FAIL\_MASK  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_FAIL\_OFFSET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_FAIL\_SIZE  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_FAIL\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_FAIL\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_RESERVED0\_9\_OFFSET  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_RESERVED0\_9\_SIZE  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_RESERVED56\_63\_OFFSET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_RESERVED56\_63\_SIZE  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_RESERVED82\_95\_OFFSET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_RESERVED82\_95\_SIZE  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_RESULT\_STATE\_OFFSET  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_STATE\_SIZE  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_STATE\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_STATE\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_MASK  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_OFFSET  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_SIZE  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_UPDATE\_TASK\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 282  
 EZDP\_HIER\_TB\_RESULT\_WORD\_COUNT  
 ezdp\_counter\_defs.h, 283  
 ezdp\_hier\_tb\_state  
 ezdp\_counter\_defs.h, 293  
 ezdp\_hier\_tb\_ug\_app\_bits, 90  
 \_\_pad0\_\_, 90  
 app\_bits, 90  
 color\_state\_g, 90  
 color\_state\_y, 90  
 eigth\_mode\_ret\_bits, 90  
 raw\_data, 90  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_APP\_BITS\_OFFSET  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_APP\_BITS\_SIZE  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STATE\_G\_OFFSET  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STATE\_G\_SIZE  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STATE\_Y\_OFFSET  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_COLOR\_STATE\_Y\_SIZE  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_EIGHTH\_MODE\_RET\_BITS\_OFFSET  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_EIGHTH\_MODE\_RET\_BITS\_SIZE  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_RESERVED24\_31\_OFFSET  
 ezdp\_counter\_defs.h, 280  
 EZDP\_HIER\_TB\_UG\_APP\_BITS\_RESERVED24\_31\_SIZE  
 ezdp\_counter\_defs.h, 280  
 ezdp\_hier\_tb\_ug\_app\_bits\_t  
 ezdp\_counter\_defs.h, 292  
 ezdp\_hier\_tb\_update, 92  
 \_\_pad0\_\_, 93  
 app\_bits, 93  
 clr\_ctr, 92  
 cond\_set\_active\_state, 92  
 raw\_data, 92  
 set\_active\_state, 92  
 set\_app\_bits, 92  
 EZDP\_HIER\_TB\_UPDATE\_APP\_BITS\_OFFSET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_APP\_BITS\_SIZE  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_BES  
 ezdp\_counter\_defs.h, 292  
 EZDP\_HIER\_TB\_UPDATE\_CLR\_CTR\_MASK  
 ezdp\_counter\_defs.h, 284  
 EZDP\_HIER\_TB\_UPDATE\_CLR\_CTR\_OFFSET  
 ezdp\_counter\_defs.h, 284

EZDP\_HIER\_TB\_UPDATE\_CLR\_CTR\_SIZE  
 ezdp\_counter\_defs.h, 284  
 EZDP\_HIER\_TB\_UPDATE\_COND\_SET\_ACTIVE\_S  
 TATE\_MASK  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_COND\_SET\_ACTIVE\_S  
 TATE\_OFFSET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_COND\_SET\_ACTIVE\_S  
 TATE\_SIZE  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_RESERVED24\_27\_OFF  
 SET  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_RESERVED24\_27\_SIZE  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_SET\_ACTIVE\_STATE\_  
 MASK  
 ezdp\_counter\_defs.h, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_ACTIVE\_STATE\_  
 OFFSET  
 ezdp\_counter\_defs.h, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_ACTIVE\_STATE\_  
 SIZE  
 ezdp\_counter\_defs.h, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_APP\_BITS\_MASK  
 ezdp\_counter\_defs.h, 284  
 EZDP\_HIER\_TB\_UPDATE\_SET\_APP\_BITS\_OFFSE  
 T  
 ezdp\_counter\_defs.h, 283  
 EZDP\_HIER\_TB\_UPDATE\_SET\_APP\_BITS\_SIZE  
 ezdp\_counter\_defs.h, 283  
 ezdp\_hier\_tb\_update\_t  
 ezdp\_counter\_defs.h, 292  
 EZDP\_HIGH\_LEVEL  
 ezdp\_job\_defs.h, 431  
 EZDP\_IMEM\_1\_CLUSTER\_DATA  
 ezdp.h, 192  
 EZDP\_IMEM\_16\_CLUSTER\_DATA  
 ezdp.h, 192  
 EZDP\_IMEM\_2\_CLUSTER\_DATA  
 ezdp.h, 192  
 EZDP\_IMEM\_4\_CLUSTER\_DATA  
 ezdp.h, 192  
 EZDP\_IMEM\_ALL\_CLUSTER\_DATA  
 ezdp.h, 193  
 EZDP\_IMEM\_HALF\_CLUSTER\_DATA  
 ezdp.h, 192  
 EZDP\_IMEM\_PRIVATE\_DATA  
 ezdp.h, 192  
 EZDP\_INACTIVE  
 ezdp\_counter\_defs.h, 293  
 ezdp\_inc\_bits\_bitwise\_ctr  
 ezdp\_counter.h, 244  
 ezdp\_inc\_bits\_bitwise\_ctr\_async  
 ezdp\_counter.h, 244  
 ezdp\_inc\_dual\_ctr  
 ezdp\_counter.h, 239  
 ezdp\_inc\_dual\_ctr\_async  
 ezdp\_counter.h, 239  
 ezdp\_inc\_hier\_tb\_ctr  
 ezdp\_counter.h, 255  
 ezdp\_inc\_hier\_tb\_ctr\_async  
 ezdp\_counter.h, 255  
 ezdp\_inc\_single\_ctr  
 ezdp\_counter.h, 234  
 ezdp\_inc\_single\_ctr\_async  
 ezdp\_counter.h, 235  
 ezdp\_inc\_tb\_ctr  
 ezdp\_counter.h, 251  
 ezdp\_inc\_tb\_ctr\_async  
 ezdp\_counter.h, 252  
 ezdp\_inc\_tm\_imem\_buf\_ctr  
 ezdp\_frame.h, 375  
 ezdp\_inc\_tm\_imem\_buf\_ctr\_async  
 ezdp\_frame.h, 375  
 ezdp\_inc\_watchdog\_ctr  
 ezdp\_counter.h, 258  
 ezdp\_inc\_watchdog\_ctr\_async  
 ezdp\_counter.h, 258  
 EZDP\_INDEX  
 ezdp\_search\_defs.h, 552  
 EZDP\_INDEX\_16B\_DATA  
 ezdp\_search\_defs.h, 552  
 EZDP\_INDEX\_32B\_DATA  
 ezdp\_search\_defs.h, 552  
 EZDP\_INDEX\_4B\_DATA  
 ezdp\_search\_defs.h, 552  
 EZDP\_INDEX\_8B\_DATA  
 ezdp\_search\_defs.h, 552  
 ezdp\_init\_alg\_tcam\_struct\_desc  
 ezdp\_search.h, 516  
 ezdp\_init\_ctr\_msg\_queue\_desc  
 ezdp\_counter.h, 259  
 ezdp\_init\_frame  
 ezdp\_frame.h, 377  
 ezdp\_init\_global  
 ezdp.h, 193  
 ezdp\_init\_hash\_struct\_desc  
 ezdp\_search.h, 510  
 ezdp\_init\_list  
 ezdp\_queue.h, 503  
 ezdp\_init\_local  
 ezdp.h, 193  
 ezdp\_init\_memory\_pool  
 ezdp\_pool.h, 495  
 ezdp\_init\_pci\_queue\_desc  
 ezdp\_pci.h, 473  
 EZDP\_INIT\_PCI\_QUEUE\_DESC\_WORK\_AREA\_SI  
 ZE  
 ezdp\_pci\_defs.h, 491  
 ezdp\_init\_pci\_queue\_desc\_working\_area\_t  
 ezdp\_pci\_defs.h, 491  
 ezdp\_init\_posted\_ctr\_msg\_queue\_desc  
 ezdp\_counter.h, 263  
 ezdp\_init\_qlock  
 ezdp\_lock.h, 436  
 ezdp\_init\_ring  
 ezdp\_queue.h, 501  
 ezdp\_init\_spinlock\_ext\_addr



ezdp\_lock.h, 435  
 ezdp\_init\_spinlock\_sum\_addr  
 ezdp\_lock.h, 435  
 ezdp\_init\_table\_struct\_desc  
 ezdp\_search.h, 507  
 ezdp\_init\_tm\_reporting\_desc  
 ezdp\_job.h, 407  
 ezdp\_init\_ultra\_ip\_struct\_desc  
 ezdp\_search.h, 514  
 ezdp\_input\_queue\_status, 94  
 \_\_pad0\_\_, 94  
 dispatched\_job, 94  
 outstanding\_job, 95  
 raw\_data, 94  
 ready, 94  
 size, 94  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_L  
 EVEL\_OFFSET  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_L  
 EVEL\_SIZE  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_L  
 EVEL\_WORD\_OFFSET  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_L  
 EVEL\_WORD\_SELECT  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JO  
 B\_OFFSET  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JO  
 B\_SIZE  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JO  
 B\_WORD\_OFFSET  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JO  
 B\_WORD\_SELECT  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_  
 JOB\_OFFSET  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_  
 JOB\_SIZE  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_  
 JOB\_WORD\_OFFSET  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_  
 JOB\_WORD\_SELECT  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_MASK  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_OFFSET  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_SIZE  
 ezdp\_job\_defs.h, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_WORD\_  
 OFFSET  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_WORD\_  
 SELECT  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_RESERVED19\_31  
 \_OFFSET  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_RESERVED19\_31  
 \_SIZE  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_OFFSET  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_SIZE  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_WORD\_OF  
 FSET  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_WORD\_SE  
 LECT  
 ezdp\_job\_defs.h, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_WORD\_COUNT  
 ezdp\_job\_defs.h, 428  
 EZDP\_INT\_MEM  
 ezdp\_frame\_defs.h, 388  
 EZDP\_INT\_MEM\_BUF\_BUDGET  
 ezdp\_job\_defs.h, 431  
 EZDP\_INTERFACE\_DEST  
 ezdp\_job\_defs.h, 432  
 ezdp\_internal\_mem\_space  
 ezdp\_memory\_defs.h, 469  
 EZDP\_INTERNAL\_MS  
 ezdp\_memory\_defs.h, 470  
 ezdp\_is\_null\_sum\_addr  
 ezdp\_memory.h, 460  
 ezdp\_job.h  
 ezdp\_alloc\_job\_id, 393  
 ezdp\_alloc\_job\_id\_async, 394  
 ezdp\_alloc\_multi\_job\_id, 394  
 ezdp\_alloc\_multi\_job\_id\_async, 394  
 ezdp\_calc\_tm\_queue\_depth\_handle, 407  
 ezdp\_cancel\_job\_request, 398  
 ezdp\_check\_notice, 404  
 ezdp\_clear\_notice, 404  
 ezdp\_container\_info, 403  
 ezdp\_container\_job\_count, 403  
 ezdp\_discard\_job, 402  
 ezdp\_discard\_job\_id, 401  
 ezdp\_discard\_job\_id\_async, 402  
 ezdp\_free\_job\_id, 394  
 ezdp\_free\_job\_id\_async, 395  
 ezdp\_get\_tm\_queue\_depth, 408  
 ezdp\_handle\_notice, 405  
 ezdp\_init\_tm\_reporting\_desc, 407  
 ezdp\_job\_alloc\_failed, 394  
 ezdp\_load\_job, 396  
 ezdp\_load\_job\_async, 396  
 ezdp\_notice\_pending, 404  
 ezdp\_notifier, 393  
 ezdp\_notifier\_t, 393  
 ezdp\_notify\_cpu, 403

ezdp\_read\_congestion\_status, 405  
 ezdp\_read\_flow\_control\_status, 405  
 ezdp\_read\_free\_job, 395  
 ezdp\_read\_global\_budget, 405  
 ezdp\_read\_pmu\_app\_schlr\_status, 407  
 ezdp\_read\_pmu\_discard\_output\_queue\_status, 406  
 ezdp\_read\_pmu\_group\_schlr\_status, 407  
 ezdp\_read\_pmu\_input\_queue\_congestion, 405  
 ezdp\_read\_pmu\_input\_queue\_status, 406  
 ezdp\_read\_pmu\_tm\_bypass\_output\_queue\_status, 406  
 ezdp\_read\_pmu\_tm\_output\_queue\_status, 406  
 ezdp\_rebudget\_job, 395  
 ezdp\_rebudget\_job\_async, 395  
 ezdp\_receive\_job, 398  
 ezdp\_request\_job\_id, 397  
 ezdp\_send\_job\_container, 403  
 ezdp\_send\_job\_id\_container, 402  
 ezdp\_send\_job\_id\_container\_async, 403  
 ezdp\_send\_job\_id\_to\_interface, 399  
 ezdp\_send\_job\_id\_to\_interface\_async, 400  
 ezdp\_send\_job\_id\_to\_queue, 398  
 ezdp\_send\_job\_id\_to\_queue\_async, 398  
 ezdp\_send\_job\_id\_to\_tm, 399  
 ezdp\_send\_job\_id\_to\_tm\_async, 399  
 ezdp\_send\_job\_to\_interface, 401  
 ezdp\_send\_job\_to\_queue, 400  
 ezdp\_send\_job\_to\_tm, 400  
 ezdp\_store\_job, 396  
 ezdp\_store\_job\_async, 396  
 ezdp\_store\_job\_container, 397  
 ezdp\_store\_job\_container\_async, 397  
 ezdp\_update\_job\_id\_queue, 400  
 ezdp\_update\_job\_queue, 401  
 ezdp\_valid\_tm\_queue\_depth\_handle, 408  
 ezdp\_wait\_for\_event, 404  
 ezdp\_wait\_for\_job\_id, 397  
 ezdp\_wait\_for\_notice, 404  
 ezdp\_job\_alloc\_failed  
 ezdp\_job.h, 394  
 EZDP\_JOB\_BUDGET  
   ezdp\_job\_defs.h, 431  
 ezdp\_job\_container\_cmd  
   ezdp\_job\_defs.h, 433  
 ezdp\_job\_container\_cmd\_desc, 96  
   \_\_pad0\_\_, 96  
   discard\_info, 97  
   job\_id, 96  
   queue\_info, 96  
   raw\_data, 96  
   transmit\_info, 96  
   u, 97  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_COMMAND\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_COMMAND\_SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_JOB\_ID\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_JOB\_ID\_SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_RESERVE\_D0\_11\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_RESERVE\_D0\_11\_SIZE  
   ezdp\_job\_defs.h, 429  
 ezdp\_job\_container\_cmd\_desc\_t  
   ezdp\_job\_defs.h, 430  
 ezdp\_job\_container\_desc, 98  
   \_\_pad0\_\_, 98  
   \_\_pad1\_\_, 98  
   info, 98  
   job\_budget\_id, 98  
   job\_commands, 98  
   raw\_data, 98  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_OFFSET  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_SIZE  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_WORD\_OFFSET  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_WORD\_SELECT  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_WORD\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_WORD\_SELECT  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_MAX\_NUM\_OF\_JOBS  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED0\_15\_OFFSET  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED0\_15\_SIZE  
   ezdp\_job\_defs.h, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED29\_31\_OFFSET  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED29\_31\_SIZE  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_WORD\_COUNT  
   ezdp\_job\_defs.h, 430  
 ezdp\_job\_container\_info\_t  
   ezdp\_job\_defs.h, 430

```

ezdp_job_defs.h
EZDP_16BITS_REPORT, 434
EZDP_32BITS_REPORT, 434
EZDP_8BITS_REPORT, 434
EZDP_ALLOW_REORDER, 433
EZDP_APP_SCHLR_STATUS_BUSY_MASK, 429
EZDP_APP_SCHLR_STATUS_BUSY_OFFSET,
    429
EZDP_APP_SCHLR_STATUS_BUSY_SIZE, 429
EZDP_APP_SCHLR_STATUS_DISPATCHED_JO
    B_OFFSET, 429
EZDP_APP_SCHLR_STATUS_DISPATCHED_JO
    B_SIZE, 429
EZDP_APP_SCHLR_STATUS_ENABLE_MASK,
    429
EZDP_APP_SCHLR_STATUS_ENABLE_OFFSET
    , 429
EZDP_APP_SCHLR_STATUS_ENABLE_SIZE,
    429
EZDP_APP_SCHLR_STATUS_RESERVED13_OF
    FSET, 429
EZDP_APP_SCHLR_STATUS_RESERVED13_SIZ
    E, 429
ezdp_app_schlr_status_t, 430
ezdp_budget_type, 431
EZDP_CAN_DROP, 432
EZDP_CHANNEL_NODE, 431
ezdp_congestion_level, 431
EZDP_CONGESTION_LEVEL_0, 432
EZDP_CONGESTION_LEVEL_1, 432
EZDP_CONGESTION_LEVEL_2, 433
EZDP_CONGESTION_LEVEL_3, 433
EZDP_CONGESTION_LEVEL_4, 433
EZDP_CONGESTION_STATUS_EMEM_BUF_CO
    NGESTION_LEVEL_OFFSET, 426
EZDP_CONGESTION_STATUS_EMEM_BUF_CO
    NGESTION_LEVEL_SIZE, 426
EZDP_CONGESTION_STATUS_EMEM_BUF_G
    UARANTEE_MASK, 426
EZDP_CONGESTION_STATUS_EMEM_BUF_G
    UARANTEE_OFFSET, 426
EZDP_CONGESTION_STATUS_EMEM_BUF_G
    UARANTEE_SIZE, 426
EZDP_CONGESTION_STATUS_IMEM_BUF_CO
    NGESTION_LEVEL_OFFSET, 426
EZDP_CONGESTION_STATUS_IMEM_BUF_CO
    NGESTION_LEVEL_SIZE, 426
EZDP_CONGESTION_STATUS_IMEM_BUF_GU
    ARANTEE_MASK, 426
EZDP_CONGESTION_STATUS_IMEM_BUF_GU
    ARANTEE_OFFSET, 426
EZDP_CONGESTION_STATUS_IMEM_BUF_GU
    ARANTEE_SIZE, 426
EZDP_CONGESTION_STATUS_JOB_CONGESTI
    ON_LEVEL_OFFSET, 427
EZDP_CONGESTION_STATUS_JOB_CONGESTI
    ON_LEVEL_SIZE, 426
EZDP_CONGESTION_STATUS_JOB_GUARANT
    EE_MASK, 427
EZDP_CONGESTION_STATUS_JOB_GUARANT
    EE_OFFSET, 427
EZDP_CONGESTION_STATUS_JOB_GUARANT
    EE_SIZE, 427
EZDP_CONGESTION_STATUS_PORT_CONGES
    TION_LEVEL_OFFSET, 427
EZDP_CONGESTION_STATUS_PORT_CONGES
    TION_LEVEL_SIZE, 427
EZDP_CONGESTION_STATUS_RESERVED11_
    OFFSET, 427
EZDP_CONGESTION_STATUS_RESERVED11_S
    IZE, 427
EZDP_CONGESTION_STATUS_RESERVED14_1
    5_OFFSET, 427
EZDP_CONGESTION_STATUS_RESERVED14_1
    5_SIZE, 427
EZDP_CONGESTION_STATUS_RESERVED3_O
    FFSET, 426
EZDP_CONGESTION_STATUS_RESERVED3_SI
    ZE, 426
EZDP_CONGESTION_STATUS_RESERVED7_O
    FFSET, 426
EZDP_CONGESTION_STATUS_RESERVED7_SI
    ZE, 426
ezdp_congestion_status_t, 430
EZDP_CRITICAL_LEVEL, 431
EZDP_DISCARD, 433
EZDP_DONT_DROP, 432
EZDP_EXPLICIT_PSID, 432
EZDP_EXT_MEM_BUF_BUDGET, 431
EZDP_FIXED_BASE, 431
ezdp_flow_control_congestion_level, 432
ezdp_flow_control_node, 430
EZDP_FLOW_CONTROL_STATUS_CONGESTIO
    N_LEVEL_OFFSET, 427
EZDP_FLOW_CONTROL_STATUS_CONGESTIO
    N_LEVEL_SIZE, 427
EZDP_FLOW_CONTROL_STATUS_ENABLE_M
    ASK, 427
EZDP_FLOW_CONTROL_STATUS_ENABLE_O
    FFSET, 427
EZDP_FLOW_CONTROL_STATUS_ENABLE_SI
    ZE, 427
EZDP_FLOW_CONTROL_STATUS_RESERVED4
    _7_OFFSET, 427
EZDP_FLOW_CONTROL_STATUS_RESERVED4
    _7_SIZE, 427
ezdp_flow_control_status_t, 430
EZDP_FREE, 433
EZDP_GLOBAL_NODE, 431
EZDP_GROUP_NODE, 431
EZDP_GROUP_SCHLR_STATUS_DISPATCHED
    _JOB_OFFSET, 429
EZDP_GROUP_SCHLR_STATUS_DISPATCHED
    _JOB_SIZE, 429
EZDP_GROUP_SCHLR_STATUS_RESERVED13
    _15_OFFSET, 429
EZDP_GROUP_SCHLR_STATUS_RESERVED13
    _15_SIZE, 429
ezdp_group_schlr_status_t, 430

```

EZDP\_HANDLE\_NOTICE, 433  
 EZDP\_HIGH\_LEVEL, 431  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_LEVEL\_OFFSET, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_LEVEL\_SIZE, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_LEVEL\_WORD\_OFFSET, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_CONGESTION\_LEVEL\_WORD\_SELECT, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JOB\_OFFSET, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JOB\_SIZE, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JOB\_WORD\_OFFSET, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_DISPATCHED\_JOB\_WORD\_SELECT, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_JOB\_OFFSET, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_JOB\_SIZE, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_JOB\_WORD\_OFFSET, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_OUTSTANDING\_JOB\_WORD\_SELECT, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_MASK, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_OFFSET, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_SIZE, 427  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_WORD\_OFFSET, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_READY\_WORD\_SELECT, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_RESERVED19\_31\_OFFSET, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_RESERVED19\_31\_SIZE, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_OFFSET, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_SIZE, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_WORD\_OFFSET, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_SIZE\_WORD\_SELECT, 428  
 EZDP\_INPUT\_QUEUE\_STATUS\_WORD\_COUNT, 428  
 EZDP\_INT\_MEM\_BUF\_BUDGET, 431  
 EZDP\_INTERFACE\_DEST, 432  
 EZDP\_JOB\_BUDGET, 431  
 ezdp\_job\_container\_cmd, 433  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_COMMAND\_OFFSET, 429  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_RESERVED0\_11\_OFFSET, 429  
 EZDP\_JOB\_CONTAINER\_CMD\_DESC\_RESERVED0\_11\_SIZE, 429  
 ezdp\_job\_container\_cmd\_desc\_t, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_OFFSET, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_SIZE, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_WORD\_OFFSET, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_INFO\_WORD\_SELECT, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_OFFSET, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_SIZE, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_WORD\_OFFSET, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_JOB\_BUDGET\_ID\_WORD\_SELECT, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_MAX\_NUM\_OF\_JOBS, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED0\_15\_OFFSET, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED0\_15\_SIZE, 429  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED29\_31\_OFFSET, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_RESERVED29\_31\_SIZE, 430  
 EZDP\_JOB\_CONTAINER\_DESC\_WORD\_COUNT, 430  
 ezdp\_job\_container\_info\_t, 430  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED0\_9\_OFFSET, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED0\_9\_SIZE, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED1\_15\_OFFSET, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED1\_15\_SIZE, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_MASK, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_OFFSET, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_SIZE, 426  
 ezdp\_job\_discard\_cmd\_info\_t, 430  
 ezdp\_job\_flags, 433  
 ezdp\_job\_id\_t, 430  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_RESERVED8\_15\_OFFSET, 425  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_RESERVED8\_15\_SIZE, 425  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_SIDE\_MASK, 425

EZDP_JOB_QUEUE_CMD_INFO_SIDE_OFFSET, 425	EZDP_JOB_RX_INFO_RESERVED112_127_SIZE, 421
EZDP_JOB_QUEUE_CMD_INFO_SIDE_SIZE, 425	EZDP_JOB_RX_INFO_SEQ_NUMBER_OFFSET, 422
ezdp_job_queue_cmd_info_t, 430	EZDP_JOB_RX_INFO_SEQ_NUMBER_SIZE, 422
EZDP_JOB_QUEUE_CMD_INFO_TARGET_QUEUE_OFFSET, 425	EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_MASK, 421
EZDP_JOB_QUEUE_CMD_INFO_TARGET_QUEUE_SIZE, 425	EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_OFFSET, 421
EZDP_JOB_RX_CONFIRMATION_INFO_RESERVED8_31_OFFSET, 419	EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_SIZE, 421
EZDP_JOB_RX_CONFIRMATION_INFO_RESERVED8_31_SIZE, 419	EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_WORD_OFFSET, 421
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_NSEC_OFFSET, 420	EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_WORD_SELECT, 421
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_NSEC_SIZE, 420	EZDP_JOB_RX_INFO_SEQ_NUMBER_WORD_OFFSET, 422
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_NSEC_WORD_OFFSET, 420	EZDP_JOB_RX_INFO_SEQ_NUMBER_WORD_SELECT, 422
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_NSEC_WORD_SELECT, 420	EZDP_JOB_RX_INFO_SIDE_MASK, 421
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_SEC_OFFSET, 419	EZDP_JOB_RX_INFO_SIDE_OFFSET, 421
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_SEC_SIZE, 419	EZDP_JOB_RX_INFO_SIDE_SIZE, 421
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_SEC_WORD_OFFSET, 419	EZDP_JOB_RX_INFO_SIDE_WORD_OFFSET, 421
EZDP_JOB_RX_CONFIRMATION_INFO_TIMES_TAMP_SEC_WORD_SELECT, 419	EZDP_JOB_RX_INFO_SIDE_WORD_SELECT, 421
EZDP_JOB_RX_CONFIRMATION_INFO_WORD_COUNT, 420	EZDP_JOB_RX_INFO_SOURCE_QUEUE_OFFSET, 421
EZDP_JOB_RX_INFO_GROSS_CHECKSUM_OFFSET, 421	EZDP_JOB_RX_INFO_SOURCE_QUEUE_SIZE, 421
EZDP_JOB_RX_INFO_GROSS_CHECKSUM_SIZE, 421	EZDP_JOB_RX_INFO_SOURCE_QUEUE_WORD_OFFSET, 421
EZDP_JOB_RX_INFO_GROSS_CHECKSUM_WORD_OFFSET, 421	EZDP_JOB_RX_INFO_SOURCE_QUEUE_WORD_SELECT, 421
EZDP_JOB_RX_INFO_GROSS_CHECKSUM_WORD_SELECT, 421	EZDP_JOB_RX_INFO_WORD_COUNT, 422
EZDP_JOB_RX_INFO_IS_SERVICE_READY_MASK, 422	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_MASK, 417
EZDP_JOB_RX_INFO_IS_SERVICE_READY_OFFSET, 422	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_OFFSET, 417
EZDP_JOB_RX_INFO_IS_SERVICE_READY_SIZE, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_SIZE, 417
EZDP_JOB_RX_INFO_IS_SERVICE_READY_WORD_OFFSET, 422	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_WORD_OFFSET, 417
EZDP_JOB_RX_INFO_IS_SERVICE_READY_WORD_SELECT, 422	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_WORD_SELECT, 417
EZDP_JOB_RX_INFO_RESERVED104_107_OFFSET, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_MASK, 417
EZDP_JOB_RX_INFO_RESERVED104_107_SIZE, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_OFFSET, 417
EZDP_JOB_RX_INFO_RESERVED108_109_OFFSET, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_SIZE, 417
EZDP_JOB_RX_INFO_RESERVED108_109_SIZE, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_WORD_OFFSET, 417
EZDP_JOB_RX_INFO_RESERVED112_127_OFFSET, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_WORD_SELECT, 417
	EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUFFER_CONGESTION_LEVEL_OFFSET, 418
	EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUFFER_CONGESTION_LEVEL_SIZE, 418

EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_WORD_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED11_OFFSET, 417
EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_WORD_SELECT, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED11_SIZE, 417
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED14_OFFSET, 418
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_SIZE, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED14_SIZE, 418
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_WORD_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED15_OFFSET, 418
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_WORD_SELECT, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED15_SIZE, 418
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCCESS_PARSING_FLAG_MASK, 418	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_OFFSET, 419
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCCESS_PARSING_FLAG_OFFSET, 417	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_SIZE, 419
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCCESS_PARSING_FLAG_SIZE, 417	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_WORD_OFFSET, 419
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCCESS_PARSING_FLAG_WORD_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_NSEC_WORD_SELECT, 419
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCCESS_PARSING_FLAG_WORD_SELECT, 418	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_PSEC_OFFSET, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_PSEC_SIZE, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_SIZE, 418	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_PSEC_WORD_OFFSET, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_WORD_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_PSEC_WORD_SELECT, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_WORD_SELECT, 418	EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_MASK, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_OFFSET, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_SIZE, 418	EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_SIZE, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_WORD_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_WORD_OFFSET, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_WORD_SELECT, 418	EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_FLAG_WORD_SELECT, 417
EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_OFFSET, 418	EZDP_JOB_RX_INTERFACE_INFO_WORD_COUNT, 419
EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_SIZE, 418	EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_OFFSET, 419
EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_WORD_OFFSET, 419	EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_SIZE, 419
EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_WORD_SELECT, 419	EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_WORD_OFFSET, 419
EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_OFFSET, 419	EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_ID_WORD_SELECT, 419
EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_SIZE, 419	EZDP_JOB_RX_LOOPBACK_INFO_RESERVED0_15_OFFSET, 419
EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_WORD_OFFSET, 419	EZDP_JOB_RX_LOOPBACK_INFO_RESERVED0_15_SIZE, 419
EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_WORD_SELECT, 419	EZDP_JOB_RX_LOOPBACK_INFO_RESERVED32_63_OFFSET, 419
EZDP_JOB_RX_INTERFACE_INFO_RESERVED10_OFFSET, 417	EZDP_JOB_RX_LOOPBACK_INFO_RESERVED32_63_SIZE, 419
EZDP_JOB_RX_INTERFACE_INFO_RESERVED10_SIZE, 417	EZDP_JOB_RX_LOOPBACK_INFO_WORD_COUNT, 419
	EZDP_JOB_RX_TIMER_INFO_EVENT_ID_OFFSET, 420

EZDP\_JOB\_RX\_TIMER\_INFO\_EVENT\_ID\_SIZE, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_EVENT\_ID\_WORD\_OFFSET, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_EVENT\_ID\_WORD\_SELECT, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_RESERVED0\_8\_OFFSET, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_RESERVED0\_8\_SIZE, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_RESERVED16\_31\_OFFSET, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_RESERVED16\_31\_SIZE, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_TIMER\_ID\_OFFSET, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_TIMER\_ID\_SIZE, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_TIMER\_ID\_WORD\_OFFSET, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_TIMER\_ID\_WORD\_SELECT, 420  
 EZDP\_JOB\_RX\_TIMER\_INFO\_WORD\_COUNT, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0\_OFFSET, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0\_SIZE, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0\_WORD\_OFFSET, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0\_WORD\_SELECT, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1\_OFFSET, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1\_SIZE, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1\_WORD\_OFFSET, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1\_WORD\_SELECT, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_WORD\_COUNT, 421  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_DESTINATION\_MASK, 426  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_DESTINATION\_OFFSET, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_DESTINATION\_SIZE, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_OUTPUT\_CHANNEL\_OFFSET, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_OUTPUT\_CHANNEL\_SIZE, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_RESERVE\_D12\_15\_OFFSET, 426  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_RESERVE\_D12\_15\_SIZE, 426  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_MASK, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_OFFSET, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_SIZE, 425  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_WORD\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_WORD\_SELECT, 423  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_SIZE, 422  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_WORD\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_WORD\_SELECT, 422  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CONTROL\_MASK, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CONTROL\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CONTROL\_SIZE, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CONTROL\_WORD\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CONTROL\_WORD\_SELECT, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_SIZE, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_WORD\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_WORD\_SELECT, 424  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_ID\_SELECT\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_ID\_SELECT\_SIZE, 422  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_ID\_SELECT\_WORD\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_ID\_SELECT\_WORD\_SELECT, 422  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_MODE\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_MODE\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_MODE\_WORD\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_PACKET\_SWITCH\_MODE\_WORD\_SELECT, 423  
 EZDP\_JOB\_TX\_INFO\_QOS\_BYPASS\_MASK, 423  
 EZDP\_JOB\_TX\_INFO\_QOS\_BYPASS\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_QOS\_BYPASS\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_QOS\_BYPASS\_WORD\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_QOS\_BYPASS\_WORD\_SELECT, 423

EZDP\_JOB\_TX\_INFO\_RESERVED102\_103\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_RESERVED102\_103\_SIZE, 424  
 EZDP\_JOB\_TX\_INFO\_RESERVED25\_29\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_RESERVED25\_29\_SIZE, 422  
 EZDP\_JOB\_TX\_INFO\_RESERVED52\_55\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_RESERVED52\_55\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_RESERVED59\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_RESERVED59\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_RESERVED61\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_RESERVED61\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_RESERVED88\_90\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_RESERVED88\_90\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_SIDE\_MASK, 423  
 EZDP\_JOB\_TX\_INFO\_SIDE\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_SIDE\_SIZE, 422  
 EZDP\_JOB\_TX\_INFO\_SIDE\_WORD\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_SIDE\_WORD\_SELECT, 422  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE1\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE1\_SIZE, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE1\_WORD\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE1\_WORD\_SELECT, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE2\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE2\_SIZE, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE2\_WORD\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_CODE\_PROFILE2\_WORD\_SELECT, 424  
 EZDP\_JOB\_TX\_INFO\_STAT\_STREAM\_ID\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_STAT\_STREAM\_ID\_SIZE, 423  
 EZDP\_JOB\_TX\_INFO\_STAT\_STREAM\_ID\_WORD\_OFFSET, 423  
 EZDP\_JOB\_TX\_INFO\_STAT\_STREAM\_ID\_WORD\_SELECT, 423  
 EZDP\_JOB\_TX\_INFO\_WORD\_COUNT, 425  
 EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_SIZE, 424  
 EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_WORD\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_WORD\_SELECT, 425  
 EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_SIZE, 422  
 EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_WORD\_OFFSET, 422  
 EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_WORD\_SELECT, 422  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_OFFSET, 424  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_SIZE, 424  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_WORD\_OFFSET, 425  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_WORD\_SELECT, 425  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_OFFSET, 425  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_SIZE, 425  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_WORD\_OFFSET, 425  
 EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_WORD\_SELECT, 425  
 EZDP\_LOW\_LEVEL, 431  
 EZDP\_MEDIUM\_LEVEL, 431  
 EZDP\_NEVER\_DROP, 432  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_MASK, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_OFFSET, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_SIZE, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_MASK, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_OFFSET, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_SIZE, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_RESERVED18\_31\_OFFSET, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_RESERVED18\_31\_SIZE, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_SIZE\_OFFSET, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_SIZE\_SIZE, 428  
 ezdp\_output\_queue\_status\_t, 430  
 EZDP\_PORT\_NODE, 431  
 EZDP\_QUEUE, 433  
 EZDP\_QUEUE\_WITH\_SEQ\_NUM, 433



ezdp\_report\_size, 433  
 EZDP\_SET\_SEQ\_NUM, 433  
 EZDP\_TM\_DEST, 432  
 EZDP\_TM\_REPORT\_WORK\_AREA\_SIZE, 416  
 EZDP\_TOPOLOGY, 431  
 EZDP\_TRANSMIT, 433  
 ezdp\_tx\_drop\_mode, 432  
 ezdp\_tx\_packet\_switch\_mode, 431  
 ezdp\_job\_desc, 100  
   frame\_desc, 100  
   rx\_info, 100  
   tx\_info, 100  
 ezdp\_job\_discard\_cmd\_info, 101  
   \_\_pad0\_\_, 101  
   \_\_pad1\_\_, 101  
   raw\_data, 101  
   side, 101  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED0\_9\_OFFSET  
   ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED0\_9\_SIZE  
   ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED11\_15\_OFFSET  
   ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_RESERVED11\_15\_SIZE  
   ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_MASK  
   ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_OFFSET  
   ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_DISCARD\_CMD\_INFO\_SIDE\_SIZE  
   ezdp\_job\_defs.h, 426  
 ezdp\_job\_discard\_cmd\_info\_t  
   ezdp\_job\_defs.h, 430  
 ezdp\_job\_flags  
   ezdp\_job\_defs.h, 433  
 ezdp\_job\_id\_t  
   ezdp\_job\_defs.h, 430  
 ezdp\_job\_queue\_cmd\_info, 102  
   \_\_pad0\_\_, 102  
   raw\_data, 102  
   side, 102  
   target\_queue, 102  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_RESERVED8\_15\_OFFSET  
   ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_RESERVED8\_15\_SIZE  
   ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_SIDE\_MASK  
   ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_SIDE\_OFFSET  
   ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_SIDE\_SIZE  
   ezdp\_job\_defs.h, 425  
 ezdp\_job\_queue\_cmd\_info\_t  
   ezdp\_job\_defs.h, 430  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_TARGET\_QUEUE\_OFFSET  
   ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_QUEUE\_CMD\_INFO\_TARGET\_QUEUE\_SIZE  
   ezdp\_job\_defs.h, 425  
 ezdp\_job\_rx\_confirmation\_info, 103  
   \_\_pad0\_\_, 103  
   raw\_data, 103  
   timestamp\_nsec, 103  
   timestamp\_sec, 103  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_RESERV  
   ED8\_31\_OFFSET  
   ezdp\_job\_defs.h, 419  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_RESERV  
   ED8\_31\_SIZE  
   ezdp\_job\_defs.h, 419  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_NSEC\_OFFSET  
   ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_NSEC\_SIZE  
   ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_NSEC\_WORD\_OFFSET  
   ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_NSEC\_WORD\_SELECT  
   ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_SEC\_OFFSET  
   ezdp\_job\_defs.h, 419  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_SEC\_SIZE  
   ezdp\_job\_defs.h, 419  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_SEC\_WORD\_OFFSET  
   ezdp\_job\_defs.h, 419  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_TIMESTA  
   MP\_SEC\_WORD\_SELECT  
   ezdp\_job\_defs.h, 419  
 EZDP\_JOB\_RX\_CONFIRMATION\_INFO\_WORD\_C  
   OUNT  
   ezdp\_job\_defs.h, 420  
 ezdp\_job\_rx\_info, 104  
   \_\_pad0\_\_, 105  
   \_\_pad1\_\_, 105  
   \_\_pad2\_\_, 105  
   confirmation\_info, 104  
   gross\_checksum, 105  
   interface\_info, 104  
   is\_service\_ready, 105  
   loopback\_info, 104  
   raw\_data, 104  
   seq\_number, 105  
   seq\_number\_valid, 105  
   side, 106  
   source\_queue, 106  
   timer\_info, 105  
   user\_info, 105

EZDP_JOB_RX_INFO_GROSS_CHECKSUM_OFFSET	EZDP_JOB_RX_INFO_SEQ_NUMBER_WORD_OFSET
ezdp_job_defs.h, 421	ezdp_job_defs.h, 422
EZDP_JOB_RX_INFO_GROSS_CHECKSUM_SIZE	EZDP_JOB_RX_INFO_SEQ_NUMBER_WORD_SELECTOR
ezdp_job_defs.h, 421	ezdp_job_defs.h, 422
EZDP_JOB_RX_INFO_GROSS_CHECKSUM_WORD_OFFSET	EZDP_JOB_RX_INFO_SIDE_MASK
ezdp_job_defs.h, 421	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_GROSS_CHECKSUM_WORD_SELECT	EZDP_JOB_RX_INFO_SIDE_OFFSET
ezdp_job_defs.h, 421	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_IS_SERVICE_READY_MASK	EZDP_JOB_RX_INFO_SIDE_SIZE
ezdp_job_defs.h, 422	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_IS_SERVICE_READY_OFFSET	EZDP_JOB_RX_INFO_SIDE_WORD_OFFSET
ezdp_job_defs.h, 422	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_IS_SERVICE_READY_SIZE	EZDP_JOB_RX_INFO_SIDE_WORD_SELECT
ezdp_job_defs.h, 421	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_IS_SERVICE_READY_WORD_OFFSET	EZDP_JOB_RX_INFO_SOURCE_QUEUE_OFFSET
ezdp_job_defs.h, 422	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_IS_SERVICE_READY_WORD_SELECT	EZDP_JOB_RX_INFO_SOURCE_QUEUE_SIZE
ezdp_job_defs.h, 422	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_RESERVED104_107_OFFSET	EZDP_JOB_RX_INFO_SOURCE_QUEUE_WORD_OFFSET
ezdp_job_defs.h, 421	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_RESERVED104_107_SIZE	EZDP_JOB_RX_INFO_SOURCE_QUEUE_WORD_SELECT
ezdp_job_defs.h, 421	ezdp_job_defs.h, 421
EZDP_JOB_RX_INFO_RESERVED108_109_OFFSET	ezdp_job_rx_interface_info, 107
ezdp_job_defs.h, 421	__pad0__, 107
EZDP_JOB_RX_INFO_RESERVED108_109_SIZE	__pad1__, 108
ezdp_job_defs.h, 421	__pad2__, 108
EZDP_JOB_RX_INFO_RESERVED112_127_OFFSET	__pad3__, 108
ezdp_job_defs.h, 421	crc_checked_flag, 108
EZDP_JOB_RX_INFO_RESERVED112_127_SIZE	crc_ok_flag, 108
ezdp_job_defs.h, 421	icu_succ_parsing_flag, 108
EZDP_JOB_RX_INFO_RESERVED112_127_SIZE	imem_buf_count, 107
ezdp_job_defs.h, 421	raw_data, 107
EZDP_JOB_RX_INFO_SEQ_NUMBER_OFFSET	timestamp_nsec, 109
ezdp_job_defs.h, 422	timestamp_sec, 108
EZDP_JOB_RX_INFO_SEQ_NUMBER_SIZE	truncation_flag, 108
ezdp_job_defs.h, 422	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_MASK
EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_MASK	ezdp_job_defs.h, 417
ezdp_job_defs.h, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_OFFSET
EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_OFSET	ezdp_job_defs.h, 417
ezdp_job_defs.h, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_SIZE
EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_SIZE	ezdp_job_defs.h, 417
ezdp_job_defs.h, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_WORD_OFFSET
EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_WORD_OFFSET	ezdp_job_defs.h, 417
ezdp_job_defs.h, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_CHECKED_FLAG_WORD_SELECT
EZDP_JOB_RX_INFO_SEQ_NUMBER_VALID_WORD_SELECT	ezdp_job_defs.h, 417
ezdp_job_defs.h, 421	EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLAG_MASK
	ezdp_job_defs.h, 417

EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLG_OFFSET ezdp_job_defs.h, 417	EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_WORD_SELECT ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLG_SIZE ezdp_job_defs.h, 417	EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_OFFSET ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLG_WORD_OFFSET ezdp_job_defs.h, 417	EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_SIZE ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_CRC_OK_FLG_WORD_SELECT ezdp_job_defs.h, 417	EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_WORD_OFFSET ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_OFFSET ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_COUNT_WORD_SELECT ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_SIZE ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_OFFSET ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_WORD_OFFSET ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_SIZE ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_EMEM_BUF_CONGESTION_LEVEL_WORD_SELECT ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_WORD_OFFSET ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_OFFSET ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_JOB_CONGESTION_LEVEL_WORD_SELECT ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_SIZE ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_OFFSET ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_WORD_OFFSET ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_SIZE ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_GLOBAL_CONGESTION_LEVEL_WORD_SELECT ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_WORD_OFFSET ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_MASK ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_PMU_QUEUE_CONGESTION_LEVEL_WORD_SELECT ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_OFFSET ezdp_job_defs.h, 417	EZDP_JOB_RX_INTERFACE_INFO_RESERVED10_OFFSET ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_SIZE ezdp_job_defs.h, 417	EZDP_JOB_RX_INTERFACE_INFO_RESERVED10_SIZE ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_WORD_OFFSET ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED11_OFFSET ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_ICU_SUCC_PARSING_FLAG_WORD_SELECT ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED11_SIZE ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_OFFSET ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED14_OFFSET ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_SIZE ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED14_SIZE ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_IMEM_BUF_CONGESTION_LEVEL_WORD_OFFSET ezdp_job_defs.h, 418	EZDP_JOB_RX_INTERFACE_INFO_RESERVED15_OFFSET ezdp_job_defs.h, 418

```

EZDP_JOB_RX_INTERFACE_INFO_RESERVED15
_SIZE
ezdp_job_defs.h, 418
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
NSEC_OFFSET
ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
NSEC_SIZE
ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
NSEC_WORD_OFFSET
ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
NSEC_WORD_SELECT
ezdp_job_defs.h, 419
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
SEC_OFFSET
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
SEC_SIZE
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
SEC_WORD_OFFSET
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TIMESTAMP_
SEC_WORD_SELECT
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_
_FLAG_MASK
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_
_FLAG_OFFSET
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_
_FLAG_SIZE
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_
_FLAG_WORD_OFFSET
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_TRUNCATION_
_FLAG_WORD_SELECT
ezdp_job_defs.h, 417
EZDP_JOB_RX_INTERFACE_INFO_WORD_COUN
T
ezdp_job_defs.h, 419
ezdp_job_rx_loopback_info, 110
__pad0__, 110
__pad1__, 110
raw_data, 110
replication_id, 110
EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_
_ID_OFFSET
ezdp_job_defs.h, 419
EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_
_ID_SIZE
ezdp_job_defs.h, 419
EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION_
_ID_WORD_OFFSET
ezdp_job_defs.h, 419

```

```

EZDP_JOB_RX_LOOPBACK_INFO_REPLICATION
_ID_WORD_SELECT
    ezdp_job_defs.h, 419
EZDP_JOB_RX_LOOPBACK_INFO_RESERVED0_
15_OFFSET
    ezdp_job_defs.h, 419
EZDP_JOB_RX_LOOPBACK_INFO_RESERVED0_
15_SIZE
    ezdp_job_defs.h, 419
EZDP_JOB_RX_LOOPBACK_INFO_RESERVED32_
63_OFFSET
    ezdp_job_defs.h, 419
EZDP_JOB_RX_LOOPBACK_INFO_RESERVED32_
63_SIZE
    ezdp_job_defs.h, 419
EZDP_JOB_RX_LOOPBACK_INFO_WORD_COUN
T
    ezdp_job_defs.h, 419
ezdp_job_rx_timer_info, 111
    __pad0__, 111
    __pad1__, 111
    event_id, 111
    raw_data, 111
    timer_id, 111
EZDP_JOB_RX_TIMER_INFO_EVENT_ID_OFFSE
T
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_EVENT_ID_SIZE
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_EVENT_ID_WORD_
OFFSET
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_EVENT_ID_WORD_
SELECT
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_RESERVED0_8_OF
FSET
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_RESERVED0_8_SIZ
E
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_RESERVED16_31_O
FFSET
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_RESERVED16_31_S
IZE
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_TIMER_ID_OFFSET
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_TIMER_ID_SIZE
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_TIMER_ID_WORD_
OFFSET
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_TIMER_ID_WORD_
SELECT
    ezdp_job_defs.h, 420
EZDP_JOB_RX_TIMER_INFO_WORD_COUNT
    ezdp_job_defs.h, 420
ezdp_job_rx_user_info, 112

```

raw\_data, 112  
 user\_data\_info0, 112  
 user\_data\_info1, 112  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0  
   \_OFFSET  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0  
   \_SIZE  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0  
   \_WORD\_OFFSET  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO0  
   \_WORD\_SELECT  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1  
   \_OFFSET  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1  
   \_SIZE  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1  
   \_WORD\_OFFSET  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_USER\_DATA\_INFO1  
   \_WORD\_SELECT  
     ezdp\_job\_defs.h, 420  
 EZDP\_JOB\_RX\_USER\_INFO\_WORD\_COUNT  
     ezdp\_job\_defs.h, 421  
 ezdp\_job\_transmit\_cmd\_info, 113  
   \_\_pad0\_\_, 113  
   output\_channel, 113  
   raw\_data, 113  
   side, 113  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_DESTINATIO  
   N\_MASK  
     ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_DESTINATIO  
   N\_OFFSET  
     ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_DESTINATIO  
   N\_SIZE  
     ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_OUTPUT\_CH  
   ANNEL\_OFFSET  
     ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_OUTPUT\_CH  
   ANNEL\_SIZE  
     ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_RESERVED1  
   2\_15\_OFFSET  
     ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_RESERVED1  
   2\_15\_SIZE  
     ezdp\_job\_defs.h, 426  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_MASK  
     ezdp\_job\_defs.h, 425  
 EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_OFFSE  
   T  
     ezdp\_job\_defs.h, 425

EZDP\_JOB\_TRANSMIT\_CMD\_INFO\_SIDE\_SIZE  
     ezdp\_job\_defs.h, 425  
 ezdp\_job\_transmit\_cmd\_info\_t  
     ezdp\_job\_defs.h, 430  
 ezdp\_job\_transmit\_dest  
     ezdp\_job\_defs.h, 432  
 ezdp\_job\_tx\_info, 114  
   \_\_pad0\_\_, 115  
   \_\_pad1\_\_, 115  
   \_\_pad2\_\_, 116  
   \_\_pad3\_\_, 116  
   \_\_pad4\_\_, 117  
   \_\_pad5\_\_, 117  
   dest\_queue, 115  
   explicit\_packet\_switch\_id, 115  
   flow\_id, 116  
   inter\_packet\_gap, 118  
   inter\_packet\_gap\_control, 118  
   packet\_switch\_id\_select, 115  
   qos\_bypass, 116  
   raw\_data, 115  
   replication\_count, 115  
   side, 116  
   stat\_code\_profile1, 116  
   stat\_code\_profile2, 116  
   stat\_stream\_id, 117  
   user\_info, 115  
   wred\_class\_scale\_profile, 117  
   wred\_class\_template\_profile, 117  
   wred\_color, 115  
   wred\_flow\_scale\_profile, 117  
   wred\_flow\_template\_profile, 117  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_OFFSET  
     ezdp\_job\_defs.h, 423  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_SIZE  
     ezdp\_job\_defs.h, 423  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_WORD\_OFFS  
   ET  
     ezdp\_job\_defs.h, 423  
 EZDP\_JOB\_TX\_INFO\_DROP\_MODE\_WORD\_SELE  
   CT  
     ezdp\_job\_defs.h, 423  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_OFFSET  
     ezdp\_job\_defs.h, 422  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_SIZE  
     ezdp\_job\_defs.h, 422  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_WORD\_OFFSET  
     ezdp\_job\_defs.h, 422  
 EZDP\_JOB\_TX\_INFO\_FLOW\_ID\_WORD\_SELECT  
     ezdp\_job\_defs.h, 422  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CON  
   TROL\_MASK  
     ezdp\_job\_defs.h, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CON  
   TROL\_OFFSET  
     ezdp\_job\_defs.h, 424  
 EZDP\_JOB\_TX\_INFO\_INTER\_PACKET\_GAP\_CON  
   TROL\_SIZE  
     ezdp\_job\_defs.h, 424

EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_WORD_OFFSET ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_RESERVED25_29_SIZE ezdp_job_defs.h, 422
EZDP_JOB_TX_INFO_INTER_PACKET_GAP_CONTROL_WORD_SELECT ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_RESERVED52_55_OFFSET ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_INTER_PACKET_GAP_OFFSET ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_RESERVED52_55_SIZE ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_INTER_PACKET_GAP_SIZE ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_RESERVED59_OFFSET ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_INTER_PACKET_GAP_WORD_OFFSET ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_RESERVED59_SIZE ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_INTER_PACKET_GAP_WORD_SELECT ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_RESERVED61_OFFSET ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_OFFSET ezdp_job_defs.h, 422	EZDP_JOB_TX_INFO_RESERVED61_SIZE ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_SIZE ezdp_job_defs.h, 422	EZDP_JOB_TX_INFO_RESERVED88_90_OFFSET ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_WORD_OFFSET ezdp_job_defs.h, 422	EZDP_JOB_TX_INFO_RESERVED88_90_SIZE ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_PACKET_SWITCH_ID_SELECT_WORD_SELECT ezdp_job_defs.h, 422	EZDP_JOB_TX_INFO_SIDE_MASK ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_OFFSET ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_SIDE_OFFSET ezdp_job_defs.h, 422
EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_SIZE ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_SIDE_SIZE ezdp_job_defs.h, 422
EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_WORD_OFFSET ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_SIDE_WORD_OFFSET ezdp_job_defs.h, 423
EZDP_JOB_TX_INFO_PACKET_SWITCH_MODE_WORD_SELECT ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_SIDE_WORD_SELECT ezdp_job_defs.h, 422
EZDP_JOB_TX_INFO_QOS_BYPASS_MASK ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_OFFSET ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_QOS_BYPASS_OFFSET ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_SIZE ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_QOS_BYPASS_SIZE ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_WORD_OFFSET ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_QOS_BYPASS_WORD_OFFSET ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE1_WORD_SELECT ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_QOS_BYPASS_WORD_SELECT ezdp_job_defs.h, 423	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_OFFSET ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_RESERVED102_103_OFFSET ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_SIZE ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_RESERVED102_103_SIZE ezdp_job_defs.h, 424	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_WORD_OFFSET ezdp_job_defs.h, 424
EZDP_JOB_TX_INFO_RESERVED25_29_OFFSET ezdp_job_defs.h, 422	EZDP_JOB_TX_INFO_STAT_CODE_PROFILE2_WORD_SELECT ezdp_job_defs.h, 424
	EZDP_JOB_TX_INFO_STAT_STREAM_ID_OFFSET ezdp_job_defs.h, 423
	EZDP_JOB_TX_INFO_STAT_STREAM_ID_SIZE ezdp_job_defs.h, 423
	EZDP_JOB_TX_INFO_STAT_STREAM_ID_WORD_OFFSET ezdp_job_defs.h, 423

EZDP\_JOB\_TX\_INFO\_STAT\_STREAM\_ID\_WORD\_SELECT  
ezdp\_job\_defs.h, 423

EZDP\_JOB\_TX\_INFO\_WORD\_COUNT  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_OFFSET  
ezdp\_job\_defs.h, 424

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_SIZE  
ezdp\_job\_defs.h, 424

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_WORD\_OFFSET  
ezdp\_job\_defs.h, 424

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_SCALE\_PROFILE\_WORD\_SELECT  
ezdp\_job\_defs.h, 424

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_TEMPLATE\_PROFILE\_OFFSET  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_TEMPLATE\_PROFILE\_SIZE  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_TEMPLATE\_PROFILE\_WORD\_OFFSET  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_CLASS\_TEMPLATE\_PROFILE\_WORD\_SELECT  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_OFFSET  
ezdp\_job\_defs.h, 422

EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_SIZE  
ezdp\_job\_defs.h, 422

EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_WORD\_OFFSET  
ezdp\_job\_defs.h, 422

EZDP\_JOB\_TX\_INFO\_WRED\_COLOR\_WORD\_SELECT  
ezdp\_job\_defs.h, 422

EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_OFFSET  
ezdp\_job\_defs.h, 424

EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_SIZE  
ezdp\_job\_defs.h, 424

EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_WORD\_OFFSET  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_SCALE\_PROFILE\_WORD\_SELECT  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_OFFSET  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_SIZE  
ezdp\_job\_defs.h, 425

EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_WORD\_OFFSET  
ezdp\_job\_defs.h, 425

- EZDP\_JOB\_TX\_INFO\_WRED\_FLOW\_TEMPLATE\_PROFILE\_WORD\_SELECT
  - ezdp\_job\_defs.h, 425
- EZDP\_LARGE\_LBD
  - ezdp\_frame\_defs.h, 390
- ezdp\_large\_linked\_buffers\_desc, 119
  - line, 119
- ezdp\_lbd\_len
  - ezdp\_frame.h, 374
- EZDP\_LINKED\_BUFFER\_DESC\_LINE\_NUMBER\_OF\_BUFFERS\_DESC
  - ezdp\_frame\_defs.h, 388
- ezdp\_linked\_buffers\_desc, 120
  - line, 120
- ezdp\_linked\_buffers\_desc\_line, 121
  - buf\_desc, 121
  - buf\_info, 121
  - ecc, 121
- ezdp\_linked\_buffers\_desc\_size
  - ezdp\_frame\_defs.h, 389
- ezdp\_list\_cfg, 122
  - head, 122
  - queue\_memory\_pool, 122
  - tail, 122
- ezdp\_list\_empty
  - ezdp\_queue.h, 503
- ezdp\_list\_t
  - ezdp\_queue\_defs.h, 505
- EZDP\_LIST\_WORK\_AREA\_SIZE
  - ezdp\_queue\_defs.h, 505
- ezdp\_load\_16\_byte\_data\_from\_ext\_addr
  - ezdp\_dma.h, 350
- ezdp\_load\_16\_byte\_data\_from\_ext\_addr\_async
  - ezdp\_dma.h, 351
- ezdp\_load\_16\_byte\_data\_from\_sum\_addr
  - ezdp\_dma.h, 354
- ezdp\_load\_16\_byte\_data\_from\_sum\_addr\_async
  - ezdp\_dma.h, 354
- ezdp\_load\_32\_byte\_data\_from\_ext\_addr
  - ezdp\_dma.h, 351
- ezdp\_load\_32\_byte\_data\_from\_ext\_addr\_async
  - ezdp\_dma.h, 351
- ezdp\_load\_32\_byte\_data\_from\_sum\_addr
  - ezdp\_dma.h, 355
- ezdp\_load\_32\_byte\_data\_from\_sum\_addr\_async
  - ezdp\_dma.h, 355
- ezdp\_load\_data\_from\_ext\_addr
  - ezdp\_dma.h, 350
- ezdp\_load\_data\_from\_ext\_addr\_async
  - ezdp\_dma.h, 350
- ezdp\_load\_data\_from\_pci
  - ezdp\_pci.h, 476
- ezdp\_load\_data\_from\_pci\_async
  - ezdp\_pci.h, 477
- ezdp\_load\_data\_from\_sum\_addr
  - ezdp\_dma.h, 353
- ezdp\_load\_data\_from\_sum\_addr\_async
  - ezdp\_dma.h, 354
- ezdp\_load\_frame\_data
  - ezdp\_frame.h, 367

ezdp\_load\_frame\_data\_async  
     ezdp\_frame.h, 367  
 ezdp\_load\_frame\_lbd  
     ezdp\_frame.h, 371  
 ezdp\_load\_frame\_lbd\_async  
     ezdp\_frame.h, 371  
 ezdp\_load\_job  
     ezdp\_job.h, 396  
 ezdp\_load\_job\_async  
     ezdp\_job.h, 396  
 ezdp\_lock.h  
     ezdp\_alloc\_qlock\_slot, 437  
     ezdp\_dequeue\_qlock, 438  
     ezdp\_destroy\_qlock, 436  
     ezdp\_enqueue\_qlock, 438  
     ezdp\_free\_qlock\_slot, 437  
     ezdp\_init\_qlock, 436  
     ezdp\_init\_spinlock\_ext\_addr, 435  
     ezdp\_init\_spinlock\_sum\_addr, 435  
     ezdp\_lock\_qlock, 437  
     ezdp\_lock\_spinlock, 436  
     ezdp\_order\_lock\_qlock, 437  
     ezdp\_try\_lock\_spinlock, 436  
     ezdp\_try\_unlock\_qlock, 438  
     ezdp\_unlock\_spinlock, 436  
 ezdp\_lock\_defs.h  
     EZDP\_NULL\_QLOCK\_SLOT, 440  
     ezdp\_qlock\_slot\_t, 440  
     ezdp\_qlock\_t, 440  
     EZDP\_QLOCK\_WORK\_AREA\_SIZE, 440  
     ezdp\_spinlock\_t, 440  
 ezdp\_lock\_qlock  
     ezdp\_lock.h, 437  
 ezdp\_lock\_spinlock  
     ezdp\_lock.h, 436  
 ezdp\_lookup\_alg\_tcaml  
     ezdp\_search.h, 517  
 ezdp\_lookup\_ext\_tcaml  
     ezdp\_search.h, 515  
 ezdp\_lookup\_ext\_tcaml\_16B\_data\_result\_element, 123  
     \_\_pad0\_\_, 124  
     assoc\_data, 124  
     lookup\_error, 123  
     match, 123  
     raw\_data, 123  
     truncated, 124  
     valid, 123  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
     ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
     ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
     ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET  
     ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT  
     ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_OFFSET  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT  
     ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET  
     ezdp\_search\_defs.h, 549



EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
 ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT  
 ezdp\_search\_defs.h, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT  
 ezdp\_search\_defs.h, 550  
 ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element, 125  
 \_\_pad0\_\_, 126  
 assoc\_data, 126  
 lookup\_error, 125  
 match, 125  
 raw\_data, 125  
 truncated, 126  
 valid, 125  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_OFFSET  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT  
 ezdp\_search\_defs.h, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
 ezdp\_search\_defs.h, 551

EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT  
 ezdp\_search\_defs.h, 551  
 ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element, 127  
 \_\_pad0\_\_, 128  
 assoc\_data, 128  
 lookup\_error, 127  
 match, 127  
 raw\_data, 127  
 truncated, 128  
 valid, 127  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE  
 ezdp\_search\_defs.h, 546  
 ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element\_t  
 ezdp\_search\_defs.h, 552  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
 ezdp\_search\_defs.h, 547  
 ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element, 129  
 \_\_pad0\_\_, 130  
 assoc\_data, 130  
 lookup\_error, 129  
 match, 129  
 raw\_data, 129  
 truncated, 130  
 valid, 129  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT  
 ezdp\_search\_defs.h, 548

EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT  
 ezdp\_search\_defs.h, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 548

EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT  
 ezdp\_search\_defs.h, 548  
 ezdp\_lookup\_ext\_tcam\_async  
 ezdp\_search.h, 516  
 ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 131  
 \_\_pad0\_\_, 132  
 assoc\_data, 132  
 device\_id, 132  
 index, 132  
 lookup\_error, 132  
 match, 131  
 raw\_data, 131  
 truncated, 132  
 valid, 131  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_OFFSET  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 542

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_SELECT  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_OFFSET  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT  
 ezdp\_search\_defs.h, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT  
 ezdp\_search\_defs.h, 544  
 ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 133  
 \_\_pad0\_\_, 134  
 assoc\_data, 134  
 device\_id, 134  
 index, 134  
 lookup\_error, 134  
 match, 133  
 raw\_data, 133  
 truncated, 134  
 valid, 133  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET  
 ezdp\_search\_defs.h, 544

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_OFFSET  
 ET  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_SELECT  
 CT  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_OFFSET  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_SELECT  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_OFFSET  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
 ezdp\_search\_defs.h, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET  
 SET  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT  
 CT  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT  
 ezdp\_search\_defs.h, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT  
 ezdp\_search\_defs.h, 546  
 ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element,  
 135  
 \_\_pad0\_\_, 136  
 assoc\_data, 136  
 device\_id, 136  
 index, 136  
 lookup\_error, 136  
 match, 135  
 raw\_data, 135  
 truncated, 136  
 valid, 135

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_OFFSET  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_SELECT  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_INDEX\_OFFSET  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSET  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_SELECT  
ezdp\_search\_defs.h, 537

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_OFFSET  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT  
ezdp\_search\_defs.h, 538

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT  
 ezdp\_search\_defs.h, 539  
 ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_element,  
 137  
 \_\_pad0\_\_, 138  
 assoc\_data, 138  
 device\_id, 138  
 index, 138  
 lookup\_error, 138  
 match, 137  
 raw\_data, 137  
 truncated, 138  
 valid, 137  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_SELECT  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_INDEX\_OFFSET  
 ezdp\_search\_defs.h, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE  
 ezdp\_search\_defs.h, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 539

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_SELECT  
 ezdp\_search\_defs.h, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT  
 ezdp\_search\_defs.h, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_OFFSET  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT  
 ezdp\_search\_defs.h, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET

ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT  
ezdp\_search\_defs.h, 541  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT  
ezdp\_search\_defs.h, 542  
ezdp\_lookup\_ext\_tcam\_index\_result\_element, 139  
\_\_pad0\_\_, 140  
any\_match, 140  
device\_id, 140  
index, 140  
lookup\_error, 139  
match, 139  
raw\_data, 139  
truncated, 140  
valid, 139  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_MASK  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_SIZE  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_INDEX\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_INDEX\_SIZE  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_MATCH\_MASK  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_MATCH\_OFFSET  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_MATCH\_SIZE  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_RESERVED23\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_RESERVED23\_SIZE  
ezdp\_search\_defs.h, 536  
ezdp\_lookup\_ext\_tcam\_index\_result\_element\_t  
ezdp\_search\_defs.h, 552  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_MASK  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_SIZE  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TYPE\_OFFSET  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TYPE\_SIZE  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_VALID\_MASK  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_VALID\_OFFSET  
ezdp\_search\_defs.h, 537  
EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_VALID\_SIZE  
ezdp\_search\_defs.h, 537  
ezdp\_lookup\_ext\_tcam\_retval, 141  
\_\_pad0\_\_, 141  
any\_match, 142  
device\_error, 142  
lookup\_error, 141  
mac\_error, 142  
multi\_match, 142  
no\_context\_match\_error, 142  
raw\_data, 141  
time\_out\_error, 142



truncated, 141  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MA  
TCH\_MASK  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MA  
TCH\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MA  
TCH\_SIZE  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_  
ERROR\_MASK  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_  
ERROR\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_  
ERROR\_SIZE  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_  
ERROR\_MASK  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_  
ERROR\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_  
ERROR\_SIZE  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ER  
ROR\_MASK  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ER  
ROR\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ER  
ROR\_SIZE  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_M  
ATCH\_MASK  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_M  
ATCH\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_M  
ATCH\_SIZE  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONT  
EXT\_MATCH\_ERROR\_MASK  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONT  
EXT\_MATCH\_ERROR\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONT  
EXT\_MATCH\_ERROR\_SIZE  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_RESERVE  
D\_BIT8\_31\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_RESERVE  
D\_BIT8\_31\_SIZE

ezdp\_search\_defs.h, 536  
ezdp\_lookup\_ext\_tcam\_retval\_t  
ezdp\_search\_defs.h, 552  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_OU  
T\_ERROR\_MASK  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_OU  
T\_ERROR\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_OU  
T\_ERROR\_SIZE  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNCA  
TED\_MASK  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNCA  
TED\_OFFSET  
ezdp\_search\_defs.h, 536  
EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNCA  
TED\_SIZE  
ezdp\_search\_defs.h, 536  
ezdp\_lookup\_hash\_entry  
ezdp\_search.h, 510  
ezdp\_lookup\_int\_tcam  
ezdp\_search.h, 514  
ezdp\_lookup\_int\_tcam\_12B\_data\_result, 143  
data0, 143  
data1, 143  
data2, 143  
match, 143  
raw\_data, 143  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA0\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA0\_SIZE  
ezdp\_search\_defs.h, 533  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA0\_WORD\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA0\_WORD\_SELECT  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA1\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA1\_SIZE  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA1\_WORD\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA1\_WORD\_SELECT  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA2\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA2\_SIZE

ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA2\_WORD\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_DATA2\_WORD\_SELECT  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_MATCH\_MASK  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_MATCH\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_MATCH\_SIZE  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_MATCH\_WORD\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_MATCH\_WORD\_SELECT  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESULT  
\_WORD\_COUNT  
ezdp\_search\_defs.h, 534  
ezdp\_lookup\_int\_tcam\_16B\_data\_result, 144  
data0, 144  
data1, 144  
data2, 144  
data3, 144  
match, 144  
raw\_data, 144  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA0\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA0\_SIZE  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA0\_WORD\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA0\_WORD\_SELECT  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA1\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA1\_SIZE  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA1\_WORD\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA1\_WORD\_SELECT  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA2\_OFFSET  
ezdp\_search\_defs.h, 535

EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA2\_SIZE  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA2\_WORD\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA2\_WORD\_SELECT  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA3\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA3\_SIZE  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA3\_WORD\_OFFSET  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_DATA3\_WORD\_SELECT  
ezdp\_search\_defs.h, 535  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_MATCH\_MASK  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_MATCH\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_MATCH\_SIZE  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_MATCH\_WORD\_OFFSET  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_MATCH\_WORD\_SELECT  
ezdp\_search\_defs.h, 534  
EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESULT  
\_WORD\_COUNT  
ezdp\_search\_defs.h, 535  
ezdp\_lookup\_int\_tcam\_4B\_data\_result, 145  
data, 145  
match, 145  
raw\_data, 145  
EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESULT\_  
DATA\_OFFSET  
ezdp\_search\_defs.h, 533  
EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESULT\_  
DATA\_SIZE  
ezdp\_search\_defs.h, 533  
EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESULT\_  
MATCH\_MASK  
ezdp\_search\_defs.h, 533  
EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESULT\_  
MATCH\_OFFSET  
ezdp\_search\_defs.h, 533  
EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESULT\_  
MATCH\_SIZE  
ezdp\_search\_defs.h, 533  
ezdp\_lookup\_int\_tcam\_4B\_data\_result\_t  
ezdp\_search\_defs.h, 552

ezdp\_lookup\_int\_tcaml\_8B\_data\_result, 146  
     data0, 146  
     data1, 146  
     match, 146  
     raw\_data, 146  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA0\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA0\_SIZE  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA0\_WORD\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA0\_WORD\_SELECT  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA1\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA1\_SIZE  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA1\_WORD\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ DATA1\_WORD\_SELECT  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ MATCH\_MASK  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ MATCH\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ MATCH\_SIZE  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ MATCH\_WORD\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ MATCH\_WORD\_SELECT  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESULT\_ WORD\_COUNT  
     ezdp\_search\_defs.h, 533  
 ezdp\_lookup\_int\_tcaml\_async  
     ezdp\_search.h, 515  
 ezdp\_lookup\_int\_tcaml\_result, 147  
     assoc\_12B\_data, 147  
     assoc\_16B\_data, 147  
     assoc\_4B\_data, 147  
     assoc\_8B\_data, 147  
     standard, 147  
 ezdp\_lookup\_int\_tcaml\_retval, 148  
     assoc\_data, 148  
     raw\_data, 148  
     standard, 148  
 ezdp\_lookup\_int\_tcaml\_retval\_t  
     ezdp\_search\_defs.h, 552  
 ezdp\_lookup\_int\_tcaml\_standard\_result, 149  
     \_\_pad0\_\_, 149  
     index, 149  
     match, 149  
     raw\_data, 149  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_INDEX\_OFFSET  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_INDEX\_SIZE  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_MATCH\_MASK  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_MATCH\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_MATCH\_SIZE  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_MAX\_NUM  
     ezdp\_search\_defs.h, 551  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_RESERVED0\_15\_OFFSET  
     ezdp\_search\_defs.h, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RESULT\_RESERVED0\_15\_SIZE  
     ezdp\_search\_defs.h, 533  
 ezdp\_lookup\_int\_tcaml\_standard\_result\_t  
     ezdp\_search\_defs.h, 552  
 EZDP\_LOOKUP\_PARITY\_BITS\_SIZE  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RESERVED\_BITS\_SIZE  
     ezdp\_search\_defs.h, 532  
 ezdp\_lookup\_retval, 150  
     data, 150  
     info, 150  
     match, 150  
     mem\_error, 150  
     raw\_data, 150  
     success, 150  
 EZDP\_LOOKUP\_RETVAL\_DATA\_OFFSET  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_DATA\_SIZE  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_INFO\_MASK  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_INFO\_OFFSET  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_INFO\_SIZE  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_MATCH\_MASK  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_MATCH\_OFFSET  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_MATCH\_SIZE  
     ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_MEM\_ERROR\_MASK

ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_MEM\_ERROR\_OFFSET  
 ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_MEM\_ERROR\_SIZE  
 ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_SUCCESS\_MASK  
 ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_SUCCESS\_OFFSET  
 ezdp\_search\_defs.h, 532  
 EZDP\_LOOKUP\_RETVAL\_SUCCESS\_SIZE  
 ezdp\_search\_defs.h, 532  
 ezdp\_lookup\_retval\_t  
 ezdp\_search\_defs.h, 552  
 ezdp\_lookup\_table\_entry  
 ezdp\_search.h, 508  
 ezdp\_lookup\_ultra\_ip\_entry  
 ezdp\_search.h, 514  
 EZDP\_LOOKUP\_VERSION\_MAJOR  
 ezdp\_search\_defs.h, 530  
 EZDP\_LOOKUP\_VERSION\_MINOR  
 ezdp\_search\_defs.h, 530  
 EZDP\_LOW\_LEVEL  
 ezdp\_job\_defs.h, 431  
 ezdp\_mac\_calculation  
 ezdp\_security.h, 566  
 ezdp\_mac\_calculation\_async  
 ezdp\_security.h, 566  
 EZDP\_MAIN\_FUNC  
 ezdp.h, 192  
 ezdp\_math.h  
 ezdp\_add, 444  
 ezdp\_add\_checksum, 457  
 ezdp\_and, 444  
 ezdp\_bit\_mode, 443  
 EZDP\_BIT\_MODE\_FALSE, 443  
 EZDP\_BIT\_MODE\_INVERSE, 443  
 EZDP\_BIT\_MODE\_TRUE, 443  
 EZDP\_BIT\_MODE\_VALUE, 443  
 ezdp\_bulk\_hash, 457  
 ezdp\_calc\_crc16, 457  
 ezdp\_calc\_crc32, 457  
 ezdp\_clear\_bit, 448  
 ezdp\_combine\_4\_bits, 453  
 ezdp\_combine\_merge\_4\_bits, 454  
 ezdp\_count\_bits, 447  
 ezdp\_div, 447  
 ezdp\_find\_first\_one, 448  
 ezdp\_find\_first\_zero, 449  
 ezdp\_fxor16, 446  
 ezdp\_fxor8, 445  
 ezdp\_get\_2\_bitfields, 450  
 ezdp\_get\_2\_bits, 451  
 ezdp\_get\_3\_bits, 452  
 ezdp\_get\_4\_bits, 452  
 ezdp\_get\_4\_bytes, 455  
 ezdp\_get\_bit, 450  
 ezdp\_get\_bitfield, 449  
 ezdp\_hash, 456  
 ezdp\_hash\_base\_matrix, 443

EZDP\_HASH\_BASE\_MATRIX\_HASH\_BASE\_M  
 ATRIX\_0, 443  
 EZDP\_HASH\_BASE\_MATRIX\_HASH\_BASE\_M  
 ATRIX\_1, 443  
 ezdp\_hash\_permutation, 444  
 EZDP\_HASH\_PERMUTATION\_0, 444  
 EZDP\_HASH\_PERMUTATION\_1, 444  
 EZDP\_HASH\_PERMUTATION\_2, 444  
 EZDP\_HASH\_PERMUTATION\_3, 444  
 ezdp\_hash32, 456  
 ezdp\_hash64, 456  
 ezdp\_merge\_2\_bitfields, 450  
 ezdp\_merge\_2\_bits, 451  
 ezdp\_merge\_3\_bits, 452  
 ezdp\_merge\_4\_bits, 453  
 ezdp\_merge\_bit, 451  
 ezdp\_merge\_bitfield, 449  
 ezdp\_merge\_pow\_of\_2, 448  
 ezdp\_mod, 447  
 ezdp\_not, 445  
 ezdp\_or, 445  
 ezdp\_pow\_of\_2, 447  
 ezdp\_reflect\_bits, 455  
 ezdp\_reflect\_resolution, 443  
 EZDP\_REFLECT\_RESOLUTION\_1\_BYTE, 443  
 EZDP\_REFLECT\_RESOLUTION\_2\_BYTE, 443  
 EZDP\_REFLECT\_RESOLUTION\_4\_BYTE, 443  
 ezdp\_set\_bit, 448  
 ezdp\_shift\_left, 446  
 ezdp\_shift\_right, 446  
 ezdp\_split\_4\_bits, 454  
 ezdp\_split\_merge\_4\_bits, 454  
 ezdp\_sub, 444  
 ezdp\_sub\_checksum, 458  
 ezdp\_xor, 445  
 EZDP\_MAX\_CPUS\_ID  
 ezdp\_processor.h, 498  
 EZDP\_MAX\_HW\_CLUSTERS  
 ezdp\_processor.h, 498  
 EZDP\_MAX\_HW\_CORES  
 ezdp\_processor.h, 498  
 EZDP\_MAX\_HW\_THREADS  
 ezdp\_processor.h, 498  
 ezdp\_mb  
 ezdp\_processor.h, 499  
 EZDP\_MD5\_ALG  
 ezdp\_security\_defs.h, 580  
 EZDP\_MD5\_BLOCK\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_MD5\_MAC\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_MD5\_STATE\_SIZE  
 ezdp\_security\_defs.h, 582  
 EZDP\_MEDIUM\_LEVEL  
 ezdp\_job\_defs.h, 431  
 EZDP\_MEM\_CFG\_EMEM\_DATA\_CACHABLE  
 ezdp.h, 192  
 EZDP\_MEM\_CFG\_IMEM\_1\_CLUSTER\_DATA\_CA  
 CHABLE  
 ezdp.h, 191

EZDP_MEM_CFG_IMEM_16_CLUSTER_DATA_CACHABLE ezdp.h, 191	ezdp_mem_space_type ezdp_memory_defs.h, 470
EZDP_MEM_CFG_IMEM_2_CLUSTER_DATA_CACHABLE ezdp.h, 191	ezdp_memory.h
EZDP_MEM_CFG_IMEM_4_CLUSTER_DATA_CACHABLE ezdp.h, 191	ezdp_calc_checksum, 459
EZDP_MEM_CFG_IMEM_ALL_CLUSTER_DATA_CACHABLE ezdp.h, 192	ezdp_calc_checksum_ext_addr, 459
EZDP_MEM_CFG_IMEM_HALF_CLUSTER_DATA_CACHABLE ezdp.h, 191	ezdp_calc_sum_addr, 460
EZDP_MEM_CFG_IMEM_PRIVATE_DATA_CACHABLE ezdp.h, 191	ezdp_calc_sum_addr_offset, 461
EZDP_MEM_CFG_USE_ALTER_CMEM ezdp.h, 191	ezdp_ext_addr_to_sum_addr, 460
EZDP_MEM_CFG_USE_ALTER_SHARED_CMEM ezdp.h, 191	ezdp_is_null_sum_addr, 460
ezdp_mem_cmp ezdp_string.h, 584	ezdp_scramble_ext_addr, 461
ezdp_mem_cmp_byte_skip ezdp_string.h, 585	ezdp_scramble_sum_addr, 461
ezdp_mem_copy ezdp_string.h, 584	ezdp_sum_addr_to_ext_addr, 460
EZDP_MEM_CTOR_FUNC ezdp.h, 192	ezdp_memory_defs.h
EZDP_MEM_FRAME_DATA_BUFFER_SIZE ezdp_frame_defs.h, 382	EZDP_1_CLUSTER_CODE, 469
ezdp_mem_pool_config, 151	EZDP_1_CLUSTER_DATA, 469
base_addr, 151	EZDP_16_CLUSTER_CODE, 470
index_pool_id, 151	EZDP_16_CLUSTER_DATA, 470
obj_size, 151	EZDP_2_CLUSTER_CODE, 469
ezdp_mem_pool_t ezdp_pool_defs.h, 497	EZDP_2_CLUSTER_DATA, 469
ezdp_mem_section_info, 152	EZDP_4_CLUSTER_CODE, 469
cache_size, 152	EZDP_4_CLUSTER_DATA, 469
emem_data_size, 153	EZDP_ALL_CLUSTER_CODE, 470
imem_1_cluster_code_size, 153	EZDP_ALL_CLUSTER_DATA, 470
imem_1_cluster_data_size, 153	EZDP_ALL_CLUSTER_DATA_EXT_MEM, 470
imem_16_cluster_code_size, 154	EZDP_ALL_CLUSTER_IO, 470
imem_16_cluster_data_size, 153	ezdp_dma_flags, 470
imem_2_cluster_code_size, 153	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE1_OFFSET, 468
imem_2_cluster_data_size, 153	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE1_SIZE, 468
imem_4_cluster_code_size, 154	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE1_WORD_OFFSET, 468
imem_4_cluster_data_size, 153	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE1_WORD_SELECT, 468
imem_all_cluster_code_size, 154	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE2_OFFSET, 468
imem_all_cluster_data_size, 153	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE2_SIZE, 468
imem_half_cluster_code_size, 153	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE2_WORD_OFFSET, 468
imem_half_cluster_data_size, 152	EZDP_DUAL_ADD32_RESULT_ORIGINAL_VA_LUE2_WORD_SELECT, 468
imem_private_data_size, 152	EZDP_DUAL_ADD32_RESULT_WORD_COUNT, 468
private_cmем_size, 152	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE1_OFFSET, 469
shared_cmем_size, 152	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE1_SIZE, 468
ezdp_mem_section_info_str ezdp.h, 194	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE1_WORD_OFFSET, 469
ezdp_mem_set ezdp_string.h, 584	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE1_WORD_SELECT, 469
	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE2_OFFSET, 468
	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE2_SIZE, 468
	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE2_WORD_OFFSET, 468
	EZDP_DUAL_ADD64_RESULT_ORIGINAL_VA_LUE2_WORD_SELECT, 468

EZDP\_DUAL\_ADD64\_RESULT\_WORD\_COUNT, 469  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_SIZE, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_WORD\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_MSB\_WORD\_SELECT, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_SIZE, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_WORD\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_ADDRESS\_WORD\_SELECT, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_MASK, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_SIZE, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_WORD\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_MEM\_TYPE\_WORD\_SELECT, 465  
 EZDP\_EXT\_ADDR\_MSID\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_MSID\_SIZE, 465  
 EZDP\_EXT\_ADDR\_MSID\_WORD\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_MSID\_WORD\_SELECT, 465  
 EZDP\_EXT\_ADDR\_RESERVED14\_15\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_RESERVED14\_15\_SIZE, 465  
 EZDP\_EXT\_ADDR\_RESERVED16\_31\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_RESERVED16\_31\_SIZE, 465  
 EZDP\_EXT\_ADDR\_RESERVED4\_7\_OFFSET, 465  
 EZDP\_EXT\_ADDR\_RESERVED4\_7\_SIZE, 465  
 EZDP\_EXT\_ADDR\_WORD\_COUNT, 465  
 EZDP\_EXTERNAL\_MS, 470  
 EZDP\_HALF\_CLUSTER\_CODE, 469  
 EZDP\_HALF\_CLUSTER\_DATA, 469  
 ezdp\_internal\_mem\_space, 469  
 EZDP\_INTERNAL\_MS, 470  
 ezdp\_mem\_space\_type, 470  
 EZDP\_MEMORY\_FLAG\_CLASS\_0, 470  
 EZDP\_MEMORY\_FLAG\_CLASS\_1, 470  
 EZDP\_MEMORY\_FLAG\_CLASS\_2, 470  
 EZDP\_MEMORY\_FLAG\_CLASS\_3, 470  
 EZDP\_MEMORY\_FLAG\_OVERWRITE, 471  
 EZDP\_MEMORY\_FLAG\_UNCACHED, 471  
 EZDP\_NULL\_SUM\_ADDR, 469  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_MASK, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_SIZE, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_WORD\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_WORD\_SELECT, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_SIZE, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_WORD\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_WORD\_SELECT, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_SIZE, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_WORD\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_WORD\_SELECT, 467  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_MASK, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_SIZE, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_WORD\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_WORD\_SELECT, 466  
 EZDP\_PCI\_ADDR\_MSID\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_MSID\_SIZE, 466  
 EZDP\_PCI\_ADDR\_MSID\_WORD\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_MSID\_WORD\_SELECT, 466  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_SIZE, 467  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_WORD\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_WORD\_SELECT, 467  
 EZDP\_PCI\_ADDR\_RESERVED14\_15\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_RESERVED14\_15\_SIZE, 466  
 EZDP\_PCI\_ADDR\_RESERVED29\_30\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_RESERVED29\_30\_SIZE, 467  
 EZDP\_PCI\_ADDR\_RESERVED4\_7\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_RESERVED4\_7\_SIZE, 466  
 ezdp\_pci\_addr\_type, 470  
 EZDP\_PCI\_ADDR\_TYPE\_TRANSLATED, 470  
 EZDP\_PCI\_ADDR\_TYPE\_UNTRANSLATED, 470  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_MASK, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_SIZE, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_WORD\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_WORD\_SELECT, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_OFFSET, 466  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_SIZE, 466  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_WORD\_OFFSET, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_WORD\_SELECT, 466  
 EZDP\_PCI\_ADDR\_WORD\_COUNT, 467  
 EZDP\_PCI\_FLAG\_ATU\_BYPASS, 471  
 EZDP\_PCI\_FLAG\_NO\_SNOOP, 471  
 EZDP\_PCI\_FLAG\_RELEX\_ORDERED, 471  
 EZDP\_PRIVATE\_DATA, 469  
 EZDP\_STACK, 469  
 EZDP\_SUM\_ADDR\_ELEMENT\_INDEX\_OFFSET, 466

EZDP\_SUM\_ADDR\_ELEMENT\_INDEX\_SIZE, 465  
 EZDP\_SUM\_ADDR\_MEM\_TYPE\_MASK, 466  
 EZDP\_SUM\_ADDR\_MEM\_TYPE\_OFFSET, 466  
 EZDP\_SUM\_ADDR\_MEM\_TYPE\_SIZE, 466  
 EZDP\_SUM\_ADDR\_MSID\_OFFSET, 466  
 EZDP\_SUM\_ADDR\_MSID\_SIZE, 466  
 ezdp\_sum\_addr\_t, 469  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_BASE\_INDEX\_OFFSET, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_BASE\_INDEX\_SIZE, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_BITS\_OFFSET, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_BITS\_SIZE, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_MASK, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_OFFSET, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_SIZE, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SIZE\_OFFSET, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SIZE\_SIZE, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MEM\_TYPE\_MASK, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MEM\_TYPE\_OFFSET, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MEM\_TYPE\_SIZE, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MSID\_OFFSET, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MSID\_SIZE, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_RESERVED2\_5\_26\_OFFSET, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_RESERVED2\_5\_26\_SIZE, 468  
 ezdp\_sum\_addr\_table\_desc\_t, 469  
 EZDP\_MEMORY\_FLAG\_CLASS\_0  
 ezdp\_memory\_defs.h, 470  
 EZDP\_MEMORY\_FLAG\_CLASS\_1  
 ezdp\_memory\_defs.h, 470  
 EZDP\_MEMORY\_FLAG\_CLASS\_2  
 ezdp\_memory\_defs.h, 470  
 EZDP\_MEMORY\_FLAG\_CLASS\_3  
 ezdp\_memory\_defs.h, 470  
 EZDP\_MEMORY\_FLAG\_OVERWRITE  
 ezdp\_memory\_defs.h, 471  
 EZDP\_MEMORY\_FLAG\_UNCACHED  
 ezdp\_memory\_defs.h, 471  
 ezdp\_merge\_2\_bitfields  
 ezdp\_math.h, 450  
 ezdp\_merge\_2\_bits  
 ezdp\_math.h, 451  
 ezdp\_merge\_3\_bits  
 ezdp\_math.h, 452  
 ezdp\_merge\_4\_bits  
 ezdp\_math.h, 453  
 ezdp\_merge\_bit  
 ezdp\_math.h, 451  
 ezdp\_merge\_bitfield  
 ezdp\_math.h, 449  
 ezdp\_merge\_pow\_of\_2  
 ezdp\_math.h, 448  
 ezdp\_mod  
 ezdp\_math.h, 447  
 ezdp\_modify\_hash\_entry  
 ezdp\_search.h, 511  
 ezdp\_modify\_table\_entry  
 ezdp\_search.h, 508  
 ezdp\_msg\_posted\_ctr\_type  
 ezdp\_counter\_defs.h, 294  
 EZDP\_MULTICAST  
 ezdp\_frame\_defs.h, 389  
 ezdp\_multicast\_control  
 ezdp\_frame\_defs.h, 389  
 EZDP\_NEVER\_DROP  
 ezdp\_job\_defs.h, 432  
 EZDP\_NONE  
 ezdp\_search\_defs.h, 553  
 ezdp\_not  
 ezdp\_math.h, 445  
 ezdp\_notice\_pending  
 ezdp\_job.h, 404  
 ezdp\_notifier  
 ezdp\_job.h, 393  
 ezdp\_notifier\_t  
 ezdp\_job.h, 393  
 ezdp\_notify\_cpu  
 ezdp\_job.h, 403  
 EZDP\_NULL\_CTR\_MSG  
 ezdp\_counter\_defs.h, 293  
 EZDP\_NULL\_FRAME  
 ezdp\_frame\_defs.h, 389  
 EZDP\_NULL\_INDEX  
 ezdp\_pool\_defs.h, 497  
 EZDP\_NULL\_POSTED\_CTR\_MSG  
 ezdp\_counter\_defs.h, 294  
 EZDP\_NULL\_QLOCK\_SLOT  
 ezdp\_lock\_defs.h, 440  
 EZDP\_NULL\_SUM\_ADDR  
 ezdp\_memory\_defs.h, 469  
 EZDP\_OPPORTUNISTIC  
 ezdp\_search\_defs.h, 553  
 ezdp\_or  
 ezdp\_math.h, 445  
 ezdp\_order\_lock\_qlock  
 ezdp\_lock.h, 437  
 ezdp\_output\_queue\_status, 155  
 \_\_pad0\_\_, 155  
 congestion, 155  
 raw\_data, 155  
 ready, 155  
 size, 155  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_MASK  
 ezdp\_job\_defs.h, 428

EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_OFFSET  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_CONGESTION\_SIZE  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_MASK  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_OFFSET  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_READY\_SIZE  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_RESERVED18\_31\_OFFSET  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_RESERVED18\_31\_SIZE  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_SIZE\_OFFSET  
     ezdp\_job\_defs.h, 428  
 EZDP\_OUTPUT\_QUEUE\_STATUS\_SIZE\_SIZE  
     ezdp\_job\_defs.h, 428  
 ezdp\_output\_queue\_status\_t  
     ezdp\_job\_defs.h, 430  
 EZDP\_PAD\_ALG\_TCAM\_WORKING\_AREA  
     ezdp\_search\_defs.h, 530  
 EZDP\_PAD\_HASH\_ENTRY  
     ezdp\_search\_defs.h, 531  
 EZDP\_PAD\_HASH\_WORKING\_AREA  
     ezdp\_search\_defs.h, 531  
 ezdp\_pci.h  
     ezdp\_copy\_frame\_data\_from\_pci, 475  
     ezdp\_copy\_frame\_data\_from\_pci\_async, 476  
     ezdp\_copy\_frame\_data\_to\_pci, 475  
     ezdp\_copy\_frame\_data\_to\_pci\_async, 475  
     ezdp\_copy\_pci\_data\_from\_ext\_addr, 478  
     ezdp\_copy\_pci\_data\_from\_ext\_addr\_async, 479  
     ezdp\_copy\_pci\_data\_to\_ext\_addr, 478  
     ezdp\_copy\_pci\_data\_to\_ext\_addr\_async, 478  
     ezdp\_get\_pci\_ctrl\_reg, 482  
     ezdp\_get\_pci\_msg, 473  
     ezdp\_get\_pci\_msgq\_read\_index, 474  
     ezdp\_get\_pci\_msgq\_write\_index, 474  
     ezdp\_init\_pci\_queue\_desc, 473  
     ezdp\_load\_data\_from\_pci, 476  
     ezdp\_load\_data\_from\_pci\_async, 477  
     ezdp\_send\_interrupt\_to\_pci, 481  
     ezdp\_send\_interrupt\_to\_pci\_async, 482  
     ezdp\_send\_message\_to\_pci, 481  
     ezdp\_send\_message\_to\_pci\_async, 481  
     ezdp\_set\_pci\_ctrl\_reg, 483  
     ezdp\_set\_pci\_msgq\_read\_index, 474  
     ezdp\_set\_pci\_msgq\_read\_index\_async, 474  
     ezdp\_store\_data\_to\_pci, 477  
     ezdp\_store\_data\_to\_pci\_async, 477  
     ezdp\_translate\_pci\_addr, 479  
     ezdp\_translate\_pci\_addr\_async, 480  
     ezdp\_translate\_pci\_addr\_to\_ext\_addr, 480  
     ezdp\_translate\_pci\_addr\_to\_ext\_addr\_async, 480  
 ezdp\_pci\_addr, 156  
     \_\_pad0\_\_, 156  
     \_\_pad1\_\_, 156  
     \_\_pad2\_\_, 156  
     address, 157  
     address\_msb, 157  
     msid, 156  
     phy\_func, 156  
     raw\_data, 156  
     virt\_func, 156  
     virt\_func\_en, 156  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_MASK  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_OFFSET  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_SIZE  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_WORD\_OFFSET  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDR\_TYPE\_WORD\_SELECT  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_OFFSET  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_SIZE  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_WORD\_OFFSET  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_MSB\_WORD\_SELECT  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_ADDRESS\_OFFSET  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_SIZE  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_WORD\_OFFSET  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_ADDRESS\_WORD\_SELECT  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_MASK  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_OFFSET  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_SIZE  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_WORD\_OFFSET  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MEM\_TYPE\_WORD\_SELECT  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MSID\_OFFSET  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MSID\_SIZE  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MSID\_WORD\_OFFSET  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_MSID\_WORD\_SELECT  
     ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_OFFSET  
     ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_SIZE



ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_WORD\_OFFSET  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_PHY\_FUNC\_WORD\_SELECT  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_RESERVED14\_15\_OFFSET  
 ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_RESERVED14\_15\_SIZE  
 ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_RESERVED29\_30\_OFFSET  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_RESERVED29\_30\_SIZE  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_RESERVED4\_7\_OFFSET  
 ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_RESERVED4\_7\_SIZE  
 ezdp\_memory\_defs.h, 466  
 ezdp\_pci\_addr\_type  
 ezdp\_memory\_defs.h, 470  
 EZDP\_PCI\_ADDR\_TYPE\_TRANSLATED  
 ezdp\_memory\_defs.h, 470  
 EZDP\_PCI\_ADDR\_TYPE\_UNTRANSLATED  
 ezdp\_memory\_defs.h, 470  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_MASK  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_OFFSET  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_SIZE  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_WORD\_OFFSET  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_EN\_WORD\_SELECT  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_OFFSET  
 ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_SIZE  
 ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_WORD\_OFFSET  
 ezdp\_memory\_defs.h, 467  
 EZDP\_PCI\_ADDR\_VIRT\_FUNC\_WORD\_SELECT  
 ezdp\_memory\_defs.h, 466  
 EZDP\_PCI\_ADDR\_WORD\_COUNT  
 ezdp\_memory\_defs.h, 467  
 ezdp\_pci\_defs.h  
 EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_SIZE, 490  
 EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_WORD\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_BUF\_DATA\_ADDR\_WORD\_SELECT, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_MASK, 489  
 EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_OFFSET, 489  
 EZDP\_DRIVER\_DESC\_FLAGS\_DATA\_SIZE, 489

EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_MASK, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_ERROR\_SIZE, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_MASK, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_OWNER\_SIZE, 489  
 EZDP\_DRIVER\_DESC\_FLAGS\_SIZE, 490  
 ezdp\_driver\_desc\_flags\_t, 491  
 EZDP\_DRIVER\_DESC\_FLAGS\_TYPE\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_TYPE\_SIZE, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_WORD\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_FLAGS\_WORD\_SELECT, 490  
 EZDP\_DRIVER\_DESC\_LEN\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_LEN\_SIZE, 490  
 EZDP\_DRIVER\_DESC\_LEN\_WORD\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_LEN\_WORD\_SELECT, 490  
 EZDP\_DRIVER\_DESC\_SUB\_TYPE\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_SUB\_TYPE\_SIZE, 490  
 EZDP\_DRIVER\_DESC\_SUB\_TYPE\_WORD\_OFFSET, 491  
 EZDP\_DRIVER\_DESC\_SUB\_TYPE\_WORD\_SELECT, 490  
 EZDP\_DRIVER\_DESC\_TOTAL\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_TOTAL\_SIZE, 490  
 EZDP\_DRIVER\_DESC\_TOTAL\_WORD\_OFFSET, 490  
 EZDP\_DRIVER\_DESC\_TOTAL\_WORD\_SELECT, 490  
 EZDP\_DRIVER\_DESC\_WORD\_COUNT, 491  
 EZDP\_INIT\_PCI\_QUEUE\_DESC\_WORK\_AREA\_SIZE, 491  
 ezdp\_init\_pci\_queue\_desc\_working\_area\_t, 491  
 EZDP\_PCI\_INFO\_ENDPOINT\_MASK, 487  
 EZDP\_PCI\_INFO\_ENDPOINT\_OFFSET, 487  
 EZDP\_PCI\_INFO\_ENDPOINT\_SIZE, 487  
 EZDP\_PCI\_INFO\_PHYS\_FUNC\_OFFSET, 487  
 EZDP\_PCI\_INFO\_PHYS\_FUNC\_SIZE, 487  
 EZDP\_PCI\_INFO\_QUEUE\_OFFSET, 487  
 EZDP\_PCI\_INFO\_QUEUE\_SIZE, 487  
 EZDP\_PCI\_INFO\_RESERVED16\_32\_OFFSET, 487  
 EZDP\_PCI\_INFO\_RESERVED16\_32\_SIZE, 487  
 ezdp\_pci\_info\_t, 491  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_MASK, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_OFFSET, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_SIZE, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_OFFSET, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_SIZE, 487

EZDP\_PCI\_INTERRUPT\_WORK\_AREA\_SIZE, 491

EZDP\_PCI\_MSG\_ATS\_INVALID, 492

EZDP\_PCI\_MSG\_CTRL\_BAR\_NUM\_OFFSET, 488

EZDP\_PCI\_MSG\_CTRL\_BAR\_NUM\_SIZE, 487

EZDP\_PCI\_MSG\_CTRL\_OFFSET, 489

EZDP\_PCI\_MSG\_CTRL\_PHY\_FUNC\_OFFSET, 487

EZDP\_PCI\_MSG\_CTRL\_PHY\_FUNC\_SIZE, 487

EZDP\_PCI\_MSG\_CTRL\_RESERVED10\_11\_OFFSET, 487

EZDP\_PCI\_MSG\_CTRL\_RESERVED10\_11\_SIZE, 487

EZDP\_PCI\_MSG\_CTRL\_RESERVED8\_OFFSET, 487

EZDP\_PCI\_MSG\_CTRL\_RESERVED8\_SIZE, 487

EZDP\_PCI\_MSG\_CTRL\_SIZE, 489

ezdp\_pci\_msg\_ctrl\_t, 491

EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_MASK, 488

EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_OFFSET, 488

EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_SIZE, 488

EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_OFFSET, 487

EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_SIZE, 487

EZDP\_PCI\_MSG\_CTRL\_WORD\_OFFSET, 489

EZDP\_PCI\_MSG\_CTRL\_WORD\_SELECT, 489

EZDP\_PCI\_MSG\_ECC\_OFFSET, 489

EZDP\_PCI\_MSG\_ECC\_SIZE, 489

EZDP\_PCI\_MSG\_ELBI, 492

EZDP\_PCI\_MSG\_ERROR, 491

EZDP\_PCI\_MSG\_FUNCTION\_LEVEL\_RESET, 492

EZDP\_PCI\_MSG\_MSG\_OFFSET, 489

EZDP\_PCI\_MSG\_MSG\_SIZE, 489

EZDP\_PCI\_MSG\_MSG\_WORD\_OFFSET, 489

EZDP\_PCI\_MSG\_MSG\_WORD\_SELECT, 489

EZDP\_PCI\_MSG\_MSIX, 492

EZDP\_PCI\_MSG\_NONE, 492

EZDP\_PCI\_MSG\_OBFF\_ACTIVE, 492

EZDP\_PCI\_MSG\_OBFF\_IDLE, 492

EZDP\_PCI\_MSG\_OBFF\_STATE, 492

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_SIZE, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_WORD\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_WORD\_SELECT, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_SIZE, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_WORD\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_WORD\_SELECT, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_RESERVED\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_RESERVED\_SIZE, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_WORD\_COUNT, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_SIZE, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_WORD\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_WORD\_SELECT, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_SIZE, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_WORD\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_WORD\_SELECT, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_RESERVED\_OFFSET, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_RESERVED\_SIZE, 488

EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_WORD\_COUNT, 488

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_32\_63\_OFFSET, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_32\_63\_SIZE, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_0\_31\_OFFSET, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_0\_31\_SIZE, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_66\_95\_OFFSET, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_66\_95\_SIZE, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_INDEX\_OFFSET, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_INDEX\_SIZE, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_INDEX\_WORD\_OFFSET, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_INDEX\_WORD\_SELECT, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_WORD\_COUNT, 489

EZDP\_PCI\_MSG\_PM, 492

EZDP\_PCI\_MSG\_RESET\_REQUEST, 492

ezdp\_pci\_msg\_type, 491

EZDP\_PCI\_MSG\_VPD\_0, 492

EZDP\_PCI\_MSG\_VPD\_1, 492

EZDP\_PCI\_MSG\_VPD\_2, 492

EZDP\_PCI\_MSG\_VPD\_3, 492

EZDP\_PCI\_MSG\_WORD\_COUNT, 489

ezdp\_pci\_queue\_desc\_t, 491  
 ezdp\_pci\_queue\_type\_t, 491  
 EZDP\_PCI\_RW\_INDEX\_WORK\_AREA\_SIZE, 491  
 EZDP\_PCI\_VERSION\_MAJOR, 487  
 EZDP\_PCI\_VERSION\_MINOR, 487  
 EZDP\_PCI\_FLAG\_ATU\_BYPASS  
   ezdp\_memory\_defs.h, 471  
 EZDP\_PCI\_FLAG\_NO\_SNOOP  
   ezdp\_memory\_defs.h, 471  
 EZDP\_PCI\_FLAG\_RELEX\_ORDERED  
   ezdp\_memory\_defs.h, 471  
 ezdp\_pci\_info, 158  
   \_\_pad0\_\_, 158  
   endpoint, 158  
   phys\_func, 158  
   queue, 158  
   raw\_data, 158  
   virt\_func, 158  
   virt\_func\_en, 158  
 EZDP\_PCI\_INFO\_ENDPOINT\_MASK  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_ENDPOINT\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_ENDPOINT\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_PHYS\_FUNC\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_PHYS\_FUNC\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_QUEUE\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_QUEUE\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_RESERVED16\_32\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_RESERVED16\_32\_SIZE  
   ezdp\_pci\_defs.h, 487  
 ezdp\_pci\_info\_t  
   ezdp\_pci\_defs.h, 491  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_MASK  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_EN\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INFO\_VIRT\_FUNC\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_INTERRUPT\_WORK\_AREA\_SIZE  
   ezdp\_pci\_defs.h, 491  
 ezdp\_pci\_msg, 159  
   \_\_pad0\_\_, 159  
   ats\_payload, 159  
   ctrl, 159  
   elbi\_payload, 159  
   msix\_payload, 159  
   raw\_data, 159  
 EZDP\_PCI\_MSG\_ATS\_INVALID  
   ezdp\_pci\_defs.h, 492  
 ezdp\_pci\_msg\_ctrl, 160  
   \_\_pad0\_\_, 160  
   \_\_pad1\_\_, 160  
   bar\_num, 160  
   phy\_func, 160  
   raw\_data, 160  
   virt\_func, 160  
   virt\_func\_en, 160  
 EZDP\_PCI\_MSG\_CTRL\_BAR\_NUM\_OFFSET  
   ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_CTRL\_BAR\_NUM\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_OFFSET  
   ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_CTRL\_PHY\_FUNC\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_PHY\_FUNC\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_RESERVED10\_11\_OFFSET  
   T  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_RESERVED10\_11\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_RESERVED8\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_RESERVED8\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_SIZE  
   ezdp\_pci\_defs.h, 489  
 ezdp\_pci\_msg\_ctrl\_t  
   ezdp\_pci\_defs.h, 491  
 EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_MASK  
   ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_OFFSET  
   ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_EN\_SIZE  
   ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_OFFSET  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_VIRT\_FUNC\_SIZE  
   ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_MSG\_CTRL\_WORD\_OFFSET  
   ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_CTRL\_WORD\_SELECT  
   ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_ECC\_OFFSET  
   ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_ECC\_SIZE  
   ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_ELBI  
   ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_ERROR  
   ezdp\_pci\_defs.h, 491  
 EZDP\_PCI\_MSG\_FUNCTION\_LEVEL\_RESET  
   ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_MSG\_OFFSET  
   ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_MSG\_SIZE  
   ezdp\_pci\_defs.h, 489

EZDP\_PCI\_MSG\_MSG\_WORD\_OFFSET  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_MSG\_WORD\_SELECT  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_MSIX  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_NONE  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_OBFF\_ACTIVE  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_OBFF\_IDLE  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_OBFF\_STATE  
     ezdp\_pci\_defs.h, 492  
 ezdp\_pci\_msg\_payload\_ats, 161  
     \_\_pad0\_\_, 161  
     data\_lsb, 161  
     data\_msb, 161  
     raw\_data, 161  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_OF  
     FSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_SIZ  
     E  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_W  
     ORD\_OFFSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_LSB\_W  
     ORD\_SELECT  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_O  
     FFSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_SI  
     ZE  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_W  
     ORD\_OFFSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_DATA\_MSB\_W  
     ORD\_SELECT  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_RESERVED\_OF  
     FSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_RESERVED\_SI  
     ZE  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ATS\_WORD\_COUNT  
     ezdp\_pci\_defs.h, 488  
 ezdp\_pci\_msg\_payload\_elbi, 162  
     \_\_pad0\_\_, 162  
     address, 162  
     data, 162  
     raw\_data, 162  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_OF  
     FSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_SIZ  
     E  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_WO  
     RD\_OFFSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_ADDRESS\_WO  
     RD\_SELECT  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_OFFSE  
     T  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_SIZE  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_WORD  
     \_OFFSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_DATA\_WORD  
     \_SELECT  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_RESERVED\_O  
     FFSET  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_RESERVED\_SI  
     ZE  
     ezdp\_pci\_defs.h, 488  
 EZDP\_PCI\_MSG\_PAYLOAD\_ELBI\_WORD\_COUN  
     T  
     ezdp\_pci\_defs.h, 488  
 ezdp\_pci\_msg\_payload\_msix, 163  
     \_\_pad0\_\_, 163  
     \_\_pad1\_\_, 163  
     \_\_pad2\_\_, 163  
     raw\_data, 163  
     vector\_index, 163  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_3  
     2\_63\_OFFSET  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED\_3  
     2\_63\_SIZE  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED0\_  
     31\_OFFSET  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED0\_  
     31\_SIZE  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED66  
     \_95\_OFFSET  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_RESERVED66  
     \_95\_SIZE  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_IND  
     EX\_OFFSET  
     ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_IND  
     EX\_SIZE  
     ezdp\_pci\_defs.h, 489

EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_IND  
     EX\_WORD\_OFFSET  
         ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_VECTOR\_IND  
     EX\_WORD\_SELECT  
         ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PAYLOAD\_MSIX\_WORD\_COUNT  
     T  
         ezdp\_pci\_defs.h, 489  
 EZDP\_PCI\_MSG\_PM  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_RESET\_REQUEST  
     ezdp\_pci\_defs.h, 492  
 ezdp\_pci\_msg\_type  
     ezdp\_pci\_defs.h, 491  
 EZDP\_PCI\_MSG\_VPD\_0  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_VPD\_1  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_VPD\_2  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_VPD\_3  
     ezdp\_pci\_defs.h, 492  
 EZDP\_PCI\_MSG\_WORD\_COUNT  
     ezdp\_pci\_defs.h, 489  
 ezdp\_pci\_queue\_desc\_t  
     ezdp\_pci\_defs.h, 491  
 ezdp\_pci\_queue\_type\_t  
     ezdp\_pci\_defs.h, 491  
 EZDP\_PCI\_RW\_INDEX\_WORK\_AREA\_SIZE  
     ezdp\_pci\_defs.h, 491  
 EZDP\_PCI\_VERSION\_MAJOR  
     ezdp\_pci\_defs.h, 487  
 EZDP\_PCI\_VERSION\_MINOR  
     ezdp\_pci\_defs.h, 487  
 ezdp\_peek\_list  
     ezdp\_queue.h, 504  
 EZDP\_PERIODIC\_CTR\_MSG  
     ezdp\_counter\_defs.h, 293  
 EZDP\_PERIODIC\_POSTED\_CTR\_MSG  
     ezdp\_counter\_defs.h, 294  
 ezdp\_pool.h  
     ezdp\_alloc\_index, 493  
     ezdp\_alloc\_multi\_index, 494  
     ezdp\_alloc\_multi\_index\_async, 494  
     ezdp\_alloc\_obj, 495  
     ezdp\_free\_index, 493  
     ezdp\_free\_index\_async, 494  
     ezdp\_free\_multi\_index, 494  
     ezdp\_free\_multi\_index\_async, 495  
     ezdp\_free\_obj, 496  
     ezdp\_get\_obj, 496  
     ezdp\_init\_memory\_pool, 495  
     ezdp\_read\_free\_indexes, 495  
     ezdp\_read\_free\_objs, 496  
 ezdp\_pool\_defs.h  
     ezdp\_mem\_pool\_t, 497  
     EZDP\_NULL\_INDEX, 497  
 EZDP\_PORT\_NODE  
     ezdp\_job\_defs.h, 431  
 ezdp\_posted\_ctr\_msg, 164  
     \_\_pad0\_\_, 164  
     \_\_pad1\_\_, 164  
     \_\_pad2\_\_, 164  
     clear, 164  
     flush, 165  
     overrun\_error\_condition, 164  
     raw\_data, 164  
     sum\_addr, 165  
     value, 165  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_MASK  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_OFFSET  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_SIZE  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_WORD\_OFFSET  
     ET  
         ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_CLEAR\_WORD\_SELECT  
     CT  
         ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_ECC\_OFFSET  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_ECC\_SIZE  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_MASK  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_OFFSET  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_SIZE  
     ezdp\_counter\_defs.h, 290  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_WORD\_OFFSET  
     ET  
         ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_FLUSH\_WORD\_SELECT  
     CT  
         ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_OFFSET  
     ezdp\_counter\_defs.h, 290  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_SIZE  
     ezdp\_counter\_defs.h, 290  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_WORD\_OFFSET  
     FFSET  
         ezdp\_counter\_defs.h, 290  
 EZDP\_POSTED\_CTR\_MSG\_MSG\_TYPE\_WORD\_SELECT  
     ELECT  
         ezdp\_counter\_defs.h, 290  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_MASK  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_OFFSET  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_SIZE  
     ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR\_CONDITION\_WORD\_OFFSET  
     ezdp\_counter\_defs.h, 291

EZDP\_POSTED\_CTR\_MSG\_OVERRUN\_ERROR\_C  
 ONDITION\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 291  
 ezdp\_posted\_ctr\_msg\_queue\_desc\_t  
 ezdp\_counter\_defs.h, 292  
 EZDP\_POSTED\_CTR\_MSG\_RESERVED5\_6\_OFFS  
 ET  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_RESERVED5\_6\_SIZE  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_RESERVED8\_23\_OFFS  
 ET  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_RESERVED8\_23\_SIZE  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_OFFSET  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_SIZE  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_WORD\_  
 OFFSET  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_SUM\_ADDR\_WORD\_S  
 ELECT  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_OFFSET  
 ezdp\_counter\_defs.h, 292  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_SIZE  
 ezdp\_counter\_defs.h, 291  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_WORD\_OFFS  
 ET  
 ezdp\_counter\_defs.h, 292  
 EZDP\_POSTED\_CTR\_MSG\_VALUE\_WORD\_SELE  
 CT  
 ezdp\_counter\_defs.h, 292  
 EZDP\_POSTED\_CTR\_MSG\_WORD\_COUNT  
 ezdp\_counter\_defs.h, 292  
 EZDP\_POSTED\_CTR\_MSG\_WORK\_AREA\_SIZE  
 ezdp\_counter\_defs.h, 292  
 ezdp\_pow\_of\_2  
 ezdp\_math.h, 447  
 ezdp\_prefetch\_bitwise\_ctr  
 ezdp\_counter.h, 249  
 ezdp\_prefetch\_bitwise\_ctr\_async  
 ezdp\_counter.h, 249  
 ezdp\_prefetch\_dual\_ctr  
 ezdp\_counter.h, 241  
 ezdp\_prefetch\_dual\_ctr\_async  
 ezdp\_counter.h, 241  
 ezdp\_prefetch\_single\_ctr  
 ezdp\_counter.h, 237  
 ezdp\_prefetch\_single\_ctr\_async  
 ezdp\_counter.h, 238  
 ezdp\_prefetch\_tb\_ctr  
 ezdp\_counter.h, 253  
 ezdp\_prefetch\_tb\_ctr\_async  
 ezdp\_counter.h, 254  
 ezdp\_prefetch\_watchdog\_ctr  
 ezdp\_counter.h, 259  
 ezdp\_prefetch\_watchdog\_ctr\_async  
 ezdp\_counter.h, 259  
 EZDP\_PRIVATE\_DATA  
 ezdp\_memory\_defs.h, 469  
 ezdp\_prm\_add\_hash\_entry  
 ezdp\_search\_prm.h, 559  
 ezdp\_prm\_compress\_hash\_entry  
 ezdp\_search\_prm.h, 561  
 ezdp\_prm\_delete\_hash\_entry  
 ezdp\_search\_prm.h, 560  
 ezdp\_prm\_delete\_table\_entry  
 ezdp\_search\_prm.h, 556  
 ezdp\_prm\_get\_hash\_base\_addr  
 ezdp\_search\_prm.h, 558  
 ezdp\_prm\_get\_hash\_first\_entry  
 ezdp\_search\_prm.h, 560  
 ezdp\_prm\_get\_hash\_next\_entry  
 ezdp\_search\_prm.h, 561  
 ezdp\_prm\_get\_table\_base\_addr  
 ezdp\_search\_prm.h, 556  
 ezdp\_prm\_get\_ultra\_ip\_base\_addr  
 ezdp\_search\_prm.h, 561  
 ezdp\_prm\_hash\_bulk\_key  
 ezdp\_search\_prm.h, 557  
 ezdp\_prm\_hash\_key32  
 ezdp\_search\_prm.h, 557  
 ezdp\_prm\_hash\_key64  
 ezdp\_search\_prm.h, 557  
 ezdp\_prm\_locate\_hash\_entry  
 ezdp\_search\_prm.h, 559  
 ezdp\_prm\_lock\_hash\_slot  
 ezdp\_search\_prm.h, 557  
 ezdp\_prm\_lock\_table\_line  
 ezdp\_search\_prm.h, 555  
 ezdp\_prm\_lookup\_alg\_tcam  
 ezdp\_search\_prm.h, 562  
 ezdp\_prm\_lookup\_hash\_entry  
 ezdp\_search\_prm.h, 558  
 ezdp\_prm\_lookup\_table\_entry  
 ezdp\_search\_prm.h, 556  
 ezdp\_prm\_lookup\_ultra\_ip\_entry  
 ezdp\_search\_prm.h, 562  
 ezdp\_prm\_modify\_hash\_entry  
 ezdp\_search\_prm.h, 560  
 ezdp\_prm\_trylock\_hash\_slot  
 ezdp\_search\_prm.h, 558  
 ezdp\_prm\_trylock\_table\_line  
 ezdp\_search\_prm.h, 555  
 ezdp\_prm\_unlock\_hash\_slot  
 ezdp\_search\_prm.h, 558  
 ezdp\_prm\_unlock\_table\_line  
 ezdp\_search\_prm.h, 555  
 ezdp\_prm\_update\_table\_entry  
 ezdp\_search\_prm.h, 556  
 ezdp\_processor.h  
 ezdp\_calc\_cpu\_id, 499  
 ezdp\_get\_cluster\_id, 499  
 ezdp\_get\_core\_id, 499  
 ezdp\_get\_cpu\_id, 498  
 ezdp\_get\_thread\_id, 498  
 EZDP\_MAX\_CPUS\_ID, 498

EZDP\_MAX\_HW\_CLUSTERS, 498  
 EZDP\_MAX\_HW\_CORES, 498  
 EZDP\_MAX\_HW\_THREADS, 498  
 ezdp\_mb, 499  
 ezdp\_rmb, 500  
 ezdp\_rsync, 499  
 ezdp\_sync, 499  
 ezdp\_wmb, 500  
 ezdp\_qlock\_slot\_t  
   ezdp\_lock\_defs.h, 440  
 ezdp\_qlock\_t  
   ezdp\_lock\_defs.h, 440  
 EZDP\_QLOCK\_WORK\_AREA\_SIZE  
   ezdp\_lock\_defs.h, 440  
 EZDP\_QUEUE  
   ezdp\_job\_defs.h, 433  
 ezdp\_queue.h  
   ezdp\_dequeue\_list, 503  
   ezdp\_dequeue\_ring, 502  
   ezdp\_destroy\_list, 504  
   ezdp\_enqueue\_list, 503  
   ezdp\_enqueue\_ring, 502  
   ezdp\_init\_list, 503  
   ezdp\_init\_ring, 501  
   ezdp\_list\_empty, 503  
   ezdp\_peek\_list, 504  
   ezdp\_ring\_empty, 501  
   ezdp\_ring\_full, 501  
   ezdp\_ring\_length, 502  
 ezdp\_queue\_defs.h  
   ezdp\_list\_t, 505  
   EZDP\_LIST\_WORK\_AREA\_SIZE, 505  
   ezdp\_ring\_t, 505  
   EZDP\_RING\_WORK\_AREA\_SIZE, 505  
 EZDP\_QUEUE\_WITH\_SEQ\_NUM  
   ezdp\_job\_defs.h, 433  
 ezdp\_read\_and\_clear\_bits\_bitwise\_ctr  
   ezdp\_counter.h, 248  
 ezdp\_read\_and\_cond\_dec\_single\_ctr  
   ezdp\_counter.h, 237  
 ezdp\_read\_and\_cond\_write\_bits\_bitwise\_ctr  
   ezdp\_counter.h, 248  
 ezdp\_read\_and\_dec\_bits\_bitwise\_ctr  
   ezdp\_counter.h, 246  
 ezdp\_read\_and\_dec\_dual\_ctr  
   ezdp\_counter.h, 240  
 ezdp\_read\_and\_dec\_single\_ctr  
   ezdp\_counter.h, 236  
 ezdp\_read\_and\_dec\_tb\_ctr  
   ezdp\_counter.h, 253  
 ezdp\_read\_and\_inc\_bits\_bitwise\_ctr  
   ezdp\_counter.h, 245  
 ezdp\_read\_and\_inc\_dual\_ctr  
   ezdp\_counter.h, 239  
 ezdp\_read\_and\_inc\_hier\_tb\_ctr  
   ezdp\_counter.h, 255  
 ezdp\_read\_and\_inc\_single\_ctr  
   ezdp\_counter.h, 235  
 ezdp\_read\_and\_inc\_tb\_ctr  
   ezdp\_counter.h, 252  
 ezdp\_read\_and\_reset\_bitwise\_ctr  
   ezdp\_counter.h, 246  
 ezdp\_read\_and\_reset\_dual\_ctr  
   ezdp\_counter.h, 241  
 ezdp\_read\_and\_reset\_single\_ctr  
   ezdp\_counter.h, 236  
 ezdp\_read\_and\_set\_bits\_bitwise\_ctr  
   ezdp\_counter.h, 247  
 ezdp\_read\_and\_update\_hier\_tb\_ctr  
   ezdp\_counter.h, 256  
 ezdp\_read\_bits\_bitwise\_ctr  
   ezdp\_counter.h, 244  
 ezdp\_read\_bitwise\_ctr  
   ezdp\_counter.h, 244  
 ezdp\_read\_bitwise\_ctr\_cfg  
   ezdp\_counter.h, 242  
 ezdp\_read\_bitwise\_ctr\_cfg\_async  
   ezdp\_counter.h, 242  
 ezdp\_read\_congestion\_status  
   ezdp\_job.h, 405  
 ezdp\_read\_ctr\_msg  
   ezdp\_counter.h, 259  
 ezdp\_read\_dual\_ctr  
   ezdp\_counter.h, 239  
 ezdp\_read\_dual\_ctr\_cfg  
   ezdp\_counter.h, 238  
 ezdp\_read\_flow\_control\_status  
   ezdp\_job.h, 405  
 ezdp\_read\_free\_buf  
   ezdp\_frame.h, 363  
 ezdp\_read\_free\_indexes  
   ezdp\_pool.h, 495  
 ezdp\_read\_free\_job  
   ezdp\_job.h, 395  
 ezdp\_read\_free\_objs  
   ezdp\_pool.h, 496  
 ezdp\_read\_global\_budget  
   ezdp\_job.h, 405  
 ezdp\_read\_hier\_tb\_ctr\_cfg  
   ezdp\_counter.h, 254  
 ezdp\_read\_mc\_buf\_counter  
   ezdp\_frame.h, 373  
 ezdp\_read\_pmu\_app\_schlr\_status  
   ezdp\_job.h, 407  
 ezdp\_read\_pmu\_discard\_output\_queue\_status  
   ezdp\_job.h, 406  
 ezdp\_read\_pmu\_group\_schlr\_status  
   ezdp\_job.h, 407  
 ezdp\_read\_pmu\_input\_queue\_congestion  
   ezdp\_job.h, 405  
 ezdp\_read\_pmu\_input\_queue\_status  
   ezdp\_job.h, 406  
 ezdp\_read\_pmu\_tm\_bypass\_output\_queue\_status  
   ezdp\_job.h, 406  
 ezdp\_read\_pmu\_tm\_output\_queue\_status  
   ezdp\_job.h, 406  
 ezdp\_read\_posted\_ctr\_msg  
   ezdp\_counter.h, 264  
 ezdp\_read\_security\_context  
   ezdp\_security.h, 576

ezdp_read_security_context_async	ezdp_report_posted_ctr
ezdp_security.h, 576	ezdp_counter.h, 262
ezdp_read_security_initial_vector	EZDP_REPORT_POSTED_CTR_MSG
ezdp_security.h, 574	ezdp_counter_defs.h, 294
ezdp_read_security_initial_vector_async	ezdp_report_size
ezdp_security.h, 574	ezdp_job_defs.h, 433
ezdp_read_security_key	ezdp_request_job_id
ezdp_security.h, 571	ezdp_job.h, 397
ezdp_read_security_key_async	ezdp_reset_bitwise_ctr
ezdp_security.h, 572	ezdp_counter.h, 246
ezdp_read_security_mac	ezdp_reset_bitwise_ctr_async
ezdp_security.h, 573	ezdp_counter.h, 246
ezdp_read_security_mac_async	ezdp_reset_dual_ctr
ezdp_security.h, 573	ezdp_counter.h, 240
ezdp_read_security_state	ezdp_reset_dual_ctr_async
ezdp_security.h, 570	ezdp_counter.h, 241
ezdp_read_security_state_async	ezdp_reset_posted_ctr
ezdp_security.h, 570	ezdp_counter.h, 262
ezdp_read_single_ctr	ezdp_reset_posted_ctr_async
ezdp_counter.h, 234	ezdp_counter.h, 263
ezdp_read_single_ctr_cfg	ezdp_reset_single_ctr
ezdp_counter.h, 233	ezdp_counter.h, 236
ezdp_read_single_ctr_cfg_async	ezdp_reset_single_ctr_async
ezdp_counter.h, 233	ezdp_counter.h, 236
ezdp_read_tb_ctr	ezdp_ring_cfg, 166
ezdp_counter.h, 250	base_addr, 166
ezdp_read_tb_ctr_async	control_addr, 166
ezdp_counter.h, 251	size, 166
ezdp_read_tb_ctr_cfg	ezdp_ring_empty
ezdp_counter.h, 250	ezdp_queue.h, 501
ezdp_read_tm_imem_buf_ctr	ezdp_ring_full
ezdp_frame.h, 376	ezdp_queue.h, 501
ezdp_read_watchdog_ctr_cfg	ezdp_ring_length
ezdp_counter.h, 257	ezdp_queue.h, 502
ezdp_rebudget_buf	ezdp_ring_t
ezdp_frame.h, 364	ezdp_queue_defs.h, 505
ezdp_rebudget_buf_async	EZDP_RING_WORK_AREA_SIZE
ezdp_frame.h, 364	ezdp_queue_defs.h, 505
ezdp_rebudget_job	ezdp_rmb
ezdp_job.h, 395	ezdp_processor.h, 500
ezdp_rebudget_job_async	ezdp_rsync
ezdp_job.h, 395	ezdp_processor.h, 499
ezdp_receive_job	ezdp_rtc, 167
ezdp_job.h, 398	nsec, 167
EZDP_RED_TRAFFIC	raw_data, 167
ezdp_counter_defs.h, 293	sec, 167
ezdp_reflect_bits	EZDP_RTC_NSEC_OFFSET
ezdp_math.h, 455	ezdp_time_defs.h, 588
ezdp_reflect_resolution	EZDP_RTC_NSEC_SIZE
ezdp_math.h, 443	ezdp_time_defs.h, 588
EZDP_REFLECT_RESOLUTION_1_BYTE	EZDP_RTC_NSEC_WORD_OFFSET
ezdp_math.h, 443	ezdp_time_defs.h, 588
EZDP_REFLECT_RESOLUTION_2_BYTE	EZDP_RTC_NSEC_WORD_SELECT
ezdp_math.h, 443	ezdp_time_defs.h, 588
EZDP_REFLECT_RESOLUTION_4_BYTE	EZDP_RTC_SEC_OFFSET
ezdp_math.h, 443	ezdp_time_defs.h, 588
EZDP_REPLICA	EZDP_RTC_SEC_SIZE
ezdp_frame_defs.h, 389	ezdp_time_defs.h, 588
ezdp_report_and_clear_posted_ctr	EZDP_RTC_SEC_WORD_OFFSET
ezdp_counter.h, 262	ezdp_time_defs.h, 588



EZDP\_RTC\_SEC\_WORD\_SELECT  
     ezdp\_time\_defs.h, 588  
 EZDP\_RTC\_WORD\_COUNT  
     ezdp\_time\_defs.h, 588  
 ezdp\_run  
     ezdp.h, 194  
 ezdp\_scan\_entry\_cb  
     ezdp\_search\_defs.h, 553  
 ezdp\_scan\_hash\_slot  
     ezdp\_search.h, 513  
 ezdp\_scan\_hash\_slot\_action  
     ezdp\_search\_defs.h, 553  
 ezdp\_scramble\_ext\_addr  
     ezdp\_memory.h, 461  
 ezdp\_scramble\_sum\_addr  
     ezdp\_memory.h, 461  
 ezdp\_search.h  
     ezdp\_add\_hash\_entry, 511  
     ezdp\_add\_table\_entry, 508  
     ezdp\_delete\_hash\_entry, 512  
     ezdp\_delete\_table\_entry, 509  
     ezdp\_get\_hash\_entry\_key, 513  
     ezdp\_init\_alg\_tcaml\_struct\_desc, 516  
     ezdp\_init\_hash\_struct\_desc, 510  
     ezdp\_init\_table\_struct\_desc, 507  
     ezdp\_init\_ultra\_ip\_struct\_desc, 514  
     ezdp\_lookup\_alg\_tcaml, 517  
     ezdp\_lookup\_ext\_tcaml, 515  
     ezdp\_lookup\_ext\_tcaml\_async, 516  
     ezdp\_lookup\_hash\_entry, 510  
     ezdp\_lookup\_int\_tcaml, 514  
     ezdp\_lookup\_int\_tcaml\_async, 515  
     ezdp\_lookup\_table\_entry, 508  
     ezdp\_lookup\_ultra\_ip\_entry, 514  
     ezdp\_modify\_hash\_entry, 511  
     ezdp\_modify\_table\_entry, 508  
     ezdp\_scan\_hash\_slot, 513  
     ezdp\_update\_hash\_entry, 512  
     ezdp\_update\_table\_entry, 509  
     ezdp\_validate\_alg\_tcaml\_struct\_desc, 516  
     ezdp\_validate\_hash\_struct\_desc, 510  
     ezdp\_validate\_table\_struct\_desc, 507  
     ezdp\_validate\_ultra\_ip\_struct\_desc, 514  
 ezdp\_search\_base\_addr\_t  
     ezdp\_search\_defs.h, 552  
 ezdp\_search\_defs.h  
     EZDP\_ACCEPT\_ENTRY, 553  
     EZDP\_ALG\_TCAML\_MAX\_KEY\_SIZE, 530  
     ezdp\_alg\_tcaml\_struct\_desc\_t, 552  
     EZDP\_ALG\_TCAML\_WORK\_AREA\_SIZE, 532  
     EZDP\_COMPRESS, 553  
     EZDP\_DELETE\_ENTRY, 553  
     ezdp\_ext\_tcaml\_result\_element\_type, 552  
     EZDP\_HASH\_HIGH\_LEVEL\_WORK\_AREA\_SIZE, 532  
     EZDP\_HASH\_LOW\_LEVEL\_WORK\_AREA\_SIZE, 532  
     ezdp\_hash\_struct\_desc\_t, 552  
     ezdp\_hashed\_key\_t, 552  
     EZDP\_INDEX, 552

EZDP\_INDEX\_16B\_DATA, 552  
 EZDP\_INDEX\_32B\_DATA, 552  
 EZDP\_INDEX\_4B\_DATA, 552  
 EZDP\_INDEX\_8B\_DATA, 552  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_ASSOC\_DATA\_COUNT, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_ASSOC\_DATA\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_ASSOC\_DATA\_SIZE, 548  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_LOOKUP\_ERROR\_MASK, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_LOOKUP\_ERROR\_SIZE, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_MATCH\_MASK, 550  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_MATCH\_OFFSET, 550  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_MATCH\_SIZE, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_MATCH\_WORD\_OFFSET, 550  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_MATCH\_WORD\_SELECT, 550  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_RESERVED24\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_RESERVED24\_SIZE, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_TRUNCATED\_MASK, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_TRUNCATED\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_TRUNCATED\_SIZE, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_TRUNCATED\_WORD\_SELECT, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_TYPE\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAML\_16B\_DATA\_RESU  
     LT\_ELEMENT\_TYPE\_SIZE, 549

EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_TYPE\_WORD\_OFFSET, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_TYPE\_WORD\_SELECT, 549  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_MASK, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_OFFSET, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_SIZE, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_WORD\_OFFSET, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_WORD\_SELECT, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_16B\_DATA\_RESU  
 LT\_ELEMENT\_WORD\_COUNT, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_ASSOC\_DATA\_COUNT, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_ASSOC\_DATA\_OFFSET, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_ASSOC\_DATA\_SIZE, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSE  
 T, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELEC  
 T, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_LOOKUP\_ERROR\_MASK, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET,  
 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_LOOKUP\_ERROR\_SIZE, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OF  
 FSET, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SE  
 LECT, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_MATCH\_MASK, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_MATCH\_OFFSET, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_MATCH\_SIZE, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_MATCH\_WORD\_OFFSET, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_MATCH\_WORD\_SELECT, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_RESERVED24\_OFFSET, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_RESERVED24\_SIZE, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TRUNCATED\_MASK, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TRUNCATED\_OFFSET, 550

EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TRUNCATED\_SIZE, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TRUNCATED\_WORD\_OFFSE  
 T, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TRUNCATED\_WORD\_SELEC  
 T, 550  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TYPE\_OFFSET, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TYPE\_SIZE, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TYPE\_WORD\_OFFSET, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_TYPE\_WORD\_SELECT, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_MASK, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_OFFSET, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_SIZE, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_WORD\_OFFSET, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_VALID\_WORD\_SELECT, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_32B\_DATA\_RESU  
 LT\_ELEMENT\_WORD\_COUNT, 551  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_ASSOC\_DATA\_COUNT, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_ASSOC\_DATA\_OFFSET, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_ASSOC\_DATA\_SIZE, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_LOOKUP\_ERROR\_MASK, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_LOOKUP\_ERROR\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_LOOKUP\_ERROR\_SIZE, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_MATCH\_MASK, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_MATCH\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_MATCH\_SIZE, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_RESERVED24\_OFFSET, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_RESERVED24\_SIZE, 546  
 ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element\_t,  
 552  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_TRUNCATED\_MASK, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_TRUNCATED\_OFFSET, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_TRUNCATED\_SIZE, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESUL  
 T\_ELEMENT\_TYPE\_OFFSET, 547

EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_4B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_RESERVED24\_SIZE, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT, 547  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT, 548  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_OFFSET, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_SELECT, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_OFFSET, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSET, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_SELECT, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET, 543

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_MASK, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_OFFSET, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_SIZE, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_OFFSET, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_MATCH\_WORD\_SELECT, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_OFFSET, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_MASK, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_16B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT, 543  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSET, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_OFFSET, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_SELECT, 546  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_OFFSET, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_SELECT, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_OFFSET, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_SIZE, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSET, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_INDEX\_WORD\_SELECT, 544  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_OFFSET, 545  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_32B\_DATA\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WORD\_SELECT, 545

EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_MASK, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_OFFSET, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_OFFSET, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_SIZE, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_SIZE, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_OFFSET, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_WORD_SELECT, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_OFFSET, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_RESERVED23_24_OFFSET, 544	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_SIZE, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_RESERVED23_24_SIZE, 544	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_OFFSET, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_MASK, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_DEVICE_ID_WORD_SELECT, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_OFFSET, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_OFFSET, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_SIZE, 544	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_SIZE, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_OFFSET, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_WORD_OFFSET, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TRUNCATED_WORD_SELECT, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_INDEX_WORD_SELECT, 537
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_OFFSET, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_MASK, 538
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_SIZE, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_OFFSET, 538
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_WORD_OFFSET, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_SIZE, 538
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_TYPE_WORD_SELECT, 545	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_OFFSET, 538
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_MASK, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_LOOKUP_ERROR_WORD_SELECT, 538
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_OFFSET, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_MASK, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_SIZE, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_OFFSET, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_WORD_OFFSET, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_SIZE, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_VALID_WORD_SELECT, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_WORD_OFFSET, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_32B_DATA_RESULT_ELEMENT_WORD_COUNT, 546	EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_MATCH_WORD_SELECT, 539
EZDP_LOOKUP_EXT_TCAM_INDEX_4B_DATA_RESULT_ELEMENT_ASSOC_DATA_COUNT, 539	

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_RESERVED23\_24\_OFFSET,  
 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE,  
 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TRUNCATED\_MASK,  
 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TRUNCATED\_OFFSET,  
 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TRUNCATED\_SIZE,  
 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TRUNCATED\_WORD\_  
 OFFSET, 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TRUNCATED\_WORD\_  
 SELECT, 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TYPE\_OFFSET, 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TYPE\_SIZE, 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET,  
 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT,  
 538  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_VALID\_MASK, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_VALID\_OFFSET, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_VALID\_SIZE, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_VALID\_WORD\_OFFSE  
 T, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_VALID\_WORD\_SELEC  
 T, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_4B\_DATA  
 \_RESULT\_ELEMENT\_WORD\_COUNT, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_ASSOC\_DATA\_COUNT  
 , 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_ASSOC\_DATA\_OFFSE  
 T, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_ASSOC\_DATA\_SIZE,  
 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_  
 OFFSET, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_ASSOC\_DATA\_WORD\_  
 SELECT, 541

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET,  
 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_OF  
 FSET, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_DEVICE\_ID\_WORD\_SE  
 LECT, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_INDEX\_OFFSET, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_INDEX\_SIZE, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_INDEX\_WORD\_OFFSE  
 T, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_INDEX\_WORD\_SELEC  
 T, 539  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MAS  
 K, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFF  
 SET, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE  
 , 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WO  
 RD\_OFFSET, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_LOOKUP\_ERROR\_WO  
 RD\_SELECT, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_MATCH\_MASK, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_MATCH\_OFFSET, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_MATCH\_SIZE, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_MATCH\_WORD\_OFFS  
 ET, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_MATCH\_WORD\_SELE  
 CT, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_RESERVED23\_24\_OFFS  
 ET, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_RESERVED23\_24\_SIZE,  
 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_TRUNCATED\_MASK,  
 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA  
 \_RESULT\_ELEMENT\_TRUNCATED\_OFFSET,  
 540

EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_SIZE, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_OFFSET, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TRUNCATED\_WORD\_SELECT, 540  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_OFFSET, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_SIZE, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_OFFSET, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_TYPE\_WORD\_SELECT, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_MASK, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_OFFSET, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_SIZE, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_OFFSET, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_VALID\_WORD\_SELECT, 541  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_8B\_DATA\_RESULT\_ELEMENT\_WORD\_COUNT, 542  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_MASK, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_ANY\_MATCH\_SIZE, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_DEVICE\_ID\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_DEVICE\_ID\_SIZE, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_INDEX\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_INDEX\_SIZE, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_MASK, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_OFFSET, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_LOOKUP\_ERROR\_SIZE, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_MATCH\_MASK, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_MATCH\_OFFSET, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_MATCH\_SIZE, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_RESERVED23\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_RESERVED23\_SIZE, 536  
 ezdp\_lookup\_ext\_tcam\_index\_result\_element\_t, 552  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_MASK, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_OFFSET, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TRUNCATED\_SIZE, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TYPE\_OFFSET, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_TYPE\_SIZE, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_VALID\_MASK, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_VALID\_OFFSET, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_INDEX\_RESULT\_ELEMENT\_VALID\_SIZE, 537  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MATCH\_MASK, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MATCH\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_ANY\_MATCH\_SIZE, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_ERROR\_MASK, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_ERROR\_OFFSET, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_DEVICE\_ERROR\_SIZE, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_ERROR\_MASK, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_ERROR\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_LOOKUP\_ERROR\_SIZE, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ERROR\_MASK, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ERROR\_OFFSET, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MAC\_ERROR\_SIZE, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_MATCH\_MASK, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_MATCH\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_MULTI\_MATCH\_SIZE, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONTEXT\_MATCH\_ERROR\_MASK, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONTEXT\_MATCH\_ERROR\_OFFSET, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_NO\_CONTEXT\_MATCH\_ERROR\_SIZE, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_RESERVED\_BIT8\_31\_OFFSET, 536

EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_RESER  
 VED\_BIT8\_31\_SIZE, 536  
 ezdp\_lookup\_ext\_tcam\_retval\_t, 552  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_O  
 UT\_ERROR\_MASK, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_O  
 UT\_ERROR\_OFFSET, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TIME\_O  
 UT\_ERROR\_SIZE, 535  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNC  
 ATED\_MASK, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNC  
 ATED\_OFFSET, 536  
 EZDP\_LOOKUP\_EXT\_TCAM\_RETVAL\_TRUNC  
 ATED\_SIZE, 536  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA0\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA0\_SIZE, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA0\_WORD\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA0\_WORD\_SELECT, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA1\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA1\_SIZE, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA1\_WORD\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA1\_WORD\_SELECT, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA2\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA2\_SIZE, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA2\_WORD\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_DATA2\_WORD\_SELECT, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_MATCH\_MASK, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_MATCH\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_MATCH\_SIZE, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_MATCH\_WORD\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_MATCH\_WORD\_SELECT, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_12B\_DATA\_RESU  
 LT\_WORD\_COUNT, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA0\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA0\_SIZE, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA0\_WORD\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA0\_WORD\_SELECT, 534

EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA1\_OFFSET, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA1\_SIZE, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA1\_WORD\_OFFSET, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA1\_WORD\_SELECT, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA2\_OFFSET, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA2\_SIZE, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA2\_WORD\_OFFSET, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA2\_WORD\_SELECT, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA3\_OFFSET, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA3\_SIZE, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA3\_WORD\_OFFSET, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_DATA3\_WORD\_SELECT, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_MATCH\_MASK, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_MATCH\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_MATCH\_SIZE, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_MATCH\_WORD\_OFFSET, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_MATCH\_WORD\_SELECT, 534  
 EZDP\_LOOKUP\_INT\_TCAM\_16B\_DATA\_RESU  
 LT\_WORD\_COUNT, 535  
 EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESUL  
 T\_DATA\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESUL  
 T\_DATA\_SIZE, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESUL  
 T\_MATCH\_MASK, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESUL  
 T\_MATCH\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_4B\_DATA\_RESUL  
 T\_MATCH\_SIZE, 533  
 ezdp\_lookup\_int\_tcam\_4B\_data\_result\_t, 552  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA0\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA0\_SIZE, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA0\_WORD\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA0\_WORD\_SELECT, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA1\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA1\_SIZE, 533



EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA1\_WORD\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_DATA1\_WORD\_SELECT, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_MATCH\_MASK, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_MATCH\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_MATCH\_SIZE, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_MATCH\_WORD\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_MATCH\_WORD\_SELECT, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_8B\_DATA\_RESUL  
 T\_WORD\_COUNT, 533  
 ezdp\_lookup\_int\_tcam\_retval\_t, 552  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_INDEX\_OFFSET, 532  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_INDEX\_SIZE, 532  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_MATCH\_MASK, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_MATCH\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_MATCH\_SIZE, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_MAX\_NUM, 551  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_RESERVED0\_15\_OFFSET, 533  
 EZDP\_LOOKUP\_INT\_TCAM\_STANDARD\_RES  
 ULT\_RESERVED0\_15\_SIZE, 533  
 ezdp\_lookup\_int\_tcam\_standard\_result\_t, 552  
 EZDP\_LOOKUP\_PARITY\_BITS\_SIZE, 532  
 EZDP\_LOOKUP\_RESERVED\_BITS\_SIZE, 532  
 EZDP\_LOOKUP\_RETVAL\_DATA\_OFFSET, 532  
 EZDP\_LOOKUP\_RETVAL\_DATA\_SIZE, 532  
 EZDP\_LOOKUP\_RETVAL\_INFO\_MASK, 532  
 EZDP\_LOOKUP\_RETVAL\_INFO\_OFFSET, 532  
 EZDP\_LOOKUP\_RETVAL\_INFO\_SIZE, 532  
 EZDP\_LOOKUP\_RETVAL\_MATCH\_MASK, 532  
 EZDP\_LOOKUP\_RETVAL\_MATCH\_OFFSET,  
 532  
 EZDP\_LOOKUP\_RETVAL\_MATCH\_SIZE, 532  
 EZDP\_LOOKUP\_RETVAL\_MEM\_ERROR\_MAS  
 K, 532  
 EZDP\_LOOKUP\_RETVAL\_MEM\_ERROR\_OFFS  
 ET, 532  
 EZDP\_LOOKUP\_RETVAL\_MEM\_ERROR\_SIZE,  
 532  
 EZDP\_LOOKUP\_RETVAL\_SUCCESS\_MASK,  
 532  
 EZDP\_LOOKUP\_RETVAL\_SUCCESS\_OFFSET,  
 532  
 EZDP\_LOOKUP\_RETVAL\_SUCCESS\_SIZE, 532  
 ezdp\_lookup\_retval\_t, 552  
 EZDP\_LOOKUP\_VERSION\_MAJOR, 530  
 EZDP\_LOOKUP\_VERSION\_MINOR, 530  
 EZDP\_NONE, 553  
 EZDP\_OPPORTUNISTIC, 553  
 EZDP\_PAD\_ALG\_TCAM\_WORKING\_AREA, 530  
 EZDP\_PAD\_HASH\_ENTRY, 531  
 EZDP\_PAD\_HASH\_WORKING\_AREA, 531  
 ezdp\_scan\_entry\_cb, 553  
 ezdp\_scan\_hash\_slot\_action, 553  
 ezdp\_search\_base\_addr\_t, 552  
 ezdp\_search\_hash\_flags, 552  
 EZDP\_TABLE\_HIGH\_LEVEL\_WORK\_AREA\_SI  
 ZE, 532  
 EZDP\_TABLE\_LOW\_LEVEL\_WORK\_AREA\_SIZ  
 E, 531  
 ezdp\_table\_struct\_desc\_t, 552  
 EZDP\_ULTRA\_IP\_WORK\_AREA\_SIZE, 532  
 EZDP\_UNCONDITIONAL, 553  
 EZDP\_UPDATE\_ENTRY, 553  
 EZDP\_USER\_DEFINED\_ASSOC\_DATA1, 552  
 EZDP\_USER\_DEFINED\_ASSOC\_DATA2, 552  
 EZDP\_USER\_DEFINED\_ASSOC\_DATA3, 552  
 ezdp\_search\_hash\_flags  
 ezdp\_search\_defs.h, 552  
 ezdp\_search\_prm.h  
 ezdp\_prm\_add\_hash\_entry, 559  
 ezdp\_prm\_compress\_hash\_entry, 561  
 ezdp\_prm\_delete\_hash\_entry, 560  
 ezdp\_prm\_delete\_table\_entry, 556  
 ezdp\_prm\_get\_hash\_base\_addr, 558  
 ezdp\_prm\_get\_hash\_first\_entry, 560  
 ezdp\_prm\_get\_hash\_next\_entry, 561  
 ezdp\_prm\_get\_table\_base\_addr, 556  
 ezdp\_prm\_get\_ultra\_ip\_base\_addr, 561  
 ezdp\_prm\_hash\_bulk\_key, 557  
 ezdp\_prm\_hash\_key32, 557  
 ezdp\_prm\_hash\_key64, 557  
 ezdp\_prm\_locate\_hash\_entry, 559  
 ezdp\_prm\_lock\_hash\_slot, 557  
 ezdp\_prm\_lock\_table\_line, 555  
 ezdp\_prm\_lookup\_alg\_tcam, 562  
 ezdp\_prm\_lookup\_hash\_entry, 558  
 ezdp\_prm\_lookup\_table\_entry, 556  
 ezdp\_prm\_lookup\_ultra\_ip\_entry, 562  
 ezdp\_prm\_modify\_hash\_entry, 560  
 ezdp\_prm\_trylock\_hash\_slot, 558  
 ezdp\_prm\_trylock\_table\_line, 555  
 ezdp\_prm\_unlock\_hash\_slot, 558  
 ezdp\_prm\_unlock\_table\_line, 555  
 ezdp\_prm\_update\_table\_entry, 556  
 ezdp\_sec\_alg  
 ezdp\_security\_defs.h, 579  
 ezdp\_sec\_block\_size  
 ezdp\_security\_defs.h, 583  
 ezdp\_sec\_initial\_vector\_size  
 ezdp\_security\_defs.h, 581  
 ezdp\_sec\_key\_size  
 ezdp\_security\_defs.h, 581  
 ezdp\_sec\_mac\_size  
 ezdp\_security\_defs.h, 582  
 ezdp\_sec\_state\_size  
 ezdp\_security\_defs.h, 582  
 ezdp\_security.h

ezdp\_decrypt, 565  
 ezdp\_decrypt\_async, 565  
 ezdp\_encrypt, 564  
 ezdp\_encrypt\_async, 565  
 ezdp\_end\_gcm\_mac\_calculation, 568  
 ezdp\_end\_gcm\_mac\_calculation\_async, 569  
 ezdp\_end\_hmac\_calculation, 567  
 ezdp\_end\_hmac\_calculation\_async, 567  
 ezdp\_expand\_security\_key, 569  
 ezdp\_expand\_security\_key\_async, 569  
 ezdp\_generate\_security\_initial\_vector, 568  
 ezdp\_generate\_security\_initial\_vector\_async, 568  
 ezdp\_mac\_calculation, 566  
 ezdp\_mac\_calculation\_async, 566  
 ezdp\_read\_security\_context, 576  
 ezdp\_read\_security\_context\_async, 576  
 ezdp\_read\_security\_initial\_vector, 574  
 ezdp\_read\_security\_initial\_vector\_async, 574  
 ezdp\_read\_security\_key, 571  
 ezdp\_read\_security\_key\_async, 572  
 ezdp\_read\_security\_mac, 573  
 ezdp\_read\_security\_mac\_async, 573  
 ezdp\_read\_security\_state, 570  
 ezdp\_read\_security\_state\_async, 570  
 ezdp\_security\_block\_size, 576  
 ezdp\_security\_initial\_vector\_size, 575  
 ezdp\_security\_key\_size, 572  
 ezdp\_security\_mac\_size, 573  
 ezdp\_security\_state\_size, 571  
 ezdp\_start\_hmac\_calculation, 566  
 ezdp\_start\_hmac\_calculation\_async, 567  
 ezdp\_write\_security\_context, 575  
 ezdp\_write\_security\_context\_async, 575  
 ezdp\_write\_security\_initial\_vector, 574  
 ezdp\_write\_security\_initial\_vector\_async, 574  
 ezdp\_write\_security\_key, 571  
 ezdp\_write\_security\_key\_async, 571  
 ezdp\_write\_security\_mac, 572  
 ezdp\_write\_security\_mac\_async, 572  
 ezdp\_write\_security\_state, 569  
 ezdp\_write\_security\_state\_async, 570  
 ezdp\_security\_block\_size  
 ezdp\_security.h, 576  
 EZDP\_SECURITY\_CLUSTER\_MAX\_CONTEXTS  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_CONTEXT\_SIZE  
 ezdp\_security\_defs.h, 578  
 ezdp\_security\_defs.h  
 EZDP\_3DES\_BLOCK\_SIZE, 583  
 EZDP\_3DES\_IV\_SIZE, 582  
 EZDP\_3DES2\_CBC\_ALG, 579  
 EZDP\_3DES2\_CFB\_ALG, 579  
 EZDP\_3DES2\_CTR\_ALG, 580  
 EZDP\_3DES2\_ECB\_ALG, 580  
 EZDP\_3DES2\_OFB\_ALG, 580  
 EZDP\_3DES2\_XXX\_KEY\_SIZE, 581  
 EZDP\_3DES3\_CBC\_ALG, 579  
 EZDP\_3DES3\_CFB\_ALG, 579  
 EZDP\_3DES3\_CTR\_ALG, 580  
 EZDP\_3DES3\_ECB\_ALG, 580  
 EZDP\_3DES3\_OFB\_ALG, 580  
 EZDP\_3DES3\_XXX\_KEY\_SIZE, 581  
 EZDP\_AES\_BLOCK\_SIZE, 583  
 EZDP\_AES\_CBC\_128\_ALG, 580  
 EZDP\_AES\_CBC\_192\_ALG, 580  
 EZDP\_AES\_CBC\_256\_ALG, 580  
 EZDP\_AES\_CCM\_128\_ALG, 580  
 EZDP\_AES\_CCM\_192\_ALG, 580  
 EZDP\_AES\_CCM\_256\_ALG, 580  
 EZDP\_AES\_CCM\_MAC\_SIZE, 582  
 EZDP\_AES\_CCM\_STATE\_SIZE, 582  
 EZDP\_AES\_CFB\_128\_ALG, 580  
 EZDP\_AES\_CFB\_192\_ALG, 580  
 EZDP\_AES\_CFB\_256\_ALG, 580  
 EZDP\_AES\_CMAC\_128\_ALG, 581  
 EZDP\_AES\_CMAC\_192\_ALG, 581  
 EZDP\_AES\_CMAC\_256\_ALG, 581  
 EZDP\_AES\_CMAC\_XXX\_MAC\_SIZE, 583  
 EZDP\_AES\_CMAC\_XXX\_STATE\_SIZE, 582  
 EZDP\_AES\_CTR\_128\_ALG, 580  
 EZDP\_AES\_CTR\_192\_ALG, 580  
 EZDP\_AES\_CTR\_256\_ALG, 580  
 EZDP\_AES\_ECB\_128\_ALG, 580  
 EZDP\_AES\_ECB\_192\_ALG, 580  
 EZDP\_AES\_ECB\_256\_ALG, 580  
 EZDP\_AES\_GCM\_128\_ALG, 580  
 EZDP\_AES\_GCM\_192\_ALG, 580  
 EZDP\_AES\_GCM\_256\_ALG, 580  
 EZDP\_AES\_GCM\_MAC\_SIZE, 583  
 EZDP\_AES\_GCM\_STATE\_SIZE, 582  
 EZDP\_AES\_IV\_SIZE, 582  
 EZDP\_AES\_OFB\_128\_ALG, 580  
 EZDP\_AES\_OFB\_192\_ALG, 580  
 EZDP\_AES\_OFB\_256\_ALG, 580  
 EZDP\_AES\_STATE\_SIZE, 582  
 EZDP\_AES\_XCBC\_MAC\_128\_ALG, 581  
 EZDP\_AES\_XCBC\_MAC\_128\_MAC\_SIZE, 583  
 EZDP\_AES\_XCBC\_MAC\_STATE\_SIZE, 582  
 EZDP\_AES\_XXX\_128\_KEY\_SIZE, 581  
 EZDP\_AES\_XXX\_192\_KEY\_SIZE, 581  
 EZDP\_AES\_XXX\_256\_KEY\_SIZE, 581  
 EZDP\_AES\_XXX\_MAC\_128\_KEY\_SIZE, 581  
 EZDP\_AES\_XXX\_MAC\_192\_KEY\_SIZE, 581  
 EZDP\_AES\_XXX\_MAC\_256\_KEY\_SIZE, 581  
 EZDP\_DES\_BLOCK\_SIZE, 583  
 EZDP\_DES\_CBC\_ALG, 579  
 EZDP\_DES\_CFB\_ALG, 579  
 EZDP\_DES\_CTR\_ALG, 580  
 EZDP\_DES\_ECB\_ALG, 580  
 EZDP\_DES\_IV\_SIZE, 582  
 EZDP\_DES\_OFB\_ALG, 579  
 EZDP\_DES\_STATE\_SIZE, 582  
 EZDP\_DES\_XXX\_KEY\_SIZE, 581  
 EZDP\_GHASH\_128\_ALG, 581  
 EZDP\_GHASH\_128\_KEY\_SIZE, 581  
 EZDP\_GHASH\_192\_ALG, 581  
 EZDP\_GHASH\_192\_KEY\_SIZE, 581  
 EZDP\_GHASH\_256\_ALG, 581  
 EZDP\_GHASH\_256\_KEY\_SIZE, 581  
 EZDP\_GHASH\_BLOCK\_SIZE, 583

EZDP\_GHASH\_MAC\_SIZE, 583  
 EZDP\_GHASH\_STATE\_SIZE, 582  
 EZDP\_GHASH\_XXX\_STATE\_SIZE, 582  
 EZDP\_MD5\_ALG, 580  
 EZDP\_MD5\_BLOCK\_SIZE, 583  
 EZDP\_MD5\_MAC\_SIZE, 583  
 EZDP\_MD5\_STATE\_SIZE, 582  
 ezdp\_sec\_alg, 579  
 ezdp\_sec\_block\_size, 583  
 ezdp\_sec\_initial\_vector\_size, 581  
 ezdp\_sec\_key\_size, 581  
 ezdp\_sec\_mac\_size, 582  
 ezdp\_sec\_state\_size, 582  
 EZDP\_SECURITY\_CLUSTER\_MAX\_CONTEXTS, 579  
 EZDP\_SECURITY\_CONTEXT\_SIZE, 578  
 EZDP\_SECURITY\_HANDLE\_ALG\_TYPE\_OFFSET, 579  
 EZDP\_SECURITY\_HANDLE\_ALG\_TYPE\_SIZE, 579  
 EZDP\_SECURITY\_HANDLE\_CONTEXT\_ID\_OFFSET, 579  
 EZDP\_SECURITY\_HANDLE\_CONTEXT\_ID\_SIZE, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED24\_31\_OFFSET, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED24\_31\_SIZE, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED8\_15\_OFFSET, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED8\_15\_SIZE, 579  
 ezdp\_security\_handle\_t, 579  
 EZDP\_SHA1\_ALG, 581  
 EZDP\_SHA1\_BLOCK\_SIZE, 583  
 EZDP\_SHA1\_MAC\_SIZE, 583  
 EZDP\_SHA1\_STATE\_SIZE, 582  
 EZDP\_SHA2\_224\_ALG, 581  
 EZDP\_SHA2\_224\_BLOCK\_SIZE, 583  
 EZDP\_SHA2\_224\_MAC\_SIZE, 583  
 EZDP\_SHA2\_224\_STATE\_SIZE, 582  
 EZDP\_SHA2\_256\_ALG, 581  
 EZDP\_SHA2\_256\_BLOCK\_SIZE, 583  
 EZDP\_SHA2\_256\_MAC\_SIZE, 583  
 EZDP\_SHA2\_256\_STATE\_SIZE, 582  
 EZDP\_SHA2\_384\_ALG, 581  
 EZDP\_SHA2\_384\_BLOCK\_SIZE, 583  
 EZDP\_SHA2\_384\_MAC\_SIZE, 583  
 EZDP\_SHA2\_384\_STATE\_SIZE, 582  
 EZDP\_SHA2\_512\_ALG, 581  
 EZDP\_SHA2\_512\_BLOCK\_SIZE, 583  
 EZDP\_SHA2\_512\_MAC\_SIZE, 583  
 EZDP\_SHA2\_512\_STATE\_SIZE, 582  
 EZDP\_XXX\_MAC\_IV\_SIZE, 582  
 ezdp\_security\_handle, 168  
 \_\_pad0\_\_, 168  
 \_\_pad1\_\_, 168  
 context\_id, 168  
 raw\_data, 168  
 EZDP\_SECURITY\_HANDLE\_ALG\_TYPE\_OFFSET, 579  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_HANDLE\_ALG\_TYPE\_SIZE, 579  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_HANDLE\_CONTEXT\_ID\_OFFSET, 579  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_HANDLE\_CONTEXT\_ID\_SIZE, 579  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED24\_31\_OFFSET, 579  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED24\_31\_SIZE, 579  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED8\_15\_OFFSET, 579  
 ezdp\_security\_defs.h, 579  
 EZDP\_SECURITY\_HANDLE\_RESERVED8\_15\_SIZE, 579  
 ezdp\_security\_defs.h, 579  
 ezdp\_security\_handle\_t, 579  
 ezdp\_security\_defs.h, 579  
 ezdp\_security\_initial\_vector\_size, 575  
 ezdp\_security.h, 575  
 ezdp\_security\_key\_size, 572  
 ezdp\_security.h, 572  
 ezdp\_security\_mac\_size, 573  
 ezdp\_security.h, 573  
 ezdp\_security\_state\_size, 571  
 ezdp\_security.h, 571  
 ezdp\_send\_interrupt\_to\_pci, 481  
 ezdp\_pci.h, 481  
 ezdp\_send\_interrupt\_to\_pci\_async, 482  
 ezdp\_pci.h, 482  
 ezdp\_send\_job\_container, 403  
 ezdp\_job.h, 403  
 ezdp\_send\_job\_id\_container, 402  
 ezdp\_job.h, 402  
 ezdp\_send\_job\_id\_container\_async, 403  
 ezdp\_job.h, 403  
 ezdp\_send\_job\_id\_to\_interface, 399  
 ezdp\_job.h, 399  
 ezdp\_send\_job\_id\_to\_interface\_async, 400  
 ezdp\_job.h, 400  
 ezdp\_send\_job\_id\_to\_queue, 398  
 ezdp\_job.h, 398  
 ezdp\_send\_job\_id\_to\_queue\_async, 398  
 ezdp\_job.h, 398  
 ezdp\_send\_job\_id\_to\_tm, 399  
 ezdp\_job.h, 399  
 ezdp\_send\_job\_id\_to\_tm\_async, 399  
 ezdp\_job.h, 399  
 ezdp\_send\_job\_to\_interface, 401  
 ezdp\_job.h, 401  
 ezdp\_send\_job\_to\_queue, 400  
 ezdp\_job.h, 400  
 ezdp\_send\_job\_to\_tm, 400  
 ezdp\_job.h, 400  
 ezdp\_send\_message\_to\_pci, 481  
 ezdp\_pci.h, 481

ezdp\_send\_message\_to\_pci\_async  
 ezdp\_pci.h, 481  
 ezdp\_set\_bit  
 ezdp\_math.h, 448  
 ezdp\_set\_bits\_bitwise\_ctr  
 ezdp\_counter.h, 247  
 ezdp\_set\_bits\_bitwise\_ctr\_async  
 ezdp\_counter.h, 247  
 ezdp\_set\_pci\_ctrl\_reg  
 ezdp\_pci.h, 483  
 ezdp\_set\_pci\_msgq\_read\_index  
 ezdp\_pci.h, 474  
 ezdp\_set\_pci\_msgq\_read\_index\_async  
 ezdp\_pci.h, 474  
 EZDP\_SET\_SEQ\_NUM  
 ezdp\_job\_defs.h, 433  
 EZDP\_SHA1\_ALG  
 ezdp\_security\_defs.h, 581  
 EZDP\_SHA1\_BLOCK\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA1\_MAC\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA1\_STATE\_SIZE  
 ezdp\_security\_defs.h, 582  
 EZDP\_SHA2\_224\_ALG  
 ezdp\_security\_defs.h, 581  
 EZDP\_SHA2\_224\_BLOCK\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_224\_MAC\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_224\_STATE\_SIZE  
 ezdp\_security\_defs.h, 582  
 EZDP\_SHA2\_256\_ALG  
 ezdp\_security\_defs.h, 581  
 EZDP\_SHA2\_256\_BLOCK\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_256\_MAC\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_256\_STATE\_SIZE  
 ezdp\_security\_defs.h, 582  
 EZDP\_SHA2\_384\_ALG  
 ezdp\_security\_defs.h, 581  
 EZDP\_SHA2\_384\_BLOCK\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_384\_MAC\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_384\_STATE\_SIZE  
 ezdp\_security\_defs.h, 582  
 EZDP\_SHA2\_512\_ALG  
 ezdp\_security\_defs.h, 581  
 EZDP\_SHA2\_512\_BLOCK\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_512\_MAC\_SIZE  
 ezdp\_security\_defs.h, 583  
 EZDP\_SHA2\_512\_STATE\_SIZE  
 ezdp\_security\_defs.h, 582  
 EZDP\_SHARED\_CMEM\_DATA  
 ezdp.h, 192  
 ezdp\_shift\_left  
 ezdp\_math.h, 446

ezdp\_shift\_right  
 ezdp\_math.h, 446  
 EZDP\_SINGLE\_BUCKET  
 ezdp\_counter\_defs.h, 293  
 EZDP\_SINGLE\_CTR  
 ezdp\_counter\_defs.h, 293  
 ezdp\_single\_ctr\_cfg, 169  
 \_\_pad0\_\_, 169  
 \_\_pad1\_\_, 169  
 \_\_pad2\_\_, 169  
 \_\_pad3\_\_, 170  
 \_\_pad4\_\_, 170  
 enable\_exceed\_message, 169  
 raw\_data, 169  
 report\_exceeded, 169  
 value, 170  
 EZDP\_SINGLE\_CTR\_CFG\_ECC\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ECC\_SIZE  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_MASK  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_SIZE  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ENABLE\_EXCEED\_MESSAGE\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDED\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDED\_SIZE  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDED\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_REPORT\_EXCEEDED\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED0\_10\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED0\_10\_SIZE  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED32\_63\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_RESERVED32\_63\_SIZE  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_SUB\_TYPE\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_SUB\_TYPE\_SIZE

ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_SIZE  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_WORD\_OFFSET  
 T  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_VALUE\_WORD\_SELECT  
 T  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_WORD\_COUNT  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ZERO\_OFFSET  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SINGLE\_CTR\_CFG\_ZERO\_SIZE  
 ezdp\_counter\_defs.h, 275  
 EZDP\_SMALL\_LBD  
 ezdp\_frame\_defs.h, 389  
 ezdp\_small\_linked\_buffers\_desc, 171  
 line, 171  
 ezdp\_spinlock\_t  
 ezdp\_lock\_defs.h, 440  
 ezdp\_split\_4\_bits  
 ezdp\_math.h, 454  
 ezdp\_split\_merge\_4\_bits  
 ezdp\_math.h, 454  
 EZDP\_SR\_TCM  
 ezdp\_counter\_defs.h, 293  
 EZDP\_STACK  
 ezdp\_memory\_defs.h, 469  
 ezdp\_start\_hmac\_calculation  
 ezdp\_security.h, 566  
 ezdp\_start\_hmac\_calculation\_async  
 ezdp\_security.h, 567  
 ezdp\_start\_watchdog\_ctr  
 ezdp\_counter.h, 257  
 ezdp\_start\_watchdog\_ctr\_async  
 ezdp\_counter.h, 257  
 EZDP\_STD\_FRAME  
 ezdp\_frame\_defs.h, 389  
 ezdp\_store\_16\_byte\_data\_to\_ext\_addr  
 ezdp\_dma.h, 352  
 ezdp\_store\_16\_byte\_data\_to\_ext\_addr\_async  
 ezdp\_dma.h, 352  
 ezdp\_store\_16\_byte\_data\_to\_sum\_addr  
 ezdp\_dma.h, 356  
 ezdp\_store\_16\_byte\_data\_to\_sum\_addr\_async  
 ezdp\_dma.h, 357  
 ezdp\_store\_32\_byte\_data\_to\_ext\_addr  
 ezdp\_dma.h, 353  
 ezdp\_store\_32\_byte\_data\_to\_ext\_addr\_async  
 ezdp\_dma.h, 353  
 ezdp\_store\_32\_byte\_data\_to\_sum\_addr  
 ezdp\_dma.h, 357  
 ezdp\_store\_32\_byte\_data\_to\_sum\_addr\_async  
 ezdp\_dma.h, 357  
 ezdp\_store\_data\_to\_ext\_addr  
 ezdp\_dma.h, 351  
 ezdp\_store\_data\_to\_ext\_addr\_async  
 ezdp\_dma.h, 352  
 ezdp\_store\_data\_to\_pci  
 ezdp\_pci.h, 477  
 ezdp\_store\_data\_to\_pci\_async  
 ezdp\_pci.h, 477  
 ezdp\_store\_data\_to\_sum\_addr  
 ezdp\_dma.h, 356  
 ezdp\_store\_data\_to\_sum\_addr\_async  
 ezdp\_dma.h, 356  
 ezdp\_store\_frame\_data  
 ezdp\_frame.h, 368  
 ezdp\_store\_frame\_data\_async  
 ezdp\_frame.h, 368  
 ezdp\_store\_frame\_lbd  
 ezdp\_frame.h, 372  
 ezdp\_store\_frame\_lbd\_async  
 ezdp\_frame.h, 372  
 ezdp\_store\_job  
 ezdp\_job.h, 396  
 ezdp\_store\_job\_async  
 ezdp\_job.h, 396  
 ezdp\_store\_job\_container  
 ezdp\_job.h, 397  
 ezdp\_store\_job\_container\_async  
 ezdp\_job.h, 397  
 ezdp\_string.h  
 ezdp\_mem\_cmp, 584  
 ezdp\_mem\_cmp\_byte\_skip, 585  
 ezdp\_mem\_copy, 584  
 ezdp\_mem\_set, 584  
 ezdp\_sub  
 ezdp\_math.h, 444  
 ezdp\_sub\_checksum  
 ezdp\_math.h, 458  
 ezdp\_sum\_addr, 172  
 element\_index, 172  
 msid, 172  
 raw\_data, 172  
 EZDP\_SUM\_ADDR\_ELEMENT\_INDEX\_OFFSET  
 ezdp\_memory\_defs.h, 466  
 EZDP\_SUM\_ADDR\_ELEMENT\_INDEX\_SIZE  
 ezdp\_memory\_defs.h, 465  
 EZDP\_SUM\_ADDR\_MEM\_TYPE\_MASK  
 ezdp\_memory\_defs.h, 466  
 EZDP\_SUM\_ADDR\_MEM\_TYPE\_OFFSET  
 ezdp\_memory\_defs.h, 466  
 EZDP\_SUM\_ADDR\_MEM\_TYPE\_SIZE  
 ezdp\_memory\_defs.h, 466  
 EZDP\_SUM\_ADDR\_MSID\_OFFSET  
 ezdp\_memory\_defs.h, 466  
 EZDP\_SUM\_ADDR\_MSID\_SIZE  
 ezdp\_memory\_defs.h, 466  
 ezdp\_sum\_addr\_t  
 ezdp\_memory\_defs.h, 469  
 ezdp\_sum\_addr\_table\_desc, 173  
 \_\_pad0\_\_, 173  
 base\_index, 174  
 key\_shuff\_bits, 173  
 key\_shuff\_en, 173  
 key\_size, 173

msid, 174  
 raw\_data, 173  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_BASE\_INDEX\_OFFSET  
   ezdp\_memory\_defs.h, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_BASE\_INDEX\_SIZE  
   ezdp\_memory\_defs.h, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_BITS\_OFFSET  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_BITS\_SIZE  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_MASK  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_OFFSET  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SHUFF\_EN\_SIZE  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SIZE\_OFFSET  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_KEY\_SIZE\_SIZE  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MEM\_TYPE\_MASK  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MEM\_TYPE\_OFFSET  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MEM\_TYPE\_SIZE  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MSID\_OFFSET  
   ezdp\_memory\_defs.h, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_MSID\_SIZE  
   ezdp\_memory\_defs.h, 467  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_RESERVED25\_26\_OFFSET  
   ezdp\_memory\_defs.h, 468  
 EZDP\_SUM\_ADDR\_TABLE\_DESC\_RESERVED25\_26\_SIZE  
   ezdp\_memory\_defs.h, 468  
 ezdp\_sum\_addr\_table\_desc\_t  
   ezdp\_memory\_defs.h, 469  
 ezdp\_sum\_addr\_to\_ext\_addr  
   ezdp\_memory.h, 460  
 ezdp\_sync  
   ezdp\_processor.h, 499  
 ezdp\_sync\_cp  
   ezdp.h, 193  
 ezdp\_sync\_frame  
   ezdp\_frame.h, 377  
 EZDP\_TABLE\_HIGH\_LEVEL\_WORK\_AREA\_SIZE  
   ezdp\_search\_defs.h, 532

EZDP\_TABLE\_LOW\_LEVEL\_WORK\_AREA\_SIZE  
   ezdp\_search\_defs.h, 531  
 ezdp\_table\_struct\_desc\_t  
   ezdp\_search\_defs.h, 552  
 ezdp\_tb\_algo  
   ezdp\_counter\_defs.h, 293  
 ezdp\_tb\_color  
   ezdp\_counter\_defs.h, 292  
 ezdp\_tb\_ctr\_cfg, 175  
   \_\_pad0\_\_, 175  
   \_\_pad1\_\_, 176  
   \_\_pad2\_\_, 176  
   \_\_pad3\_\_, 176  
   color\_aware, 175  
   commit\_profile\_id, 176  
   coupling\_flag, 175  
   excess\_profile\_id, 175  
   raw\_data, 175  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_OFFSET  
   T  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_SIZE  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_WORD\_OFFSET  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_ALGORITHM\_TYPE\_WORD\_SELECT  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_MASK  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_OFFSET  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_SIZE  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_WORD\_OFFSET  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COLOR\_AWARE\_WORD\_SELECT  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_OFFSET  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_SIZE  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_WORD\_OFFSET  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COMMIT\_PROFILE\_ID\_WORD\_SELECT  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_MASK  
   ezdp\_counter\_defs.h, 280  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_OFFSET  
   ezdp\_counter\_defs.h, 280  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_SIZE  
   ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_WORD\_OFFSET

ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_COUPLING\_FLAG\_WORD\_SELECT  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_OFFSET  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_SIZE  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_WORD\_OFFSET  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_CFG\_EXCESS\_PROFILE\_ID\_WORD\_SELECT  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_CFG\_RESERVED26\_31\_OFFSET  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_RESERVED26\_31\_SIZE  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_RESERVED32\_63\_OFFSET  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_RESERVED32\_63\_SIZE  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_RESERVED64\_95\_OFFSET  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_RESERVED64\_95\_SIZE  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_RESERVED96\_127\_OFFSET  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_RESERVED96\_127\_SIZE  
ezdp\_counter\_defs.h, 280  
EZDP\_TB\_CTR\_CFG\_WORD\_COUNT  
ezdp\_counter\_defs.h, 280  
ezdp\_tb\_ctr\_result, 177  
\_\_pad0\_\_, 177  
\_\_pad1\_\_, 177  
\_\_pad2\_\_, 178  
empty\_commit\_bucket, 178  
empty\_commit\_bucket\_ug, 177  
empty\_excess\_bucket, 177  
empty\_excess\_bucket\_ug, 177  
raw\_data, 177  
EZDP\_TB\_CTR\_RESULT\_COLOR\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_COLOR\_SIZE  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_COLOR\_WORD\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_COLOR\_WORD\_SELECT  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_MASK  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_SIZE

ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_UG\_MASK  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_UG\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_UG\_SIZE  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_UG\_WORD\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_UG\_WORD\_SELECT  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_WORD\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_COMMIT\_BUCKET\_WORD\_SELECT  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_MASK  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_SIZE  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_UG\_MASK  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_UG\_OFFSET  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_UG\_SIZE  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_UG\_WORD\_OFFSET  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_UG\_WORD\_SELECT  
ezdp\_counter\_defs.h, 279  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_WORD\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_EMPTY\_EXCESS\_BUCKET\_WORD\_SELECT  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_RESERVED0\_31\_OFFSET  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_RESERVED0\_31\_SIZE  
ezdp\_counter\_defs.h, 278  
EZDP\_TB\_CTR\_RESULT\_RESERVED34\_57\_OFFSET  
ezdp\_counter\_defs.h, 278

EZDP\_TB\_CTR\_RESULT\_RESERVED34\_57\_SIZE  
     ezdp\_counter\_defs.h, 278  
 EZDP\_TB\_CTR\_RESULT\_RESERVED60\_63\_OFFSET  
     ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_RESULT\_RESERVED60\_63\_SIZE  
     ezdp\_counter\_defs.h, 279  
 EZDP\_TB\_CTR\_RESULT\_WORD\_COUNT  
     ezdp\_counter\_defs.h, 279  
 EZDP\_THRESHOLD\_CTR\_MSG  
     ezdp\_counter\_defs.h, 293  
 EZDP\_THRESHOLD\_POSTED\_CTR\_MSG  
     ezdp\_counter\_defs.h, 294  
 ezdp\_time.h  
     ezdp\_get\_real\_time\_clock, 586  
     ezdp\_get\_real\_time\_clock\_async, 586  
     ezdp\_get\_system\_tick, 586  
     ezdp\_get\_system\_tick\_async, 586  
 ezdp\_time\_defs.h  
     EZDP\_RTC\_NSEC\_OFFSET, 588  
     EZDP\_RTC\_NSEC\_SIZE, 588  
     EZDP\_RTC\_NSEC\_WORD\_OFFSET, 588  
     EZDP\_RTC\_NSEC\_WORD\_SELECT, 588  
     EZDP\_RTC\_SEC\_OFFSET, 588  
     EZDP\_RTC\_SEC\_SIZE, 588  
     EZDP\_RTC\_SEC\_WORD\_OFFSET, 588  
     EZDP\_RTC\_SEC\_WORD\_SELECT, 588  
     EZDP\_RTC\_WORD\_COUNT, 588  
 EZDP\_TM\_DEST  
     ezdp\_job\_defs.h, 432  
 EZDP\_TM\_REPORT\_WORK\_AREA\_SIZE  
     ezdp\_job\_defs.h, 416  
 EZDP\_TOPOLOGY  
     ezdp\_job\_defs.h, 431  
 EZDP\_TR\_TCM  
     ezdp\_counter\_defs.h, 293  
 EZDP\_TR\_TCM\_MEF  
     ezdp\_counter\_defs.h, 293  
 ezdp\_translate\_pci\_addr  
     ezdp\_pci.h, 479  
 ezdp\_translate\_pci\_addr\_async  
     ezdp\_pci.h, 480  
 ezdp\_translate\_pci\_addr\_to\_ext\_addr  
     ezdp\_pci.h, 480  
 ezdp\_translate\_pci\_addr\_to\_ext\_addr\_async  
     ezdp\_pci.h, 480  
 EZDP\_TRANSMIT  
     ezdp\_job\_defs.h, 433  
 ezdp\_trim\_frame\_head\_working\_area\_t  
     ezdp\_frame\_defs.h, 388  
 ezdp\_try\_lock\_spinlock  
     ezdp\_lock.h, 436  
 ezdp\_try\_unlock\_qlock  
     ezdp\_lock.h, 438  
 ezdp\_tx\_drop\_mode  
     ezdp\_job\_defs.h, 432  
 ezdp\_tx\_packet\_switch\_mode  
     ezdp\_job\_defs.h, 431  
 EZDP\_UG\_FAST\_PATH  
     ezdp\_counter\_defs.h, 293  
 EZDP\_UG\_SLOW\_PATH  
     ezdp\_counter\_defs.h, 293  
 EZDP\_ULTRA\_IP\_WORK\_AREA\_SIZE  
     ezdp\_search\_defs.h, 532  
 EZDP\_UNCONDITIONAL  
     ezdp\_search\_defs.h, 553  
 EZDP\_UNICAST  
     ezdp\_frame\_defs.h, 389  
 ezdp\_unlock\_spinlock  
     ezdp\_lock.h, 436  
 EZDP\_UPDATE\_ENTRY  
     ezdp\_search\_defs.h, 553  
 ezdp\_update\_hash\_entry  
     ezdp\_search.h, 512  
 ezdp\_update\_hier\_tb\_ctr  
     ezdp\_counter.h, 255  
 ezdp\_update\_hier\_tb\_ctr\_async  
     ezdp\_counter.h, 256  
 ezdp\_update\_job\_id\_queue  
     ezdp\_job.h, 400  
 ezdp\_update\_job\_queue  
     ezdp\_job.h, 401  
 ezdp\_update\_table\_entry  
     ezdp\_search.h, 509  
 ezdp\_update\_tb\_ctr  
     ezdp\_counter.h, 250  
 ezdp\_update\_tb\_ctr\_async  
     ezdp\_counter.h, 250  
 EZDP\_USER\_DEFINED\_ASSOC\_DATA1  
     ezdp\_search\_defs.h, 552  
 EZDP\_USER\_DEFINED\_ASSOC\_DATA2  
     ezdp\_search\_defs.h, 552  
 EZDP\_USER\_DEFINED\_ASSOC\_DATA3  
     ezdp\_search\_defs.h, 552  
 ezdp\_valid\_tm\_queue\_depth\_handle  
     ezdp\_job.h, 408  
 ezdp\_validate\_alg\_tcam\_struct\_desc  
     ezdp\_search.h, 516  
 ezdp\_validate\_hash\_struct\_desc  
     ezdp\_search.h, 510  
 ezdp\_validate\_table\_struct\_desc  
     ezdp\_search.h, 507  
 ezdp\_validate\_ultra\_ip\_struct\_desc  
     ezdp\_search.h, 514  
 ezdp\_version, 179  
     build\_number, 180  
     creation\_date, 180  
     creation\_time, 180  
     major\_patch\_version, 180  
     major\_version, 179  
     micro\_patch\_version, 180  
     minor\_patch\_version, 180  
     minor\_version, 179  
     module\_name, 179  
     project\_name, 179  
     version\_char, 179  
     version\_string, 179  
 ezdp\_version.h  
     \_SCM\_REV\_, 589  
     EZDP\_VERSION\_DEF\_MAJOR\_VER, 589



EZDP\_VERSION\_DEF\_MINOR\_VER, 589  
 EZDP\_VERSION\_DEF\_PATCH\_MAJOR\_VER, 589  
 EZDP\_VERSION\_DEF\_PATCH\_MICRO\_VER, 589  
 EZDP\_VERSION\_DEF\_PATCH\_MINOR\_VER, 589  
 EZDP\_VERSION\_DEF\_PROJECT\_NAME, 589  
 EZDP\_VERSION\_DEF\_VERSION\_CHAR, 589  
 EZDP\_VERSION\_DEF\_VERSION\_STRING, 589  
 ezdp\_version\_get\_string, 589  
 EZDP\_VERSION\_MAX\_STRING\_LENGTH, 589  
 EZDP\_VERSION\_STR, 589  
 EZDP\_VERSION\_XSTR, 589  
 EZDP\_VERSION\_DEF\_MAJOR\_VER  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_DEF\_MINOR\_VER  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_DEF\_PATCH\_MAJOR\_VER  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_DEF\_PATCH\_MICRO\_VER  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_DEF\_PATCH\_MINOR\_VER  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_DEF\_PROJECT\_NAME  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_DEF\_VERSION\_CHAR  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_DEF\_VERSION\_STRING  
   ezdp\_version.h, 589  
 ezdp\_version\_get\_string  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_MAX\_STRING\_LENGTH  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_STR  
   ezdp\_version.h, 589  
 EZDP\_VERSION\_XSTR  
   ezdp\_version.h, 589  
 ezdp\_wait\_for\_event  
   ezdp\_job.h, 404  
 ezdp\_wait\_for\_job\_id  
   ezdp\_job.h, 397  
 ezdp\_wait\_for\_notice  
   ezdp\_job.h, 404  
 ezdp\_watchdog\_accumulative\_window\_cfg, 181  
   \_\_pad0\_\_, 181  
   \_\_pad1\_\_, 181  
   \_\_pad2\_\_, 182  
   \_\_pad3\_\_, 182  
   \_\_pad4\_\_, 182  
   \_\_pad5\_\_, 182  
   \_\_pad6\_\_, 182  
   accumulative\_events, 182  
   curr\_events, 182  
   last\_events, 182  
   profile\_id, 182  
   raw\_data, 181  
   valid, 182  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_ACCUMULATIVE\_EVENTS\_OFFSET

  ezdp\_counter\_defs.h, 284  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_ACCUMULATIVE\_EVENTS\_SIZE  
   ezdp\_counter\_defs.h, 284  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_ACCUMULATIVE\_EVENTS\_WORD\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_ALERT\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_ALERT\_SIZE  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_CURR\_EVENTS\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_CURR\_EVENTS\_SIZE  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_CURR\_EVENTS\_WORD\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_CURR\_EVENTS\_WORD\_SELECT  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_INIT\_BIT\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_INIT\_BIT\_SIZE  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_LAST\_EVENTS\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_LAST\_EVENTS\_SIZE  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_LAST\_EVENTS\_WORD\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_LAST\_EVENTS\_WORD\_SELECT  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MAX\_THRESHOLD\_ALERT\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MAX\_THRESHOLD\_ALERT\_SIZE  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MIN\_THRESHOLD\_ALERT\_OFFSET  
   ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_MIN\_THRESHOLD\_ALERT\_SIZE  
   ezdp\_counter\_defs.h, 285

EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PARITY\_OFFSET  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PARITY\_SIZE  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_OFFSET  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_SIZE  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_PROFILE\_ID\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED5\_28\_OFFSET  
 ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED5\_28\_SIZE  
 ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED63\_OFFSET  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_RESERVED63\_SIZE  
 ezdp\_counter\_defs.h, 286  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_MASK  
 ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_OFFSET  
 ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_SIZE  
 ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_VALID\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 285  
 EZDP\_WATCHDOG\_ACCUMULATIVE\_WINDOW\_CFG\_WORD\_COUNT  
 ezdp\_counter\_defs.h, 286  
 ezdp\_watchdog\_ctr\_cfg, 183  
 \_\_pad0\_\_, 183  
 \_\_pad1\_\_, 183  
 \_\_pad2\_\_, 183  
 \_\_pad3\_\_, 183  
 accumulative\_window, 183  
 raw\_data, 183  
 sliding\_window, 183  
 EZDP\_WATCHDOG\_CTR\_CFG\_ECC\_OFFSET  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_ECC\_SIZE  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED0\_18\_OFFSET  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED0\_18\_SIZE  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED32\_63\_OFFSET  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_RESERVED32\_63\_SIZE  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_SUB\_TYPE\_OFFSET  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_SUB\_TYPE\_SIZE  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CFG\_WORD\_COUNT  
 ezdp\_counter\_defs.h, 287  
 ezdp\_watchdog\_ctr\_check\_result, 185  
 \_\_pad0\_\_, 186  
 \_\_pad1\_\_, 186  
 \_\_pad2\_\_, 186  
 \_\_pad3\_\_, 186  
 \_\_pad4\_\_, 186  
 \_\_pad5\_\_, 186  
 \_\_pad6\_\_, 186  
 \_\_pad7\_\_, 186  
 \_\_pad8\_\_, 186  
 alert, 185  
 max\_threshold\_alert, 185  
 min\_threshold\_alert, 185  
 raw\_data, 185  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ACCUMULATIVE\_EVENTS\_OFFSET  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ACCUMULATIVE\_EVENTS\_SIZE  
 ezdp\_counter\_defs.h, 287  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_MASK  
 ezdp\_counter\_defs.h, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_OFFSET  
 ezdp\_counter\_defs.h, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_SIZE  
 ezdp\_counter\_defs.h, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_WORD\_OFFSET  
 ezdp\_counter\_defs.h, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_ALERT\_WORD\_SELECT  
 ezdp\_counter\_defs.h, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_CURR\_EVENTS\_OFFSET  
 ezdp\_counter\_defs.h, 288  
 EZDP\_WATCHDOG\_CTR\_CHECK\_RESULT\_CURR\_EVENTS\_SIZE  
 ezdp\_counter\_defs.h, 288

EZDP_WATCHDOG_CTR_CHECK_RESULT_INIT_BIT_OFFSET ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_CHECK_RESULT_VALID_OFFSET ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_INIT_BIT_SIZE ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_CHECK_RESULT_VALID_SIZE ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_LAST_EVENTS_OFFSET ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_CHECK_RESULT_WINDOW_RELATED_OFFSET ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_LAST_EVENTS_SIZE ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_CHECK_RESULT_WINDOW_RELATED_SIZE ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_MASK ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_CHECK_RESULT_WORD_COUNT ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_OFFSET ezdp_counter_defs.h, 288	ezdp_watchdog_ctr_start_result, 187
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_SIZE ezdp_counter_defs.h, 288	lsb, 187
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_WORD_OFFSET ezdp_counter_defs.h, 288	msb, 187
EZDP_WATCHDOG_CTR_CHECK_RESULT_MAX_THRESHOLD_ALERT_WORD_SELECT ezdp_counter_defs.h, 288	raw_data, 187
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_MASK ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_START_RESULT_LSB_OFFSET ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_OFFSET ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_START_RESULT_LSB_SIZE ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_SIZE ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_START_RESULT_LSB_WORD_OFFSET ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_WORD_OFFSET ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_START_RESULT_LSB_WORD_SELECT ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_MIN_THRESHOLD_ALERT_WORD_SELECT ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_START_RESULT_MSB_OFFSET ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_PROFILE_ID_OFFSET ezdp_counter_defs.h, 289	EZDP_WATCHDOG_CTR_START_RESULT_MSB_SIZE ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_PROFILE_ID_SIZE ezdp_counter_defs.h, 289	EZDP_WATCHDOG_CTR_START_RESULT_MSB_WORD_OFFSET ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_RESET_RVED4_28_OFFSET ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_START_RESULT_MSB_WORD_SELECT ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_RESET_RVED4_28_SIZE ezdp_counter_defs.h, 288	EZDP_WATCHDOG_CTR_START_RESULT_WORD_COUNT ezdp_counter_defs.h, 289
EZDP_WATCHDOG_CTR_CHECK_RESULT_RESET_RVED62_OFFSET ezdp_counter_defs.h, 289	ezdp_watchdog_sliding_window_cfg, 188
EZDP_WATCHDOG_CTR_CHECK_RESULT_RESET_RVED62_SIZE ezdp_counter_defs.h, 289	__pad0__, 188
	__pad1__, 188
	__pad2__, 188
	__pad3__, 189
	__pad4__, 189
	__pad5__, 189
	__pad6__, 189
	counters, 189
	profile_id, 189
	raw_data, 188
	valid, 189
	valid_windows, 189

EZDP_WATCHDOG_SLIDING_WINDOW_CFG_ALERT_OFFSET ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED63_OFFSET ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_ALERT_SIZE ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED63_SIZE ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_OFFSET ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_MASK ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_SIZE ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_OFFSET ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_WORD_OFFSET ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_SIZE ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_COUNTERS_WORD_SELECT ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_OFFSET ezdp_counter_defs.h, 286
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MAX_THRESHOLD_ALERT_OFFSET ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_SIZE ezdp_counter_defs.h, 286
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MAX_THRESHOLD_ALERT_SIZE ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_WORD_OFFSET ezdp_counter_defs.h, 286
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MIN_THRESHOLD_ALERT_OFFSET ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WINDOWS_WORD_SELECT ezdp_counter_defs.h, 286
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_MIN_THRESHOLD_ALERT_SIZE ezdp_counter_defs.h, 286	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WORD_OFFSET ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PARITY_OFFSET ezdp_counter_defs.h, 287	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_VALID_WORD_SELECT ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PARITY_SIZE ezdp_counter_defs.h, 287	EZDP_WATCHDOG_SLIDING_WINDOW_CFG_WORD_COUNT ezdp_counter_defs.h, 287
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PRFILE_ID_OFFSET ezdp_counter_defs.h, 287	ezdp_wmb ezdp_processor.h, 500
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PRFILE_ID_SIZE ezdp_counter_defs.h, 287	ezdp_write_bits_bitwise_ctr ezdp_counter.h, 243
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PRFILE_ID_WORD_OFFSET ezdp_counter_defs.h, 287	ezdp_write_bits_bitwise_ctr_async ezdp_counter.h, 243
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_PRFILE_ID_WORD_SELECT ezdp_counter_defs.h, 287	ezdp_write_bitwise_ctr_cfg ezdp_counter.h, 242
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED5_28_OFFSET ezdp_counter_defs.h, 286	ezdp_write_bitwise_ctr_cfg_async ezdp_counter.h, 242
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED5_28_SIZE ezdp_counter_defs.h, 286	ezdp_write_dual_ctr_cfg ezdp_counter.h, 238
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED56_OFFSET ezdp_counter_defs.h, 287	ezdp_write_dual_ctr_cfg_async ezdp_counter.h, 238
EZDP_WATCHDOG_SLIDING_WINDOW_CFG_RESERVED56_SIZE ezdp_counter_defs.h, 286	ezdp_write_hier_tb_ctr_cfg ezdp_counter.h, 254
	ezdp_write_hier_tb_ctr_cfg_async ezdp_counter.h, 254
	ezdp_write_mc_buf_counter ezdp_frame.h, 373
	ezdp_write_mc_buf_counter_async ezdp_frame.h, 373
	ezdp_write_posted_ctr ezdp_counter.h, 260

ezdp_write_posted_ctr_async	ezdp_job_desc, 100
ezdp_counter.h, 260	frame_length
ezdp_write_security_context	ezdp_frame_desc, 83
ezdp_security.h, 575	free_bytes
ezdp_write_security_context_async	ezdp_2step_1588_header, 12
ezdp_security.h, 575	ezdp_buffer_info, 16
ezdp_write_security_initial_vector	ezdp_frame_desc, 83
ezdp_security.h, 574	global_addresses
ezdp_write_security_initial_vector_async	ezdp_decode_ipv6_result, 41
ezdp_security.h, 574	ezdp_decode_ipv6_retval, 43
ezdp_write_security_key	gre
ezdp_security.h, 571	ezdp_decode_ip_next_protocol, 24
ezdp_write_security_key_async	ezdp_decode_ip_protocol_retval, 27
ezdp_security.h, 571	gross_checksum
ezdp_write_security_mac	ezdp_job_rx_info, 105
ezdp_security.h, 572	gross_checksum_flag
ezdp_write_security_mac_async	ezdp_frame_desc, 82
ezdp_security.h, 572	head
ezdp_write_security_state	ezdp_list_cfg, 122
ezdp_security.h, 569	header_length_gt_frame_length
ezdp_write_security_state_async	ezdp_decode_ipv4_errors, 32
ezdp_security.h, 570	header_length_lt_5
ezdp_write_single_ctr	ezdp_decode_ipv4_errors, 32
ezdp_counter.h, 233	header_offset
ezdp_write_single_ctr_async	ezdp_2step_1588_header, 12
ezdp_counter.h, 234	ezdp_frame_desc, 83
ezdp_write_single_ctr_cfg	icmp
ezdp_counter.h, 232	ezdp_decode_ipv4_control, 30
ezdp_write_single_ctr_cfg_async	icmp_igmp
ezdp_counter.h, 233	ezdp_decode_ip_next_protocol, 24
ezdp_write_tb_ctr_cfg	ezdp_decode_ip_protocol_retval, 27
ezdp_counter.h, 249	icu_succ_parsing_flag
ezdp_write_tb_ctr_cfg_async	ezdp_job_rx_interface_info, 108
ezdp_counter.h, 249	id
ezdp_write_watchdog_ctr_cfg	ezdp_buffer_desc, 15
ezdp_counter.h, 256	igmp
ezdp_write_watchdog_ctr_cfg_async	ezdp_decode_ipv4_control, 29
ezdp_counter.h, 257	imem_1_cluster_code_size
ezdp_xchg_bits_bitwise_ctr	ezdp_mem_section_info, 153
ezdp_counter.h, 243	imem_1_cluster_data_size
ezdp_xchg_single_ctr	ezdp_mem_section_info, 153
ezdp_counter.h, 234	imem_16_cluster_code_size
ezdp_xor	ezdp_mem_section_info, 154
ezdp_math.h, 445	imem_16_cluster_data_size
EZDP_XXX_MAC_IV_SIZE	ezdp_mem_section_info, 153
ezdp_security_defs.h, 582	imem_2_cluster_code_size
EZDP_YELLOW_TRAFFIC	ezdp_mem_section_info, 153
ezdp_counter_defs.h, 293	imem_2_cluster_data_size
fail	ezdp_mem_section_info, 153
ezdp_hier_tb_result, 88	imem_4_cluster_code_size
first_fragment	ezdp_mem_section_info, 154
ezdp_decode_ipv4_result, 33	imem_4_cluster_data_size
ezdp_decode_ipv4_retval, 35	ezdp_mem_section_info, 153
flags	imem_all_cluster_code_size
ezdp_driver_desc, 69	ezdp_mem_section_info, 154
flow_id	imem_all_cluster_data_size
ezdp_job_tx_info, 116	ezdp_mem_section_info, 153
flush	imem_buf_count
ezdp_posted_ctr_msg, 165	ezdp_job_rx_interface_info, 107
frame_desc	imem_buf_guarantee

ezdp\_congestion\_status, 18  
 imem\_half\_cluster\_code\_size  
 ezdp\_mem\_section\_info, 153  
 imem\_half\_cluster\_data\_size  
 ezdp\_mem\_section\_info, 152  
 imem\_private\_data\_size  
 ezdp\_mem\_section\_info, 152  
 index  
 ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 132  
 ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 134  
 ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element, 136  
 ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_element, 138  
 ezdp\_lookup\_ext\_tcam\_index\_result\_element, 140  
 ezdp\_lookup\_int\_tcam\_standard\_result, 149  
 index\_pool\_id  
 ezdp\_mem\_pool\_config, 151  
 info  
 ezdp\_job\_container\_desc, 98  
 ezdp\_lookup\_retval, 150  
 inject\_checksum\_flag  
 ezdp\_1step\_1588\_header, 10  
 inter\_packet\_gap  
 ezdp\_job\_tx\_info, 118  
 inter\_packet\_gap\_control  
 ezdp\_job\_tx\_info, 118  
 interface\_info  
 ezdp\_job\_rx\_info, 104  
 internetwork\_multicast\_range  
 ezdp\_decode\_ipv4\_control, 30  
 ezdp\_decode\_ipv6\_control, 38  
 ip\_version\_mismatch\_in\_pppoe  
 ezdp\_decode\_mac\_errors, 48  
 ipv4  
 ezdp\_decode\_eth\_type\_retval, 22  
 ezdp\_decode\_ip\_next\_protocol, 24  
 ezdp\_decode\_ip\_protocol\_retval, 27  
 ezdp\_decode\_mac\_protocol\_type, 52  
 ipv4\_in\_pppoe  
 ezdp\_decode\_mac\_result, 54  
 ezdp\_decode\_mac\_retval, 55  
 ipv4\_multicast  
 ezdp\_decode\_mac\_control, 46  
 ipv6  
 ezdp\_decode\_eth\_type\_retval, 22  
 ezdp\_decode\_ip\_next\_protocol, 24  
 ezdp\_decode\_ip\_protocol\_retval, 27  
 ezdp\_decode\_mac\_protocol\_type, 51  
 ipv6\_in\_pppoe  
 ezdp\_decode\_mac\_result, 53  
 ezdp\_decode\_mac\_retval, 55  
 ipv6\_multicast  
 ezdp\_decode\_mac\_control, 46  
 is\_service\_ready  
 ezdp\_job\_rx\_info, 105  
 job\_budget\_id  
 ezdp\_frame\_desc, 84  
 ezdp\_job\_container\_desc, 98  
 job\_commands  
 ezdp\_job\_container\_desc, 98  
 job\_guarantee  
 ezdp\_congestion\_status, 17  
 job\_id  
 ezdp\_job\_container\_cmd\_desc, 96  
 key\_shuffle\_bits  
 ezdp\_sum\_addr\_table\_desc, 173  
 key\_shuffle\_en  
 ezdp\_sum\_addr\_table\_desc, 173  
 key\_size  
 ezdp\_sum\_addr\_table\_desc, 173  
 label1\_ttl\_is\_one  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 label1\_ttl\_is\_zero  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 label2\_ttl\_is\_one  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 label2\_ttl\_is\_zero  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 label3\_ttl\_is\_one  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 label3\_ttl\_is\_zero  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 label4\_ttl\_is\_one  
 ezdp\_decode\_mpls\_result, 61  
 ezdp\_decode\_mpls\_retval, 64  
 label4\_ttl\_is\_zero  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 last\_entry\_in\_stack  
 ezdp\_decode\_mpls\_result, 62  
 ezdp\_decode\_mpls\_retval, 65  
 last\_events  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 last\_tag\_protocol\_type  
 ezdp\_decode\_mac\_result, 54  
 layer2\_size  
 ezdp\_decode\_mac\_result, 54  
 len  
 ezdp\_driver\_desc, 69  
 length  
 ezdp\_decode\_eth\_type\_retval, 22  
 ezdp\_decode\_mac\_protocol\_type, 51  
 likely  
 ezdp\_defs.h, 346  
 line  
 ezdp\_ext\_linked\_buffers\_desc, 79  
 ezdp\_large\_linked\_buffers\_desc, 119  
 ezdp\_linked\_buffers\_desc, 120  
 ezdp\_small\_linked\_buffers\_desc, 171  
 link\_local\_address  
 ezdp\_decode\_ipv6\_result, 42

ezdp\_decode\_ipv6\_retval, 43  
 link\_local\_multicast\_range  
 ezdp\_decode\_ipv4\_control, 30  
 ezdp\_decode\_ipv6\_control, 38  
 logical\_id  
 ezdp\_frame\_desc, 83  
 lookup\_error  
 ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element, 123  
 ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element, 125  
 ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element, 127  
 ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element, 129  
 ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 132  
 ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 134  
 ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element, 136  
 ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_element, 138  
 ezdp\_lookup\_ext\_tcam\_index\_result\_element, 139  
 ezdp\_lookup\_ext\_tcam\_retval, 141  
 loopback\_info  
 ezdp\_job\_rx\_info, 104  
 lsb  
 ezdp\_watchdog\_ctr\_start\_result, 187  
 mac\_control\_lsb\_0x  
 ezdp\_decode\_mac\_control, 46  
 mac\_control\_lsb\_1x  
 ezdp\_decode\_mac\_control, 46  
 mac\_control\_lsb\_2x  
 ezdp\_decode\_mac\_control, 46  
 mac\_control\_other  
 ezdp\_decode\_mac\_control, 46  
 mac\_error  
 ezdp\_lookup\_ext\_tcam\_retval, 142  
 major\_patch\_version  
 ezdp\_version, 180  
 major\_version  
 ezdp\_version, 179  
 match  
 ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element, 123  
 ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element, 125  
 ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element, 127  
 ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element, 129  
 ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 131  
 ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 133  
 ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element, 135  
 ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_element, 137  
 ezdp\_lookup\_ext\_tcam\_index\_result\_element, 139  
 ezdp\_lookup\_ext\_tcam\_retval, 141  
 max\_threshold\_alert  
 ezdp\_watchdog\_ctr\_check\_result, 185  
 mem\_error  
 ezdp\_lookup\_retval, 150  
 micro\_patch\_version  
 ezdp\_version, 180  
 min\_threshold\_alert  
 ezdp\_watchdog\_ctr\_check\_result, 185  
 minor\_patch\_version  
 ezdp\_version, 180  
 minor\_version  
 ezdp\_version, 179  
 module\_name  
 ezdp\_version, 179  
 mpls  
 ezdp\_decode\_ip\_next\_protocol, 25  
 ezdp\_decode\_ip\_protocol\_retval, 27  
 mpls\_multicast  
 ezdp\_decode\_eth\_type\_retval, 22  
 ezdp\_decode\_mac\_protocol\_type, 51  
 mpls\_unicast  
 ezdp\_decode\_eth\_type\_retval, 22  
 ezdp\_decode\_mac\_protocol\_type, 51  
 msb  
 ezdp\_watchdog\_ctr\_start\_result, 187  
 msid  
 ezdp\_ext\_addr, 77  
 ezdp\_pci\_addr, 156  
 ezdp\_sum\_addr, 172  
 ezdp\_sum\_addr\_table\_desc, 174  
 msix\_payload  
 ezdp\_pci\_msg, 159  
 multi\_match  
 ezdp\_lookup\_ext\_tcam\_retval, 142  
 my\_mac  
 ezdp\_decode\_mac\_control, 47  
 next\_protocol  
 ezdp\_decode\_ipv4\_result, 34  
 ezdp\_decode\_ipv6\_result, 42  
 no\_context\_match\_error  
 ezdp\_lookup\_ext\_tcam\_retval, 142  
 not\_ipv4\_version  
 ezdp\_decode\_ipv4\_errors, 32  
 not\_ipv6\_version  
 ezdp\_decode\_ipv6\_errors, 40  
 nsec  
 ezdp\_rtc, 167  
 number\_of\_tags  
 ezdp\_decode\_mac\_result, 53  
 ezdp\_decode\_mac\_retval, 55  
 obj\_size  
 ezdp\_mem\_pool\_config, 151  
 one\_step

ezdp_1588_header, 8	ezdp_list_cfg, 122
option_exist	raw_data
ezdp_decode_ipv4_result, 34	ezdp_1step_1588_header, 9
ezdp_decode_ipv4_retval, 36	ezdp_2step_1588_header, 11
options_exist	ezdp_app_schlr_status, 13
ezdp_decode_ipv6_result, 42	ezdp_bitwise_ctr_cfg, 14
ezdp_decode_ipv6_retval, 44	ezdp_buffer_desc, 15
ezdp_decode_tcp_retval, 68	ezdp_congestion_status, 17
original_value1	ezdp_ctr_msg, 19
ezdp_dual_add32_result, 71	ezdp_decode_eth_type_retval, 21
ezdp_dual_add64_result, 72	ezdp_decode_ip_next_protocol, 24
original_value2	ezdp_decode_ip_protocol_retval, 26
ezdp_dual_add32_result, 71	ezdp_decode_ipv4_control, 29
ezdp_dual_add64_result, 72	ezdp_decode_ipv4_errors, 31
other	ezdp_decode_ipv4_result, 33
ezdp_decode_eth_type_retval, 21	ezdp_decode_ipv4_retval, 35
ezdp_decode_ip_next_protocol, 24	ezdp_decode_ipv6_control, 37
ezdp_decode_ip_protocol_retval, 26	ezdp_decode_ipv6_errors, 39
output_channel	ezdp_decode_ipv6_result, 41
ezdp_job_transmit_cmd_info, 113	ezdp_decode_ipv6_retval, 43
outstanding_job	ezdp_decode_mac_control, 45
ezdp_input_queue_status, 95	ezdp_decode_mac_errors, 48
overflow	ezdp_decode_mac_protocol_type, 50
ezdp_ctr_msg, 19	ezdp_decode_mac_result, 53
overrun_error_condition	ezdp_decode_mac_retval, 55
ezdp_ctr_msg, 19	ezdp_decode_mpls_label_result, 57
ezdp_posted_ctr_msg, 164	ezdp_decode_mpls_label_retval, 59
owner	ezdp_decode_mpls_result, 61
ezdp_driver_desc_flags, 70	ezdp_decode_mpls_retval, 64
packet_switch_id_select	ezdp_decode_tcp_errors, 67
ezdp_job_tx_info, 115	ezdp_decode_tcp_retval, 68
payload_gt_frame_length	ezdp_driver_desc, 69
ezdp_decode_ipv6_errors, 40	ezdp_driver_desc_flags, 70
payload_missing	ezdp_dual_add32_result, 71
ezdp_decode_ipv6_errors, 39	ezdp_dual_add64_result, 72
phy_func	ezdp_dual_ctr, 73
ezdp_pci_addr, 156	ezdp_dual_ctr_cfg, 74
ezdp_pci_msg_ctrl, 160	ezdp_dual_ctr_result, 76
phys_func	ezdp_ext_addr, 77
ezdp_pci_info, 158	ezdp_flow_control_status, 80
pppoe_discovery	ezdp_frame_desc, 81
ezdp_decode_eth_type_retval, 22	ezdp_group_schlr_status, 85
ezdp_decode_mac_protocol_type, 51	ezdp_hier_tb_ctr_cfg, 86
pppoe_session	ezdp_hier_tb_result, 88
ezdp_decode_eth_type_retval, 22	ezdp_hier_tb_ug_app_bits, 90
ezdp_decode_mac_protocol_type, 51	ezdp_hier_tb_update, 92
private_cmem_size	ezdp_input_queue_status, 94
ezdp_mem_section_info, 152	ezdp_job_container_cmd_desc, 96
profile_id	ezdp_job_container_desc, 98
ezdp_watchdog_accumulative_window_cfg, 182	ezdp_job_discard_cmd_info, 101
ezdp_watchdog_sliding_window_cfg, 189	ezdp_job_queue_cmd_info, 102
project_name	ezdp_job_rx_confirmation_info, 103
ezdp_version, 179	ezdp_job_rx_info, 104
qos_bypass	ezdp_job_rx_interface_info, 107
ezdp_job_tx_info, 116	ezdp_job_rx_loopback_info, 110
queue	ezdp_job_rx_timer_info, 111
ezdp_pci_info, 158	ezdp_job_rx_user_info, 112
queue_info	ezdp_job_transmit_cmd_info, 113
ezdp_job_container_cmd_desc, 96	ezdp_job_tx_info, 115
queue_memory_pool	



ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element, 123  
 ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element, 125  
 ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element, 127  
 ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element, 129  
 ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 131  
 ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 133  
 ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element, 135  
 ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_element, 137  
 ezdp\_lookup\_ext\_tcam\_index\_result\_element, 139  
 ezdp\_lookup\_ext\_tcam\_retval, 141  
 ezdp\_lookup\_int\_tcam\_12B\_data\_result, 143  
 ezdp\_lookup\_int\_tcam\_16B\_data\_result, 144  
 ezdp\_lookup\_int\_tcam\_4B\_data\_result, 145  
 ezdp\_lookup\_int\_tcam\_8B\_data\_result, 146  
 ezdp\_lookup\_int\_tcam\_retval, 148  
 ezdp\_lookup\_int\_tcam\_standard\_result, 149  
 ezdp\_lookup\_retval, 150  
 ezdp\_output\_queue\_status, 155  
 ezdp\_pci\_addr, 156  
 ezdp\_pci\_info, 158  
 ezdp\_pci\_msg, 159  
 ezdp\_pci\_msg\_ctrl, 160  
 ezdp\_pci\_msg\_payload\_atl, 161  
 ezdp\_pci\_msg\_payload\_elbi, 162  
 ezdp\_pci\_msg\_payload\_msix, 163  
 ezdp\_posted\_ctr\_msg, 164  
 ezdp\_rtc, 167  
 ezdp\_security\_handle, 168  
 ezdp\_single\_ctr\_cfg, 169  
 ezdp\_sum\_addr, 172  
 ezdp\_sum\_addr\_table\_desc, 173  
 ezdp\_tb\_ctr\_cfg, 175  
 ezdp\_tb\_ctr\_result, 177  
 ezdp\_watchdog\_accumulative\_window\_cfg, 181  
 ezdp\_watchdog\_ctr\_cfg, 183  
 ezdp\_watchdog\_ctr\_check\_result, 185  
 ezdp\_watchdog\_ctr\_start\_result, 187  
 ezdp\_watchdog\_sliding\_window\_cfg, 188  
 ready  
   ezdp\_input\_queue\_status, 94  
   ezdp\_output\_queue\_status, 155  
 replication\_count  
   ezdp\_job\_tx\_info, 115  
 replication\_id  
   ezdp\_job\_rx\_loopback\_info, 110  
 report\_exceeded  
   ezdp\_single\_ctr\_cfg, 169  
 reserved\_label  
   ezdp\_decode\_mpls\_label\_result, 58  
   ezdp\_decode\_mpls\_label\_retval, 60  
 rx\_info  
   ezdp\_job\_desc, 100  
 sec  
   ezdp\_rtc, 167  
 seq\_number  
   ezdp\_job\_rx\_info, 105  
 seq\_number\_valid  
   ezdp\_job\_rx\_info, 105  
 set\_active\_state  
   ezdp\_hier\_tb\_update, 92  
 set\_app\_bits  
   ezdp\_hier\_tb\_update, 92  
 shared\_cmem\_size  
   ezdp\_mem\_section\_info, 152  
 side  
   ezdp\_job\_discard\_cmd\_info, 101  
   ezdp\_job\_queue\_cmd\_info, 102  
   ezdp\_job\_rx\_info, 106  
   ezdp\_job\_transmit\_cmd\_info, 113  
   ezdp\_job\_tx\_info, 116  
 single\_ctr\_value  
   ezdp\_ctr\_msg, 20  
 sip\_dip\_hash  
   ezdp\_decode\_ipv4\_result, 34  
   ezdp\_decode\_ipv6\_result, 42  
 sip\_equal\_dip  
   ezdp\_decode\_ipv4\_errors, 31  
   ezdp\_decode\_ipv6\_errors, 40  
 sip\_is\_multicast  
   ezdp\_decode\_ipv4\_errors, 32  
   ezdp\_decode\_ipv6\_errors, 39  
 sip\_is\_one  
   ezdp\_decode\_ipv6\_errors, 40  
 sip\_is\_zero  
   ezdp\_decode\_ipv4\_errors, 32  
   ezdp\_decode\_ipv6\_errors, 40  
 site\_local\_address  
   ezdp\_decode\_ipv6\_result, 41  
   ezdp\_decode\_ipv6\_retval, 43  
 size  
   ezdp\_input\_queue\_status, 94  
   ezdp\_output\_queue\_status, 155  
   ezdp\_ring\_cfg, 166  
 sliding\_window  
   ezdp\_watchdog\_ctr\_cfg, 183  
 smac\_equals\_dmac  
   ezdp\_decode\_mac\_control, 46  
 smac\_is\_not\_unicast  
   ezdp\_decode\_mac\_errors, 49  
 smac\_is\_zero  
   ezdp\_decode\_mac\_errors, 48  
 solicited\_node\_multicast\_range  
   ezdp\_decode\_ipv6\_control, 38  
 source\_queue  
   ezdp\_job\_rx\_info, 106  
 standard  
   ezdp\_lookup\_int\_tcam\_result, 147  
   ezdp\_lookup\_int\_tcam\_retval, 148  
 stat\_code\_profile1  
   ezdp\_job\_tx\_info, 116  
 stat\_code\_profile2  
   ezdp\_job\_tx\_info, 116

stat\_stream\_id  
     ezdp\_job\_tx\_info, 117  
 stop\_bit  
     ezdp\_decode\_mpls\_label\_result, 57  
     ezdp\_decode\_mpls\_label\_retval, 59  
 sub\_type  
     ezdp\_driver\_desc, 69  
 success  
     ezdp\_lookup\_retval, 150  
 sum\_addr  
     ezdp\_ctr\_msg, 20  
     ezdp\_posted\_ctr\_msg, 165  
 syn\_and\_fin\_eq\_1  
     ezdp\_decode\_tcp\_errors, 67  
 tag1\_protocol\_type  
     ezdp\_decode\_mac\_result, 54  
 tag2\_protocol\_type  
     ezdp\_decode\_mac\_result, 54  
 tag3\_protocol\_type  
     ezdp\_decode\_mac\_result, 54  
 tail  
     ezdp\_list\_cfg, 122  
 target\_queue  
     ezdp\_job\_queue\_cmd\_info, 102  
 tcp  
     ezdp\_decode\_ip\_next\_protocol, 25  
     ezdp\_decode\_ip\_protocol\_retval, 27  
 time\_out\_error  
     ezdp\_lookup\_ext\_tcam\_retval, 142  
 timer\_id  
     ezdp\_job\_rx\_timer\_info, 111  
 timer\_info  
     ezdp\_job\_rx\_info, 105  
 timestamp\_flag  
     ezdp\_frame\_desc, 82  
 timestamp\_nsec  
     ezdp\_job\_rx\_confirmation\_info, 103  
     ezdp\_job\_rx\_interface\_info, 109  
 timestamp\_sec  
     ezdp\_job\_rx\_confirmation\_info, 103  
     ezdp\_job\_rx\_interface\_info, 108  
 timestamp\_threshold  
     ezdp\_hier\_tb\_ctr\_cfg, 87  
 total  
     ezdp\_driver\_desc, 69  
 total\_length\_gt\_frame\_length  
     ezdp\_decode\_ipv4\_errors, 32  
 transmit\_confirmation\_flag  
     ezdp\_frame\_desc, 82  
 transmit\_info  
     ezdp\_job\_container\_cmd\_desc, 96  
 transmit\_keep\_buf\_flag  
     ezdp\_frame\_desc, 82  
 truncated  
     ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element, 124  
     ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element, 126  
     ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element, 128  
     ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element, 130  
     ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_element, 132  
     ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_element, 134  
     ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_element, 136  
     ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_element, 138  
     ezdp\_lookup\_ext\_tcam\_index\_result\_element, 140  
     ezdp\_lookup\_ext\_tcam\_retval, 141  
 truncation\_flag  
     ezdp\_job\_rx\_interface\_info, 108  
 ttl\_is\_one  
     ezdp\_decode\_mpls\_label\_result, 58  
     ezdp\_decode\_mpls\_label\_retval, 60  
 ttl\_is\_zero  
     ezdp\_decode\_mpls\_label\_result, 58  
     ezdp\_decode\_mpls\_label\_retval, 60  
 two\_step  
     ezdp\_1588\_header, 8  
 tx\_info  
     ezdp\_job\_desc, 100  
 type  
     ezdp\_driver\_desc\_flags, 70  
 u  
     ezdp\_1588\_header, 8  
     ezdp\_job\_container\_cmd\_desc, 97  
 udp  
     ezdp\_decode\_ip\_next\_protocol, 25  
     ezdp\_decode\_ip\_protocol\_retval, 27  
 unlikely  
     ezdp\_defs.h, 346  
 update\_task  
     ezdp\_hier\_tb\_result, 88  
 user\_config\_sip  
     ezdp\_decode\_ipv4\_result, 34  
     ezdp\_decode\_ipv4\_retval, 35  
 user\_config\_vlan0  
     ezdp\_decode\_mac\_protocol\_type, 52  
 user\_config\_vlan1  
     ezdp\_decode\_mac\_protocol\_type, 52  
 user\_config\_vlan2  
     ezdp\_decode\_mac\_protocol\_type, 51  
 user\_config0  
     ezdp\_decode\_ipv4\_control, 29  
     ezdp\_decode\_mac\_control, 46  
     ezdp\_decode\_mac\_protocol\_type, 51  
     ezdp\_decode\_mpls\_label\_result, 58  
     ezdp\_decode\_mpls\_label\_retval, 60  
 user\_config1  
     ezdp\_decode\_ipv4\_control, 29  
     ezdp\_decode\_mac\_control, 46  
     ezdp\_decode\_mac\_protocol\_type, 51  
     ezdp\_decode\_mpls\_label\_result, 58  
     ezdp\_decode\_mpls\_label\_retval, 60  
 user\_config2  
     ezdp\_decode\_ipv4\_control, 29  
     ezdp\_decode\_mac\_control, 46

ezdp\_decode\_mac\_protocol\_type, 51  
 ezdp\_decode\_mpls\_label\_result, 58  
 ezdp\_decode\_mpls\_label\_retval, 60  
 user\_config3  
   ezdp\_decode\_mac\_control, 46  
   ezdp\_decode\_mac\_protocol\_type, 51  
   ezdp\_decode\_mpls\_label\_result, 58  
   ezdp\_decode\_mpls\_label\_retval, 60  
 user\_data\_info0  
   ezdp\_job\_rx\_user\_info, 112  
 user\_data\_info1  
   ezdp\_job\_rx\_user\_info, 112  
 user\_def0  
   ezdp\_decode\_eth\_type\_retval, 22  
 user\_def1  
   ezdp\_decode\_eth\_type\_retval, 22  
 user\_info  
   ezdp\_job\_rx\_info, 105  
   ezdp\_job\_tx\_info, 115  
 valid  
   ezdp\_lookup\_ext\_tcam\_16B\_data\_result\_element,  
     123  
   ezdp\_lookup\_ext\_tcam\_32B\_data\_result\_element,  
     125  
   ezdp\_lookup\_ext\_tcam\_4B\_data\_result\_element,  
     127  
   ezdp\_lookup\_ext\_tcam\_8B\_data\_result\_element,  
     129  
   ezdp\_lookup\_ext\_tcam\_index\_16B\_data\_result\_ele  
     ment, 131  
   ezdp\_lookup\_ext\_tcam\_index\_32B\_data\_result\_ele  
     ment, 133  
   ezdp\_lookup\_ext\_tcam\_index\_4B\_data\_result\_eleme  
     nt, 135  
   ezdp\_lookup\_ext\_tcam\_index\_8B\_data\_result\_eleme  
     nt, 137  
   ezdp\_lookup\_ext\_tcam\_index\_result\_element, 139  
 ezdp\_watchdog\_accumulative\_window\_cfg, 182  
 ezdp\_watchdog\_sliding\_window\_cfg, 189  
 valid\_data\_buf  
   ezdp\_buffer\_desc, 15  
 valid\_windows  
   ezdp\_watchdog\_sliding\_window\_cfg, 189  
 value  
   ezdp\_dual\_ctr\_cfg, 75  
   ezdp\_posted\_ctr\_msg, 165  
   ezdp\_single\_ctr\_cfg, 170  
 vector\_index  
   ezdp\_pci\_msg\_payload\_msix, 163  
 version\_char  
   ezdp\_version, 179  
 version\_string  
   ezdp\_version, 179  
 virt\_func  
   ezdp\_pci\_addr, 156  
   ezdp\_pci\_info, 158  
   ezdp\_pci\_msg\_ctrl, 160  
 virt\_func\_en  
   ezdp\_pci\_addr, 156  
   ezdp\_pci\_info, 158  
   ezdp\_pci\_msg\_ctrl, 160  
 vrrp\_mac  
   ezdp\_decode\_mac\_control, 46  
 wrap\_around\_condition  
   ezdp\_1step\_1588\_header, 9  
 wred\_class\_scale\_profile  
   ezdp\_job\_tx\_info, 117  
 wred\_class\_template\_profile  
   ezdp\_job\_tx\_info, 117  
 wred\_color  
   ezdp\_job\_tx\_info, 115  
 wred\_flow\_scale\_profile  
   ezdp\_job\_tx\_info, 117  
 wred\_flow\_template\_profile  
   ezdp\_job\_tx\_info, 117