9725

Acceleration Processor

Data Sheet

Licensing and Government Use

Any Exar software ("Licensed Programs") based on Hifn Technology described in this document is furnished under a license and may be used and copied only in accordance with the terms of such license and with the inclusion of this copyright notice. Distribution of this document or any copies thereof and the ability to transfer title or ownership of this document's contents are subject to the terms of such license.

Such Licensed Programs and their documentation may contain public open-source software that would be licensed under open-source licenses. Refer to the applicable product release notes for open-source licenses and proprietary notices. Use, duplication, disclosure, and acquisition by the U.S. Government of such Licensed Programs is subject to the terms and definitions of their applicable license.

Disclaimer

Exar reserves the right to make changes to its products, including the contents of this document, or to discontinue any product or service without notice. Exar advises its customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied upon is current. Every effort has been made to keep the information in this document current and accurate as of the date of this document's publication or revision.

Limited Warranty

Exar warrants Products based on the Hifn Technology, including cards, against defects in materials and workmanship for a period of twelve (12) months from the delivery date. Exar's sole liability shall be limited to either, replacing, repairing or issuing credit, at its option, for the Product if it has been paid for. Exar will not be liable under this provision unless: (a) Exar is promptly notified in writing upon discovery of claimed defects by Buyer; (b) The claimed defective Product is returned to Exar, insurance and transportation charges prepaid, by Buyer; (c) The claimed defective Product is received within twelve (12) months from the delivery date; and (d) Exar's examination of the Product discloses to its satisfaction that the alleged defect was not caused by misuse, neglect, improper installation, repair, alteration, accident or other hazard. THIS WARRANTY DOES NOT COVER PRODUCT DAMAGE WHICH RESULTS FROM ACCIDENT, MISUSE, ABUSE, IMPROPER LINE VOLTAGE, FIRE, FLOOD, LIGHTNING OR OTHER ACTS OF GOD OR DAMAGE RESULTING FROM ANY MODIFICATIONS, REPAIRS OR ALTERATIONS PERFORMED OTHER THAN BY EXAR OR EXAR'S AUTHORIZED AGENT OR RESULTING FROM FAILURE TO STRICTLY COMPLY WITH EXAR'S WRITTEN OPERATING AND MAINTENANCE INSTRUCTIONS. BUYER ACKNOWLEDGES THAT THE PRODUCT ARE HIGHLY SENSITIVE ELECTRONIC PRODUCT REQUIRING SPECIAL HANDLING AND THAT THIS WARRANTY DOES NOT APPLY TO IMPROPERLY HANDLED PRODUCT. PRODUCT MANUFACTURED TO MEET BUYER'S SPECIFIC PERFORMANCE SPECIFICATIONS ACCEPTED BY EXAR ARE WARRANTED ONLY TO PERFORM IN CONFORMITY WITH SUCH SPECIFICATIONS, AND ARE WARRANTED ONLY AGAINST DEFECTS NOT RELATED TO SUCH SPECIFICATIONS IN ACCORDANCE WITH THE TERMS AND CONDITIONS SET FORTH HEREIN ABOVE.

Life Support Policy

Exar's Product are not authorized for use as critical components in life support devices or systems. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury or death to human life. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness. Buyer agrees to indemnify, defend and hold Exar harmless for any cost, loss, liability, or expense (including without limitation attorneys' fees and other costs of litigation or threatened litigation) arising out of violation of the above prohibition by Buyer or any person or entity receiving Exar's Product through Buyer.

Patent Infringement - Indemnification

Exar agrees, at its own expense, to defend Buyer from and against any claim, suit or proceeding, and to pay all judgments and costs finally awarded against Buyer by reason of claim, suit or proceeding insofar as it is based upon an allegation that the Product as furnished by Exar infringes any United States letter patent, provided that Exar is notified promptly of such claim in writing and is given authority and full and proper information and assistance (at Exar's expense) for defense of same. In case such Product are finally constituted an infringement and the use of Product is enjoined, Exar shall at its sole discretion and at its own expense: (1) procure for Buyer the right to continue using the Product; (2) replace or modify the same so that it becomes non-infringing; or (3) remove such Product and grant Buyer a credit for the depreciated value of the same.

Buyer shall have the right to employ separate counsel in any claim, suit or proceeding and to participate in the defense thereof, but the fees and expenses of Buyer's counsel shall not be borne by Exar unless: (1) Exar specifically so agrees; or (2) Exar, after written request and without cause, does not assume such defense. Exar shall not be liable to indemnify Buyer for any settlement effected without Exar's written consent, unless Exar failed, after notice and without cause, to defend such claim, suit or proceeding.

The indemnification shall not apply and Buyer shall indemnify Exar and hold it harmless from all liability or expense (including costs of suit and attorney's fees) if the infringement arises from, or is based upon Exar's

compliance with particular requirements of Buyer or Buyer's customer that differ from Exar's standard specifications (Custom Product) for the Product, or modifications or alterations of the Product, or a combination of the Product with other items not furnished or manufactured by Exar.

Buyer agrees that Exar shall not be liable for any collateral, incidental or consequential damages arising out of patent infringement.

The foregoing states the entire liability of Exar for patent infringement.

Motorola

The use of this product in stateful compression protocols (for example, PPP or multi-history applications) with certain configurations may require a license from Motorola. In such cases, a license agreement for the right to use Motorola patents (US05,245,614, US05,130,993) may be obtained directly from Motorola.

Patents

May include one or more of the following United States patents: 4,930,142; 4,996,690; 4,701,745; 5,003,307; 5,016,009; 5,126,739; 5,146,221; 5,414,425; 5,414,850; 5,463,390; 5,506,580; 5,532,694; 6,320,846; 6,816,459; 6,651,099; 6,665,725; 6,771,646; 6,789,116; 6,954,789; 6,839,751; 7,299,282; 7,260,558. Other patents pending.

Trademarks

Hi/fn®, MeterFlow®, MeterWorks®, and LZS®, are registered trademarks of Exar Corporation. Hifn™, Hifn Technology, FlowThrough™, BitWackr, and the Hifn logo are trademarks of Hi/fn, Inc. All other trademarks and trade names are the property of their respective holders.

IBM, IBM Logo, and IBM PowerPC are trademarks of International Business Machines Corporation in the United States, or other countries.

Microsoft, Windows, Windows XP, Windows Vista, Windows Server 2003, Windows Server 2008 and the Windows logo are trademarks of Microsoft Corporation in the United States, and/or other countries.

Intel QuickAssist is a trademark of Intel Corporation in the United States and in other countries.

Exporting

This product may only be exported from the United States in accordance with applicable Export Administration Regulations. Diversion contrary to United States laws is prohibited.

Exar Confidential

If you have signed a Exar Confidential Disclosure Agreement that includes this document as part of its subject matter, please use this document in accordance with the terms of the agreement. If not, please destroy the document.

Table of Contents

List of Figures

List of Tables

Preface

Welcome to the Data Sheet for the 9725 Acceleration Processor. This document describes the features, operation, interfaces, configuration, and specifications for the 9725 device.

Audience

This document is intended for integrators and application developers responsible for and familiar with software and hardware architecture of a target system.

Prerequisite

Before proceeding, you should generally understand:

- Software and hardware of the target system
- General networking concepts

Document Organization

This document is organized as follows:

Chapter 1, "Product Description" provides an overview of the 9725 processor.

Chapter 2, "Operation" describes key features of the 9725 operations.

Chapter 3, "Data Structures" defines the format of the data structures used by the 9725.

Chapter 4, "Modules" describes the internal 9725 modules in more detail.

Chapter 5, "PCIe Configuration Register Definition" details the syntax and usage of the 9725 PCIe registers.

Chapter 6, "9725 Register Definition" provides the syntax and usage o the internal 9725 configuration registers.

Chapter 7, "Interface Definition" defines the external interfaces for the 9725 device.

Chapter 8, "DC Specifications" defines the 9725 DC specifications.

Chapter 9, "AC Specifications" defines the 9725 AC specifications.

Chapter 10, "Package Information" defines the 9725 package specifications.

Appendix A, "Temperature Monitoring" describes the optional thermal monitoring device.

Related Documents

The following documents can be used as a reference to this document.

Express DR SDK 4.1.2L Getting Started Guide, UG-0197

Express DR SDK User Guide, UG-0198

Express DR SDK Release Notes, RN-0124

Express DR SDK 4.1.2L Performance Application Note, AN-0190

Customer Support

For technical support about this product, please contact your local Exar sales office, representative, or distributor.

For general information about Exar and Exar products refer to: www.exar.com

Glossary

Term	Definition
AAD	Additional Authentication Data
AES	Advanced Encryption Standard
CBC	Cipher Block Chaining
CPR	Command Pointer Ring
CRC	Cyclic Redundancy Check
DES	Data Encryption Standard
DH	Diffie-Hellman key exchange protocol
DIF	Data Integrity Field for AES-XTS
DSA	Digital Signature Algorithm
ECC	Error correction code
ECRC	End-to-End cyclic Redundancy Check
eLZS	Enhanced LZS
GMAC	Galois Message Authentication Code
HMAC	Hash Message Authentication Code
IHV	Initial Hash Value
IPAD	Inner Padding
IPsec	IP Security Protocol
IV	Initial Vector
JTAG	Joint Test Action Group
LZS	Lempel-Ziv Stac
MAC	Message Authentication Code
OPAD	Outer Padding
PHY	Physical-Layer interface - usually for PCI express
PLL	Phase-Locked Loop
RC	PCIe Root Complex
RSA	Ron Rivest, Adi Shamir and Leonard Adleman
S0	Initial Value for Hash Engine GCM-MAC and GAMC calculation
SPI	Serial Peripheral Interface
SSL	Security Socket Layer
TLS	Transport Layer Security
XTS	XES-based Tweaked CodeBook mode with Cipher Text Stealing

1 Product Description

The 9725 Acceleration Processor performs compression, encryption and hashing for multi-sequenced operations in an integrated secure session environment. All data paths within the 9725 Acceleration Processor support full data integrity.

1.1 Features

- Configurable for 4-lane or 8-lane PCI Express 1.1 compliant interface
- Multiple hardware acceleration engines deliver high performance
    - Six Compression/Decompression, Encryption/Decryption, and Hash Engines
    - Compression/Decompression engine with LZS and eLZS algorithms (300MB/s per engine)
    - Encryption/Decryption engine with AES-GCM, AES-CBC and AES-XTS algorithm (AES-GCM and AES-XTS with 300MB/s per engine, AES-CBC with 170MB/s per engine)
    - Hash engine with SHA1, SHA256, MD5 algorithm (SHA1 and SHA256 with 300MB/s per engine, MD5 with 180MB/s per engine)
- Compression/Decompression engine features
    - Industry-standard LZS algorithm
    - Enhanced LZS (eLZS) algorithm with anti-expansion compression
    - Record CRC enhances data integrity
- Encryption/Decryption engine features
    - Supports maximum 96-bit IV for GCM/CBC/XTS mode
    - Supports 128/256-bit key AES GCM/CBC mode
    - Supports 256/512-bit key AES XTS mode
    - Supports AES CBC with or without HMAC-SHA1
    - Supports AES-XTS with multi-sector operations
    - Key storage in on chip SRAM
- Hash engine features
    - Supports slice and stateful hash operations
    - Supports slice or file Hash
- DMA supports 12 operations
    - Compression only
    - Decompression only
    - Pass through only

- Compression + Encryption (compress then encrypt)
- Decryption + Decompression (decrypt then decompress)
- Encryption only
- Decryption only
- Write Key
- Hash only
- Compression + Hash
- Encryption + Hash (except AES-CBC/HMAC-SHA1 + Hash)
- Compression + Encryption + Hash
  (except Compression +AES-CBC/HMAC-SHA1 + Hash and
  Compression + AES-XTS)

- DMA supports command/result ring and data conversion with 4 endian modes
  - No swap
  - Byte swap in word
  - Word swap, no byte swap in word
  - Word swap, and byte swap in word
- DMA supports direct and indirect command addressing modes
  - In direct command addressing mode, the command ring resides in contiguous memory
  - In indirect command addressing mode, the command ring does not need to reside in contiguous memory
- DMA supports a 32-bit or 64-bit result ring
  - 32-bit result ring
  - 64-bit result ring reports more command processing information to the host
- DMA supports scatter-gather operation
  - Supports 512 descriptors in direct command addressing mode
  - Supports unlimited descriptors in indirect command addressing mode
- DMA supports a command ring with maximum of 16K commands
- DMA supports two command rings with Round-Robin software arbitration
- End-to-end data integrity/validation including data path, on chip memory and transform engines
  - Full data path protection with ECC or Parity
  - On-chip/off-chip data integrity validation with CRC
  - All key protection with CRC
  - Real time verification of encryption/decryption operation

- Real time verification of compression/decompression operation
- Real time verification of hash operation
- ECC check for on chip memory

- Internal key SRAM

1.2 Applications

The 9725 is targeted at motherboard or embedded applications that require the same level of performance and functionality as Exar's Express DR 1625 card, such as primary and secondary storage applications for VTL, Back-up, NAS, SAN, DAS and replication. The functionality of the 9725 provides data compression, encryption and hashing for deduplication, thus freeing up valuable CPU resources for applications.

The 9725 software architecture enables higher availability by providing failover protection in case of device failure. In addition, the 9725 has built-in functionality to enable end-to-end data protection.

Figure 1-1 shows a typical application example.

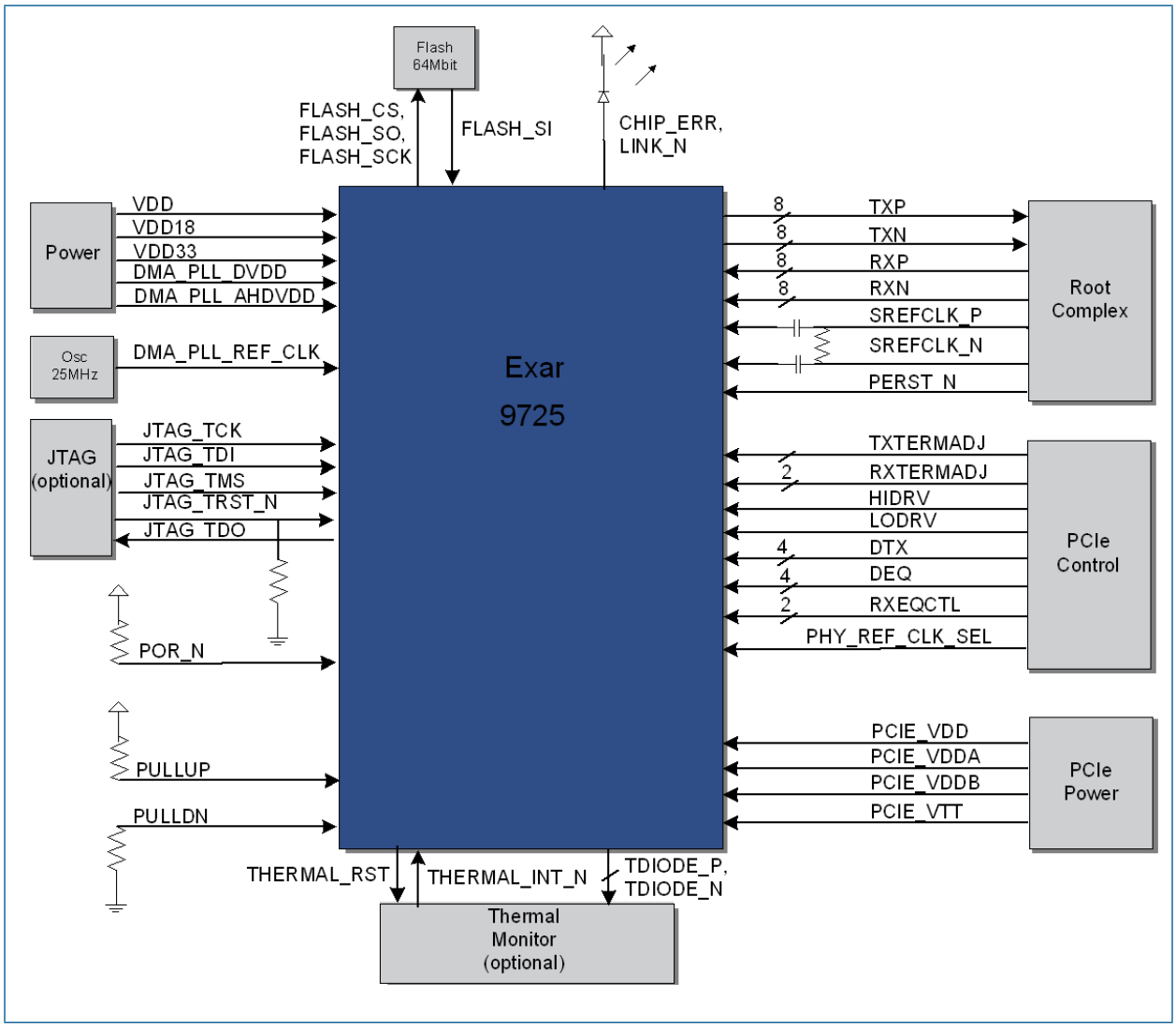


Figure 1-1. Typical 9725 Application

1.3 Ordering Information

Table 1-1. Ordering Information

Part Number	Package	Description
9725HG	396-ball FCBGA	9725 Acceleration Processor

2 Operation

The 9725 performs compression, encryption and hashing for multi-sequenced operations in an integrated secure session environment. All data paths within the 9725 Acceleration Processor support full data validation.

2.1 Functional Description

The 9725 contains six Compression/Encryption/Hash engine modules. Each Compression/ Encryption/Hash engine module is dedicated to a corresponding Engine Manager module, providing powerful hardware parallel processing. Figure 2-1 illustrates a functional level block diagram for Exar's 9725 Acceleration Processor.

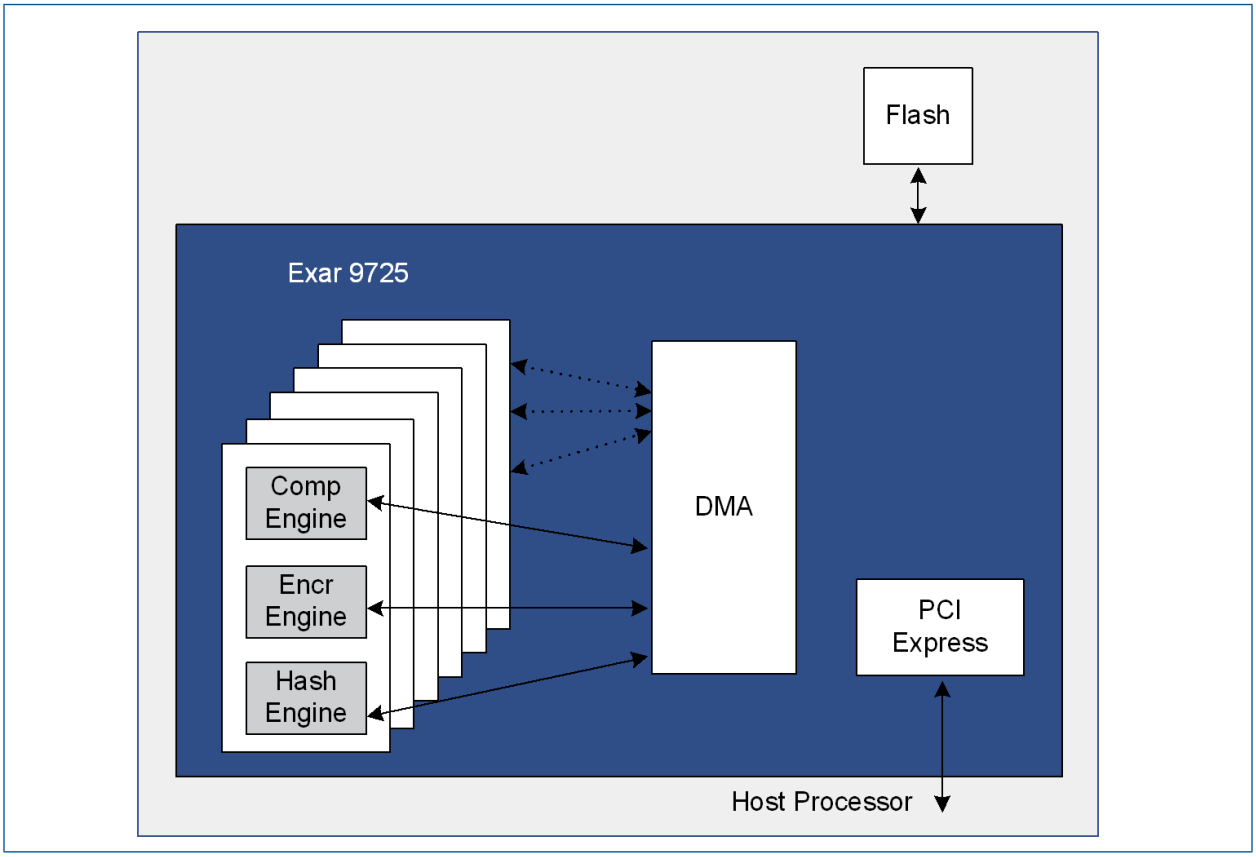


Figure 2-1. 9725 Interface Block Diagram

A file of source data that is sent to the 9725 is divided into commands as shown in Figure 2-2. Each command can be further sub-divided into slices for hash operations. Commands within a file may have a different number of slices. The number of slices per command may vary but must be evenly divisible by the command size. With a maximum command size of 4GB, and slices that must align to 64B, the maximum number of slices per command would

be 4G/64=64M, therefore the number of slices per command may vary from 1 to 64M. The last slice of a command may be smaller than the command SLICE_SIZE if the data does not divide evenly between the number of slices.

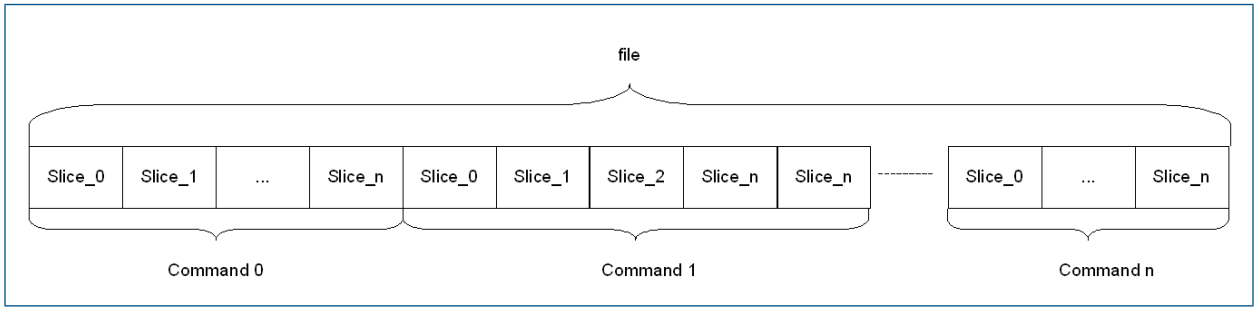


Figure 2-2. File Structure Example

Compression Engine

The Compression Engine is a high performance lossless data compression processor that uses the industry-standard Lempel-Ziv-Stac (LZS®) compression algorithm, as well as Exar's "Enhanced LZS" (eLZS) algorithm that uses an anti-expansion algorithm to limit worst case expansion during compression to 0.2%. If the output expands during compression, the original uncompressed input along with certain header bytes is passed through as the output.

Encryption Engine

The Encryption Engine performs real time encryption. The Encryption Engine supports AES-GCM, AES-CBC and AES-XTS protocols and the IEEE1619 and IEEE1619.1 encryption/ decryption reference standards.

Hash Engine

The Hash engine contains two internal cores; each core supports SHA1, SHA256, MD5, and HMAC-SHA1 hash algorithms. Typically, a command is sent to both Hash cores. The output of each core is compared for real time data verification. If instead, two hash operations are selected, they must use the same Hash algorithm. HMAC-SHA1 cannot be calculated in parallel with any other Hash calculation.

Real time verification is performed on all Slice, file and HMAC Hash operations. Slice+file Hash operations do not have real time verification because both hash cores are required. Slice, file, and slice+file operations may use SHA1, SHA256 and MD5 hash algorithms. HMAC operations may only select the SHA1 hash algorithm.

2.2 Data Integrity

The 9725 provides robust data integrity with a combination of Error Code Correction (ECC), Parity, Cyclic Redundancy Check (CRC), and Real Time Verification (RTV). These features combine to create strong data protection for encode and decode operations and high levels of assurance that data is not corrupted on-chip without being detected and reported.

2.2.1 ECC & Parity Protection

All data is protected with 6-bit Error Code Correction (ECC6), 8-bit Error Code Correction (ECC8) or parity.

- ECC6 is used to protect 16 bit width data path
- ECC8 is used to protect 64 bit width data path
- Parity is used to protect the PCIe Core RAMs (1 bit per byte)

ECC6 and ECC8 provide detection of all single, double, and triple bit errors anywhere in the data path. If an error is detected, the 9725 will report a parity error and stop processing that command.

Figure 2-3 illustrates the ECC and parity data path protection for a generic compression/ encryption/HMAC operation where the host does not attach a CRC to the raw input data. Figure 2-4 illustrates the ECC and parity data path protection for a generic compression/ encryption/HMAC operation where the host does attach a CRC to the raw input data.

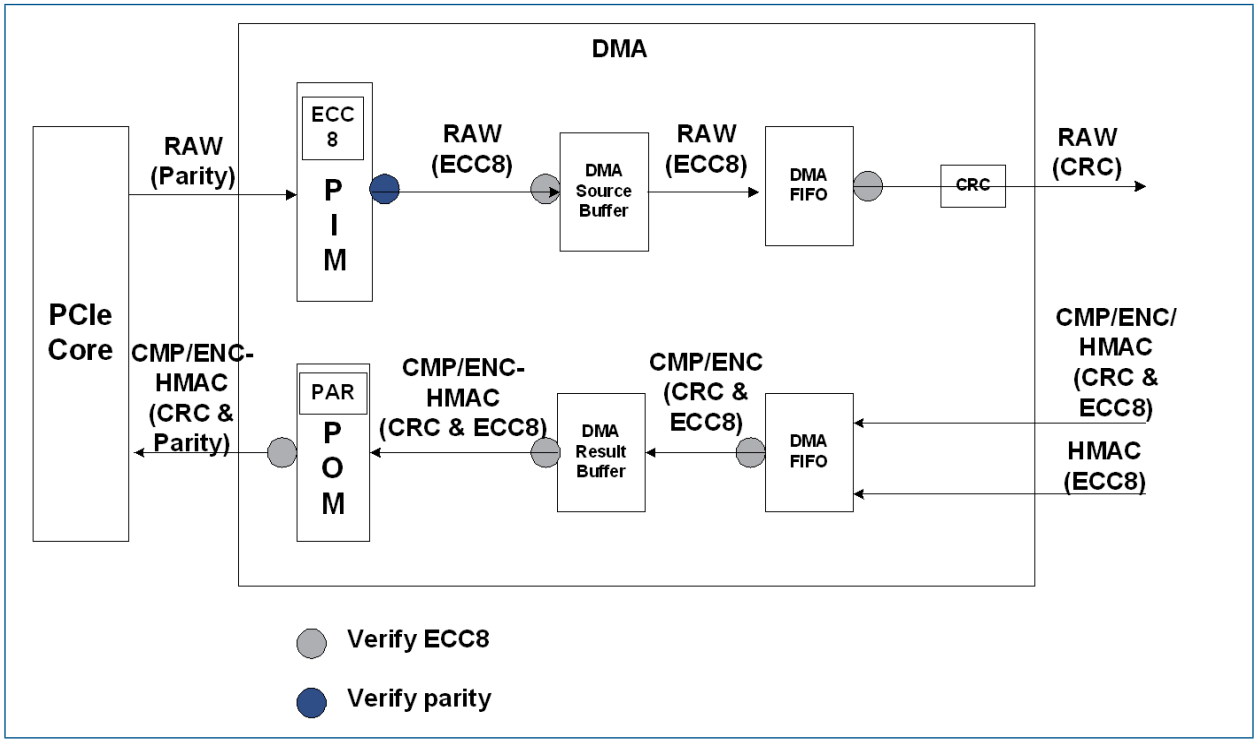


Figure 2-3. Data Path Protection Example - No Attached Host CRC

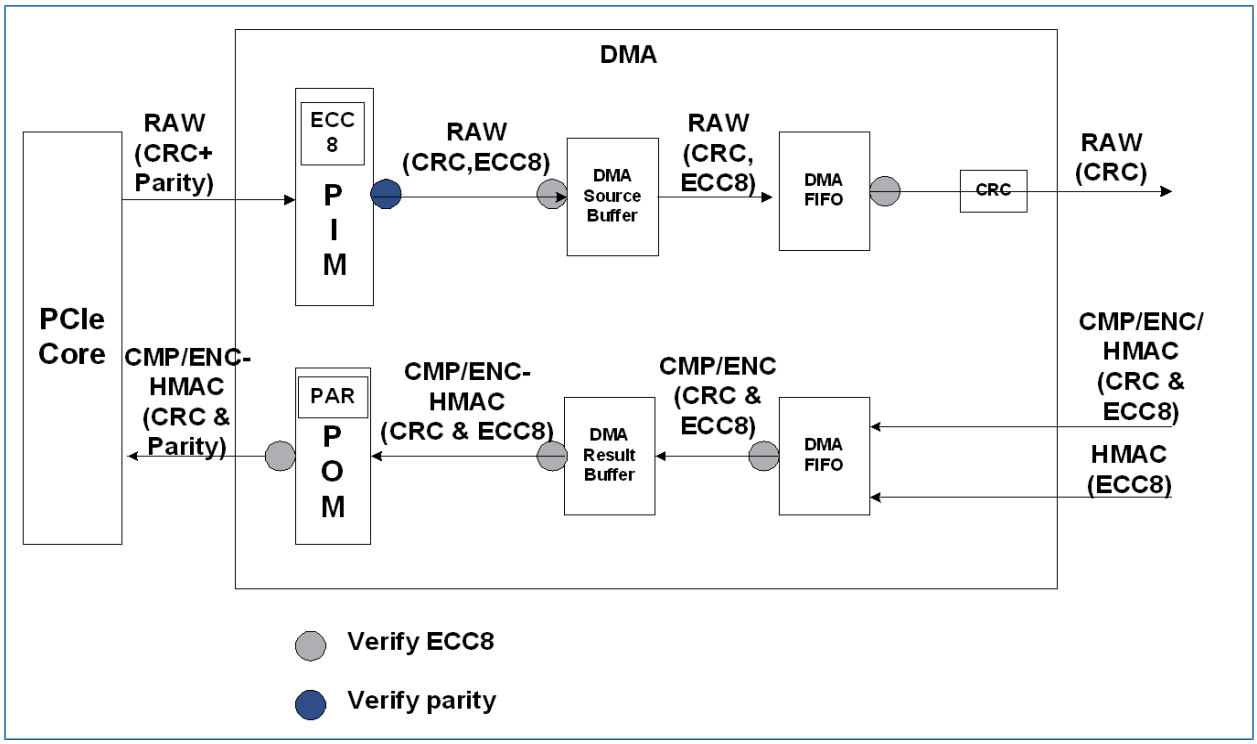


Figure 2-4. Data Path Protection Example - With Attached Host CRC

2.3 CRC Protection

The 9725 supports Cyclic Redundancy Check (CRC) to protect the compression/ decompression, encryption/decryption engines and to detect off-chip data corruption.

Data CRC can be generated by the host or by the 9725 device. The CRC can then be verified by the 9725 or the host, and may be sent from the 9725 to the host or stripped from the data stream. There are three CRC configuration modes:

No CRC

In this mode, no CRC operation is performed.

Host generated/checked CRC

For compression operations, the host calculates and appends the CRC to the data stream. The 9725 verifies the CRC, compresses the data and sends the CRC value to the host embedded in the data stream.

For decompression operations, the 9725 receives the compressed data stream, decompresses the data and verifies the CRC. The decompressed data stream with the CRC is passed to the host.

9725 generated/checked CRC

For compression operations, the 9725 calculates and appends the CRC to the data stream. The 9725 verifies the CRC, compresses the data and sends the CRC value to the host embedded in the data stream.

For decompression operations, the 9725 receives the compressed data stream, decompresses the data, verifies the CRC, and strips the CRC from the data stream. The decompressed data stream is passed to the host.

Exar does not recommend using the host generated/checked CRC mode due to the amount of overhead imposed on the host software.

The CRC configuration is set internally in the 9725 using three control signals: EM_CRC_EN, HOST_CRC_EN, and RCRC_OUT.

EM_CRC_EN:

Bit 8 of the DMA Configuration Register; enables the Engine Manager to generate the CRC.

HOST_CRC_EN:

Bit 9 of the DMA Configuration Register; enables the Host to generate the CRC.

RCRC_OUT:

Bit 7 of the DMA Configuration Register; controls if the CRC will be stripped from the decompressed data after verification or passed to the Host with the decompressed data.

Refer to Table 2-1 for how these control signals should be set for the proper CRC operation.

Table 2-1. CRC Configuration Settings

HOST_CRC_EN	EM_CRC_EN	RCRC_OUT	CRC Behavior - Encode	CRC Behavior - Decode
1	0	1	Host calculates and appends CRC to data stream. 9725 verifies CRC. CRC value sent to host embedded in the encoded data stream.	9725 verifies CRC. CRC value is appended to the destination data stream and transferred to the host.
0	0	0	No CRC operation performed.	No CRC operation performed.
0	1	0	9725 generates CRC. CRC value sent to host embedded in the encoded data stream.	9725 verifies CRC. CRC value is stripped from the destination data stream.

Please refer to the *Express DR SDK Getting Started Guide* for more information on the driver configuration file settings. The *Express DR SDK User Guide* contains detailed information on the CRC behavior.

2.4 PCIe CRC Protection

The 9725 also supports End-to-end Cyclic Redundancy Check (ECRC) for transfers over the PCIe bus. The host may configure the 9725 to either enable or disable ECRC protection using the PCIe Advanced Error Report Capabilities and Control Register (refer to the PCIe specification for further information). If enabled, the 9725 will generate the ECRC on the fly to protect data on the PCIe bus.

2.5 Real Time Verification

The Compression, Encryption and Hash engines in the 9725 contain internal real time verification.

2.5.1 Compression Engine Real Time Verification

Each Compression engine contains one compression core and one decompression core for LZS and eLZS operations. If CRC checking is enabled, data for compression operations in the encode direction will be automatically decompressed to verify that the decompressed CRC matches the original CRC of the raw data. For compression operations in the decode direction, the decompressed CRC will automatically be verified against the original CRC attached to the compressed raw data. The Compression engine also employs an ECC in its FIFO.

Figure 2-5 illustrates LZS/eLZS real time verification in the Compression engine for encode and decode operations. In the following figures, a CRC in parenthesis, as in RAW (CRC), signifies that the CRC is embedded into the data.

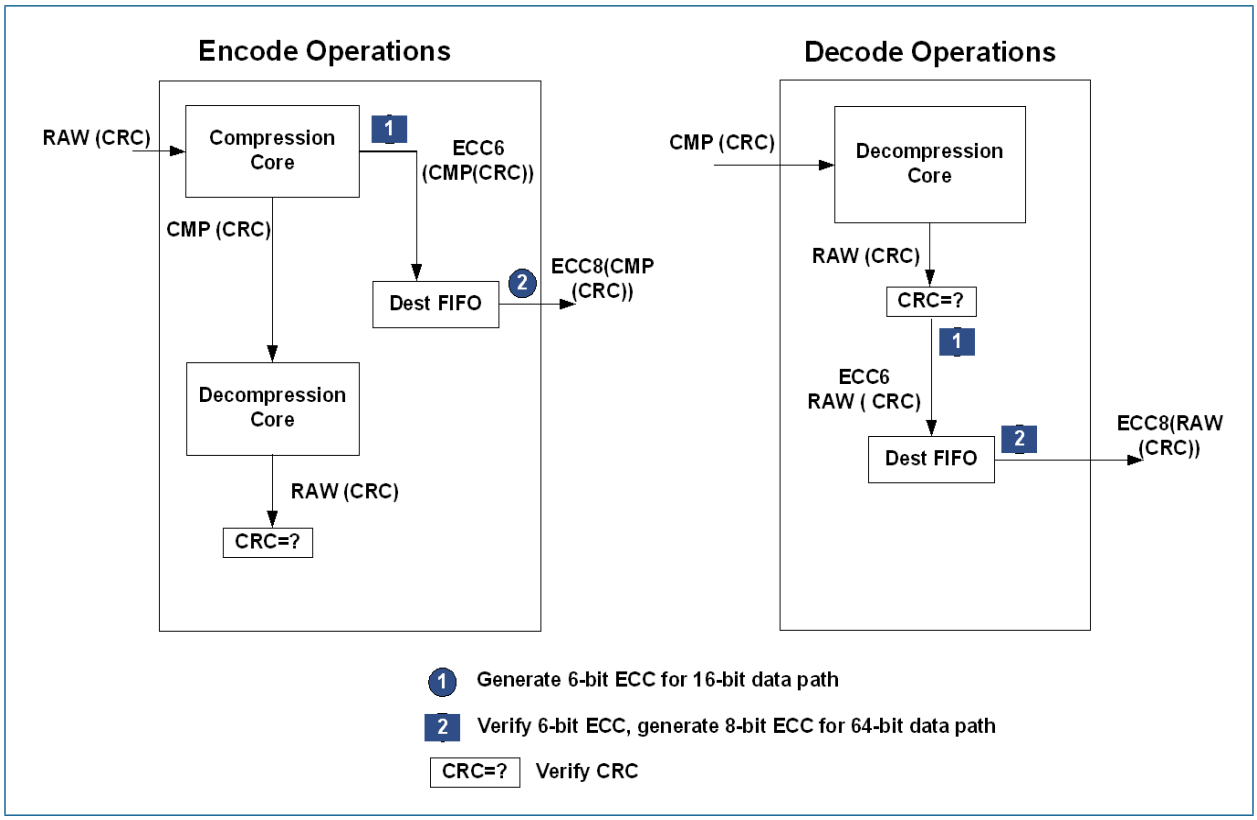


Figure 2-5. Compression Engine eLZS Real Time Verification

2.5.2 Encryption Engine Real Time Verification

The Encryption engine contains one encryption core and one decryption core for AES operations. If CRC checking is enabled, the 9725 will automatically perform real time verification of the data in the encryption engine. For encryption operations in the encode direction, the Encryption engine will verify the raw data CRC, encrypt the raw data, automatically decrypt the encrypted data, and compare the decrypted data CRC against the original CRC. For encryption operations in the decode direction, the decryption engine verifies the decrypted data CRC against the CRC attached to the encrypted raw data.

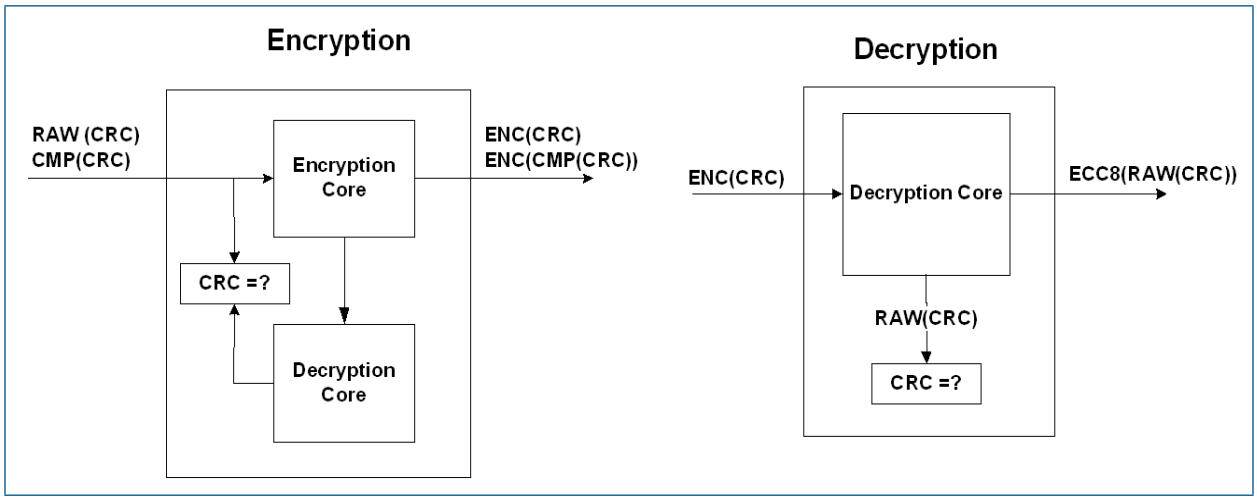


Figure 2-6. Encryption Engine Real Time Verification

2.5.3 Hash Engine Real Time Verification

The Hash engine contains two hash cores to implement real time verification of the data. For single File hash or Slice hash operations, both Hash cores will simultaneously verify the raw data CRC and calculate the Hash and then compare the two results. If an error is detected it will be reported to the host. For "File hash + Slice hash" operations, one core calculates the "File hash" and the other core calculates the "Slice Hash". Figure 2-7 illustrates the real time verification for File hash or Slice hash operations. Figure 2-8 illustrates the data flow for File + Slice hash operations. Note that for File + Slice hash operations, there is no spare hash engine for real time verification. Figure 2-9 shows a HMAC-SHA1 operation. In Figure 2-9, RAW data is first encrypted by the Encryption engine and real time verification is performed if CRC checking is enabled. The Hash engine then verifies the encrypted data CRC while calculating the MAC and then compares the two results.

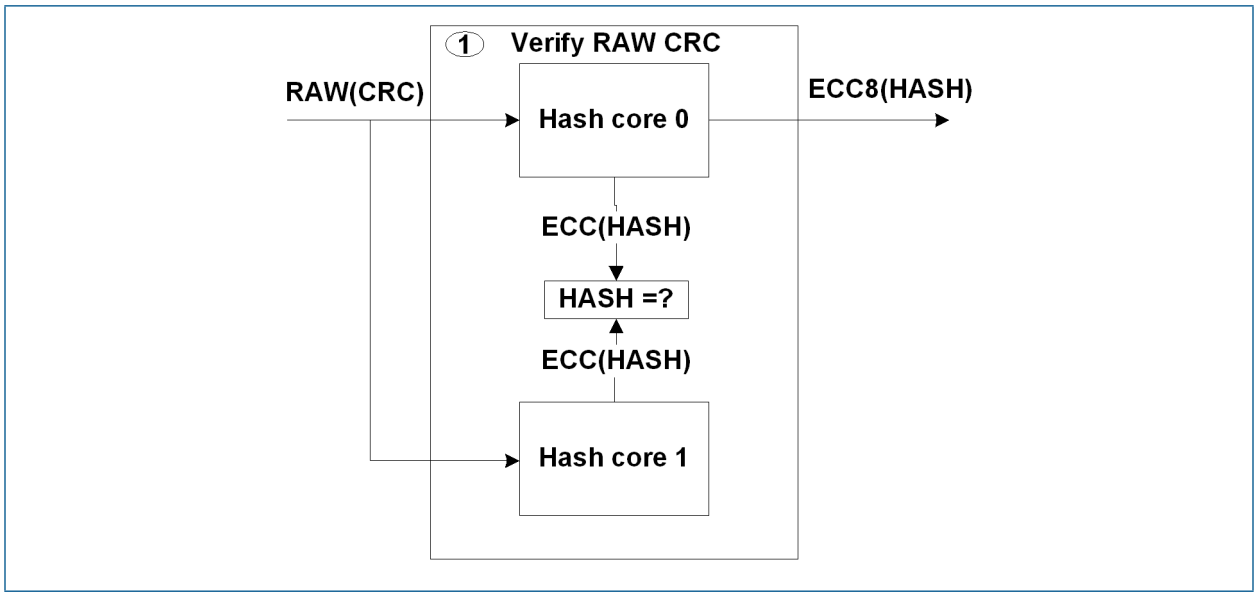


Figure 2-7. File or Slice Hash Engine Real Time Verification

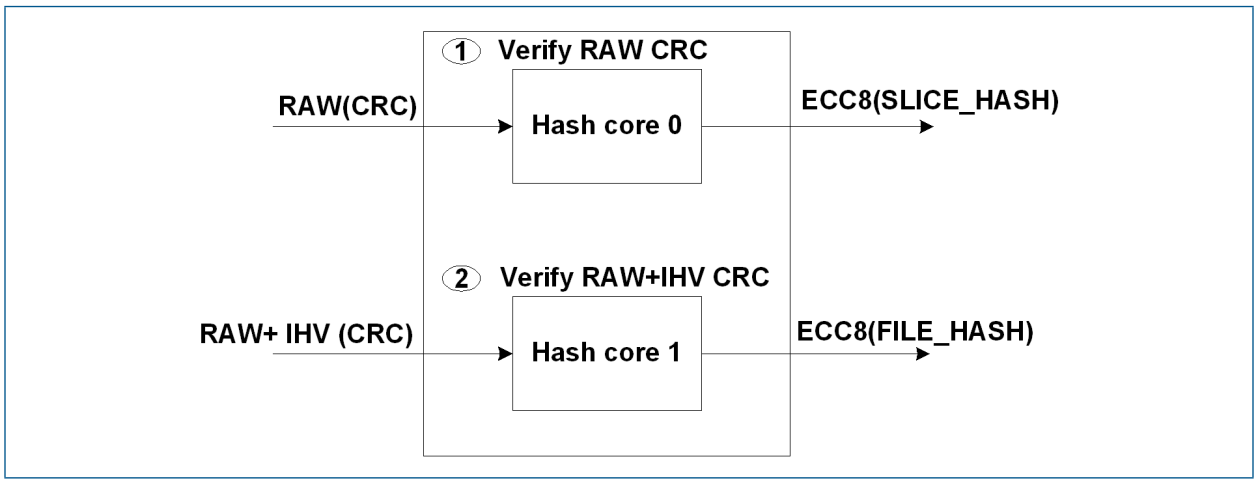


Figure 2-8. Slice+File Hash Disables Real Time Verification

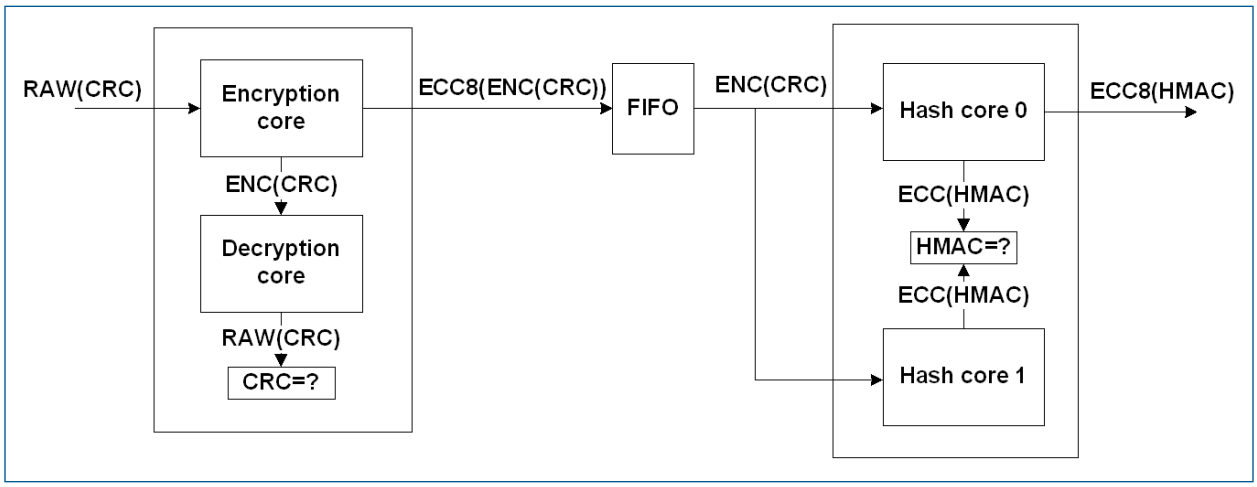


Figure 2-9. HMAC Real Time Verification

2.6 Endian Settings

Network computers support a variety of endian formats. In order to accommodate most network systems, the 9725 supports the endian formats listed below.

- 32-bit and 64-bit Little endian (No swap)
- 32-bit and 64-bit Big endian (Byte swap in word)
- 64-bit Little endian double word swap (Word swap, no byte swap in word
- 64-bit Little endian double word swap (Word swap and byte swap in word)

These formats are illustrated in Figure 2-10.

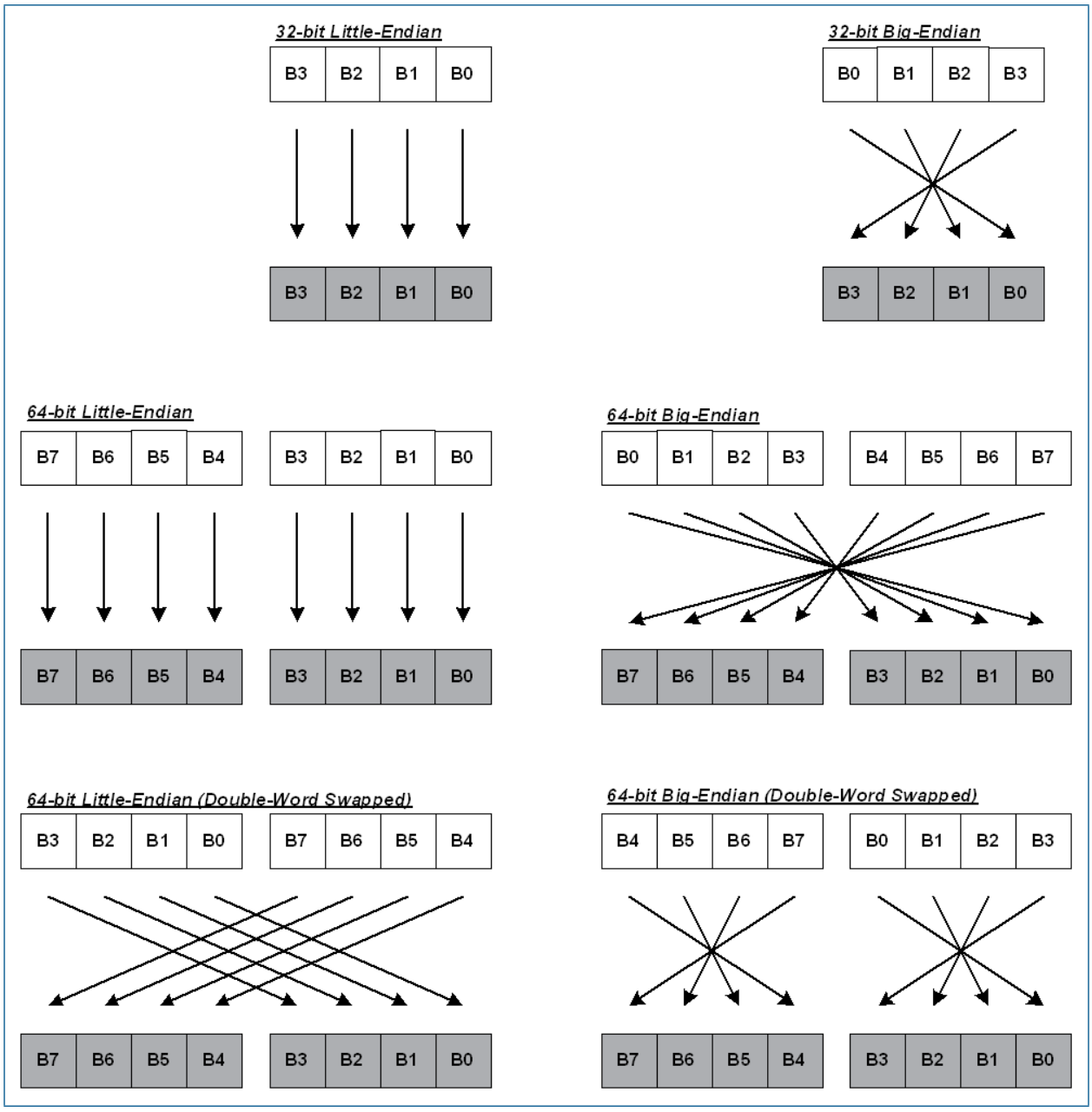


Figure 2-10. Supported Endian Formats

No latency is induced by the 9725 to perform these endian format translations. Internally, endian format may be configured for the following data paths:

- LZS Engine

  The host must set to little endian using bit 1 of the LZS Configuration Register.

- Source and destination descriptors

The host may set two endian formats during initialization in the DMA Configuration Register fields EF0 and EF1. The Endian Pointer field of the source/destination descriptor can then be used to select between these two endian formats on a per command basis. Note a command must use the same endian format for both the source and destination descriptor.

- Command/Result Ring

  The host sets this format using bits [5:4] of the DMA Configuration Register.

3 Data Structures

This section describes the 9725 data structures for the command ring, result ring, as well as the command structure format.

Note that the 9725 SDK and its API abstracts the user application from the level of detail described in this chapter. However, this information will help the user understand the 9725 operation in greater detail.

3.1 Definitions

A *record* is a portion of the data stream with a definite start and end. Compression and decompression operate on a record-by-record basis.

A *descriptor* defines the format for a record.

A *command* is a single 9725 instruction, launched when the command is written to the LZS Command register. A command may consist of many descriptors.

A *command ring* is a circular queue of descriptors containing commands and pointers to data in the queue.

A *completion code* is a 32-bit or 64-bit result summary that the 9725 writes into the result ring.

A *result ring* is a circular queue of completion codes that are read by the host, containing indices that point to the corresponding command ring index descriptor.

A *source buffer* is memory structure that contains the source input data. For each command, the source buffer contains exactly one record.

A *destination buffer* is a memory structure that contains the output data.

3.2 Command Ring

Command rings are implemented in host memory. The 9725 SDK writes a complete command structure as an entry in the command ring. The 9725 reads the command structure and interprets its various fields to perform the appropriate operations. Once the 9725 has completed executing a command, it updates the appropriate entries within the corresponding command structure and signals an interrupt to the host. The SDK then reads the entries in the command structure that have been updated by the 9725 to fetch the results.

The 9725 does not maintain a “full” bit for the command ring, however, the SDK manages the command ring to avoid an overflow condition.

The maximum number of commands in the command ring is configurable by the host using the SDK configuration file. After the host software determines the memory location of the command pointer ring and command structure, the SDK will write the command ring address and write pointer into 9725 DMA registers. The 9725 will fetch the command pointer, and then fetch the command structure.

The 9725 supports both direct and indirect command addressing to execute commands. The performance for both addressing mode is the same.

3.2.1 Direct Addressing Mode

In direct command addressing mode, the command ring occupies contiguous physical memory. The maximum number of command structures in the command ring is set in the SDK configuration file when the system driver is initialized. The command ring should be aligned to a 128 byte boundary to avoid splitting the command read request.

Figure 3-1 illustrates a typical command ring structure using direct addressing when the maximum number of descriptors per command is set to 8. Note that any unused bytes in the command structure are padded with spare bytes.

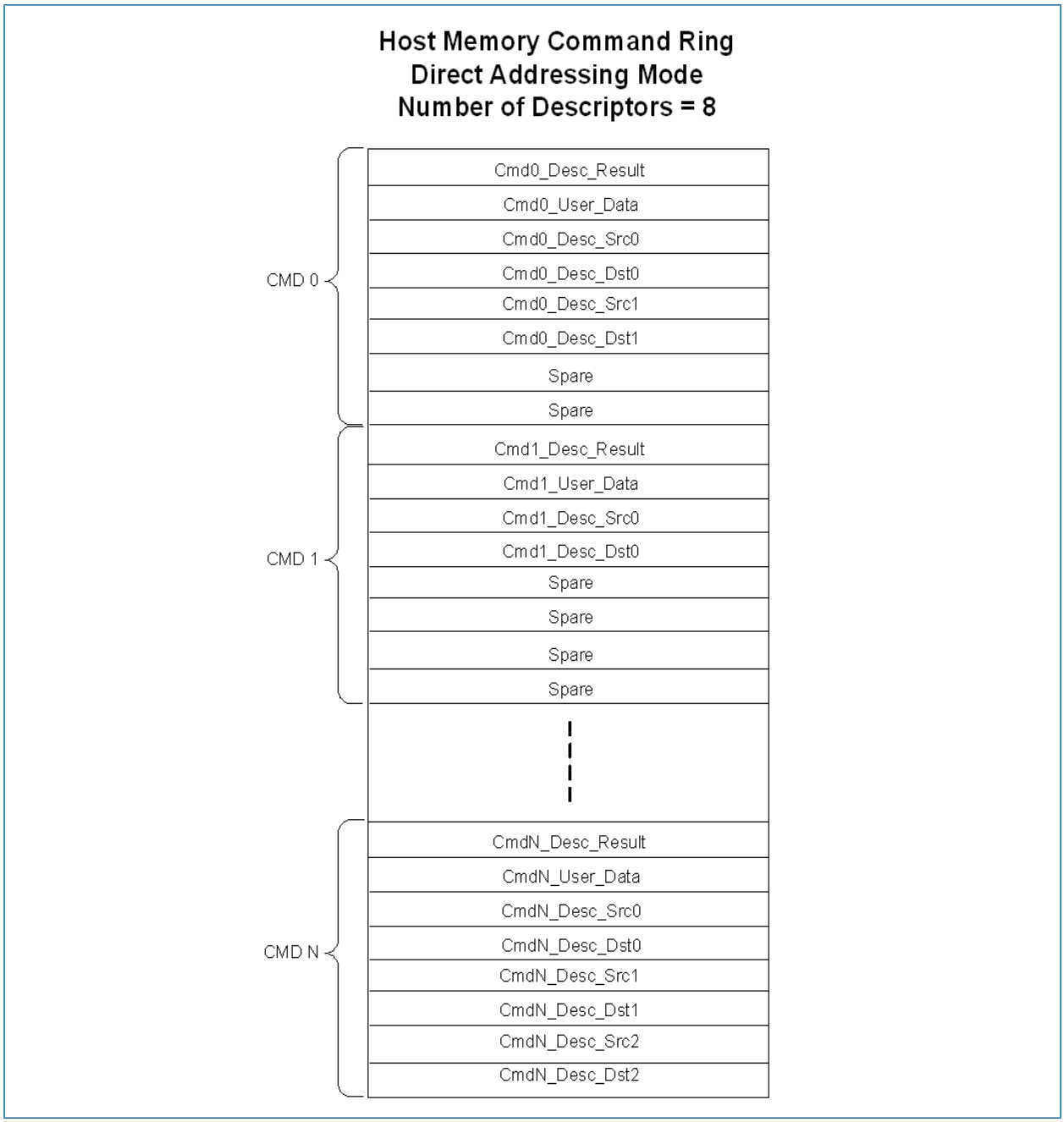


Figure 3-1. Direct Addressing Mode Example

Direct command addressing mode is recommended for applications with a relatively small number of commands and a small number of descriptor per command. Direct addressing should be used for backward compatibility to the Exar Express DR 6x0, 10x0 cards as they only support this addressing mode.

3.2.2 Indirect Addressing Mode

For indirect command addressing, the host maintains a command ring as well as a linear array command pointer ring that points to the commands. In indirect addressing, the command pointer ring must reside in contiguous memory but the commands themselves may be located anywhere in host memory. The maximum number of command pointers in the command pointer ring is set in the SDK configuration file when the system driver is initialized.

The command pointer ring must be aligned to a 8 byte boundary and must reside in contiguous physical memory. In 32-bit addressing mode, every command pointer is 4 bytes; in 64-bit addressing mode, every command pointer is 8 bytes.

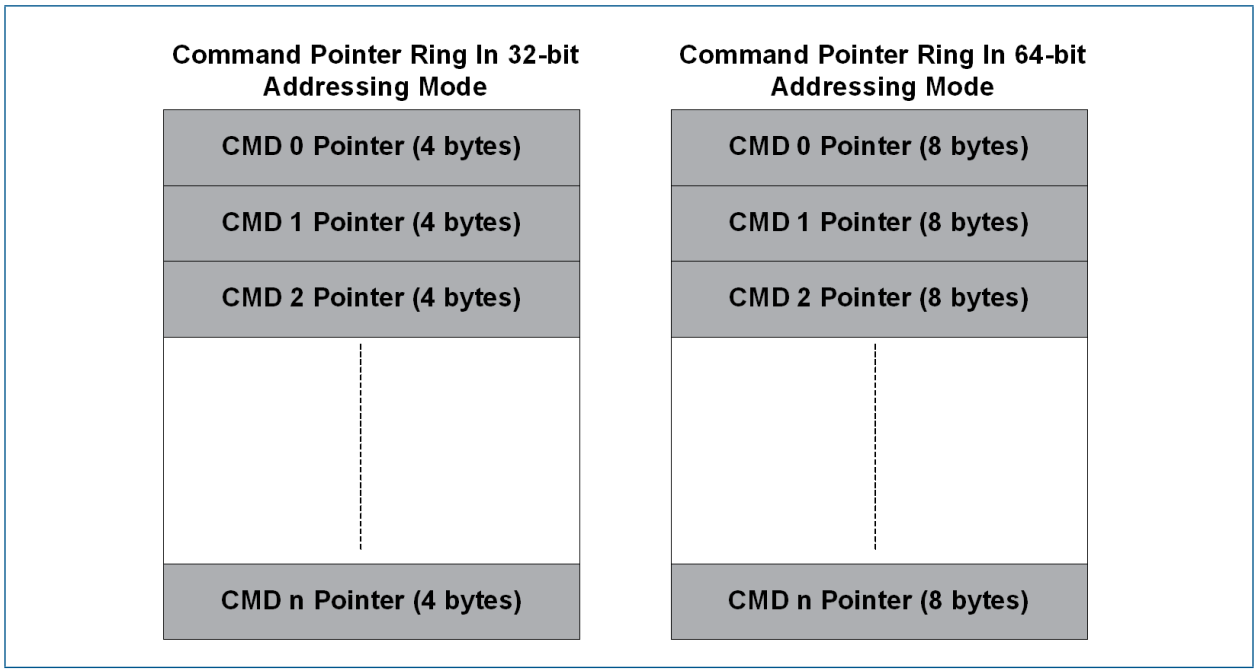


Figure 3-2. Command Pointer Ring Format

Figure 3-3 illustrates a typical command ring structure when using indirect addressing. Note that in indirect addressing mode, the command structure size of each command can vary and no spare bytes are required.

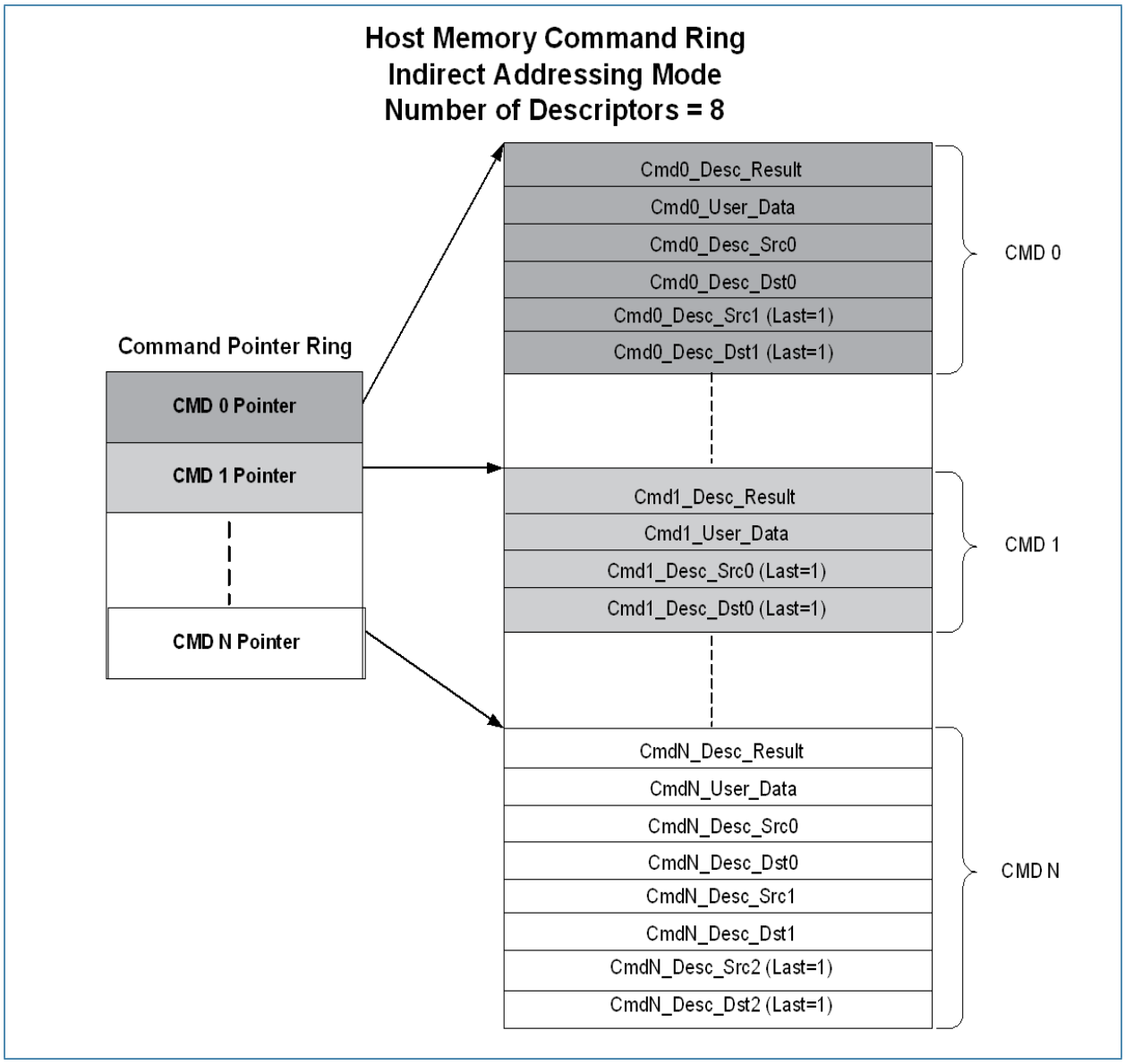


Figure 3-3. Indirect Addressing Mode Example

Indirect command addressing mode is recommended for applications with a large number of commands or a large number of descriptors per command.

3.3 Command Structure

The host must ensure that the starting address of all command structures are 512-byte aligned. This constraint will avoid the PCIe 4K boundary read request limitation because the 9725 maximum command read request is 512 bytes.

A command structure consists of a linear array of descriptors. The size of one command structure is only limited by the amount of host contiguous physical memory. Each command may have a unique command structure.

The command structure includes one result descriptor, one user data descriptor and several pairs of source and destination descriptors.

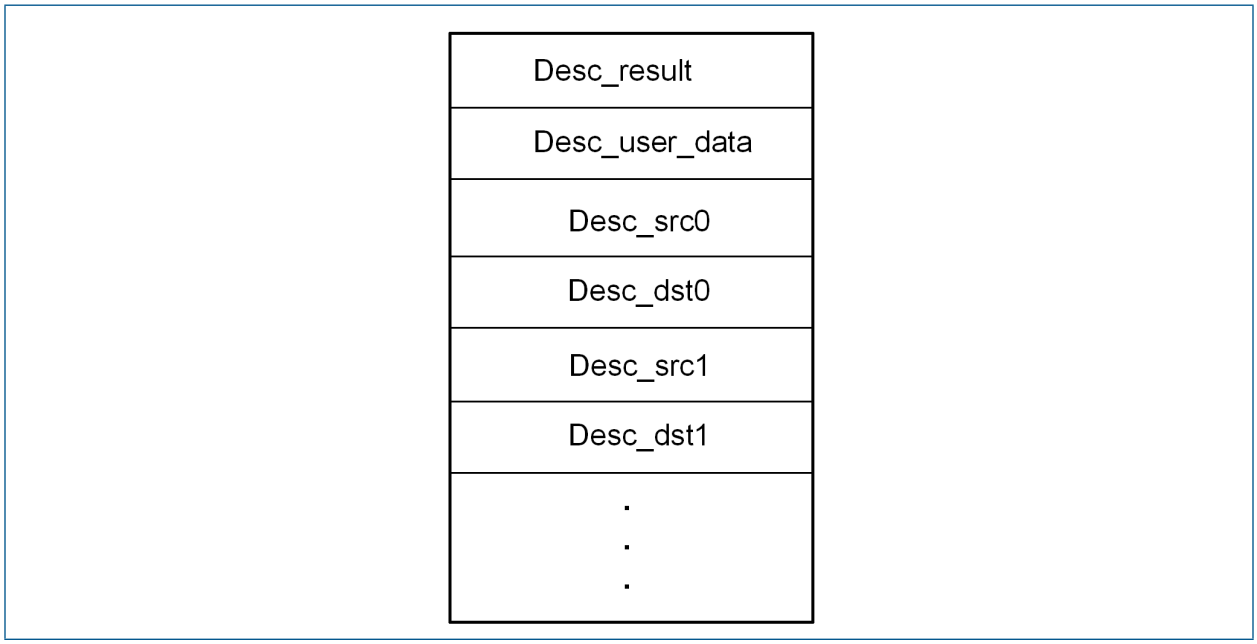
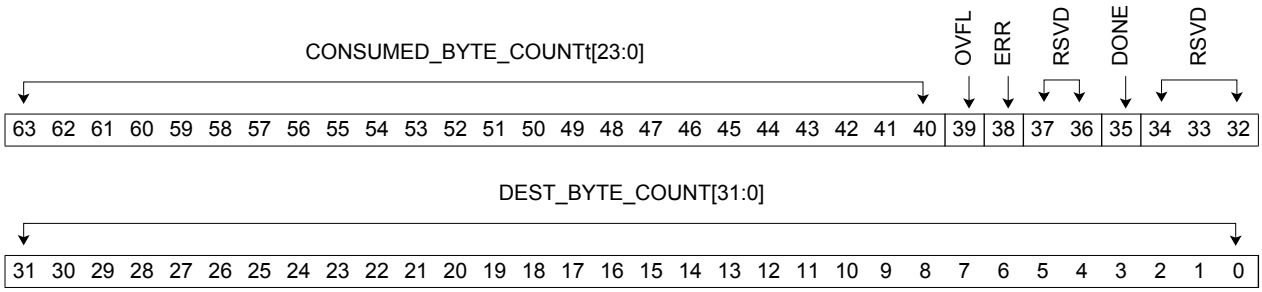


Figure 3-4. Command Structure

3.3.1 Desc_result

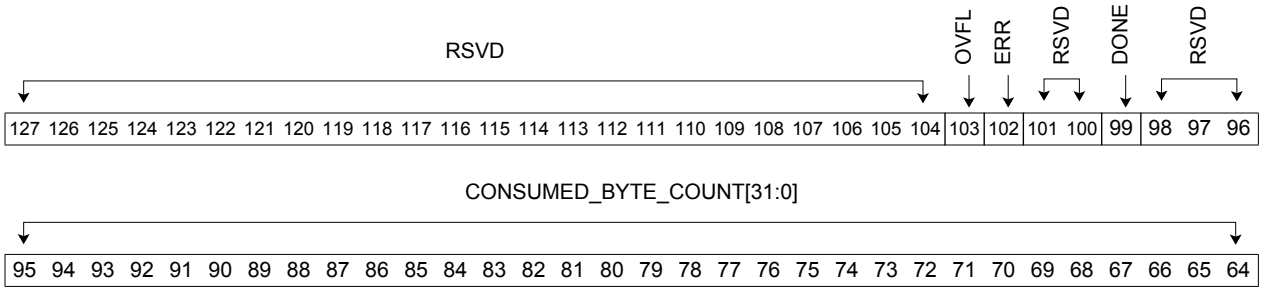
The result descriptor holds the location of the command result. The 9725 will write to this descriptor after a command completes. The Desc_result format is not the same for 32-bit and 64-bit addressing modes.

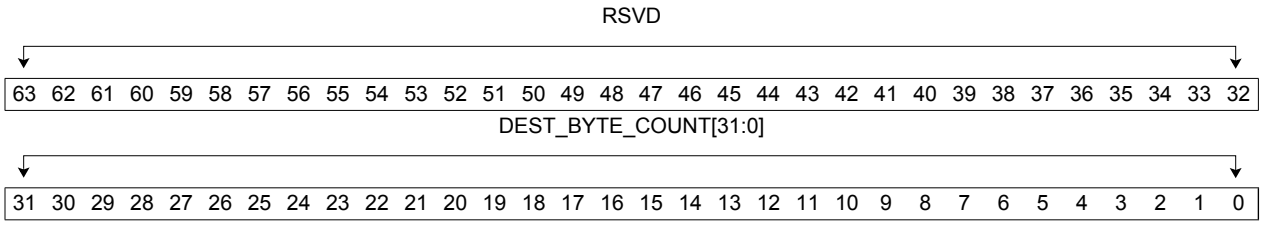
32-bit addressing mode:



Field Name	Bits	Default	Description
CONSUMED_BYTE_ COUNT[23:0]	63:40	0	Consumed Byte Count. This field contains the number of source data bytes that were processed by the hardware. Normally, this value will be the same as the total source data length. However, if this command does not have enough destination buffer space, then consumed_byte_count will be less than the total_source_count.
OVFL	39	0	Overflow bit. The 9725 will set this bit if the destination data size exceeded the total destination buffer space. 0 = No destination buffer overflow occurred 1 = Destination buffer overflow occurred
ERR	38	0	Error bit. The 9725 will set this bit if any type of error occurred during command processing. 0 = No error occurred 1 = Error occurred
RSVD	37:36	0	Reserved.
DONE	35	0	Done bit. This bit is written to and read by both the host and 9725. To use this bit to determine whether a command is finished, the host software would set this bit when the command structure is written. The 9725 will write a zero to this bit when it has completed a command. 0 = 9725 has completed the command 1 = Host has set up the command for the 9725
RSVD	34:32	0	Reserved.
DEST_BYTE_COUNT [31:0]	31:0	0	Destination Data Byte Count. This field contains the number of bytes written into the destination data buffer.

64-bit addressing mode:





Field Name	Bits	Default	Description
RSVD	127:104	0	Reserved.
OVFL	103	0	Overflow bit. The 9725 will set this bit if the destination data size exceeded the total destination buffer space. 0 = No destination buffer overflow occurred 1 = Destination buffer overflow occurred
ERR	102	0	Error bit. The 9725 will set this bit if any type of error occurred during command processing. 0 = No error occurred 1 = Error occurred
RSVD	101:100	0	Reserved.
DONE	99	0	Done bit. This bit is written to and read by both the host and 9725. To use this bit to determine whether a command is finished, the host software would set this bit when the command structure is written. The 9725 will write a zero to this bit when it has completed a command. 0 = 9725 has completed the command 1 = Host has set up the command for the 9725
RSVD	98:96	0	Reserved.
CONSUMED_BYTE_ COUNT[31:0]	95:64	0	Consumed Byte Count. This field contains the number of source data bytes that were processed by the hardware. The 9725 updates this field after each command finishes. Normally, this value will be the same as the total source data length. However, if this command does not have enough destination buffer space, then consumed_byte_count will be less than the total_source_count.
RSVD	63:32	0	Reserved.
DEST_BYTE_COUNT [31:0]	31:0	0	Destination Data Byte Count. This field contains the number of bytes written into the destination data buffer.

3.3.2 Desc_User_Data

This descriptor may be used as needed by the user application. The 9725 does not modify the content of these descriptors.

3.3.3 Desc_src and Desc_dst

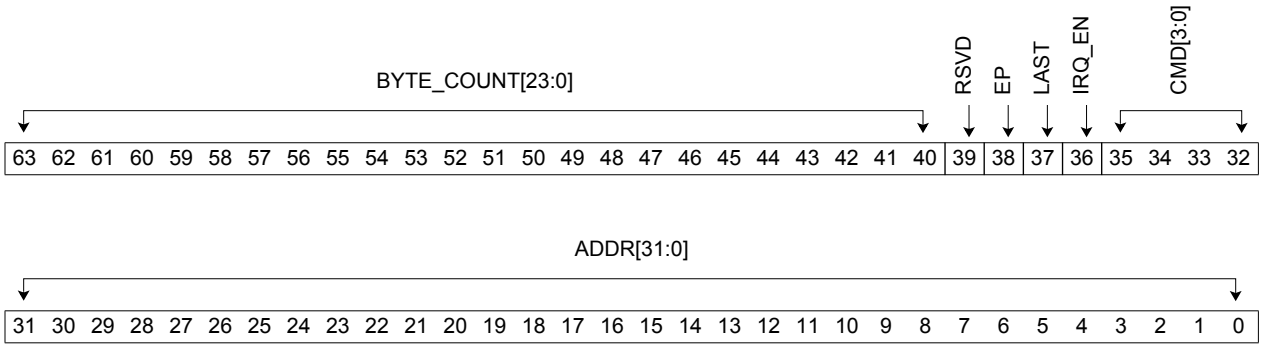
Generally, the source data descriptors point to source data buffers. For some commands, however, the first and second source descriptors may be used to store command-related configuration data and then the subsequent descriptors will be used to point to the source buffers. The source buffer address length may be an arbitrary number of bytes. The source buffer address is byte aligned except for when the source buffer is used to store the information fields for the "Key", "IHV", "MAC", or "IV", in which case its address are 8-byte aligned.

Generally, the destination data descriptors point to the destination buffers. For some commands, however, the first destination descriptor may be used to store command-related configuration data and then the subsequent descriptors will be used to point to the destination buffers. The destination buffer starting address and size have either a 4-byte or 8-byte alignment restriction, depending on the value of the data endian format in the driver configuration file. If the value of data_endian_format is 0x2 or 0x3, the destination buffer will be 8-byte aligned, otherwise, it will be 4-byte aligned. Usually, the destination buffers are used to store the result data, and must be fully consumed in order. For hash related operations, the first destination buffer is used to store hash values only.

Results from a Hash operation are stored in the hash buffer. The hash buffer is actually the first descriptor in the destination descriptor and must be located in physically contiguous memory. The hash buffer starting address and length must be 32-byte aligned.

The source and destination descriptors both contain 16 bytes in the 32-bit addressing mode and 32 bytes in the 64-bit addressing mode.

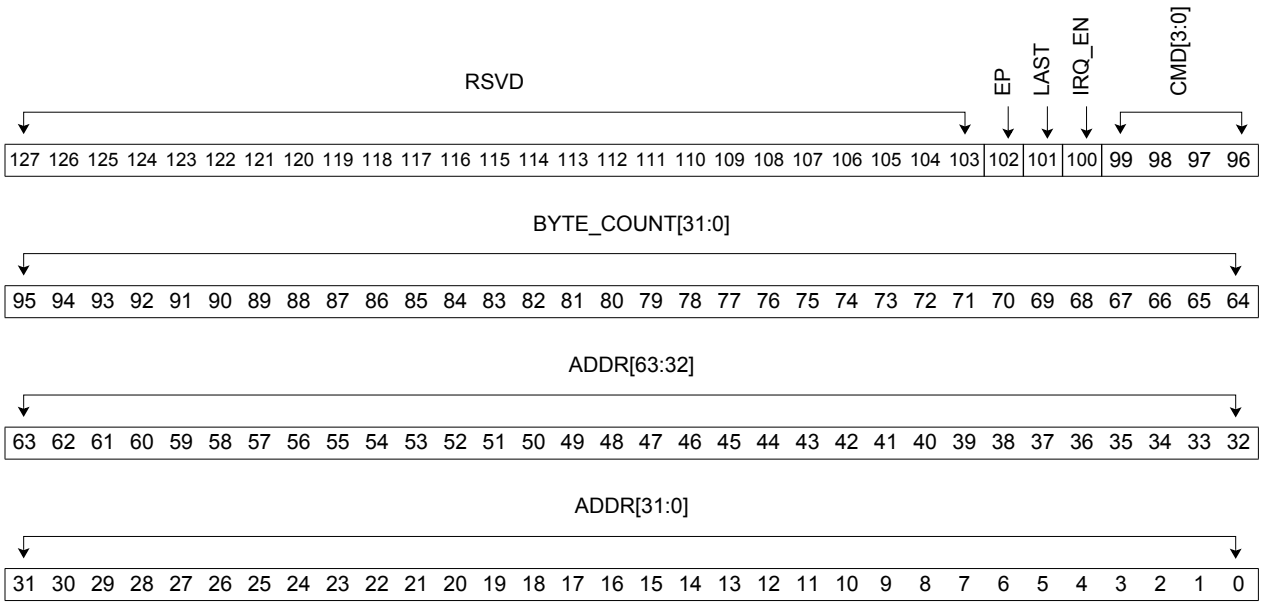
32-bit Addressing Mode Format:



Field Name	Bits	Default	Description
BYTE_COUNT[23:0]	63:40	0	Byte Count. The size of the source or destination buffer in bytes. Byte Count indicates the size of the data block defined by this descriptor. For the source descriptor, the byte count is an arbitrary number of bytes. For the destination descriptor, the byte count must be a multiple of 8 bytes.
RSVD	39	0	Reserved.
EP	38	0	Endian Pointer This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, "DMA Configuration Register". See Endian Settings for more information. This bit is only valid for the first Desc_src and Desc_dst.
LAST	37	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.
IRQ_EN	36	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. This bit is only valid for the first Desc_src and Desc_dst. 0 = Interrupt disabled 1 = Interrupt enabled
CMD[3:0]	35:32		Command. This field is used to set the command type. 0000 = Compression only 0001 = Decompression only 0010 = Passthrough only 0011 = Write Key 0100 = Compression + Encryption 0101 = Decompression + Decryption 0111 = Decryption only 1000 = Compression + Hash 1010 = Hash only 1100 = Compression + Encryption + Hash* 1110 = Encryption + Hash* All other values are reserved. These four command field bits are only valid for the first Desc_src and Desc_dst. *Compression + AES-CBC/HMAC-SHA1 + Hash and Compression + AES-XTS are not supported.

Field Name	Bits	Default	Description
ADDR[31:0]	31:0	0	Address. The starting physical address of the source or destination descriptor. For source descriptors, including those that only contain AAD, the address is byte aligned. If the source buffers include the information fields “Key”, “IV”, “IHV”,”MAC”, or “GZIP History”, the address must be on an 8-byte boundary. For destination descriptors, the address is either 4-byte or 8-byte aligned, depending on the value of the data endian format in the driver configuration file.

64-bit Addressing Mode Format:



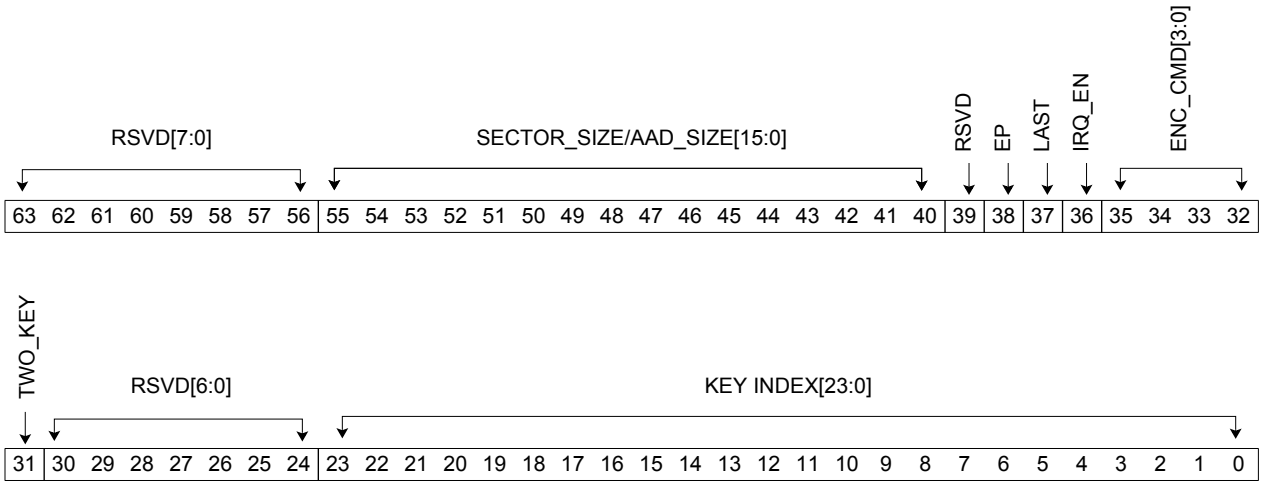
Field Name	Bits	Default	Description
RSVD	127:103	0	Reserved
EP	102	0	Endian Pointer. This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, “DMA Configuration Register”. See Endian Settings for more information. This bit is only valid for the first Desc_src and Desc_dst.
LAST	101	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.

Field Name	Bits	Default	Description
IRQ_EN	100	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. This bit is only valid for the first Desc_src and Desc_dst. 0 = Interrupt disabled 1 = Interrupt enabled
CMD[3:0]	99:96	0	Command. This field indicates the command. 0000 = Compression only 0001 = Decompression only 0010 = Passthrough only 0011 = Write Key 0100 = Compression + Encryption 0101 = Decompression + Decryption 0111 = Decryption only 1000 = Compression + Hash 1010 = Hash only 1100 = Compression + Encryption + Hash* 1110 = Encryption + Hash* All other values are reserved. These four command field bits are only valid for the first Desc_src and Desc_dst. *Compression + AES-CBC/HMAC-SHA1 + Hash and Compression + AES-XTS not supported
BYTE_COUNT[31:0]	95:64	0	Byte Count. The size of the source or destination buffer in bytes. Byte Count indicates the block size defined by the descriptor. For the source descriptor, the byte count is an arbitrary number of bytes. For the destination descriptor, the byte count must be a multiple of 8 bytes.
ADDR[63:0]	63:0	0	Address. The starting physical address of the source or destination descriptor. For source descriptors, including those that only contain AAD, the address is byte aligned. If the source buffers include the information fields “Key”, “IV”, “IHV”,”MAC”, or “GZIP History”, the address must be on an 8-byte boundary. For destination descriptors, the address is either 4-byte or 8-byte aligned, depending on the value of the data endian format in the driver configuration file.

3.3.3.1 Desc_src0 for Encryption/Decryption Commands

This special source descriptor is required for encryption and decryption commands to set the type of encryption command, and the location of the encryption key in the 9725 Key SRAM by means of an index and sector size. This descriptor must be the first descriptor (Desc_src0) for all encryption and decryption commands.

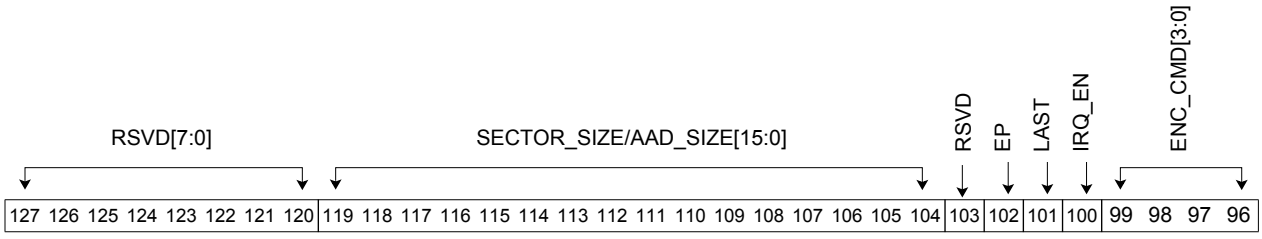
32-bit Addressing Mode Format

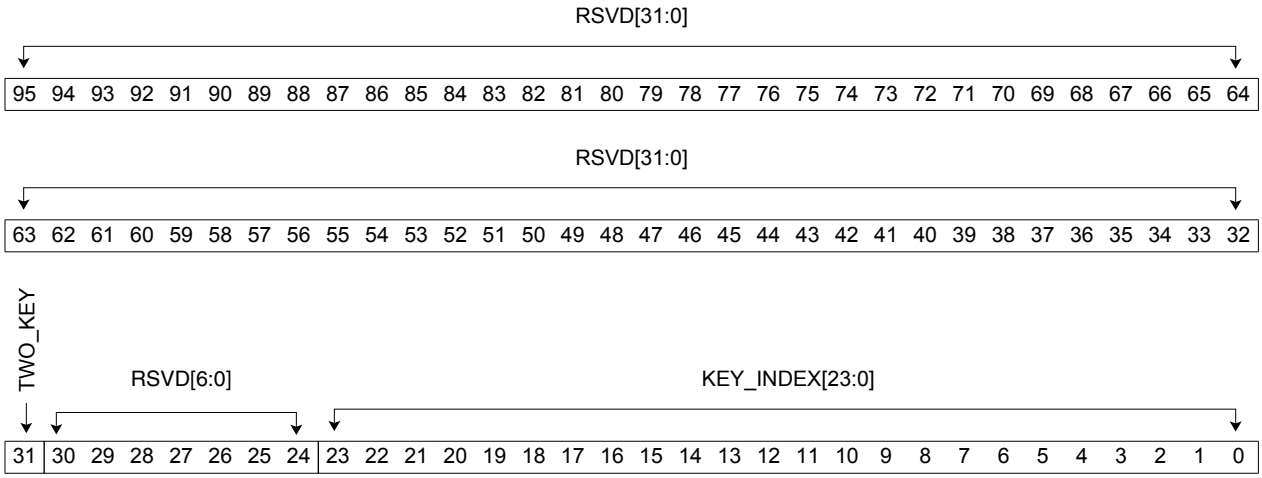


Field Name	Bits	Default	Description
RSVD	63:56	0	Reserved.
SECTOR_SIZE/ AAD_SIZE[15:0]	55:40	0	Sector Size / AAD Size. The behavior of this field depends on the encryption algorithm. For XTS mode, this field represents the Sector Size. For GCM mode, this field represents the AAD length in bytes. Sector size in XTS mode: 0x0000 - 0x0080 = 0x10000 - 0x10080 bytes 0x0081 - 0xFFFF = 0x0081 - 0xFFFF bytes The sector size must be 4-byte aligned and greater than 128 bytes, and does not include the DIF (fixed 8 bytes) when in XTS DIF mode. AAD size in GCM mode: 0x0000 - 0xFFFF = 0 - 65536 bytes.
RSVD	39	0	Reserved.
EP	38	0	Endian Pointer. This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, "DMA Configuration Register". See Endian Settings for more information.
LAST	37	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.

Field Name	Bits	Default	Description
IRQ_EN	36	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. 0 = Interrupt disabled 1 = Interrupt enabled
ENC_CMD[3:0]	35:32	0	Encryption/Decryption Command. This field is used to set the encryption or decryption command type. 0100 = Compression + Encryption 0101 = Decompression + Decryption 0111 = Decryption only 1100 = Compression + Encryption + Hash* 1110 = Encryption + Hash* All other values are reserved. *Compression + AES-CBC/HMAC-SHA1 + Hash and Compression + AES-XTS are not supported.
TWO_KEY	31	0	Two Key. Indicates whether the Key Manager should fetch one key or two keys for the current command. 0: Fetch one key 1: Fetch two keys for AES-XTS 256/512-bit key.
RSVD	30:24	0	Reserved.
KEY_INDEX[23:0]	23:0	0	Key Index. The index of the keys stored in the 9725 Key SRAM. 0 = Use Key SRAM index 0 for the encryption/ decryption command 1 = Use Key SRAM index 1 for the encryption/ decryption command ... 255 = Use Key SRAM index 255 for the encryption/ decryption command All other values are reserved for future expansion.

64 bit Mode Addressing Format





Field Name	Bits	Default	Description
RSVD	127:120	0	Reserved.
SECTOR_SIZE/ AAD_SIZE[15:0]	119:104	0	Sector Size / AAD Size. The behavior of this field depends on the encryption algorithm. For XTS mode, this field represents the Sector Size. For GCM mode, this field represents the AAD length in bytes. Sector size in XTS mode: 0x0000 - 0x0080 = 0x10000 - 0x10080 bytes 0x0081 - 0xFFFF = 0x0081 - 0xFFFF bytes The sector size must be 4-byte aligned and greater than 128 bytes, and does not include the DIF (fixed 8 bytes) when in XTS DIF mode. AAD size in GCM mode: 0x0000 - 0xFFFF = 0 - 65536 bytes.
RSVD	103	0	Reserved
EP	102	0	Endian Pointer. This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, "DMA Configuration Register". See Endian Settings for more information.
LAST	101	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.
IRQ_EN	100	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. 0 = Interrupt disabled 1 = Interrupt enabled

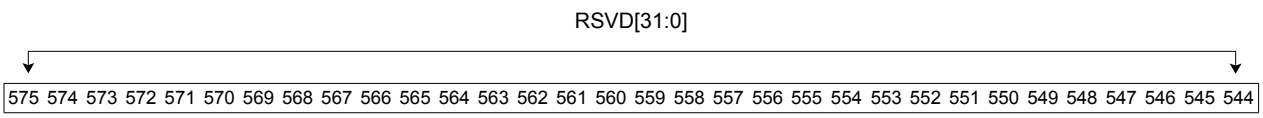
Field Name	Bits	Default	Description
ENC_CMD[3:0]	99:96	0	Encryption/Decryption Command. This field is used to set the encryption/decryption command type. 0100 = Compression + Encryption 0101 = Decompression + Decryption 0111 = Decryption only 1100 = Compression + Encryption + Hash* 1110 = Encryption + Hash* All other values are reserved. *Compression + AES-CBC/HMAC-SHA1 + Hash and Compression + AES-XTS not supported.
RSVD	95:32	0	Reserved.
TWO_KEY	31		Two Key. Indicates whether the Key Manager should fetch one key or two keys for the current command. 0: Fetch one key 1: Fetch two keys for AES-XTS 256/512-bit key
RSVD	30:24	0	Reserved.
KEY_INDEX[23:0]	23:0	0	Key Index. The index of the keys stored in the 9725 Key SRAM. 0 = Use Key SRAM index 0 for the encryption/ decryption command 1 = Use Key SRAM index 1 for the encryption/ decryption command ... 255 = Use Key SRAM index 255 for the encryption/ decryption command All other values are reserved for future expansion.

3.3.3.2 Desc_src0 for Write Key Commands

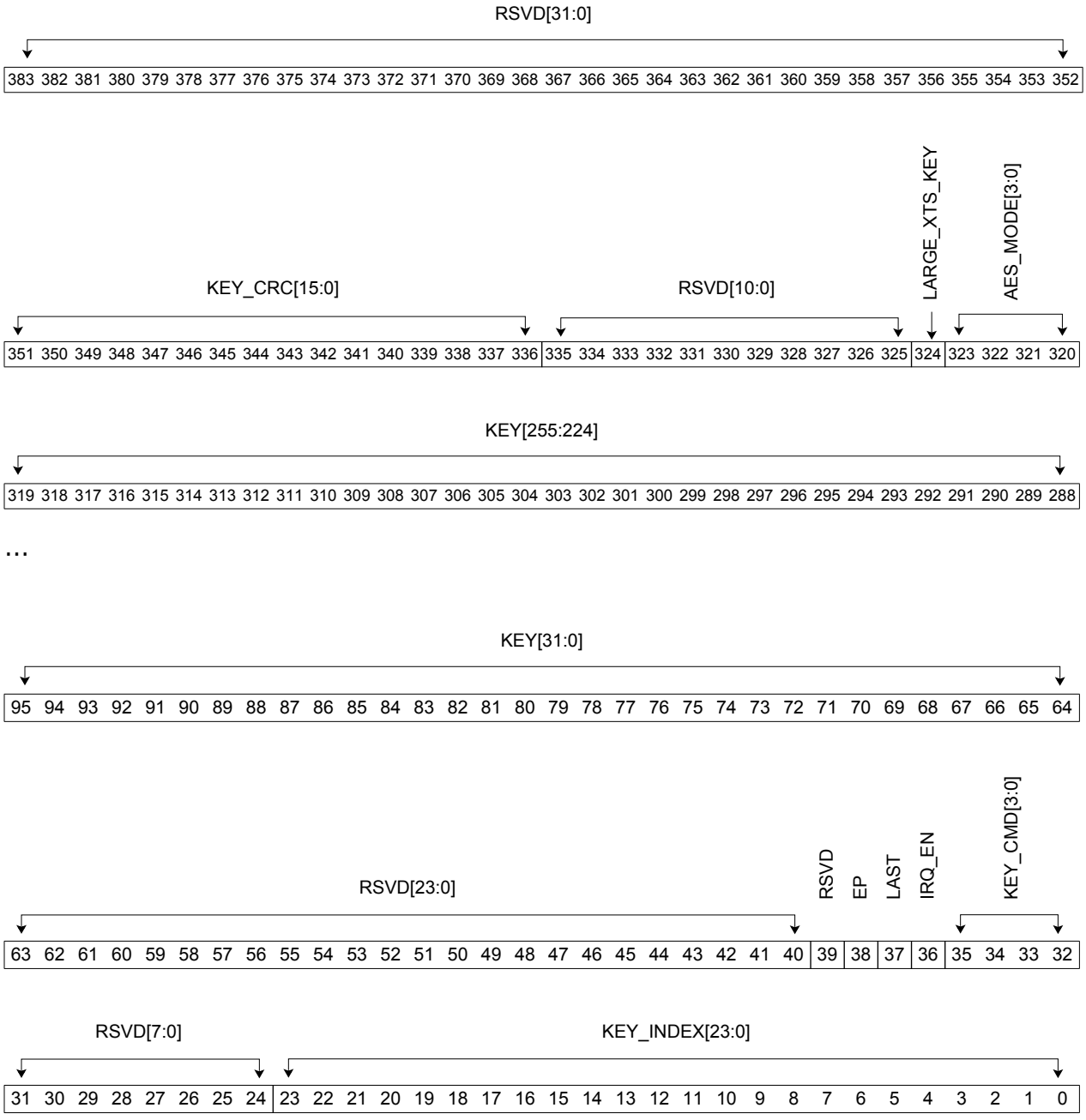
This special descriptor is required when writing a new 256-bit key to the 9725 Key SRAM at a specified index location. No source or destination buffers are needed for this command.

To write a new key to the 9725 Key SRAM, the host must provide the Key SRAM index where the key will be stored, the 256-bit key value, and the AES mode that is used with this key. Note that writing an AES-XTS 512-bit key is a two-step process.

32-bit Addressing Mode Format



...

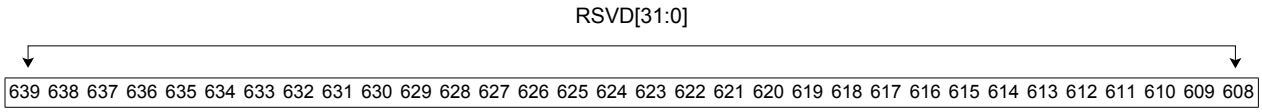


Field Name	Bits	Default	Description
RSVD	575:352	0	Reserved.

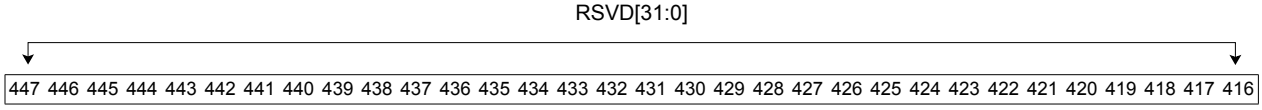
Field Name	Bits	Default	Description
KEY_CRC[15:0]	351:336	0	Key CRC. This field is generated by the host software to validate the key and is stored in the 9725 internal key SRAM with the key. The 9725 will check this value when the key is written to the Key SRAM or when the Key is used by encryption/decryption commands. The Key CRC is based on 256 bits key, 4 bits AES_Mode, 1 bit Large_XTS_Key and 11 bits reserved.
RSVD	335:325	0	Reserved.
LARGE_XTS_KEY	324	0	Large XTS Key. This field is only valid when AES_Mode = 0011. Writing the AES-XTS 512-bit key is a two step process. The host must first write the ECB key with Large_XTS_KEY set to zero and then write the Tweak key with Large_XTS_KEY set to one. 1 = Indicates the current key is the 256-bit Key1 (ECB Key) of the XTS key; the next write key command must be Key2 (Tweak Key) of the XTS key 0 = Indicates the current key is the 256-bit Key2 (Tweak Key) of the XTS key Note: the Key1 and Key2 XTS indexes must be (2N) and (2N + 1).
AES_MODE[3:0]	323:320	0	AES Mode. 0000 = AES-GCM 256-bit key 0001 = AES-XTS 256-bit key 0010 = AES-GCM 128-bit key 0011 = AES-XTS 512-bit key 0100 = AES-CBC 256-bit key 0101 = AES-CBC 256-bit key with HMAC-SHA1 0110 = AES-CBC 128-bit key 0111 = AES-CBC 128-bit key with HMAC-SHA1 All other values are undefined. The 9725 uses key_index[n] to store the AES key, key_index[n+1] to store the HMAC IPAD, and key_index[n+2] to store the HMAC OPAD. The host software must preprocess the HMAC key to generate the IPAD and OPAD. If the 9725 encounters an undefined AES Mode, it will report "unsupported command" interrupt to the host.
KEY[255:0]	319:64	0	Key. The encryption/decryption that will be stored in the 9725 Key SRAM.
RSVD	63:40	0	Reserved.
RSVD	39	0	Reserved.

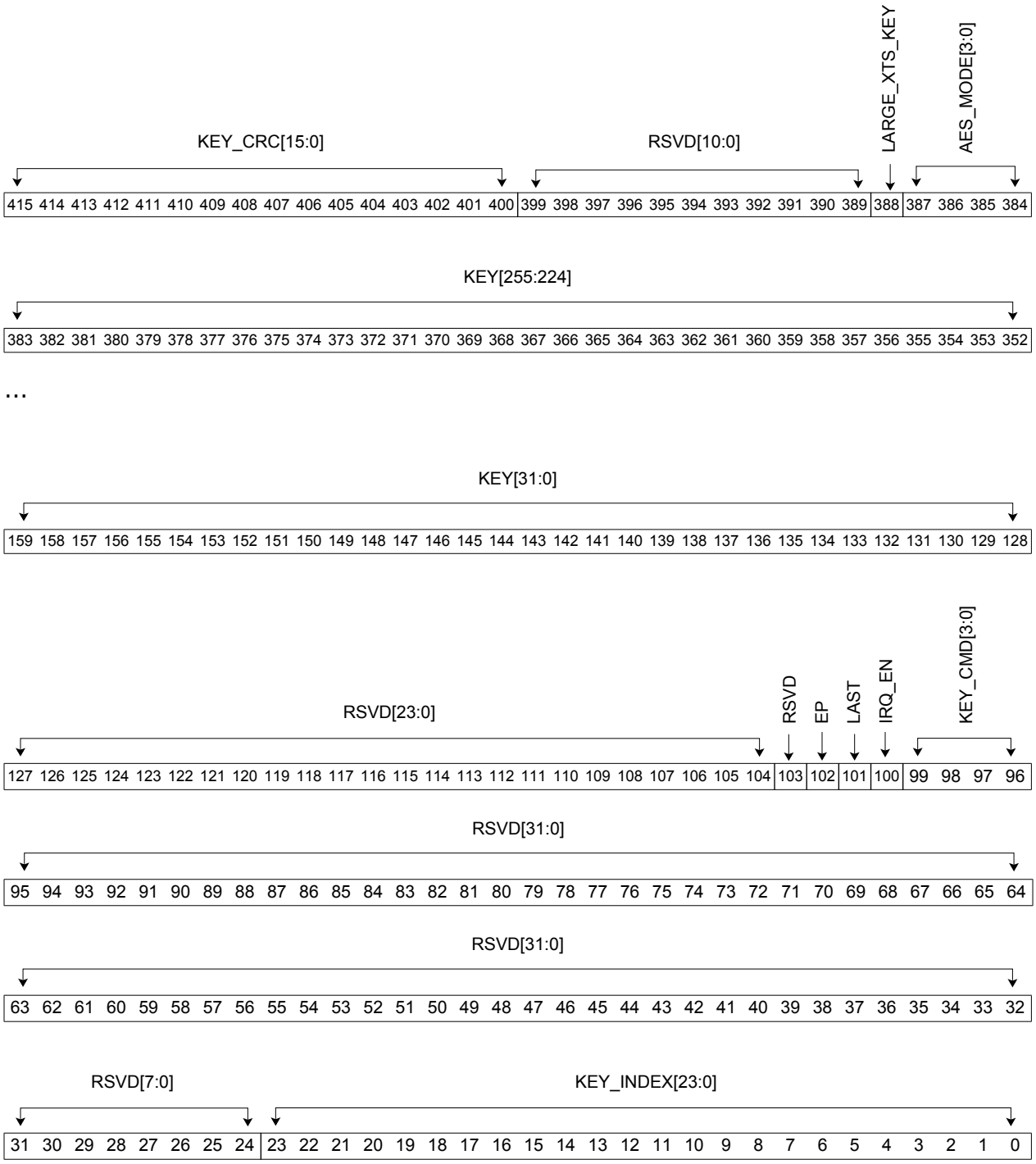
Field Name	Bits	Default	Description
EP	38	0	Endian Pointer This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, "DMA Configuration Register". See Endian Settings for more information.
LAST	37	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.
IRQ_EN	36	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. 0 = Interrupt disabled 1 = Interrupt enabled
KEY_CMD[3:0]	35:32	0	Write Key Command. This field is used to set the command type. 0011 = Write Key All other values are reserved.
RSVD	31:24	0	Reserved.
KEY_INDEX[23:0]	23:0	0	Key Index. The index of the keys stored in the 9725 Key SRAM. 0 = Use Key SRAM index 0 for the encryption/ decryption command 1 = Use Key SRAM index 1 for the encryption/ decryption command ... 255 = Use Key SRAM index 255 for the encryption/ decryption command All other values are reserved for future expansion.

64 bit Mode Addressing Format



...





Field Name	Bits	Default	Description
RSVD	639:416	0	Reserved.

Field Name	Bits	Default	Description
KEY_CRC[15:0]	415:400	0	Key CRC. This field is generated by the host software to validate the key and is stored in the 9725 internal key SRAM with the key. The 9725 will check this value when the key is written to the Key SRAM or when the Key is used by encryption/decryption commands. The Key CRC is based on 256 bits key, 4 bits AES_Mode, 1 bit Large_XTS_Key and 11 bits reserved.
RSVD	399:389	0	Reserved.
LARGE_XTS_KEY	388	0	Large XTS Key. This field is only valid when AES_Mode = 0011. Writing the AES-XTS 512-bit key is a two step process. The host must first write the ECB key with Large_XTS_KEY set to zero and then write the Tweak key with Large_XTS_KEY set to one. 1 = indicates the current key is the 256-bit Key1 (ECB Key) of the XTS key; the next write key command must be Key2 (Tweak Key) of the XTS key 0 = indicates the current key is the 256-bit Key2 (Tweak Key) of the XTS key Note: the Key1 and Key2 XTS indexes must be (2N) and (2N + 1).
AES_MODE[3:0]	387:384	0	AES Mode. 0000 = AES-GCM 256-bit key 0001 = AES-XTS 256-bit key 0010 = AES-GCM 128-bit key 0011 = AES-XTS 512-bit key 0100 = AES-CBC 256-bit key 0101 = AES-CBC 256-bit key with HMAC-SHA1 0110 = AES-CBC 128-bit key 0111 = AES-CBC 128-bit key with HMAC-SHA1 All other values are undefined. The 9725 uses key_index[n] to store the AES key, key_index[n+1] to store the HMAC IPAD, and key_index[n+2] to store the HMAC OPAD. The host software must preprocess the HMAC key to generate the IPAD and OPAD. If the 9725 encounters an undefined AES Mode, it will report "unsupported command" interrupt to host.
KEY[255:0]	383:128	0	Key. The encryption/decryption that will be stored in the 9725 Key SRAM.
RSVD	127:104	0	Reserved.
RSVD	103	0	Reserved.

Field Name	Bits	Default	Description
EP	102	0	Endian Pointer This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, "DMA Configuration Register". See Endian Settings for more information.
LAST	101	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.
IRQ_EN	100	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. 0 = Interrupt disabled 1 = Interrupt enabled
KEY_CMD[3:0]	99:96	0	Write Key Command. This field is used to set the command type. 0011 = Write Key All other values are reserved.
RSVD	95:24	0	Reserved.
KEY_INDEX[23:0]	23:0	0	Key Index. The index of the keys stored in the 9725 Key SRAM. 0 = Use Key SRAM index 0 for the encryption/ decryption command 1 = Use Key SRAM index 1 for the encryption/ decryption command ... 255 = Use Key SRAM index 255 for the encryption/ decryption command All other values are reserved for future expansion.

3.3.3.3 Desc_src0/Desc_src1 for Hash Commands

For hash only commands, Desc_src0 is used to indicate the Hash slice size and the number of bytes of data in the source buffer (HASH_SRC_COUNT). HASH_SRC_COUNT is used to calculate the padding required for the last block of a stateful hash operation.

For the concatenated hash commands (Encryption + Hash) and (Compression + Encryption + Hash), the first source descriptor (Desc_src0) will be used to indicate the encryption key index and sector size in the 9725 Key SRAM and sector size. The second source descriptor (Desc_src1) will be used to indicate the Hash slice size and the number of bytes of data in the source buffer (HASH_SRC_COUNT). Desc_src2 will be the first source buffer descriptor for the source data.

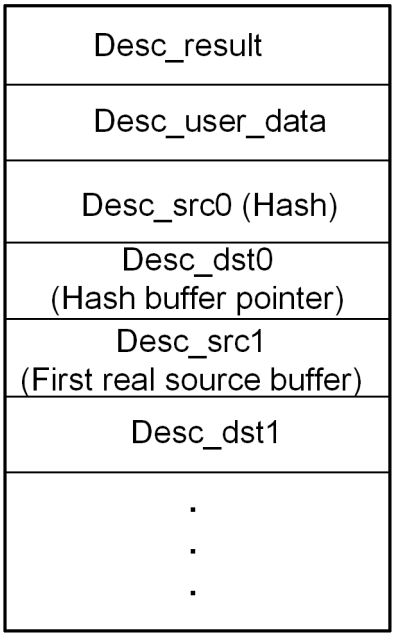


Figure 3-5. Command Structure for Hash only Command

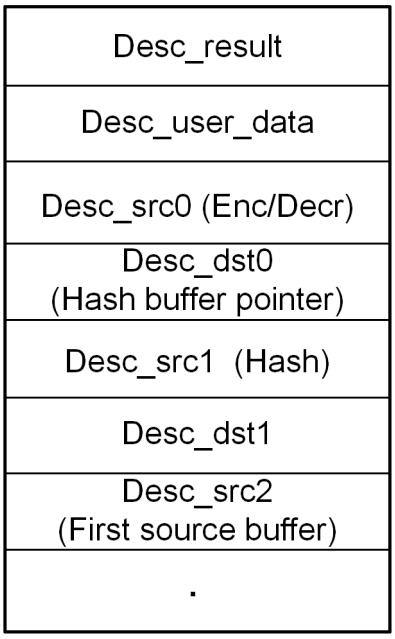
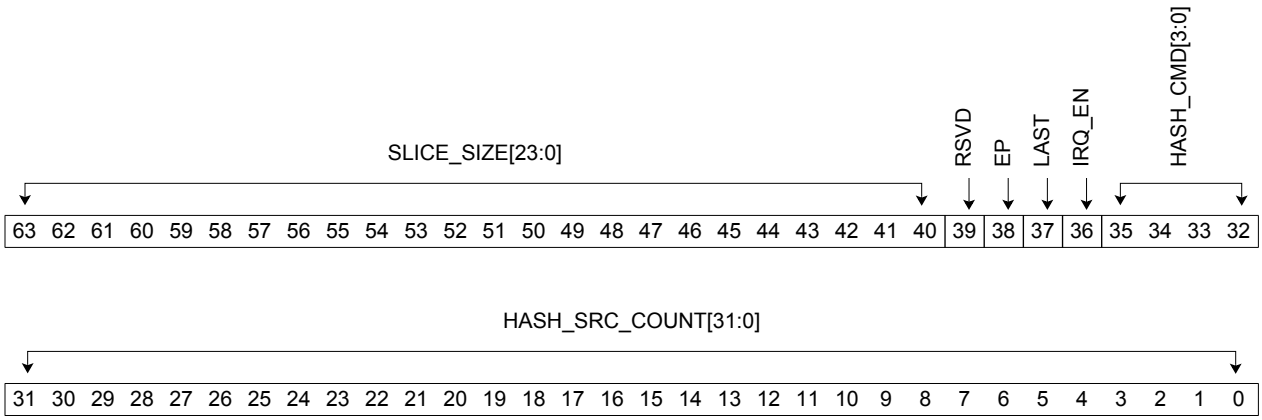


Figure 3-6. Command Structure for Encryption + Hash Command

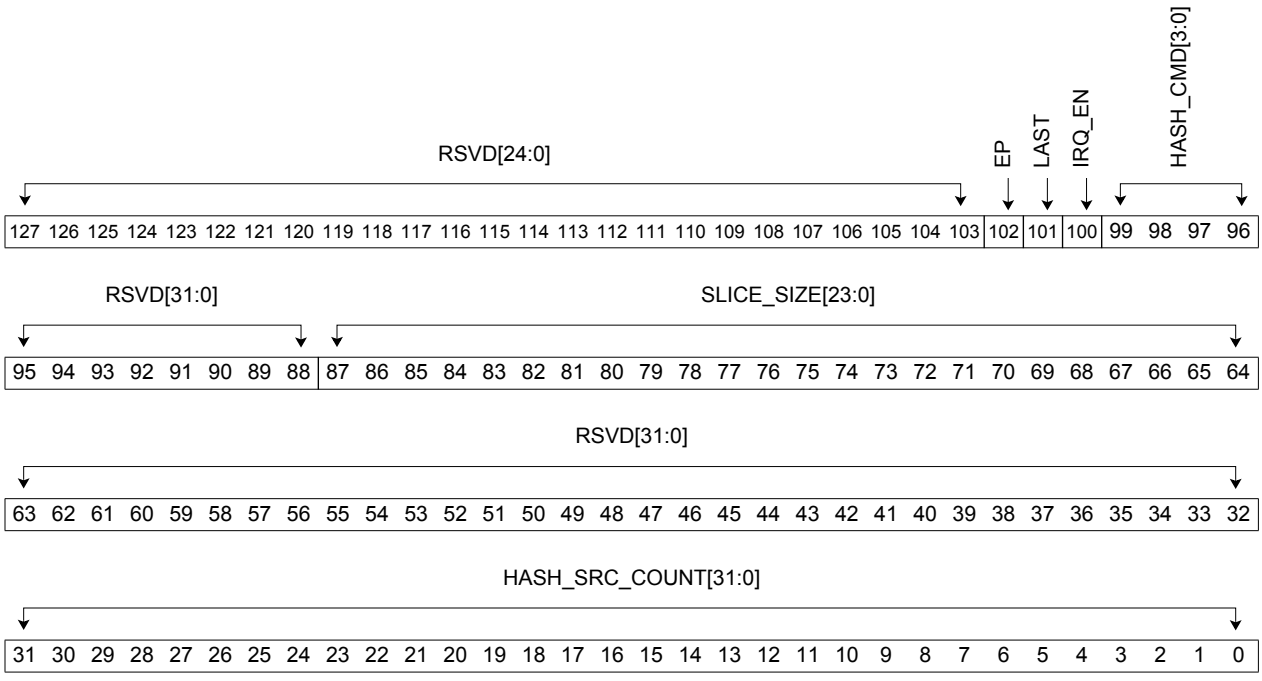
32-bit Addressing Mode Format



Field Name	Bits	Default	Description
SLICE_SIZE[23:0]	63:40	0	Hash Slice Size. This field is the slice size in bytes. The slice size must be on a 512-bit boundary. The 9725 calculates the slice size of the last slice of the command since the last slice may be smaller than the configured slice size. The 9725 will compute the complete hash operation and output the padded result for each data slice.
RSVD	39		Reserved.
EP	38	0	Endian Pointer. This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, "DMA Configuration Register". See Endian Settings for more information.
LAST	37	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.
IRQ_EN	36	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. 0 = Interrupt disabled 1 = Interrupt enabled

Field Name	Bits	Default	Description
HASH_CMD[3:0]	35:32	0	Hash Command. This field is used to set the hash command type. 1000 = Compression + Hash 1010 = Hash only 1100 = Compression + Encryption + Hash* 1110 = Encryption + Hash* Compression + AES-CBC/HMAC-SHA1 + Hash and Compression + AES-XTS are not supported Note that for hash-related commands, the host software must also configure Desc_dst0.
HASH_SRC_COUNT [31:0]	31:0	0	Hash Source Count. The number of bytes of data in the source buffer. This value is used to calculate the padding required for the last block of a stateful hash operation. If the host masks the data in the first source data buffer, then the value of HASH_SRC_COUNT should not include the mask data.

64 bit Mode Addressing Format



Field Name	Bits	Default	Description
RSVD	127:103	0	Reserved

Field Name	Bits	Default	Description
EP	102	0	Endian Pointer. This bit indicates the endian pointer for the source and destination descriptors as defined in the fields EF0 and EF1 in Section 6.2.5, "DMA Configuration Register". See Endian Settings for more information.
LAST	101	0	Last. Used to identify the current descriptor as the last valid descriptor in the command.
IRQ_EN	100	0	Interrupt Enable. This bit allows the 9725 to interrupt the host when a command completes. 0 = Interrupt disabled 1 = Interrupt enabled
HASH_CMD[3:0]	99:96	0	Hash Command. This field is used to set the hash command type. 1000 = Compression + Hash 1010 = Hash only 1100 = Compression + Encryption + Hash* 1110 = Encryption + Hash* All other values are reserved. *Compression + AES-CBC/HMAC-SHA1 + Hash and Compression + AES-XTS are not supported Note that for hash-related commands, the host software must also configure Desc_dst0.
RSVD	95:88	0	Reserved.
SLICE_SIZE[23:0]	87:64	0	Hash Slice Size. This field is the slice size in bytes. The slice size must be on a 512-bit boundary. The 9725 calculates the slice size of the last slice of the command since the last slice may be smaller than the configured slice size. The 9725 will compute the complete hash operation and output the padded result for each data slice.
RSVD	63:32	0	Reserved.
HASH_SRC_COUNT [31:0]	31:0	0	Hash Source Count. The number of bytes of data in the source buffer. This value is used to calculate the padding required for the last block of a stateful hash operation. If the host masks the data in the first source data buffer, then the value of HASH_SRC_COUNT should not include the mask data.

3.3.3.4 Desc_dst0 for Hash Commands

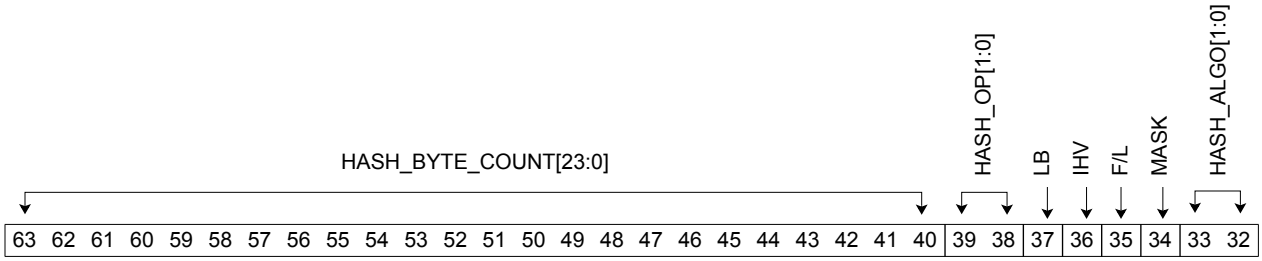
This special descriptor is used to configure the hash algorithm and point to the hash buffer which is used to store the hash result. If used, the second destination descriptor (Desc_dst1) will be used to store the result data.

The host must indicate the last block of data so that the 9725 will pad the last hash result out to 512-bits.

The host may set an Initial Hash Value (IHV) if desired. The IHV should be inserted into the first hash source buffer and the IHV bit should be set.

The host may also mask the first or last N bytes of the source data to be useful in deduplication. The value of N is the size in bytes of the source buffer to be masked.

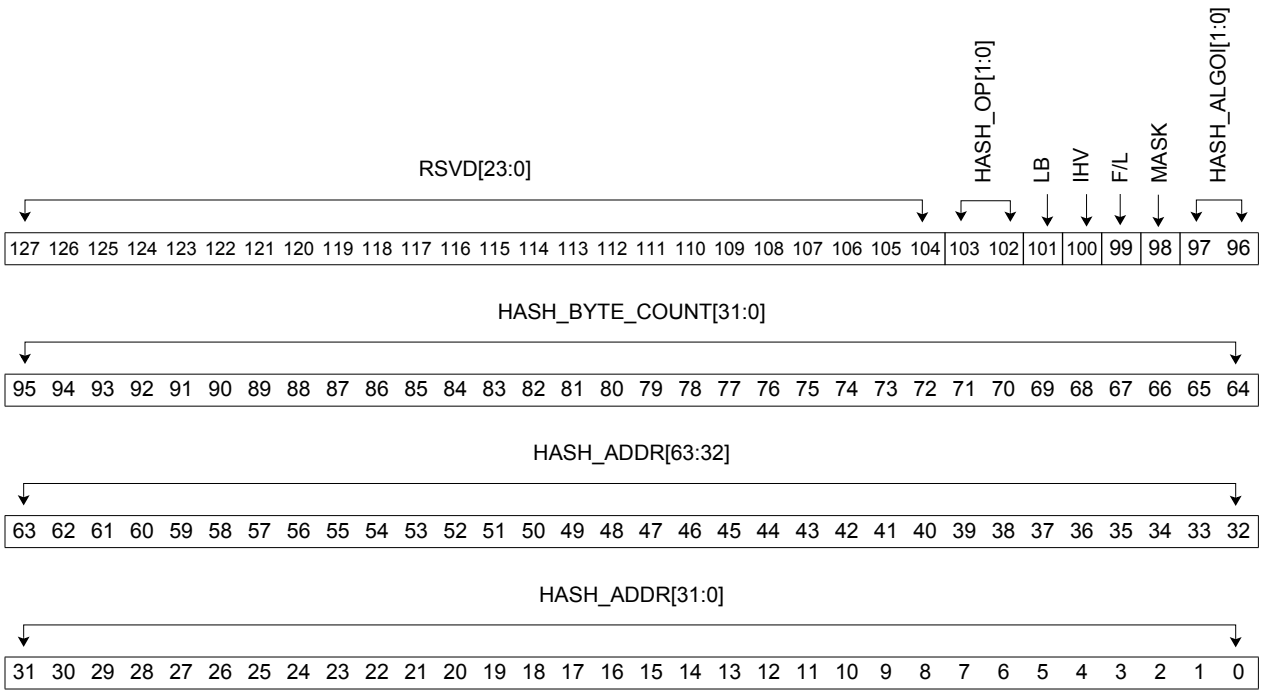
32-bit Addressing Mode Format



Field Name	Bits	Default	Description
HASH_BYTE_COUNT [23:0]	63:40	0	Hash Buffer Count. This field indicates the hash buffer size. The value must fall on a 256-bit boundary and be large enough to store the entire slice hash.
HASH_OP[1:0]	39:38	0	Hash Operation. This field is only valid for the hash operations with the special Desc_dst0. 00 = slice hash only with internal integrity verification 01 = file hash only with internal integrity verification 10 = slice hash + file hash with no internal integrity verification 11 = Reserved When calculating the file hash, the host must wait for one file hash command to finish before proceeding to update the file hash on the next block of the file.

Field Name	Bits	Default	Description
LB	37	0	Last Block. This bit is only valid for the special hash Desc_dst0 descriptor. For stateful hash operation, each command occupies one block of data. This bit indicates the last block for file hash operation. 1 = This block is the last block of source data. The source data may be an arbitrary number of bytes because the HASH engine will pad the result to fill 512 bits. 0 = This block is not the last block in the source data. The source data length must be aligned to a 512-bit boundary because the HASH engine will not pad the result.
IHV	36	0	Initial Hash Value. Only valid in Desc_dst0 for hash related operation If an IHV is configured, the 9725 Hash engine will load this value before making the hash calculation. The IHV should be the first hash source buffer. For SHA-1, SHA-256 and MD5, the IHV is 256 bits. 1 = An IHV exists and should be used 0 = Use the default initial hash chaining values If the command is encrypt + hash, the first source buffer should include IHV (Initial Hash Value), IV (Initial Value for encryption/decryption engine) and AAD (Additional Authenticated Data for AES-GCM), and this source buffer should 8-byte aligned. No data should be put into this source buffer.
F/L	35	0	First/Last. 0 = Indicates that the 9725 will mask data in the first source buffer for a hash operation 1 = Indicates that the 9725 will mask data in the last source buffer for a hash operation The maximum mask byte count is 16 MBytes.
MASK	34	0	Mask. 0 = Indicates that the 9725 will not mask data for the hash operation 1 = Indicates that the 9725 should mask data for the hash operation If the first source buffer is used for storing the IHV/IV/AAD, the data in the second source buffer will be masked.
HASH_ALGO[1:0]	33:32	0	Hash Algorithm Select. 00 = SHA-1 01 = SHA-256 10 = MD5 11 = Reserved
HASH_ADDR[31:0]	31:0	0	Hash Buffer Address. This field defines the hash buffer starting address and must be 32-byte aligned.

64 bit Mode Addressing Format



Field Name	Bits	Default	Description
RSVD	127:104	0	Reserved
HASH_OP[1:0]	103:102	0	Hash Operation. This field is only valid for the hash operations with the special Desc_dst0. 00 = slice hash only with internal integrity verification 01 = file hash only with internal integrity verification 10 = slice hash + file hash with no internal integrity verification 11 = Reserved When calculating the file hash, the host must wait for one file hash command to finish before proceeding to update the file hash on the next block of the file.

Field Name	Bits	Default	Description
LB	101	0	Last Block. Data of one command is one block for stateful hash operation. This bit is only valid for the special hash Desc_dst0 descriptor and indicates the last block for file hash operation. 1 = This block is the last block of source data. The source data may be an arbitrary number of bytes because the HASH engine will pad the result to fill 512 bits. 0 = This block is not the last block in the source data. The source data length must be aligned to a 512-bit boundary because the HASH engine will not pad the result.
IHV	100	0	Initial Hash Value. Only valid in Desc_dst0 for hash related operation If an IHV is configured, the 9725 Hash engine will load this value before making the hash calculation. The IHV should be the first hash source buffer. For SHA-1, SHA-256 and MD5, the IHV is 256 bits. 1 = An IHV exists and should be used 0 = Use the default initial hash chaining values If the command is encrypt + hash, the first source buffer should include IHV (Initial Hash Value), IV (Initial Value for encryption/decryption engine) and AAD (Additional Authenticated Data for AES-GCM), and this source buffer should 8-byte aligned. No data should be put into this source buffer.
F/L	99	0	First/Last. 0 = Indicates that the 9725 will mask data in the first source buffer for a hash operation 1 = Indicates that the 9725 will mask data in the last source buffer for a hash operation The maximum mask byte count is 16 MBytes.
MASK	98	0	Mask. 0 = Indicates that the 9725 will not mask data for the hash operation 1 = Indicates that the 9725 should mask data for the hash operation If the first source buffer is used for storing the IHV/IV/AAD, the data in the second source buffer will be masked.
HASH_ALGO[1:0]	97:96	0	Hash Algorithm Select. 00 = SHA-1 01 = SHA-256 10 = MD5 11 = Reserved

Field Name	Bits	Default	Description
HASH_BYTE_COUNT [31:0]	95:64	0	Hash Buffer Count. This field indicates the hash buffer size. The value must fall on a 256-bit boundary and be large enough to store the entire slice hash.
HASH_ADDR[63:0]	63:0	0	Hash Buffer Address. This field defines the hash buffer starting address and must be 32-byte aligned.

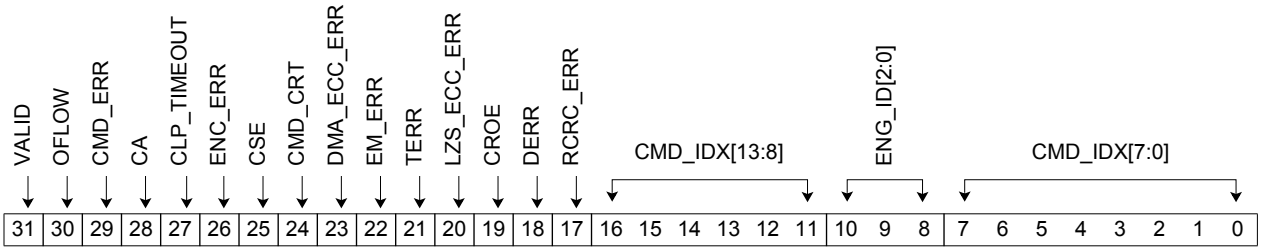
3.4 Result Ring

The result ring contains a completed command's resultant data pointers and execution status (successful completion or errors occurred). The 9725 writes to the result ring and the host reads from the result ring. The 9725 will write to the result ring when a command completes and then update its result ring write pointer appropriately.

The size of the result ring is always identical to the size of command pointer ring. The result ring must be aligned to an 8-byte boundary and reside entirely in a physically contiguous range of memory.

The 9725 supports both 32-bit and 64-bit result rings.

32-bit Result Ring



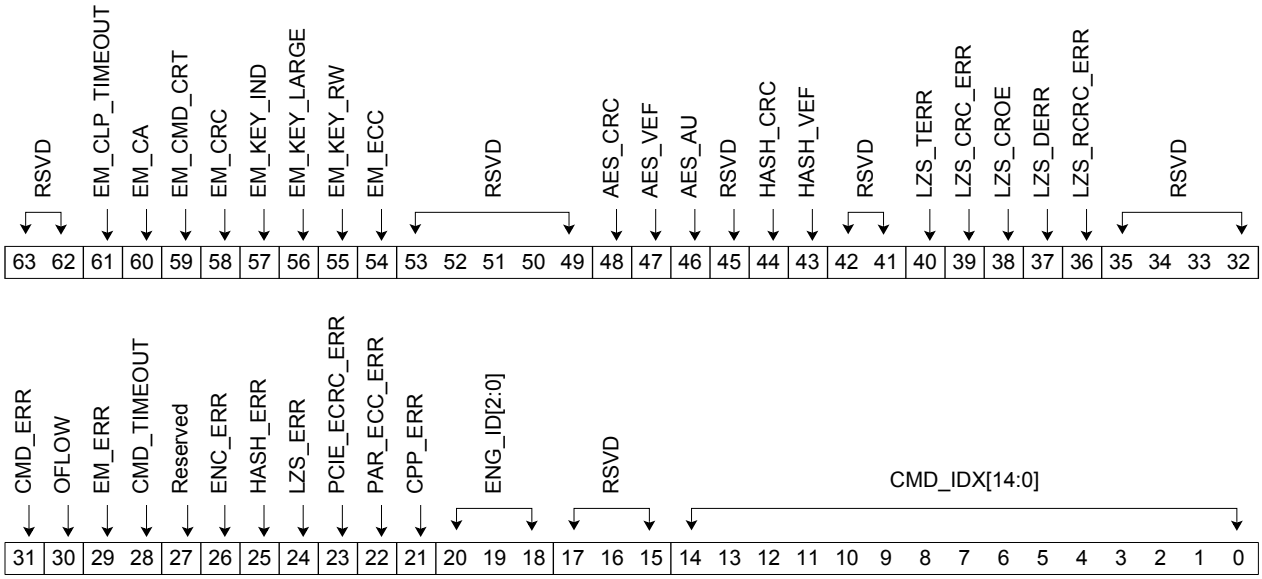
Field Name	Bits	Default	Description
VALID	31	0	Valid bit. This bit indicates there are valid entries in the result ring. This bit toggles each time the write pointer has wrapped the result ring.
OFLOW	30	0	Overflow. This bit indicates if an overflow error occurred in the destination data buffer. 0 No overflow error occurred for this command 1 Overflow error occurred for this command

Field Name	Bits	Default	Description
CMD_ERR	29	0	Command Error. This bit summarizes the command error status. It will be set if any of the error bits, excluding the overflow error bit, for this entry are set. 0 No error occurred for this command 1 Error occurred for this command
CA	28	0	Host Completion Abort. This will be set if the host could not respond to a read request issued by the 9725. The 9725 will terminate the command associated with the completion packet marked with a completion abort error. 0 No Host Completion Abort error occurred 1 Host Completion Abort Error occurred
CLP_TIMEOUT	27	0	Completion Timeout. This bit is set if the completion packet for an outstanding read request does not return to the 9725 within the timeout window valued defined in the PCIe specification. 0 No Completion Timeout error occurred 1 Completion Timeout Error occurred
ENC_ERR	26	0	Encryption Engine Error. This bit indicates if an error occurred in the Encryption engine. 0 No Encryption engine error occurred for this command 1 Encryption engine error occurred for this command
CSE	25	0	Host Completion Sequence Error. This bit will be set if there are multiple completion packets for one outstanding read request, and the packets returned out of order. 0 No Host Completion sequence error occurred 1 Host Completion sequence error occurred
CMD_CRT	24	0	Command Corruption Error. This bit will be set if the compression engine did not assert EOR (end of record), or output data after asserting EOR (multiple record error). 0 No Compression engine error occurred for this command 1 Compression engine error occurred for this command
DMA_ECC_ERR	23	0	DMA ECC/Parity Error. This bit will be set if the DMA detects an ECC or parity error. 0 No ECC/Parity error occurred 1 ECC/Parity error occurred

Field Name	Bits	Default	Description
EM_ERR	22	0	This bit summarizes the error status. It will be set if any of the error bits are set. 0 No error have occurred 1 Error has occurred
TERR	21	0	Compression Engine Token Error. 0 No Token errors detected 1 Token Error occurred
LZS_CRC_ERR	20	0	Compression Engine CRC Error. 0 No Compression Engine CRC errors detected 1 Compression Engine CRC Error occurred
CROE	19	0	Compression Engine Command Result Overrun Error. 0 No Parity errors detected 1 Parity Error occurred
DERR	18		Compression Engine Data Error. 0 No Data errors detected 1 Data Error occurred
RCRC_ERR	17		Record CRC Error. 0 No Record CRC errors detected 1 Record CRC Error occurred
CMD_IDX[13:8]	16:11	0	Command Index. See description for bits [7:0] below.
ENG_ID[2:0]	10:8	0	Engine Manager Identifier. This bit identifies which Engine Manager reported the error. 000 = Engine Manager 0 reported the error 001 = Engine Manager 1 reported the error 010 = Engine Manager 2 reported the error 011 = Engine Manager 3 reported the error 100 = Engine Manager 4 reported the error 101 = Engine Manager 5 reported the error All other combinations reserved.
CMD_IDX[7:0]	7:0	0	Command Index. The command index field contains the index number of the completed command in the result ring.

64-bit Result Ring

For a 64-bit result ring, bits [31:17] summarize the source of the error, while bits [63:32] provide a more detailed identification of the error.



Field Name	Bits	Default	Description
RSVD	63:62	0	Reserved.
EM_CLP_TIMEOUT	61	0	Engine Manager Completion Timeout error. This bit will be set if completion packets of outstanding read requests do not return to the 9725 within the timeout window valued defined in the PCIe specification. 0 Engine Manager did not detect Completion Timeout error 1 Engine Manager detected Completion Timeout error
EM_CA	60	0	Engine Manager Host Completion Abort. This bit will be set if the host could not respond to a read request issued by the 9725. The 9725 will terminate the command associated with the completion packet marked with a completion abort error. 0 Engine Manager did not detect Host Completion Abort error 1 Engine Manager detected Host Completion Abort error
EM_CMD_CRT	59	0	Engine Manager Command Corruption Error. This bit will be set if the Engine manager detects that the Compression engine did not assert EOR (end of record) or output data after asserting EOR (multiple record error). 0 Engine Manager did not detect Command Corruption error 1 Engine Manager detected Command Corruption error

Field Name	Bits	Default	Description
EM_CRC	58	0	Engine Manager CRC Error. This bit will be set if the Engine manager detects a CRC error. 0 Engine Manager did not detect CRC error 1 Engine Manager detected CRC error
EM_KEY_IND	57	0	Engine Manager Key Index Error. This bit will be set if the Engine manager detected that the command key index exceeded the key index Mask. 0 Engine Manager did not detect Key Index error 1 Engine Manager detected Key Index error
EM_KEY_LARGE	56	0	Engine Manager Large Key Error. This bit will be set if the Engine manager detected that the host sent an illegal key index command when using large key. 0 Engine Manager did not detect Large Key error 1 Engine Manager detected Large Key error
EM_KEY_RW	55	0	Engine Manager Key Read/Write Error. This bit will be set if the Engine manager detected a write/read key CRC/ECC error. 0 Engine Manager did not detect Key Read/Write error 1 Engine Manager detected Key Read/Write error
EM_ECC	54	0	Engine Manager ECC/Parity Error. This bit will be set if the Engine manager detected an ECC/Parity error. 0 Engine Manager did not detect ECC/Parity error 1 Engine Manager detected ECC/Parity error
RSVD	53:49	0	Reserved.
AES_CRC	48	0	Encryption Engine CRC Error. This bit will be set if the encryption engine detects an error when verifying the CRC. 0 Encryption engine did not detect CRC error 1 Encryption engine detected CRC error
AES_VEF	47	0	Encryption Engine Real time Verification Error. This bit will be set if the encryption engine detects an encryption error. 0 Encryption engine did not detect Verification error 1 Encryption engine detected Verification error

Field Name	Bits	Default	Description
AES_AU	46	0	Encryption Engine Authentication Error. This bit will be set if the encryption engine detects an authentication error in AES-GCM mode. 0 Encryption engine did not detect Authentication error 1 Encryption engine detected Authentication error
RSVD	45	0	Reserved.
HASH_CRC	44	0	Hash Engine CRC Error. This bit will be set if the hash engine detects an error when verifying the CRC that was set by the Engine Manager. 0 Hash engine did not detect CRC error 1 Hash engine detected CRC error
HASH_VEF	43	0	Hash Engine Real time Verification Error. This bit will be set if the Hash engine detects an encryption error. 0 Hash engine did not detect Verification error 1 Hash engine detected Verification error
RSVD	42:41	0	Reserved.
LZS_TERR	40	0	Compression Engine Token Error. 0 Compression Engine did not detect token error 1 Compression Engine detected token error
LZS_CRC_ERR	39	0	Compression Engine CRC Error. 0 Compression Engine did not detect CRC error 1 Compression Engine detected CRC error
LZS_CROE	38	0	Compression Engine Command/Result Overrun Error. 0 Compression engine did not detect Command/Result Overrun error 1 Compression engine detected Command/Result Overrun error
LZS_DERR	37	0	Compression Engine Data Error. 0 Compression engine did not detect data error 1 Compression engine detected data error
LZS_RCRC_ERR	36	0	Compression Engine RCRC Error. 0 Compression engine did not detect RCRC error in decode operation 1 Compression engine detected RCRC error in decode operation
RSVD	34:32	0	Reserved.

Field Name	Bits	Default	Description
CMD_ERR	31	0	Command Error. This bit summarizes the command error status. It will be set if any of the error bits, excluding the overflow error bit, for this entry are set. 0 No error occurred for this command 1 Error occurred for this command
OFLOW	30	0	Overflow. This bit indicates if an overflow error occurred in the destination data buffer. 0 No overflow error occurred for this command 1 Overflow error occurred for this command
EM_ERR	29	0	Engine Manager Error Summary. This bit will be set if any error bit are detected by the Engine manager. 0 No errors detected by the Engine Manager 1 Error detected by the Engine Manager
CMD_TIMEOUT	28	0	Command Completion Timeout. This bit is set if the completion packet for an outstanding read request does not return to the 9725 within the timeout window valued defined in the PCIe specification. 0 No Command Completion Timeout error occurred 1 Command Completion Timeout Error occurred
RSVD	27	0	Reserved.
ENC_ERR	26	0	Encryption Engine Error. This bit indicates if an error occurred in the Encryption engine. 0 No Encryption engine error occurred for this command 1 Encryption engine error occurred for this command
HASH_ERR	25	0	Hash Engine Error. This bit indicates if an error occurred in the Hash engine. 0 No Hash engine error occurred for this command 1 Hash engine error occurred for this command
LZS_ERR	24	0	Compression Engine Error. This bit indicates if an error occurred in the Compression engine. 0 No Compression engine error occurred for this command 1 Compression engine error occurred for this command

Field Name	Bits	Default	Description
PCIE_ECRC_ERR	23	0	PCIe ECRC Error. This bit will be set if the PCIe core detects an ECRC error. 0 No PCIe ECRC error occurred 1 PCIe ECRC error occurred
PAR_ECC_ERR	22	0	PIM/POM/PCIE Parity or ECC error. This bit will be set if either the PIM, POM, or PCIe core detects a parity or ECC error. The 9725 must be reset after this unrecoverable error. 0 No PCIE Parity or ECC error occurred 1 PCIE Parity or ECC error occurred
CPP_ERR	21	0	Command Pre-Processor Detects Error. This bit will be set if the CPP detects a Completion abort or timeout in indirect addressing mode. The host should reset the 9725 because the command pointer will be lost. 0 CPP did not detect an error 1 CPP detected an error
ENG_ID[2:0]	20:18	0	Engine Manager Identifier. This bit identifies which Engine Manager reported the error. 000 = Engine Manager 0 reported the error 001 = Engine Manager 1 reported the error 010 = Engine Manager 2 reported the error 011 = Engine Manager 3 reported the error 100 = Engine Manager 4 reported the error 101 = Engine Manager 5 reported the error All other combinations reserved.
RSVD	17:15	0	Reserved.

Field Name	Bits	Default	Description
CMD_IDX	14:0	0	Command Index. The command index field contains a Valid bit and the index number of the completed command in the result ring. The bit position of the Valid bit depends on the command ring size, as described below. The Valid bit toggles each time the write pointer wraps the result ring. For a command ring size of 16K: bit 14 Valid bit if command ring size is 16K bit 13:0 Command index of completed command For a command ring size of 8K: bit 13 Valid bit if command ring size is 8K bit 12:0 Command index of completed command For a command ring size of 4K: bit 12 Valid bit if command ring size is 4K bit 11:0 Command index of completed command A similar pattern can be applied for command ring sizes < 4.

The Valid bit in the CMD_IDX[14:0] field informs the Express DR SDK that the corresponding command has finished and indicates that there are valid entries in the result ring. The value of the Valid bit toggles every time the result ring is completely used, i.e., the 9725 write pointer has wrapped the ring. For example, for the first pass through the result ring the value of the Valid bit indicating valid entries is one. After the 9725's write pointer wraps around the result ring, the Valid bit will be zero, and so on for every complete pass through the result ring. During initialization, the Express DR SDK will write a zero to the Valid bit. This simple mechanism is sufficient to allow the Express DR SDK to differentiate unambiguously whether an entry in the result ring has been updated.

3.5 Command /Result Ring Example

Figure 3-7 shows an example of how the result ring and command pointer ring are used to process a command using indirect addressing. To simplify the illustration, this example shows a command pointer ring only with 4 entries, but the actual minimum command pointer ring size is 16 entries.

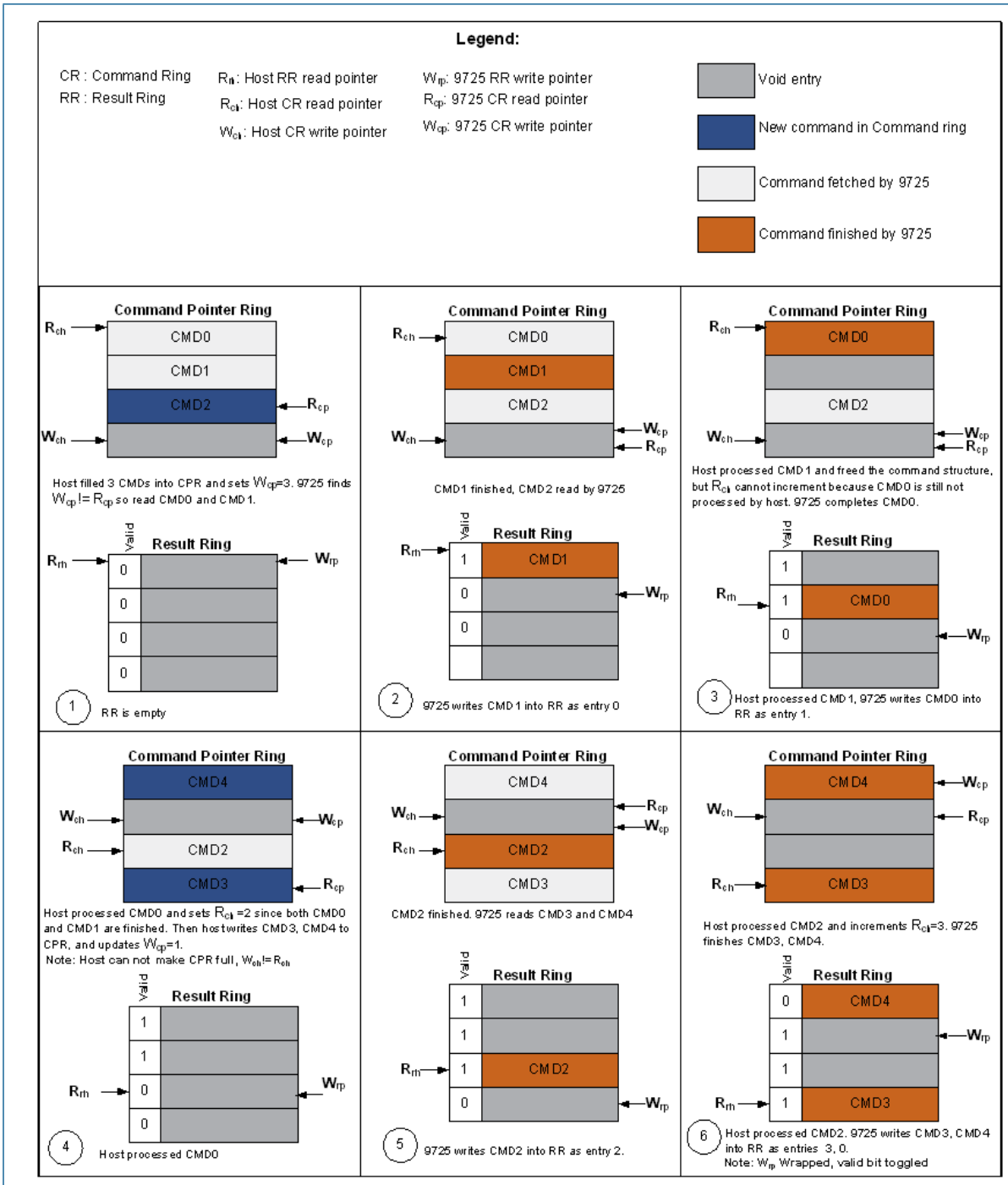


Figure 3-7. Result Ring Example

3.6 Dual Command/Result Rings

Typically the host would maintain only one command ring or one command pointer ring, but the 9725 also supports a dual command ring mode. In this mode, the host maintains two command rings and two result rings in host memory. The 9725 fetches commands from both command rings using a round-robin scheme where both command rings have the same priority. The two command pointer rings and result rings have the same structures and configuration as shown in Figure 3-8.

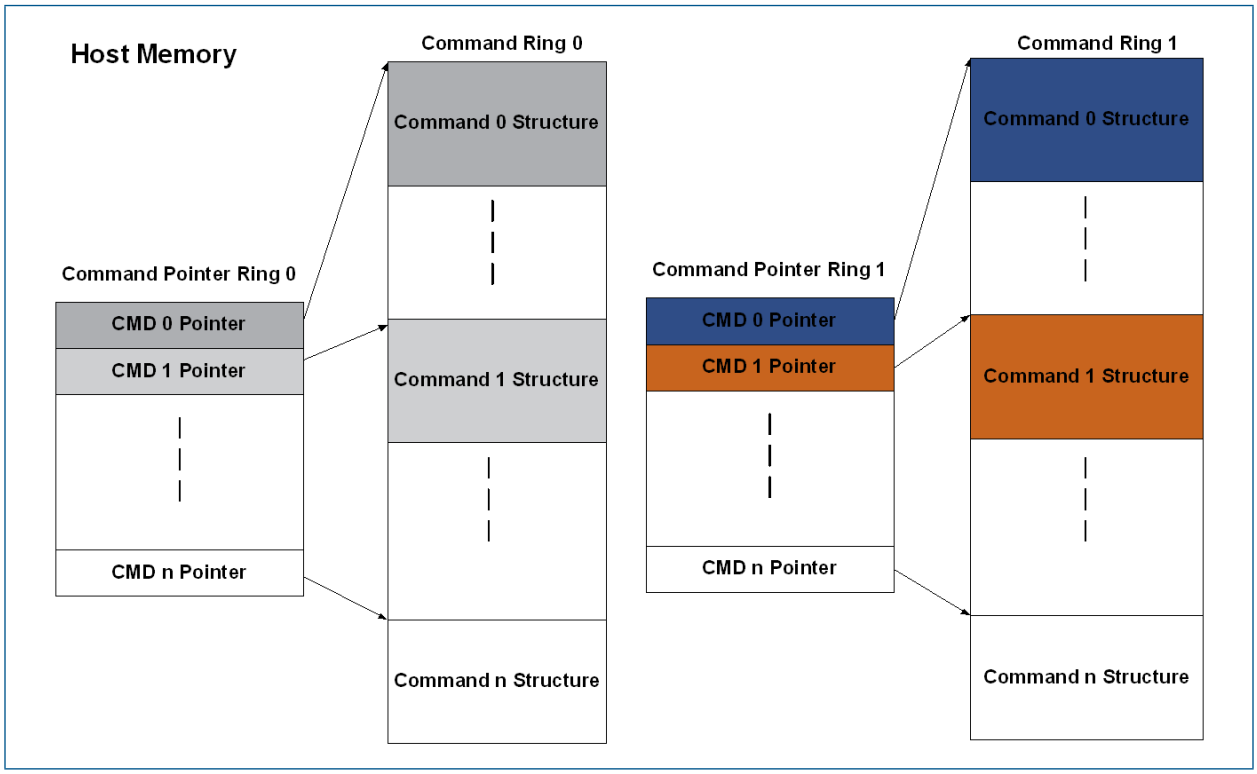


Figure 3-8. Dual Command Ring Indirect Mode

4 Modules

This chapter provides a more detailed description of some of the key 9725 internal modules: DMA, Engine Managers, Key Manager, Compression engine, Encryption engine, Hash engine, and the clock generator.

Figure 4-1 shows a detailed block diagram of the 9725. Each module is described in Table 4-1.

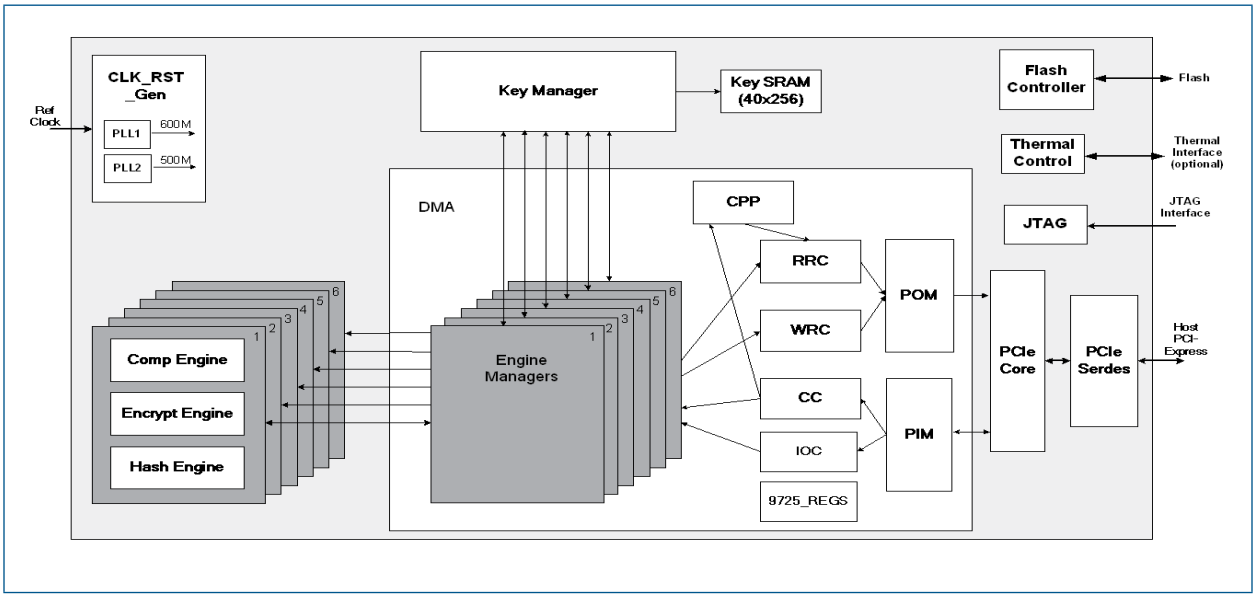


Figure 4-1. 9725 Detailed Block Diagram

Table 4-1. Description of 9725 Modules (Sheet 1 of 2)

Module	Description
9725_REGS	9725 Control Registers Internal registers that control the 9725 device.
CC	Completion Controller Receives the completion data from the PCIe Inbound Manager, and then sends the data to the proper Engine Manager.
CLK_RST_Gen	Clock and Reset Generation Generates the clock and reset signals for all modules.
Comp Engine	Compression/Decompression Engine Compresses/Decompresses the data stream using the LZS or enhanced LZS (eLZS) algorithm. There are six Compression Engines in the 9725.
CPP	Command Pre-Processor Prefetches the command pointer from the command ring.
Encrypt Engine	Encryption Engine Encrypts/Decrypts the data stream using the AES algorithm. There are six Encryption Engines in the 9725.

Table 4-1. Description of 9725 Modules (Sheet 2 of 2)

Module	Description
Engine Manager	Engine Manager Controls all command processing: fetching command structures and data from host memory into the source buffer, transmitting results from the result buffer to host memory. There are six Engine Managers; one for each set of Compression, Encryption and Hash Engines.
Flash Controller	The Flash controller connects the 9725 to an external flash device.
Hash Engine	Hash Engine Calculates the hash value of the data stream using SHA-1, SHA-256 or MD5. There are six Hash Engines in the 9725.
IOC	I/O Access Controller Receives the read/write requests from the PIM module and dispatch the request to the appropriate Engine manager.
JTAG	JTAG Test Function
Key Manager	Key Manager Arbitrates encryption key access to the internal SRAM.
Key SRAM	Key SRAM Internal SRAM for storing the encryption keys.
PCIe Core	PCI express end point controller The PCIe core includes the Memory mapped 9725 PCIe configuration and control registers
PCIe Serdes	PCI express physical layer module
PIM	PCIe Inbound Manager Receives the PCIe completion from the PCIe Core, and sends/receives memory read/memory write to/from the PCIe Core.
POM	PCIe Outbound Manager Sends the PCIe write data and read requests to the PCIe Core.
RRC	Read Request Controller Arbitrates read requests from the CPP, Engine Managers, and sends read requests to the PCIe Outbound Manager.
WRC	Write Request Controller Arbitrates the write requests from the Engine Managers, and sends the write requests to the PCIe Outbound Manager.

4.1 DMA

4.1.1 PCIe Outbound Manager

The PCIe Outbound Manager (POM) provides a transmit interface between the PCIe Core and DMA modules. The main functions of the POM are:

- Send write requests to the PCIe Core transmit interface
- Send read requests to the PCIe Core transmit interface
- Convert output data to the correct endian format

4.1.2 PCIe Inbound Manager

The PCIe Inbound Manager (PIM) provides a receive interface between the PCIe Core and DMA modules. The main functions of the PIM are:

- Decode the completion PCIe TLP (Transaction Layer Packet) from the PCIe Core completion receive interface, and send the completion data to Completion Controller
- Decode memory writes and memory reads from the PCIe Core ELBI interface (a local bus of the PCIe Core for reading or writing application-owned register space), and send write and read requests to the 9725 Control Registers or IO Access Controller module
- Send interrupts to the host through the PCIe Core interface
- Convert input data to the correct endian format

4.1.3 Command Pre-Processor

The Command Pre-Processor (CPP) module is used to prefetch command pointers from the host command pointer ring. The CPP implements a 128 byte Command Pointer Ring (CPR) buffer which can accommodate 16 command pointers in 64-bit addressing mode (8 bytes per pointer) and 32 command pointers in 32-bits addressing mode (4 bytes per pointer).

When the Command Pointer Ring buffer is half empty and the CPR Read Pointer is not equal to the CPR write pointer, the CPP module sends a read request to the Read Request controller (RRC, discussed in Section 4.1.4 below) to prefetch the maximum command pointer ring entries from host memory (8 entries for 64-bit addressing mode or 16 entries for 32-bit addressing mode). This CPP request has the highest priority in the RRC.

Figure 4-2 illustrates how the CPP pre-fetches 8 command pointers using 64-bit addressing. Initially, the host software writes 9 command pointers into the command pointer ring and updates the CPR write pointer for the 9725. At this point, the 9725 CPR read pointer is zero because it has not read any of the commands. Once the CPP discovers that the CPR buffer is empty and CPR Read Pointer is not equal to the CPR write pointer, the 9725 will fetch the 8 CPR entries and store them into the CPR buffer.

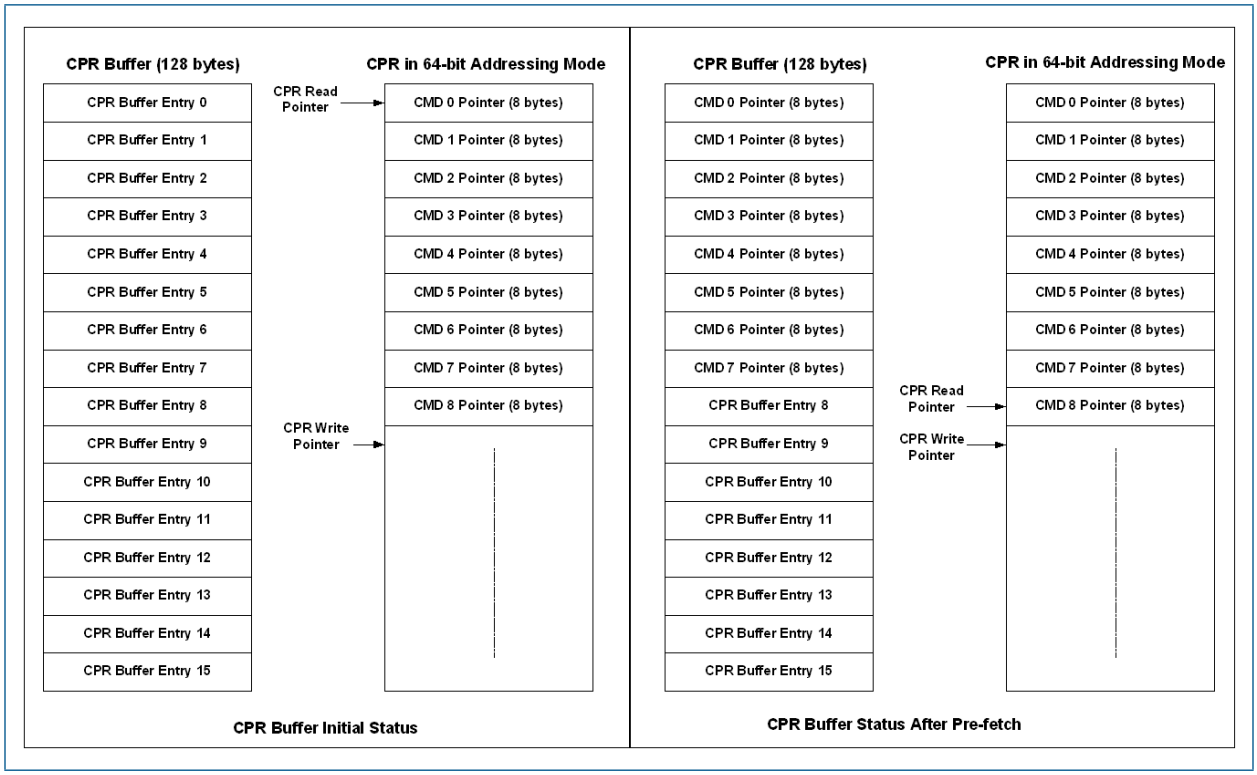


Figure 4-2. DMA Command Pointer Prefetch Example

Note: If dual command ring mode is enabled, the CPP will maintain two CPR buffers. The arbitration of the two CPR buffer read requests uses a round-robin scheme.

4.1.4 Read Request Controller

The Read Request controller (RRC) arbitrates the read requests from the Engine Managers and the CPP, builds the TLP, and sends a read request to the POM. The main functions of the RRC are:

- Store the source descriptor starting address and byte count
- Split the source descriptor into TLP read requests and send requests to the POM
- Arbitrate command processing requests from the dual command ring using a round-robin scheme
- Arbitrate read data requests from the Engine Managers

4.1.5 Write Request Controller

The Write Request controller (WRC) arbitrates the write requests from the Engine Managers, builds the TLP, and sends write requests to the POM. The main functions of the WRC are:

- Store the destination descriptor starting address and byte count
- Split the destination descriptor into TLP write requests and send those requests to the POM
- Generate the command structure result field and result ring write requests when a command completes
- Write the slice hash value and file chaining hash value into the hash buffer
- Update the result ring write pointer after a command completes
- Arbitrate write data requests from the Engine Managers

4.1.6 Completion Controller

The Completion Controller (CC) distributes completion data to the proper destination according to the PCIe tags, which tag a completion. The main functions of the Completion Controller are:

- Decode the command structure and send the command to an engine manager
- Update the command ring write pointer in polling mode
- Verify the completion TLPs for unexpected completions
- Feed data into the source buffer

4.1.7 I/O Access Controller

The main functions of the I/O Access Controller (IOC) are:

- Receive the read/write requests from the PIM module and dispatch the request to the appropriate Engine manager
- Construct the opcode for I/O access to the Engine managers

4.2 9725 Control Registers

The 9725 Control Registers (9725_REGS) module implements all 9725 registers accessed by the host through the PCIe bus. The main functions of the 9725 Registers are:

- Implement all registers
- Respond to PCIe memory write and read request from the PIM
- Provide global control signals to all other modules

4.3 Engine Managers

There are six engine managers in the 9725. Each engine manager controls one set of Compression, Encryption and Hash engines. Each of the Engine interface control blocks and their associated FIFO structures will be described in the section for that engine.

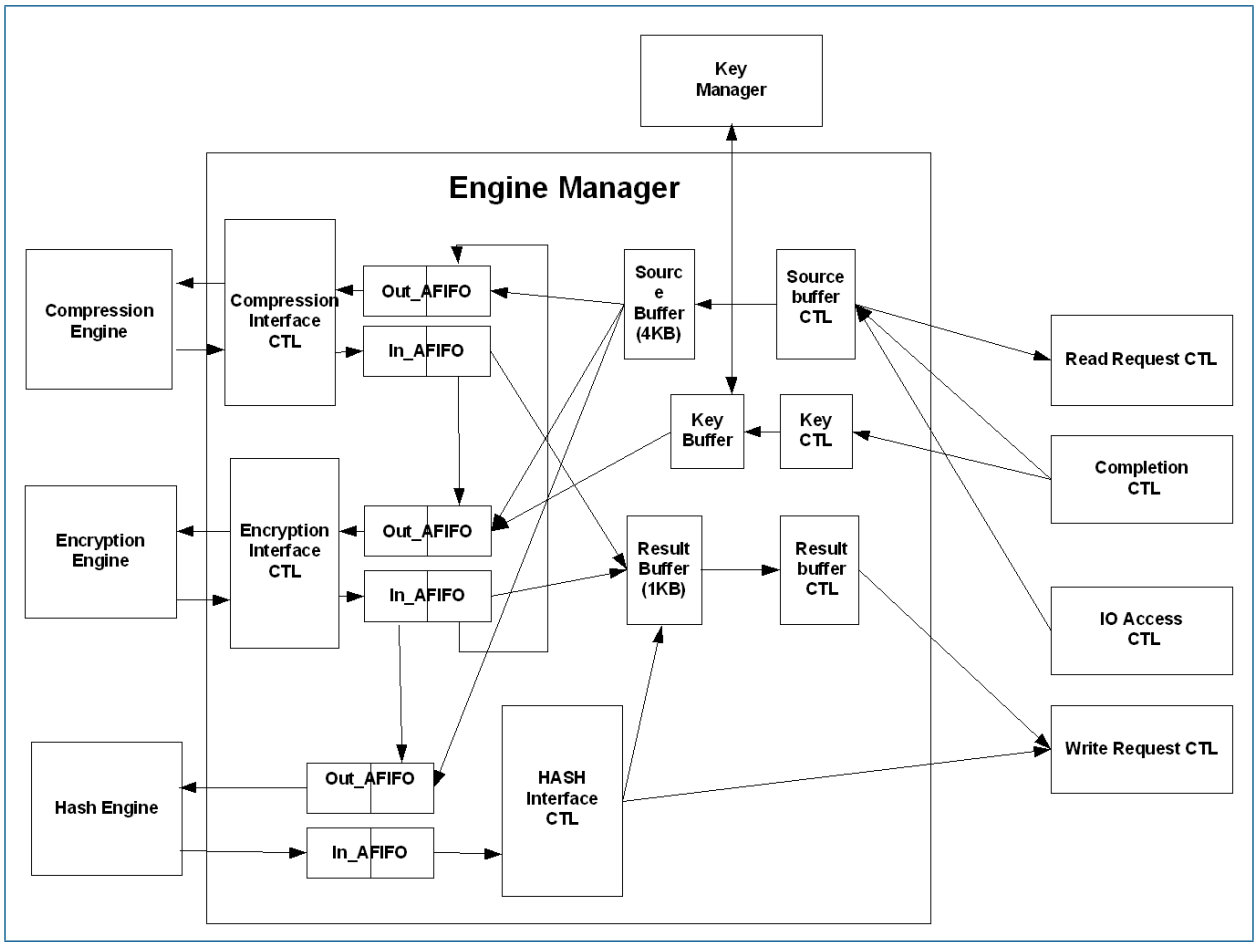


Figure 4-3. Engine Manager Block Diagram

4.3.1 Engine Manager Source Buffer

The source buffer is implemented as a two-port 72-bit memory (64 bits of data and 8 bits of ECC). The source buffer memory is segmented into multiple blocks; each block comprises multiple cells. The cell size is fixed at 128-bytes, the minimum max_read_request_size.

When an Engine Manager requests a block of source data, the length will be the block size, except the first and the last requests of a source descriptor because the starting address of a source descriptor is arbitrary aligned. The number of cells in a block are a power of 2 and depend on the parameter max_read_request_size which may be 128, 256, 512 bytes. Each Engine Manager has a 4K byte deep source buffer, and supports max_read_request_size equal to 512 bytes, therefore each block may consist of 1, 2, or 4 cells.

For data integrity, every 64 bits of data are protected by 8 bits of ECC.

Note: The default maximum read request size, max_read_request_size for the PCIe specification is 512 bytes. If the host configures the max read request size to 1KB or 2KB, the 9725 will still send 512B read requests over the PCIe interface.

4.3.2 Key Control

The Key Control module comprises the glue logic between the DMA and the Key Manager. The following functions are implemented in this module:

- Issue key write requests to the Key Manager for write key commands.
- Read the key from the Key Manager for encryption/decryption related commands.

4.3.3 Engine Manager Source Buffer Controller

The Source Buffer Controller (SBC) controls the command structure and fetching the source data. The main functions of the Source Buffer Controller are:

- Issue command structure read requests and put the command into the command buffer after the command completes
- Issue source descriptor read requests
- Issue source data read requests while maintaining the source data read request tags and ensuring the source buffer is available before issuing the source data read request (The SBC can generate a maximum 8 outstanding data read requests)
- Maintain two entries into the command buffers to improve performance by prefetching the next command's command structure and source data if all the current command’s source data is consumed

4.3.4 Engine Manager Result Buffer

The Result Buffer is used to temporarily store the destination data generated by the processing engines. Unlike source data, destination data is stored into the result buffer in the order that the commands complete and is fetched using the same sequence, therefore it mimics a FIFO.

The size of the write request payload must not exceed the parameter max_payload_size. The host software sets this parameter during initialization and is typically set to 128 bytes.

Note: The 9725 supports a maximum payload size, max_payload_size, of 128, 256 or 512 bytes. If the host configures the max read request size to 1KB or 2KB, the 9725 will still send 512B read requests over the PCIe interface.

4.3.5 Engine Manager Result Buffer Controller

The Result Buffer Controller (RBC) controls the result data and command process status writes to host. The main functions of the Result Buffer Controller are:

- Issue data write requests to the POM when data in the result buffer exceeds the maximum read request size
- Issue destination descriptor read requests
- Issue hash/MAC values write requests to the POM if used

- Issue command structure result field write requests and result ring write requests when a command completes
- Organize the errors reported by the processing engines

4.4 Key Manager and Key SRAM

Encryption keys are stored in the internal Key SRAM of the 9725. The Key Manger arbitrates access to the internal key SRAM by the six Engine Managers using a round-robin scheme. The keys are transferred from the host using a inband WRITE_KEY command over the PCIe interface. The Key Manager blocks read requests for keys that are actively being updated.

The internal SRAM supports up to 256 key entries of 40 bytes each. The key size may vary based on the key encryption format (see Table 4-2) and may require more than one key entry. To create a uniform implementation, all keys are stored in the internal SRAM using 256 bits. For example, an AES-XTS key will occupy two 256 bit key entries, and an AES-CBC-HMC-SHA1 key will occupy three key entries. A key index is used to point to the key starting address. Figure 4-4 illustrates the key format.

Table 4-2. Number of Keys Entries Required per Encryption Mode

AES Mode	Number of encryption key entries	Decryption key Required?	Number of decryption key entries	Total number of required key entries
AES-GCM 256 bit	1	No		1
AES-XTS 256 bit	1	Yes	1	2
AES-GCM 128 bit	1	No		1
AES-XTS 512 bit	2	Yes	2	4
AES-CBC 256 bit	1	Yes	1	2
AES-CBC 256 bit with HMAC-SHA1	3	Yes	3	6
AES-CBC 128 bit	1	Yes	1	2
AES-CBC 128 bit with HMAC-SHA1	3	Yes	3	6

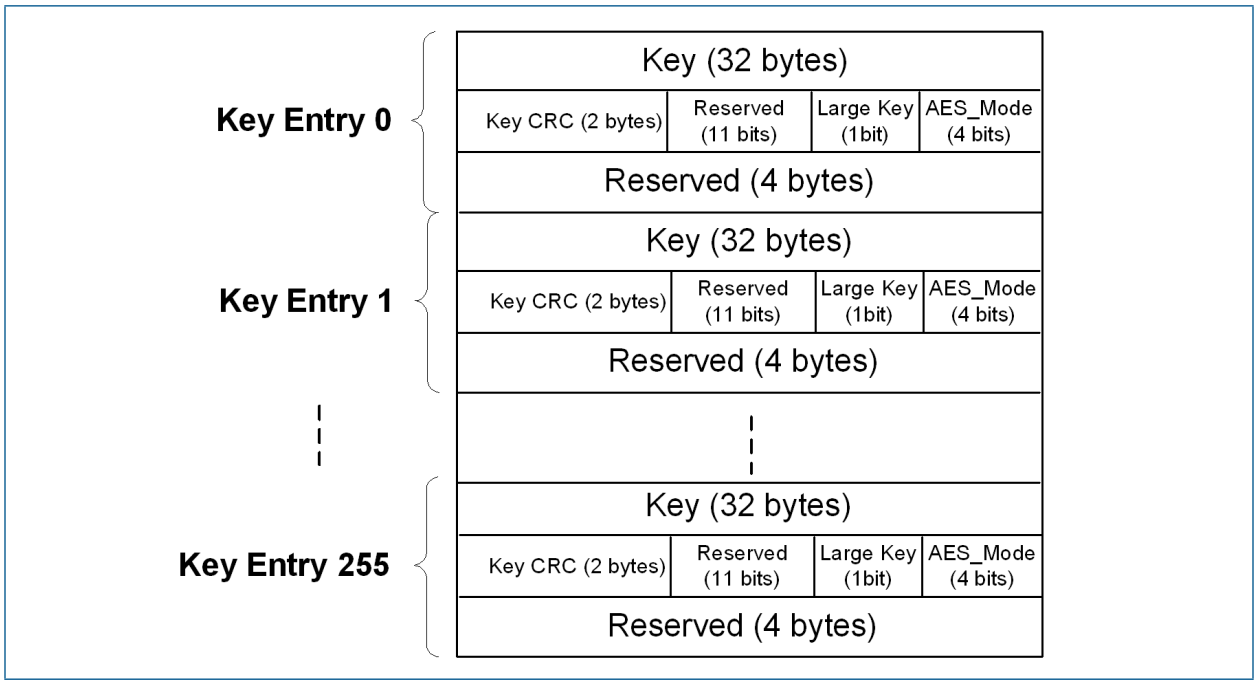


Figure 4-4. Key Format

The software preprocess the decryption key for some encryption/decryption operations before the data is submitted to the hardware. Note that XTS and CBC related operations require additional entries for decryption.

4.5 Compression Engine

The Compression Engine compresses/decompresses source data using the LZS or Enhanced eLZS algorithm. The Compression engine comprises an input FIFO, output FIFO, LZS interface controller and LZS core.

4.5.1 Compression Engine In_AFIFO & Out_AFIFO

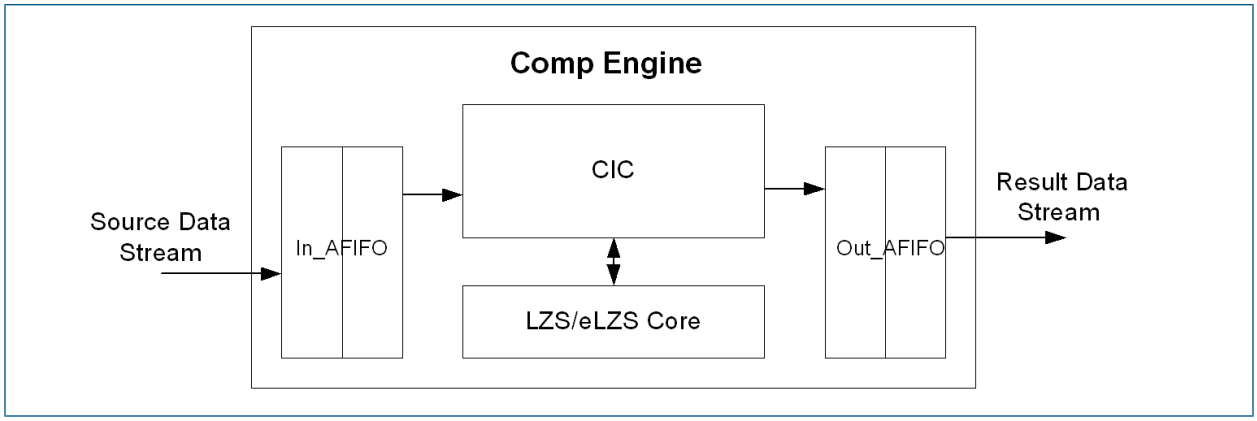


Figure 4-5. Compression Engine Block Diagram

The Compression Engine In_AFIFO and Out_AFIFO are used to transfer data and control signals between the DMA clock domain (250M) and the Compression Engine interface controller clock domain (300M).

4.5.2 Compression Engine Interface Controller

The Compression Engine Interface Controller (CIC) is the communication bridge between the In_AFIFO, Out_AFIFO, and LZS/eLZS core. The main functions of the CIC are:

- Read the source data from the In_AFIFO, and send the remaining data to the LZS/eLZS core.
- Write to the LZS/eLZS core command register at the beginning of each command.
- Read the result data from the LZS/eLZS Core, merge the tail of the source data stream and the result data into a double word, and send it to the Out_AFIFO.
- Report the LZS/eLZS core status to the Engine Manager after each command completes.

4.5.3 LZS Core

The Compression Engine performs high performance lossless data compression using the industry-standard Lempel-Ziv-Stac (LZS®) compression algorithm, as well as the "Enhanced LZS" algorithm that limits expansion during compression to 0.2%. The Compression Engine compresses or decompresses data at up to 300 MBytes/sec. The output of the compression engine is tied to the input of the decompression engine for compression operation verification.

The Compression Engine uses an anti-expansion algorithm that prevents the output from expanding during compression. If the output expands during compression the original uncompressed input along with certain header bytes is passed through as the output.

4.6 Encryption Engine

The Encryption Engine encrypts/decrypts source data with the AES algorithm. The Encryption Engine is comprised of the input FIFO, output FIFO, Encryption interface controller, and AES Core.

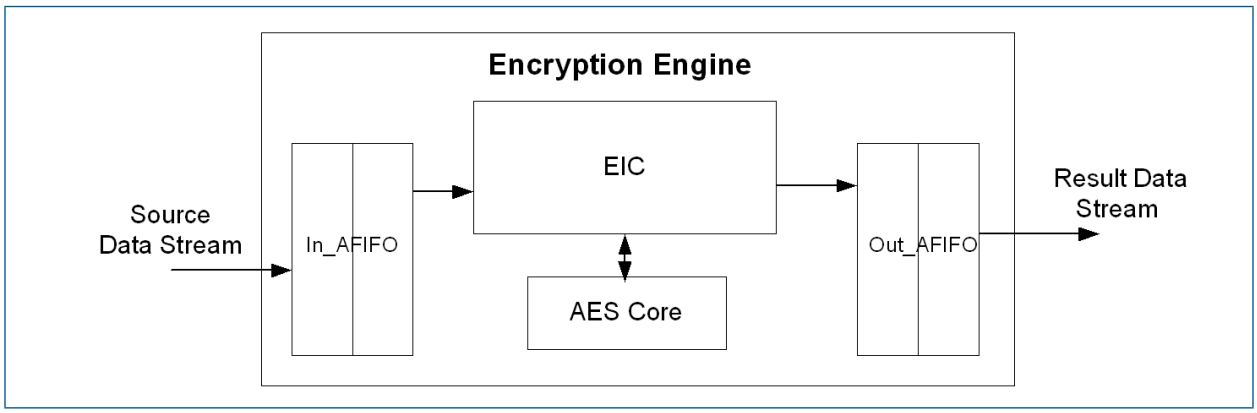


Figure 4-6. Encryption Engine Block Diagram

4.6.1 Encryption In_AFIFO & Out_AFIFO

The Encryption In_AFIFO and Out_AFIFO are used to transfer data and control signals between the DMA clock domain (250M) and the Encryption Engine interface controller clock domain (150M).

4.6.2 Encryption Interface Controller

The Encryption Interface Controller (EIC) is the communication bridge between the In_AFIFO, Out_AFIFO, and AES core. The main functions of the EIC are:

- Read the source data from the In_AFIFO, and send the remaining data to the AES core.
- Control the operation flow of the AES core.
- Read the result data from the AES Core, merge the tail of the source data stream and the result data into a double word, and send it to the Out_AFIFO.
- Report the AES core status to the Engine Manager after each command completes.

4.6.3 Encryption AES Core

The AES core supports AES-GCM, AES-CBC, AES-XTS, and CTS for AES-XTS mode.

4.7 Hash Engine

The Hash Engine calculates the hash or MAC values. The Hash Engine comprises an input FIFO, output FIFO, Hash FIFO, Hash interface controller and two Hash cores.

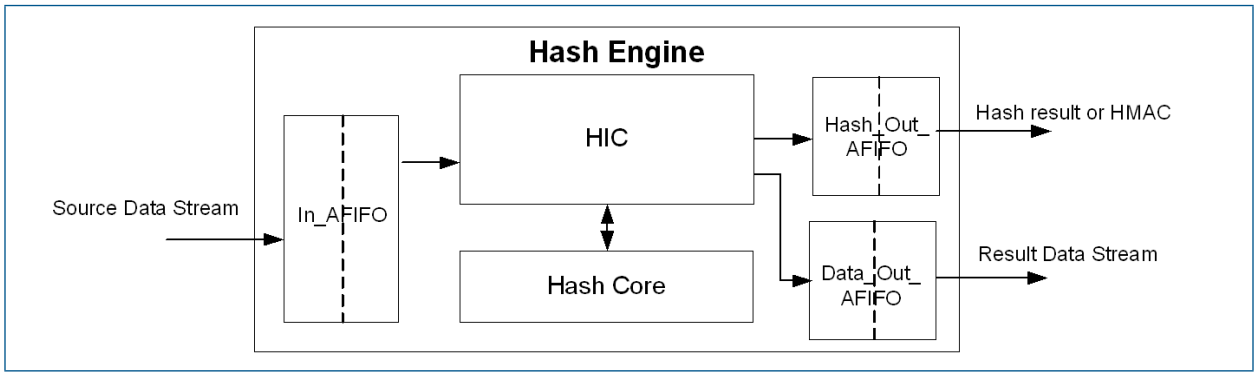


Figure 4-7. Hash Engine Block Diagram

4.7.1 Hash In_AFIFO, Hash Out_AFIFO, Data Out_FIFO

The Hash In_AFIFO is used to transfer data and control signals between the DMA clock domain (250M) and the Hash Engine clock domain (200M). The Hash OUT_AFIFO is used to store the calculated Hash or MAC value. The Data OUT_AFIFO is used to store processed data from the Hash engine.

4.7.2 Hash Interface Controller

The Hash Interface Controller (HIC) is the communication bridge between the In_FIFO, Hash_FIFO and Hash cores. The main functions of the HIC are:

- Read the source data from the In_FIFO, truncate the header and tail from the data stream, and then send the truncated data to the Hash core
- Control the real time verification flow
- Control the two Hash core's execution of the required operation
- Read the hash values from the Hash Cores and send them to the Hash_FIFO
- Combine the source data with the MAC value, and then send it to the Out_FIFO
- Report the Hash core status to the Engine Manager after each command completes

4.7.3 Hash Core

The Hash engine contains two Hash cores. Each core supports 4 operation modes:

- slice hash only
- file hash only

- slice hash + file hash
- HMAC (only SHA1 algorithm)

The Hash cores supports 3 Hash algorithms:

- SHA1
- SHA256
- MD5

4.8 Clock and Reset Generator

The Clock Generator (CLK_RST_Gen) module resides in the core of the 9725, and is responsible for the generation and management of all clocks and resets throughout the device. The major components of the CLK_RST_Gen module are:

- DMA PLL (25MHz)
- PCIe PHY PLL (100 MHz)
- Clock dividers, distribution and gating logic
- Reset generation logic

The DMA PLL is used to multiply the input clock to a frequency of 600MHz, which is then divided down to 200MHz, 150MHz, and 300MHz for the processing engines. The PCIe PHY PLL is used to generate a 250MHz clock for most of the DMA related modules.

To conserve power, the CLK_RST_Gen modules uses clock gating cells to statically disable the clock.

CLK_RST_Gen also generates the power on reset and software reset signals for all modules.

4.9 JTAG

The JTAG module implements a standard JTAG controller that is fully IEEE 1149.1 compliant.

4.10 PCIe Core

The PCIe Core is fully PCIe 1.1 protocol compliant. It receives raw PCIe packets from the PCIe SerDes module, extracts the application data, and sends the data to others modules.

The host may configure the 9725 to either enable/disable ECRC protection using the PCIe Advanced Error Report Capabilities and Control Register (refer to the PCIe specification for further information). If enabled, the 9725 will generate the ECRC on the fly to protect data on the PCIe bus. ECRC must also be enabled in the Root Complex of the host chip.

The PCIe Core also contains the PCIe Configuration Registers that memory map the 9725 PCIe configuration and control registers. The main functions of the PCIe Configuration Registers are:

- PCIe 3.0 Type 0 Configuration Header
- PCIe Power Management Structure
- PCIe Capabilities Structure

4.11 PCIe SerDes

The PCIe SerDes module provides a PCIe serial interface to PIPE interface conversion. The PCIe SerDes module supports x4 and x8 PCIe lanes.

4.12 Flash Controller

The Flash controller interfaces an external Flash to the 9725. The external Flash can be used to store log information written by the SDK and user configuration data such as subsystem vendor ID, subsystem device ID and expansion ROM size.

5 PCIe Configuration Register Definition

This section describes the 9725 PCIe configuration registers. The registers are mapped into 4K bytes of PCIe configuration space that can be accessed through the PCIe bus. The offset values in this section are given in hex.

The host should not read/write to/from reserved registers. If the host writes to a reserved register, the write will be ignored by the 9725. Likewise, if the host reads a reserved register, the return value will be all zeros.

Table 5-1 lists the register types definitions used to describe the PCIe configuration registers in this chapter. Fields marked as 'Read Only' usually indicate the 9725 capability and may not be altered by host; fields marked as 'Read/Write' may be altered by the host (usually BIOS or OS) for special purposes.

Table 5-1. Register Type Definitions

Register Type	Description
RO	Read only. Register bits are read-only and cannot be altered by software.
ROS	Sticky - Read only. Registers are read-only and cannot be altered by software. Registers are not initialized or modified by hot reset.
RW	Read/Write. Register bits are read-write and may be either set or cleared by software to the desired state.
RW1C	Read-only status, Write-1-to-clear status. Register bits indicate status when read, a set bit indicating a status event may be cleared by writing a 1. Writing a 0 to RW1C bits has no effect.
RWS	Sticky - Read-Write. Registers are read-write and may be either set or cleared by software to the desired state. Bits are not initialized or modified by hot reset.
HWINIT	Hardware Initialized. Register bits are initialized by firmware or hardware mechanisms such as pin strapping or serial EEPROM. Bits are read-only after initialization and can only be reset with a power on reset.

Figure 5-1 illustrates the 9725 PCIe Configuration registers.

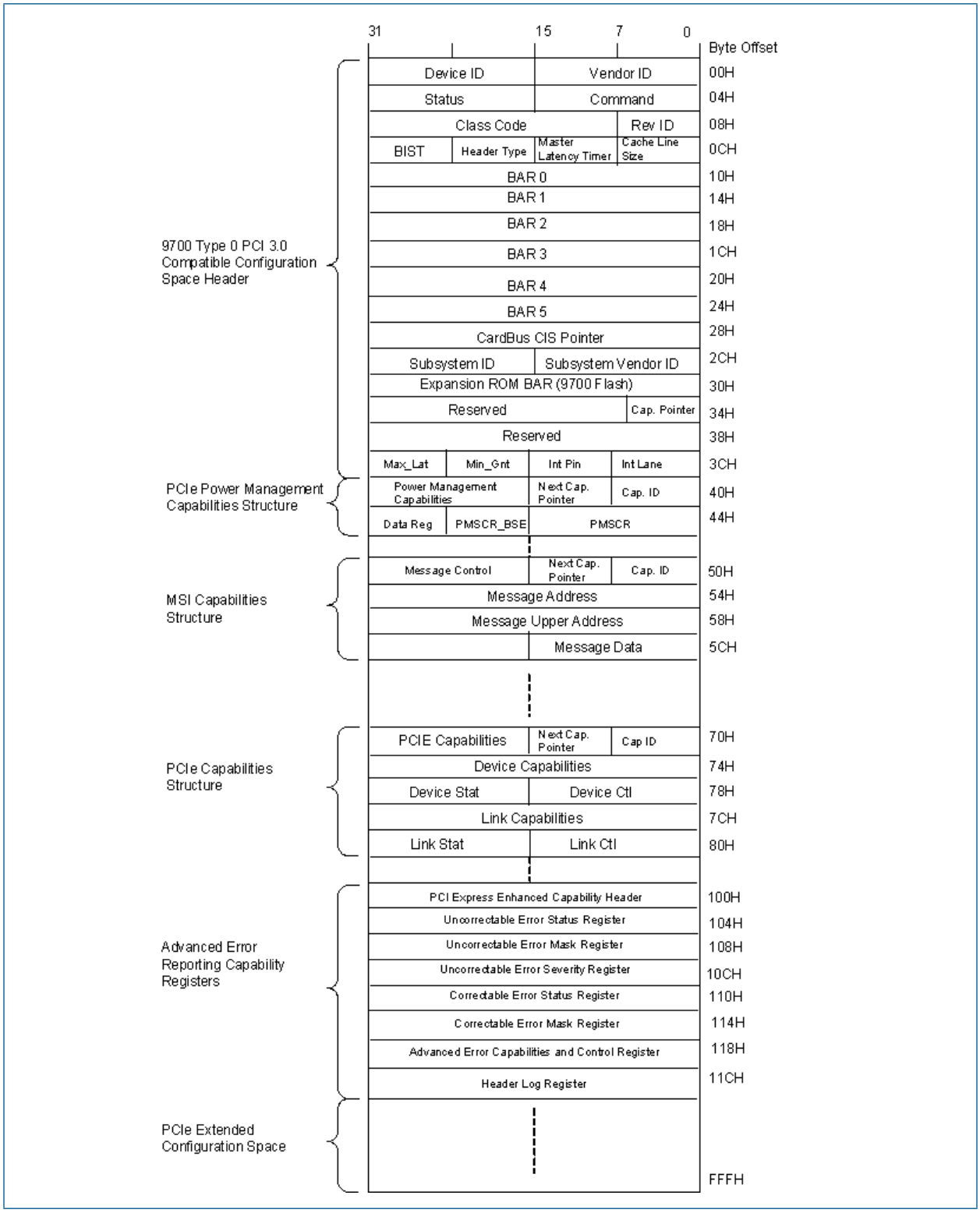


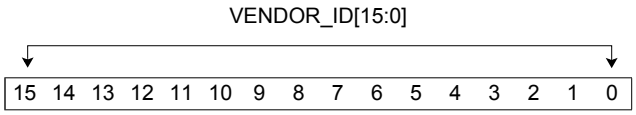
Figure 5-1. 9725 PCI-Express Configuration Space

5.1 Type 0 PCIe Compatible Configuration Space

5.1.1 Vendor ID Register

The Vendor ID register identifies Exar as the manufacturer of the 9725 device.

Offset x‘0000’



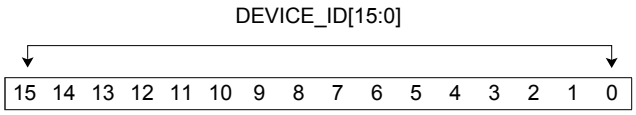
Field Name	Bits	Type	9725 Value	Description
Vendor_ID[15:0]	15:0	RO	0x13A3	This 16-bit register identifies Exar as the manufacturer of the 9725. The value hardwired in this read-only register is assigned by a central authority (the PCI SIG) that controls issuance of the numbers.

5.1.2 Device ID Register

The Device ID register identifies the 9725.

Type: Read only

Offset x‘0002’

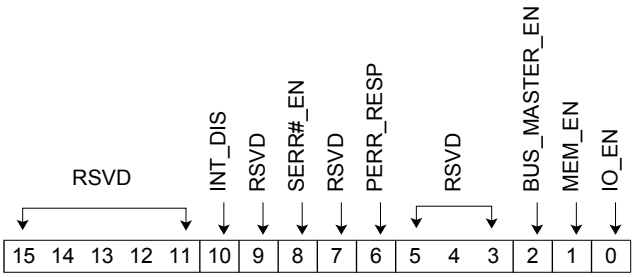


Field Name	Bits	Type	9725 Value	Description
Device_ID[15:0]	15:0	RO	0x002F	This 16-bit value is assigned by Exar and identifies the 9725. In conjunction with the Vendor ID and Revision ID, the Device ID can be used to locate a 9725-specific driver for the 9725.

5.1.3 Command Register

The Command register is used to enable or disable The I/O space, Memory space, Bus Master, Parity Error Response, System Errors, and Interrupts.

Offset x‘0004’



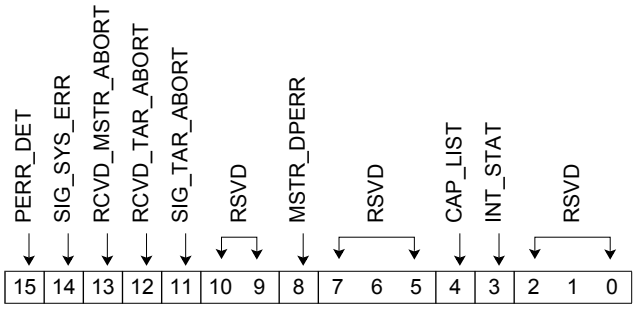
Field Name	Bits	Type	9725 Value	Description
RSVD	15:11	RO	0	Reserved
INT_DIS	10	RW	0	Interrupt Disable. Controls whether the 9725 can generate INTx interrupt messages. 0 9725 enabled to generate INTx interrupt messages. 1 9725's disabled to generate INTx interrupt messages is disabled. If the 9725 had already transmitted an Assert_INTx emulation interrupt messages and this bit is then set, the 9725 must transmit a corresponding Deassert_INTx message for each previously transmitted assert message Note that INTx emulation interrupt messages forwarded by Root and Switch Ports from devices downstream of the Root or Switch Port are not affected by this bit.
RSVD	9	RO	0	Reserved
SERR#_EN	8	RW	0	System Error Enable. This active low bit enables or disables the reporting of errors detected by the 9725 to the Root Complex. 0 9725 may send detected errors to the Root Complex 1 9725 may not send detected errors to the Root Complex
RSVD	7	RO	0	Reserved

Field Name	Bits	Type	9725 Value	Description
PERR_RESP	6	RW	0	Parity Error Response. This bit is used to enable or disable the Master Data Parity Error bit in the Status register. 0 Disable MSTR_DPERR 1 Enable MSTR_DPERR
RSVD	5:3	RO	0	Reserved
BUS_MASTER_EN	2	RW	0	Bus Master Enable. 0 Disables the 9725 from issuing memory or IO requests, and from generating MSI messages. 1 Enables the 9725 to issue memory or IO requests, including MSI messages. Requests other than memory or IO requests are not controlled by this bit.
MEM_EN	1	RW	0	Memory Address Space Decoder Enable. 0 Memory decoder is disabled and Memory transactions targeting the 9725 are not recognized. 1 Memory decoder is enabled and Memory transactions targeting the 9725 are accepted.
IO_EN	0	RW	0	IO Address Space Decoder Enable. 0 IO decoder is disabled and IO transactions targeting the 9725 are not recognized. 1 IO decoder is enabled and IO transactions targeting the 9725 are accepted.

5.1.4 Status Register

The Status register is used to report errors and interrupts from the 9725 to the host. The read only fields of this register are automatically updated by the 9725 to reflect their internal status.

Offset x‘0006’



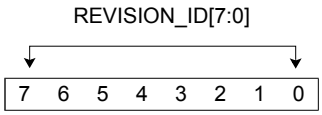
Field Name	Bits	Type	9725 Value	Description
PERR_DET	15	RO	0	Parity Error Detected. Regardless of the state the Parity Error Enable bit in the 9725's Command register, this bit is set if the 9725 receives a Poisoned TLP. 0 9725 has not received a Poisoned TLP. 1 9725 received a Poisoned TLP.
SIG_SYS_ERR	14	RO	0	Signaled System Error. 0 9725 has not sent a fatal or non-fatal message. 1 The 9725 sent an ERR_FATAL or ERR_NONFATAL message, and the SERR Enable bit in the Command register is set to one.
RCVD_MSTR_ABORT	13	RO	0	Received Master Abort. 0 9725 has not sent a master abort message. 1 9725 received a Completion with Unsupported Request Completion Status.
RCVD_TAR_ABORT	12	RO	0	Received Target Abort. 0 9725 has not sent a received target abort message. 1 9725 received a Completion with Completer Abort Completion Status.

Field Name	Bits	Type	9725 Value	Description
SIG_TAR_ABORT	11	RO	0	Signaled Target Abort. 0 9725 has not sent a signal target abort message. 1 The 9725, acting as a Completer, terminated a request by issuing Completer Abort Completion Status to the Requester.
RSVD	10:9	RO		Reserved
MSTR_DPERR	8	RO	0	Master Data Parity Error. The Master Data Parity Error bit is set by a 9725 if the Parity Error Enable bit is set in the Command register and either of the following two conditions occurs: -the 9725 receives a poisoned Completion. -the 9725 poisons a write request. If the Parity Error Enable bit is cleared, the Master Data Parity Error status bit is never set. 0 No Master Data Parity Error occurred. 1 Master Data Parity Error occurred.
RSVD	7:5	RO	0	Reserved
CAP_LIST	4	RO	1	Capabilities List. Indicates the presence of one or more extended capability register sets in the lower 48 dwords of the 9725's PCI-compatible configuration space. 0 Capabilities List not present. 1 Capabilities List present.
INT_STAT	3	RO	0	Interrupt Status. Indicates that the 9725 has an interrupt request outstanding (that is, the 9725 transmitted an interrupt message that is waiting to be serviced). Note that INTx emulation interrupts forwarded by Root and Switch Ports from devices downstream of the Root or Switch Port are not reflected in this bit. Note: this bit is only associated with INTx messages, and has no meaning if the device is using Message Signaled Interrupts. 0 9725 has no interrupt request outstanding. 1 9725 has an interrupt request outstanding.
RSVD	2:0	RO	N/A	Reserved

5.1.5 Revision ID Register

The Revision ID register identifies the revision number assigned to each 9725 device.

Offset x‘0008’

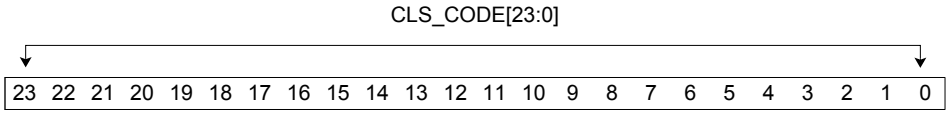


Field Name	Bits	Type	9725 Value	Description
REVISION_ID[7:0]	7:0	RO	0x00	This 8-bit value is assigned by the Exar to identify the revision number of the 9725.

5.1.6 Class Code Register

The Class Code register defines the 9725 encryption/decryption controller.

Offset x‘0009’



Field Name	Bits	Type	Default	Description
CLS_CODE[23:0]	23:0	RO	0x108000	Encryption/Decryption controller.

5.1.7 Cache Line Size Register

Offset x‘00C’

The Cache Line Size register is implemented as a read-write field for legacy compatibility purposes but has no impact on the 9725's functionality. The typical read value is 0x10.

5.1.8 Master Latency Timer Register

Offset x‘00D’

The Master Latency Timer register is implemented for legacy compatibility purposes but has no impact on the 9725's functionality. This register is hard-wired to 0x00.

5.1.9 Header Type Register

Offset x‘00E’

The Header Type register is a read-only optional register whose value is hard-wired to 0x00.

5.1.10 BIST Register

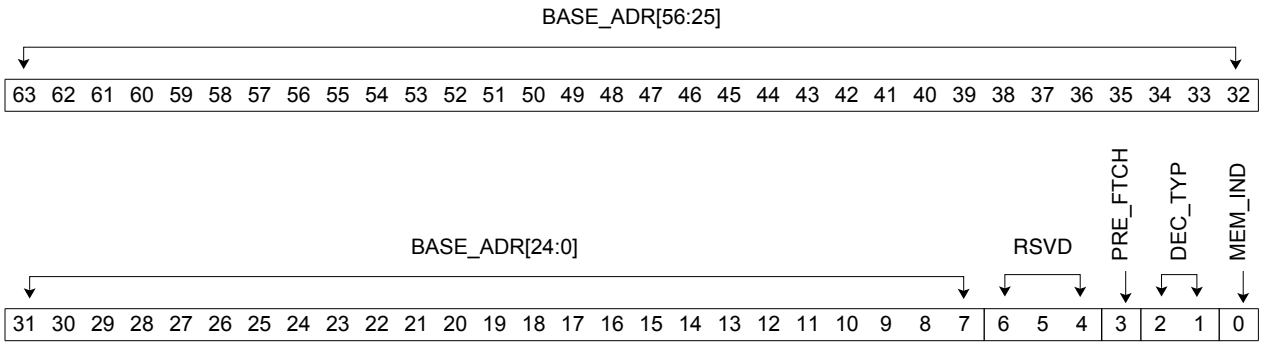
Offset x‘00F’

The BIST register is used for control and status of the BIST function. This register is hard-wired to 0x00.

5.1.11 Base Address Register 0, 1

The Base Address 0 register provides the 9725 base address on a 4KB boundary and some memory configuration parameters. This address in memory space is where the standard 9725 register set will reside and be accessed. For 32-bit systems, the base address is set using only BAR 0. For 64-bit systems, the base address is set using both BAR 0 and BAR 1. In both cases, a 4KB memory space will be requested for the BAR registers.

Offset x‘0010’



Field Name	Bits	Type	9725 Value	Description
BASE_ADR[56:0]	63:7	RW	0	Base Address. [63:32] Only used for 64-bit 4KB memory base address [31:7] Indicates 32-bit 4KB memory base address
RSVD	6:4	RO	0	Reserved
PRE_FTCH	3	RO	0	Prefetchable memory. 0 Non-Prefetchable memory 1 Prefetchable memory
DEC_TYP	2:1	RW	00 for a 32-bit system; 10 for a 64-bit system	Decoder Type. 00 32 bit decoder; locate memory anywhere in lower 4GB; use only BAR0 01 Reserved 10 64-bit decoder; locate memory anywhere in 264 memory space; uses BAR0 and BAR1 11 Reserved
MEM_IND	0	RO	0	Memory Space Indicator. x0 Base Address Register0 is a memory address decoder x1 Base Address Register0 is an IO address decoder

5.1.12 Base Address Register 2-5

The Base Address Registers 2-5 are not used by the 9725, are read only, and will be read as all zeroes.

5.1.13 Cardbus CIS Pointer Register

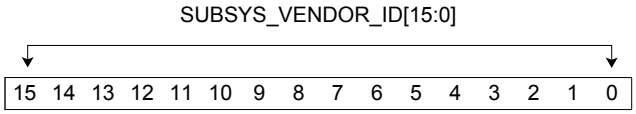
Offset x‘0028’

This optional read-only register is used by devices that contain a CardBus and PCIe interface. The 9725 does not support Cardbus so this register is hard-wired to 0x00000000.

5.1.14 Sub-System Vendor ID Register

The Sub-System Vendor ID register identifies Exar as the manufacturer of the 9725 device. This value is loaded internally from the Flash. The 9725 may overwrite the Sub-system Vendor ID value from the Flash.

Offset x‘002C’

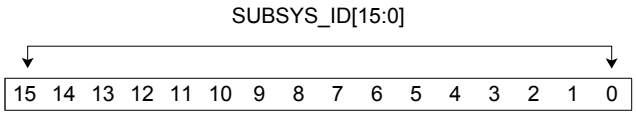


Field Name	Bits	Type	9725 Value	Description
SUBSYS_VENDOR_ID[15:0]	15:0	RO	0x13A3	This 16-bit register identifies the manufacturer of the subsystem. The value hardwired in this read-only register is assigned by a central authority (the PCI SIG) that controls issuance of the numbers.

5.1.15 Sub-System ID Register

The Sub-System ID register identifies the 9725. This value is loaded internally from the Flash. The 9725 may overwrite the Sub-system ID value from the Flash.

Offset x‘002E’

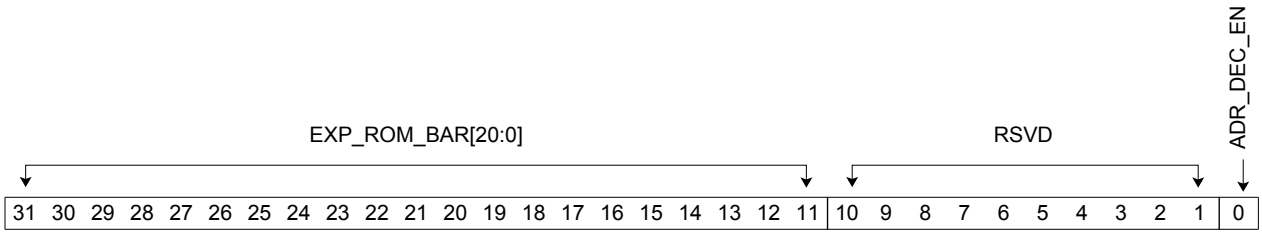


Field Name	Bits	Type	9725 Value	Description
SUBSYS_ID[15:0]	15:0	RO	0x9725	This 16-bit value is assigned by the subsystem manufacturer and identifies the type of subsystem.

5.1.16 Expansion ROM Base Address Register

The ROM Base address register (ROM BAR) provides the 9725 expansion ROM Base address and address decode enable. This register sets the address space in memory for the attached Flash device.

Offset x‘0030’



Field Name	Bits	Type	9725 Value	Description
EXP_ROM_BAR [20:0]	31:11	RW		Expansion ROM Base Address. [31:11] The base address of the expansion ROM
RSVD	10:1	RO	0	Reserved.
ADR_DEC_EN	0	RW	0	Address Decode Enable. 0 Disable expansion ROM 1 Enable expansion ROM

5.1.17 Capabilities Pointer Register

Offset x‘0034’

The Capabilities Pointer register is used to point to a linked list of capabilities implemented by the 9725. This read-only register value is hard-wired to 0x40, the Power Management Capabilities structure.

5.1.18 Interrupt Line Register

Offset x‘003C’

The Interrupt Line register communicates interrupt line routing information to device drivers and operating systems. Values in this register are programmed by system software and are system architecture specific. The typical read value is 0x0B.

5.1.19 Interrupt Pin Register

Offset x‘003D’

The Interrupt Pin register identifies the legacy interrupt Message(s) used by the 9725. This register is a read-only register whose value will be read as 0x01, indicating the 9725 uses INTA.

5.1.20 Min_Gnt Register

Offset x‘003E’

This legacy register is a read-only register whose value is hard-wired to 0x00.

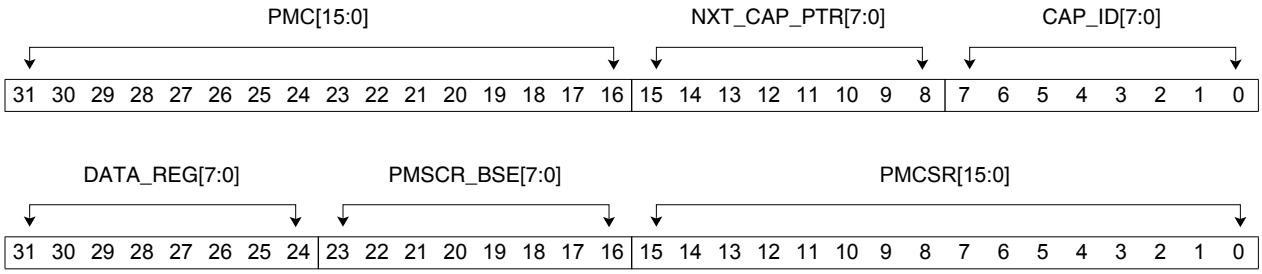
5.1.21 Max_Lat Register

Offset x‘003F’

This legacy register is a read-only register whose value is hard-wired to 0x00.

5.2 Power Management Capabilities Registers

The Power Management registers indicate the 9725 power management capabilities.



5.2.1 Capability ID Register

The Capability Identifier register, when read by system software as 01h, indicates that the data structure currently being pointed to is the PCI Power Management data structure.

Offset x‘0040’

Field Name	Bits	Type	9725 Value	Description
CAP_ID	7:0	RO	0x01	Power Management Capability ID. 01h = identifies the linked list item as being the PCI Power Management registers

5.2.2 Next Capabilities Pointer Register

The Next Capabilities register describes the location of the next item in the 9725’s capability list. The value given is an offset into the 9725’s PCI Configuration Space.

Offset x‘0041’

Field Name	Bits	Type	9725 Value	Description
NXT_CAP_PTR	7:0	RO	0x50	Power Management Next Capabilities Pointer. 50h = PCIe MSI capabilities structure

5.2.3 Power Management Capabilities Register

The Power Management Capabilities register is a 16-bit read-only register which provides information on the capabilities of the function related to power management. The information in this register is generally static and known at design time.

Offset x‘0042’

Field Name	Bits	Type	9725 Value	Description
PME Support	15:11	RO	0x0B	PME Support. Indicates the PM states supported by the 9725. A one in a bit indicates the 9725 is capable of sending a Power Management Event (PME) message. A zero in a bit indicates PME notification is not supported in the respective PM state. Bit PM State 11 D0 (supported by 9725) 12 D1 (supported by 9725) 13 D2 (not supported by 9725) 14 D3hot (supported by 9725) 15 D3cold (not supported by 9725)
D2 Support	10	RO	0	D2 Support. 0 = D2 PM state not supported
D1 Support	9	RO	1	D1 Support. 1 = D1 PM state supported
Aux Current	8:6	RO	111	Aux Current. The Aux_Current field reports the 3.3Vaux current requirements for the 9725. The 9725 reports 375mA max.
Device-Specific Initialization	5	RO	0	Device-Specific Initialization. A one in this bit indicates that immediately after entry into the D0 Uninitialized state, the 9725 requires additional configuration above and beyond setup of its PCI configuration Header registers before the Class driver can use the 9725. Microsoft OSs do not use this bit. Rather, the determination and initialization is made by the Class driver.
RSVD	4	RO	0	Reserved.
PME Clock	3	RO	0	The 9725 does not use PME Clock.
Version Field	2:0	RO	0x3	Version Field. This field indicates the version of the PCI Bus PM Interface spec that the 9725 complies with.

5.2.4 Power Management Control/Status Register

The Power Management Control/Status (PMCSR) register is used to manage the PCI function's power management state as well as to enable/monitor Power Management Events (PMEs).

Offset x'0044'

Field Name	Bits	Type	9725 Value	Description
PME_STAT	15	RW1C	0	PME Status. This bit is set to "0" because the 9725 does not support PME# generation from D3cold.
DAT_SCALE	14:13	RO	0	Data Scale. This field is set to "00b" in the 9725 because the PM Data register is not implemented.
DAT_SEL	12:9	RW	0000	Data Select. This field is set to "0000b" in the 9725 because the PM Data register is not implemented.
PME_EN	8	RW	0	PME Enable. This bit is set to "0" because the 9720 does not support PME# generation from D3cold.
RSVD	7:4	RO	0	Reserved.
NO_SFT_RST	3	RO	0	No Soft Reset. The 9725 sets this bit to 0 to indicate it will perform an internal reset upon transitioning from D3hot to D0 via software control of the PowerState bits. Configuration Context is lost when performing the soft reset. Upon transition from the D3hot to the D0 state, full reinitialization sequence is needed to return the device to D0 Initialized.
RSVD	2	RO	0	Reserved.

Field Name	Bits	Type	9725 Value	Description
PWR_STATE	1:0	RW	00	Power State. This 2-bit field is used both to determine the current power state of the 9725 and to set the 9725 into a new power state. The definition of the field values is given below. 00b - D0 01b - D1 10b - D2 11b - D3hot (not supported) If software attempts to write an unsupported state to this field, the write operation must complete normally on the bus; however, the data is discarded and no state change will occur.

5.2.5 PMCSR-BSE Register

Offset x‘0046’

The PMCSR PCI-to-PCI Bridge Support Extensions register does not apply to the 9725. When read, the register will return zeroes.

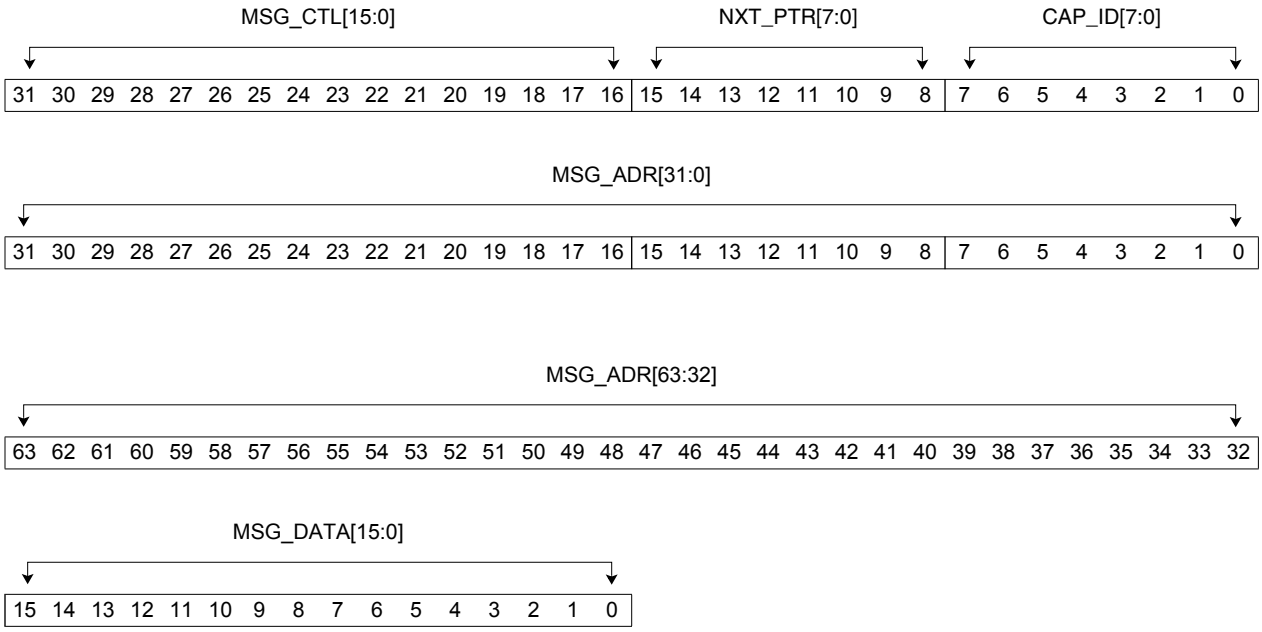
5.2.6 Data Register

Offset x‘0047’

The Data register is not used by the 9725 but may be written to by system software. When read, the register will typically return all zeroes.

5.3 MSI Capability Registers

The MSI Capability registers indicate the 9725 Message Signaled Interrupt capabilities. The 9725 uses a 64-bit Message Address.



5.3.1 Capability ID Register

The Capability Identifier register, when read by system software as 05h, indicates that the 9725 is MSI capable.

Offset x‘0050’

Field Name	Bits	Type	9725 Value	Description
CAP_ID	7:0	RO	0x05	MSI Capability. 05h = MSI capable

5.3.2 Next Capabilities Pointer Register

The Next Capabilities register describes the location of the next item in the 9725's capability list. The value given is an offset into the 9725's PCI Configuration Space.

Offset x'0051'

Field Name	Bits	Type	9725 Value	Description
NXT_PTR	7:0	RO	0x70	MSI Next Pointer. 70h = PCIe capabilities structure

5.3.3 Message Control Register

Offset x‘0053’

Field Name	Bits	Type	9725 Value	Description
RSVD	15:9	RO	0	Reserved.
PER_VEC_MSK	8	RO	0	Per-vector masking. The 9725 does not support MSI per-vector masking. 0 = Per-vector masking not supported
64_ADR_CAP	7	RO	1	64-bit Address. The 9725 is capable of generating a 64-bit message address. 1 = 64-bit address capable
MULT_MESS_EN	6:4	RW	000	Multiple Message Enable. System software writes to this field to indicate the number of allocated vectors (equal to or less than the number of requested vectors). 000 = 1 vector allowed
MULT_MESS_CAP	3:1	RO	000	Multiple Message Capable. System software reads this field to determine the number of requested vectors. 000 = 1 vector requested
MSI Enable	0	RW	0	MSI Enable. System configuration software sets this bit to enable MSI. 0 = 9725 is prohibited from using MSI to request service 1 = If the MSI-X Enable bit in the MSI-X Message Control register is 0, the 9725 is permitted to use MSI to request service.

5.3.4 Message Address Register

Offset x‘0054 - x0058’

Field Name	Bits	Type	9725 Value	Description
MSG_ADR	63:2	RW		MSI Message Address. If the Message Enable bit (bit 0 of the Message Control register) is set, the contents of this register specifies the DWORD aligned address (AD[63:02]) for the MSI memory write transaction. AD[1:0] are driven to zero during the address phase.
RSVD	1:0	RO	00	Reserved. Always returns 0 on read. Write operations have no effect.

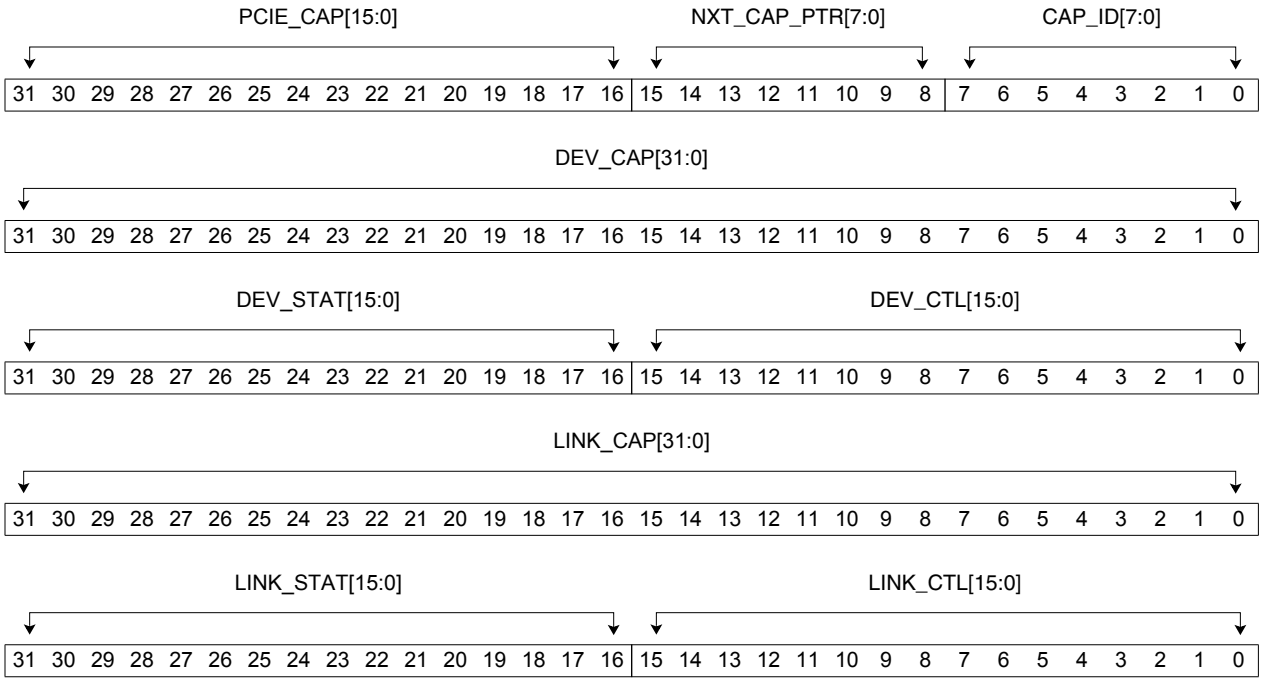
5.3.5 Message Data Register

Offset x‘005C’

Field Name	Bits	Type	9725 Value	Description
MSG_DATA	15:0	RW		MSI Message Data. If the Message Enable bit (bit 0 of the Message Control register) is set, the message data is driven onto the lower word (MSG_ADR[15:0]) of the memory write transaction’s data phase. MSG_ADR[31:16] are driven to zero during the memory write transaction’s data phase. C/BE[3:0]# are asserted during the data phase of the memory write transaction.

5.4 PCI Express Capability Registers

The PCI Express Capability registers are required for PCI Express devices. The capability structure is a mechanism for enabling PCI software transparent features requiring support on legacy operating systems. In addition to identifying a PCI Express device, the PCI Express Capability structure is used to provide access to PCI Express specific Control/Status registers and related Power Management enhancements.



5.4.1 Capability ID Register

The Capability Identifier register, when read by system software as 10h, indicates that the 9725 is PCIe capable.

Offset x‘0070’

Field Name	Bits	Type	9725 Value	Description
CAP_ID	7:0	RO	0x10	PCIe Capable. 10h = 9725 is PCIe capable

5.4.2 Next Capabilities Pointer Register

The Next Capabilities register describes the location of the next item in the 9725's capability list. The value given is an offset into the 9725's PCI Configuration Space.

Offset x'0071'

Field Name	Bits	Type	9725 Value	Description
NXT_PTR	7:0	RO	0x00	Next Capability Pointer. 00h = Advanced Error structure

5.4.3 PCIe Capabilities Register

The PCI Express Capabilities (PCIE_CAP) register identifies the PCI Express device type and its associated capabilities.

Offset x'0072'

Field Name	Bits	Type	9725 Value	Description
RSVD	15:14	RO	00	Reserved.
INT_MSG_NUM	13:9	RO	0x0	Interrupt Message Number. This field is not used by the 9725.
SLOT_IMPL	8	HWINIT	0	Slot Implemented. This bit is not used by the 9725.
DEV_PORT_TYP	7:4	RO	0000	Device/Port Type. Indicates the type of PCI Express logical device. 0000 = 9725 is PCI Express Endpoint device
CAP_VER	3:0	RO	01	Capability Version. Indicates the PCI Express capability structure version number.

5.4.4 Device Capabilities (DEV_CAP) Register

The Device Capabilities register identifies the PCI Express device specific capabilities.

Offset x‘0074’

Field Name	Bits	Type	9725 Value	Description
RSVD	31:28	RO	0	Reserved
Captured Slot Power Limit Scale	27:26	RO		Captured Slot Power Limit Scale. This field specifies the scale used for the Slot Power Limit Value, which is set by the Set_Slot_Power_Limit Message for the 9725. Most systems will read 0x0, which means 1.0x.
Captured Slot Power Limit Value	25:18	RO		Captured Slot Power Limit Value. This field is used in combination with the Slot Power Limit Scale value to specify the upper limit on the power supplied by the slot. The power limit (in Watts) is calculated by multiplying the value in this field by the value in the Slot Power Limit Scale field. This value is set by Set_Slot_Power_Limit Message for the 9725. Most systems will read 0x19, which represents 25W.
RSVD	17:15	RO	0x0	Reserved
Power Indicator Present	14	RO	0	Power Indicator Present. When set to one, indicates a Power Indicator is implemented on the card or module. Valid for the following PCI Express device types: Express Endpoint device Legacy Express Endpoint device Switch upstream port Express-to-PCI/PCI-X bridge
Attention Indicator Present	13	RO	0	Attention Indicator Present. When set to one, indicates an Attention Indicator is implemented on the card or module. Valid for the following PCI Express device Types: Express Endpoint device Legacy Express Endpoint device Switch upstream port Express-to-PCI/PCI-X bridge

Field Name	Bits	Type	9725 Value	Description
Attention Button Present	12	RO	0	Attention Button Present. When set to one, indicates an Attention Button is implemented on the card or module. Valid for the following PCI Express device types: Express Endpoint device Legacy Express Endpoint device Switch upstream port Express-to-PCI/PCI-X bridge
Endpoint L1s Acceptable Latency[2:0]	11:9	RO	0x1	Endpoint L1s Acceptable Latency. Acceptable latency that an Endpoint can withstand due to the transition from L1 state to the L0 state. This value is an indirect indication of the amount of the Endpoint's internal buffering. Power management software uses this value to compare against the L1 Exit Latencies reported by all components in the path between this Endpoint and its parent Root Port to determine whether ASPM L1 entry can be used with no loss of performance. The 9725 requires an endpoint L1 acceptable latency of 1µs to less than 2µs.
Endpoint L0s Acceptable Latency[2:0]	8:6	RO	0x1	Endpoint L0s Acceptable Latency. Acceptable total latency that an Endpoint can withstand due to the transition from the L0s state to the L0 state. This value is an indirect indication of the amount of the Endpoint's internal buffering. Power management software uses this value to compare against the L0s exit latencies reported by all components in the path between this Endpoint and its parent Root Port to determine whether ASPM L0s entry can be used with no loss of performance. The 9725 requires an endpoint L0 acceptable latency of 64ns to less than 128ns.
Extended Tag Field Supported	5	RO	0x1	Extended Tag Field Supported. Max supported size of the Tag field when this function acts as a Requester. 0 = 5-bit Tag field supported (max of 32 outstanding request per Requester). 1 = 8-bit Tag field supported (max of 256 outstanding request per Requester). If 8-bit Tags are supported and will be used, this feature is enabled by setting the Extended Tag Field Enable bit in the Device Control register to one.

Field Name	Bits	Type	9725 Value	Description
Phantom Functions Supported[1:0]	4:3	RO	0x0	Phantom Functions Supported. When the device within which a function resides does not implement all eight functions, a non-zero value in this field indicates that this is so. Assuming all functions are not implemented and that the programmer has set the Phantom Function Enable bit in the Device Control register, a function may issue request packets using its own function number as well as one or more additional function numbers. This field indicates the number of msbs of the function number portion of Requester ID that are logically combined with the Tag identifier. 00 = The Phantom Function feature is not available within this device. 01 = The msb of the function number in the Requestor ID is used for Phantom Functions. The device designer may implement functions 0-3. When issuing request packets, Functions 0, 1, 2, and 3 may also use function numbers 4, 5, 6, and 7, respectively, in the packet's Requester ID. 10 = The two msbs of the function number in the Requestor ID are used for Phantom Functions. The device designer may implement functions 0 and 1. When issuing request packets, Function 0 may also use function numbers 2, 4, and 6 in the packet's Requester ID. Function 1 may also use function numbers 3, 5, and 7 in the packet's Requester ID. 11 =All three bits of the function number in the Requestor ID are used for Phantom Functions. The device designer must only implement Function 0 (and it may use any function number in the packet's Requester ID).
Max Payload Size Supported[2:0]	2:0	RO	0x2	Max Payload Size Supported. Defines the Max data payload size that the 9725 supports for TLPs. The 9725 supports 512 bytes max payload size.

5.4.5 Device Control (DEV_CTL) Register

The Device Control register controls PCI Express device specific parameters. The system software may read and write the fields of this register to control the operation of the 9725.

Offset x‘0078’

Field Name	Bits	Type	9725 Value	Description
RSVD	15	RO	00	Reserved.
MAX_RD_REQ_SZ	14:12	RW	010	Maximum Read Request Size. This field sets the maximum Read Request size for the Device as a Requester. The Device must not generate read requests with size exceeding the set value. Supported 9725 read request sizes: 000 = 128 bytes 001 = 256 bytes 010 = 512 bytes
EN_NO_SNOOP	11	RW	1	Enable No Snoop. This bit is set to 1 in the 9725, indicating the 9725 is permitted to set the No Snoop bit in the Requester Attributes of transactions it initiates that do not require hardware enforced cache coherency.
AUX_PWR_EN	10	RWS	0	Auxiliary (AUX) Power PM Enable. This bit is set to 0 by the 9725, disabling the 9725 to draw AUX power independent of PME AUX power.
PHANT_EN	9	RW	0	Phantom Functions Enable. This bit is hardwired to 0 as the 9725 device does not implement this capability.
EXT_TAG_EN	8	RW	1	Extended Tag Field Enable. This bit is set to 1 in the 9725 to enable the 9725 to use an 8-bit Tag field as a requester.

Field Name	Bits	Type	9725 Value	Description
MAX_PAY_SZ	7:5	RW	000	Maximum Payload Size. This field sets maximum TLP payload size for the device/function. As a Receiver, the device must handle TLPs as large as the set value; as Transmitter, the device must not generate TLPs exceeding the set value. Supported 9725 payload sizes: 000 = 128 bytes max payload size 001 = 256 bytes max payload size 010 = 512 bytes max payload size
EN_RLX_ORD	4	RW	0	Enable Relaxed Ordering. This bit is hardwired to 0 as the 9725 device never sets the Relaxed Ordering attribute in transactions it initiates as a requester.
NON_SPT_REQ_REP_EN	3	RW	0	Unsupported Request Reporting Enable. This bit, in conjunction with other bits, controls the signaling of Unsupported Requests by sending Error Messages.
FTL_ERR_REP_EN	2	RW	0	Fatal Error Reporting Enable. This bit, in conjunction with other bits, controls sending ERR_FATAL Messages.
NON_FTL_ERR_REP_EN	1	RW	0	Non-Fatal Error Reporting Enable. This bit, in conjunction with other bits, controls sending ERR_NONFATAL Messages.
COR_ERR_REP_EN	0	RW	0	Correctable Error Reporting Enable. This bit, in conjunction with other bits, controls sending ERR_COR Messages.

5.4.6 Device Status Register

The Device Status register provides information about PCI Express device specific parameters.

Offset x‘007A’

Field Name	Bits	Type	9725 Value	Description
RSVD	15:6	RO	00	Reserved.
TRNS_PEND	5	RO	0	Transactions Pending. This bit when set indicates that the 9725 has issued Non- Posted Requests which have not been completed. The 9725 reports this bit cleared only when all outstanding Non-Posted Requests have completed or have been terminated by the Completion Timeout mechanism.
AUX_PWR_DET	4	RO	0	AUX Power Detected. This bit is set to 0 in the 9725, indicating the 9725 does not use AUX power.
UNSUP_REQ_DET	3	RW1C	0	Unsupported Request Detected. This bit indicates that the 9725 received an Unsupported Request. Errors are logged in this register regardless of whether error reporting is enabled or disabled in the Device Control register.
FTL_ERR_DET	2	RW1C	0	Fatal Error Detected. This bit indicates the status of the fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or disabled in the Device Control register. Errors are logged in this register regardless of the settings of the correctable error mask register in the Advanced Error Handling registers.
NON_FTL_ERR_DET	1	RW1C	0	Non-Fatal Error Detected. This bit indicates the status of the non-fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or disabled in the Device Control register. Errors are logged in this register regardless of the settings of the correctable error mask register in the Advanced Error Handling registers.

Field Name	Bits	Type	9725 Value	Description
COR_ERR_DET	0	RW1C	0	Correctable Error Detected. This bit indicates the status of the correctable errors detected. Errors are logged in this register regardless of whether error reporting is enabled or disabled in the Device Control register. Errors are logged in this register regardless of the settings of the correctable error mask register in the Advanced Error Handling registers.

5.4.7 Link Capabilities Register

The Link Capabilities register identifies PCI Express Link specific capabilities.

Offset x‘007C’

Field Name	Bits	Type	9725 Value	Description
Port Number	31:24	RO	0	Port Number. The 9725 has a single PCIe port; the port number is zero.
RSVD	23:18	RO	0	Reserved.
L1 Exit Latency	17:15	RO	0x6	L1 Exit Latency. Indicates the L1 exit latency for the Link (i.e., the length of time this Port requires to complete a transition from L1 to L0). The 9725 L1 Exit latency required is 32µs to 64µs.
L0 Exit Latency	14:12	RO	0x3	L0 Exit Latency. Indicates the L0s exit latency for the Link (i.e., the length of time this Port requires to complete a transition from L0s to L0). The 9725 L0 Exit latency required is 256ns to less than 512ns.
Active State Link PM Support	11:10	RO	0x3	Active State Power Management (ASPM) Support. Indicates the level of ASPM supported on this Link. The 9725 supports L0 and L1.
Max Link Width	9:4	RO	0x8	Max Link Width. The 9725 max link width is x8.
Max Link Speed	3:0	RO	0x1	Max Link Speed. 0001 = 2.5 Gb/s

5.4.8 Link Control Register

The Link Control register controls PCI Express Link specific parameters.

Offset x‘0080’

Field Name	Bits	Type	9725 Value	Description
RSVD	15:9	RO	0	Reserved.
EN_CLK_PWR_MAN	8	RW	0	Enable Clock Power Management. This bit is hardwired to 0 as the 9725 device does not support Clock Power Management.
EXT_SYNC	7	RW	0	Extended Synch. This bit when set forces the transmission of additional ordered sets when exiting the L0s state and when in the Recovery state. This bit is not used by the 9725.
CLK_CFG	6	RW	1	Common Clock Configuration. This bit when set indicates that the 9725 and the component at the opposite end of this Link are operating with a distributed common reference clock.
RTN_LINK	5	RW	0	Retrain Link. This field is not applicable and is reserved for the 9725 device.
LINK_DIS	4	RW	0	Link Disable. This field is not applicable and is reserved for the 9725 device.
RCB	3	RW	0	Read Completion Boundary. This bit is hardwired to 0 as the 9725 device does not support RCB.
RSVD	2	RO	0	Reserved.
ASPM_CTL	1:0	RW	00	Active State Power Management (ASPM) Control. This field controls the level of ASPM supported on the given PCI Express Link. 9725 supported values are: 00 = Disabled

5.4.9 Link Status Register

The Link Status register provides information about PCI Express Link specific parameters.

Offset x‘0082’

Field Name	Bits	Type	9725 Value	Description
RSVD	15:14	RO	0	Reserved.
DAT_LINK_ACT	13	RO	0	Data Link Layer Link Active. This bit indicates the status of the Data Link Control and Management State Machine. It returns a 1b to indicate the DL_Active state, 0b otherwise. This bit must be implemented if the corresponding Data Link Layer Active Capability bit is implemented. Otherwise, this bit must be hardwired to 0b.
SLT_CLK_CFG	12	HWINIT	1	Slot Clock Configuration. This bit indicates that the 9725 uses the same physical reference clock that the platform provides on the connector.
LINK_TRN	11	RO	0	Link Training. This field is not applicable to the 9725 device and is hardwared to zero.
UNDEF	10	RO	0	Undefined. This legacy bit is no longer used.
NEG_LINK_WDTH	9:4	RO		Negotiated Link Width. This field indicates the negotiated width of the given PCI Express Link. Defined encodings are: 000001 = x1 000010 = x2 000100 = x4 001000 = x8 (typical value) 001100 = x12 010000 = x16 100000 = x32 All other encodings are reserved. The value in this field is undefined when the Link is not up.
LINK_SP	3:0	RO	0001	Link Speed. This field indicates the negotiated Link speed of the given PCI Express Link. Defined encodings are: 0001b 2.5 Gb/s PCI Express Link

5.5 Advanced Error Reporting Capability Registers

The Advanced Error Reporting Capability registers is an optional extended capability that may be implemented by PCI Express devices supporting advanced error control and reporting.

ENH_CAP_ HEAD[31:0]

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

UNC_ERR_STAT[31:0]

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

UNC_ERR_MASK[31:0]

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

UNC_ERR_SEV[31:0]

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

COR_ERR_STAT[31:0]

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

COR_ERR_MASK[31:0]

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

ERR_CAP_CTL[31:0]

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

HEAD_LOG[127:96]

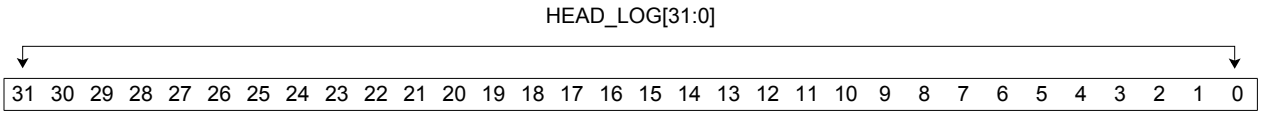
127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96

HEAD_LOG[95:64]

95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64

HEAD_LOG[63:32]

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32



5.5.1 Enhanced Capability Header Register

Offset x‘0100’

Field Name	Bits	Type	9725 Value	Description
NXT_CAP_OFF	31:20	RO	000h	Next Capability Offset. This field contains the offset to the next PCI Express capability structure or 000h if no other items exist in the linked list of capabilities. 000h = Terminating list of capabilities
CAP_VER	19:16	RO	1h	Capability Version. This field is the version number of the capability structure present.
EXT_CAP_ID	15:0	RO	0001h	PCI Express Extended Capability ID. The Extended Capability ID for the Advanced Error Reporting Capability is 0001h.

5.5.2 Uncorrectable Error Status Register

The Uncorrectable Error Status register indicates error detection status of individual errors on a 9725 device. An individual error status bit that is set indicates that a particular error was detected; software may clear an error status by writing a one to the respective bit.

Offset x‘0104’

Field Name	Bits	Type	9725 Value	Description
RSVD	31:21	RO	0	Reserved.
REQ_ERR_STAT	20	RW1CS	0	Unsupported Request Error Status.
ECRC_ERR_STAT	19	RW1CS	0	ECRC Error Status (Optional).
TLP_STAT	18	RW1CS	0	Malformed TLP Status.
RV_OVF_STAT	17	RW1CS	0	Receiver Overflow Status (Optional).
CMPL_STAT	16	RW1CS	0	Unexpected Completion Status.
CMPL_ABORT_STAT	15	RW1CS	0	Completer Abort Status (Optional).
CMPL_TMOUT_STAT	14	RW1CS	0	Completion Timeout Status.
FC_ERR_STAT	13	RW1CS	0	Flow Control Protocol Error Status (Optional).
POIS_TLP_STAT	12	RW1CS	0	Poisoned TLP Status.
RSVD	11:6	RO	0	Reserved.
DWN_ERR_STAT	5	RW1CS	0	Surprise Down Error Status (Optional).
DLNK_PROT_STAT	4	RW1CS	0	Data Link Protocol Error Status.
RSVD	3:1	RO	0	Reserved.
UNDEF	0	RO	0	Undefined.

5.5.3 Uncorrectable Error Mask Register

The Uncorrectable Error Mask register controls reporting of individual errors by the device to the PCI Express Root Complex via a PCI Express error Message. A masked error (respective bit set to 1b in the mask register) is not logged in the Header Log register, does not update the First Error Pointer, and is not reported to the PCI Express Root Complex by an individual device.

Offset x‘0108’

Field Name	Bits	Type	9725 Value	Description
RSVD	31:21	RO	0	Reserved.
REQ_ERR_MSK	20	RWS	0	Unsupported Request Error Mask.
ECRC_ERR_MSK	19	RWS	0	ECRC Error Mask (Optional).
TLP_MSK	18	RWS	1	Malformed TLP Mask.
RV_OVF_MSK	17	RWS	1	Receiver Overflow Mask (Optional).
CMPL_MSK	16	RWS	0	Unexpected Completion Mask.
CMPL_ABORTMSK	15	RWS	0	Completer Abort Mask (Optional).
CMPL_TMOUT_MSK	14	RWS	0	Completion Timeout Mask.
FC_ERR_MSK	13	RWS	1	Flow Control Protocol Error Mask (Optional).
POIS_TLP_MSK	12	RWS	0	Poisoned TLP Mask.
RSVD	11:6	RO	0	Reserved.
DWN_ERR_MSK	5	RWS	1	Surprise Down Error Mask (Optional).
DLNK_PROT_MSK	4	RWS	0	Data Link Protocol Error Mask.
RSVD	3:1	RO	0	Reserved.
UNDEF	0	RO	0	Undefined.

5.5.4 Uncorrectable Error Severity Register

The Uncorrectable Error Severity register controls whether an individual error is reported as a Nonfatal or Fatal error. An error is reported as fatal when the corresponding error bit in the severity register is set. If the bit is cleared, the corresponding error is considered non-fatal.

Offset x‘010C’

Field Name	Bits	Type	9725 Value	Description
RSVD	31:21	RO	0	Reserved.
REQ_ERR_SEV	20	RWS	0	Unsupported Request Error Severity.
ECRC_ERR_SEV	19	RWS	0	ECRC Error Severity (Optional).
TLP_SEV	18	RWS	1	Malformed TLP Severity.
RV_OVF_SEV	17	RWS	1	Receiver Overflow Severity (Optional).
CMPL_SEV	16	RWS	0	Unexpected Completion Severity.
CMPL_ABORT_SEV	15	RWS	0	Completer Abort Severity (Optional).
CMPL_TMOUT_SEV	14	RWS	0	Completion Timeout Severity.
FC_ERR_SEV	13	RWS	1	Flow Control Protocol Error Severity (Optional).
POIS_TLP_SEV	12	RWS	0	Poisoned TLP Severity.
RSVD	11:6	RO	0	Reserved.
DWN_ERR_SEV	5	RWS	1	Surprise Down Error Severity (Optional).
DLNK_PROT_SEV	4	RWS	1	Data Link Protocol Error Severity.
RSVD	3:1	RO	0	Reserved.
UNDEF	0	RO	0	Undefined.

5.5.5 Correctable Error Status Register

The Correctable Error Status register reports error status of individual correctable error sources on a 9725 device. When an individual error status bit is set, it indicates that a particular error occurred; software may clear an error status by writing a 1 to the respective bit.

Offset x‘0110’

Field Name	Bits	Type	9725 Value	Description
RSVD	31:14	RO	0	Reserved.
ADV_ERR_STAT	13	RW1CS	0	Advisory Non-Fatal Error Status.
REPLY_TMOUT_STAT	12	RW1CS	0	Replay Timer Timeout Status.
RSVD	11:9	RO	0	Reserved.
REPLAY_NUM_STAT	8	RW1CS	0	REPLAY_NUM Rollover Status.
BAD_DLLP_STAT	7	RW1CS	0	Bad DLLP Status.
BAD_TLP_STAT	6	RW1CS	0	Bad TLP Status.
RSVD	5:1	RO	0	Reserved.
RX_ERR_STAT	0	RW1CS	0	Receiver Error Status.

5.5.6 Correctable Error Mask Register

The Correctable Error Mask register controls reporting of individual correctable errors by the 9725 to the PCI Express Root Complex via a PCI Express error Message. A masked error (respective bit set in mask register) is not reported to the PCI Express Root Complex by an individual device.

Offset x‘0114’

Field Name	Bits	Type	9725 Value	Description
RSVD	31:14	RO	0	Reserved.
ADV_ERR_MSK	13	RWS	1	Advisory Non-Fatal Error Mask.
REPLY_TMOUT_MSK	12	RWS	0	Replay Timer Timeout Mask.
RSVD	11:9	RO	0	Reserved.
REPLAY_NUM_MSK	8	RWS	0	REPLAY_NUM Rollover Mask.
BAD_DLLP_MSK	7	RWS	0	Bad DLLP Mask.
BAD_TLP_MSK	6	RWS	0	Bad TLP Mask.
RSVD	5:1	RO	0	Reserved.
RX_ERR_MSK	0	RWS	0	Receiver Error Mask.

5.5.7 Advanced Error Capabilities and Control Register

Offset x‘0118’

Field Name	Bits	Type	9725 Value	Description
RSVD	31:9	RO	0	Reserved.
ECRC_CHK_CAP	8	RWS	0	ECRC Check Enable. This bit when set enables ECRC checking.
ECRC_CHK_CAP	7	RO	1	ECRC Check Capable. This bit indicates that the 9725 is capable of checking ECRC.
ECRC_EN	6	RWS	0	ECRC Generation Enable. This bit when set enables ECRC generation.
ECRC_CAP	5	RO	1	ECRC Generation Capable. This bit indicates that the 9725 is capable of generating ECRC
FRST_ERR_PTR	4:0	ROS	0	First Error Pointer. The First Error Pointer is a read-only register that identifies the bit position of the first error reported in the Uncorrectable Error Status register.

5.5.8 Header Log Register

The Header Log register captures the Header for the TLP corresponding to a detected error.

The header is captured such that the fields of the header read by software in the same way the headers are presented in this document, when the register is read using DW accesses. Therefore, byte 0 of the header is located in byte 3 of the Header Log register, byte 1 of the header is in byte 2 of the Header Log register and so forth. For 12-byte headers, only bytes 0 through 11 of the Header Log register are used and values in bytes 12 through 15 are undefined.

The allocation of register fields in the Header Log register is shown below.

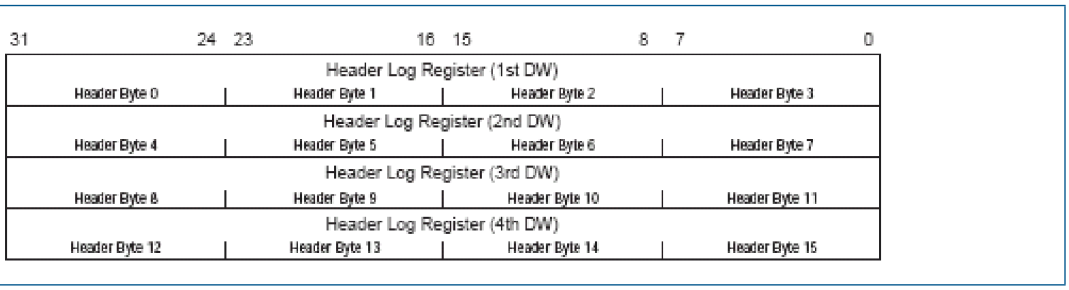


Figure 5-2. Header Log Register Layout

Offset x‘011C’

Field Name	Bits	Read/ Write	9725 Value	Description
TLP_ERR_HDR	127:0	ROS		Header of TLP associated with error.

6 9725 Register Definition

This section describes the 9725 internal registers for those customers who wish to write their own software. Exar's 9725 SDK eliminates the user from having to understand the 9725 internal register details.

The host should not read from or write to reserved registers; doing so may cause unexpected behavior from the device. Writing to reserved fields within a defined register will be ignored; reading from a reserved field within a defined register will return a zero.

Table 6-1 lists the register type definitions used to describe the 9725 internal registers in this chapter. Fields marked as 'Read Only' usually indicate the 9725 capability and may not be altered by host; fields marked as 'Read/Write' may be altered by the host (usually BIOS or OS) for special purposes.

Table 6-1. Register Type Definitions

Register Type	Description
RO	Read only. Register bits are read-only and cannot be altered by software.
RW	Read/Write. Register bits are read-write and may be either set or cleared by software to the desired state.
RW1C	Read-only status, Write-1-to-clear status. Register bits indicate status when read, a set bit indicating a status event may be cleared by writing a 1. Writing a 0 to RW1C bits has no effect.

Figure 6-1 illustrates the 9725 register map.

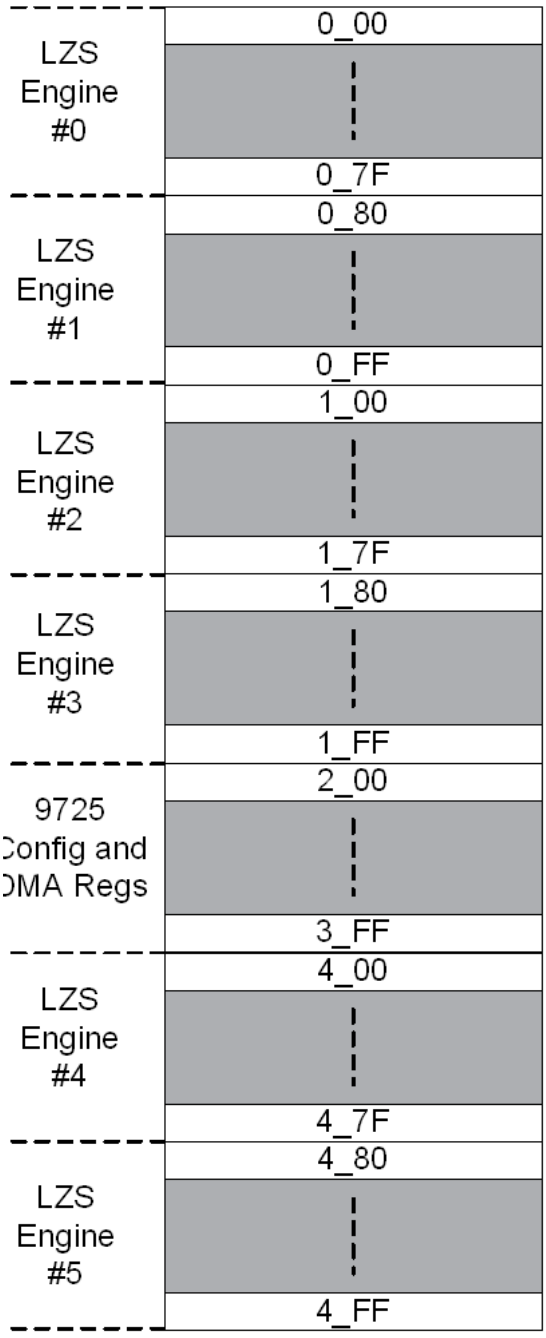


Figure 6-1. 9725 Memory Map

The registers are mapped into 4K bytes of PCIe configuration space that can be accessed through the PCIe bus. Each LZS engine is allocated 128 byte memory space for its configuration registers. For backward compatibility with the Exar legacy VTL software, the address space of LZS engine 4 and LZS engine 5 is allocated to 0x400 to 0x4FF respectively. Throughout this section, the offset values for each register are given in hex.

Table 6-2 details the complete 9725 register list.

Table 6-2. Complete 9725 Register List

Register Name	Description	Type	Address Offset
RSVD	Reserved	RW	0x000 - 0x021
LZS0_CONFIG	LZS Configuration Register	RW	0x022 - 0x023
LZS0_IE	LZS Interrupt Enable Register	RW	0x024 - 0x025
RSVD	Reserved	RW	0x026 - 0x029
LZS0_FIFO	FIFO Configuration Register	RW	0x02A - 0x02D
RSVD	Reserved	RW	0x02E - 0x0A1
LZS1_CONFIG	LZS Configuration Register	RW	0x0A2 - 0x0A3
LZS1_IE	LZS Interrupt Enable Register	RW	0x0A4 - 0x0A5
RSVD	Reserved	RW	0x0A6 - 0x0A9
LZS1_FIFO	FIFO Configuration Register	RW	0x0AA - 0x0AD
RSVD	Reserved	RW	0x0AE - 0x121
LZS2_CONFIG	LZS Configuration Register	RW	0x122 - 0x123
LZS2_IE	LZS Interrupt Enable Register	RW	0x124 - 0x125
RSVD	Reserved	RW	0x126 - 0x129
LZS2_FIFO	FIFO Configuration Register	RW	0x12A - 0x12D
RSVD	Reserved	RW	0x12E - 0x1A1
LZS3_CONFIG	LZS Configuration Register	RW	0x1A2 - 0x1A3
LZS3_IE	LZS Interrupt Enable Register	RW	0x1A4 - 0x1A5
RSVD	Reserved	RW	0x1A6 - 0x1A9
LZS3_FIFO	FIFO Configuration Register	RW	0x1AA - 0x1AD
RSVD	Reserved	RW	0x1AE - 0x1FF
CPR0_BASEADR	Command Ring / Command Pointer Ring 0 Base Address Register	RW	0x200 - 0x207
RSVD	Reserved	RW	0x208 - 0x20F
RR0_BADDR	Result Ring 0 Base Address Register	RW	0x210 - 0x217
RR0_WP	Result Ring 0 Write Pointer Register	RW	0x218 - 0x21B
RSVD	Reserved	RW	0x21C - 0x21F
DMA_CONFIG	DMA Configuration Register	RW	0x220 - 0x223
CR0_WP	Command Ring 0 Write Pointer Register	RW	0x224 - 0x227
SW_CTL	Software Control Register	RW	0x228 - 0x22B
CPR0_RP	Command Pointer Ring 1 Read Pointer Register	RW	0x22C - 0x22F
RSVD	Reserved	RW	0x230 - 0x23F
ISLAND_RST	Island Reset Register	RW	0x240 - 0x243
EM0_STATUS	Engine Manager 0-5 Status Register	RO	0x244 - 0x247
EM1_STATUS	Engine Manager 0-5 Status Register	RO	0x248 - 0x24B
EM2_STATUS	Engine Manager 0-5 Status Register	RO	0x24C - 0x24F

Table 6-2. Complete 9725 Register List

Register Name	Description	Type	Address Offset
EM3_STATUS	Engine Manager 0-5 Status Register	RO	0x250 - 0x253
RSVD	Reserved	RW	0x254 - 0x283
INT_STATUS	Interrupt Status Register	RO	0x284 - 0x287
INT_EN	Interrupt Enable Register	RW	0x288 - 0x28B
RSVD	Reserved	RW	0x28C - 0x29B
CFG1	Config1 Register	RW	0x29C - 0x29F
FLASH_WR_EN	Flash Write Enable Register	RW	0x2A0 - 0x2A3
FLASH_ADDR	Flash Address Register	RW	0x2A4 - 0x2A7
FLASH_DATA	Flash Data Register	RW	0x2A8 - 0x2AB
CFG2	Config2 Register	RW	0x2AC - 0x2AF
FLASH_STATUS	Flash Status Register	RO	0x2B0 - 0x2B3
RSVD	Reserved	RW	0x2B4 - 0x2FB
CFG3	Config3 Register	RW	0x2FC - 0x2FF
RSVD	Reserved	RW	0x300 - 0x343
9725_STATUS	9725 Status Register	RO	0x344 - 0x347
RSVD	Reserved	RW	0x348 - 0x3BF
EM4_STATUS	Engine Manager 0-5 Status Register	RO	0x3C0 - 0x3C3
EM5_STATUS	Engine Manager 0-5 Status Register	RO	0x3C4 - 0x3C7
CPR1_BASEADR	Command Ring / Command Pointer Ring 1 Base Address Register	RW	0x3C8 - 0x3CF
RSVD	Reserved	RW	0x3D0 - 0x3D7
RR1_BADDR	Result Ring 1 Base Address Register	RW	0x3D8 - 0x3DF
RR1_WP	Result Ring 1 Write Pointer Register	RO	0x3E0 - 0x3E3
CR1_WP	Command Ring 1 Write Pointer Register	RW	0x3E4 - 0x3E7
CPR1_RP	Command Pointer Ring 1 Read Pointer Register	RW	0x3E8 - 0x3EB
EM4/5_EN	Engine Manager 4/5 Enable Register	RW	0x3F0 - 0x3F3
CMD_ADDR_MODE	Command Addressing Mode Register	RW	0x3F4 - 0x3F7
RR_MODE	Result Ring Mode Register	RW	0x3F8 - 0x3FB
RSVD	Reserved	RW	0x3FC - 0x421
LZS4_CONFIG	LZS Configuration Register	RW	0x422 - 0x423
LZS4_IE	LZS Interrupt Enable Register	RW	0x424 - 0x425
RSVD	Reserved	RW	0x426 - 0x429
LZS4_FIFO	FIFO Configuration Register	RW	0x42A - 0x42D
RSVD	Reserved	RW	0x42E - 0x4A1
LZS5_CONFIG	LZS Configuration Register	RW	0x4A2 - 0x4A3
LZS5_IE	LZS Interrupt Enable Register	RW	0x4A4 - 0x4A5

Table 6-2. Complete 9725 Register List

Register Name	Description	Type	Address Offset
RSVD	Reserved	RW	0x4A6 - 0x4A9
LZS5_FIFO	FIFO Configuration Register	RW	0x4AA - 0x4AD
RSVD	Reserved	RW	0x4AE - 0x4FF

6.1 LZS Engine Registers

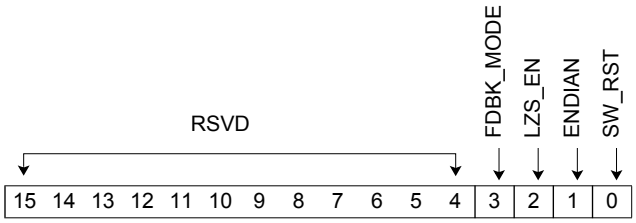
This section describes the LZS Engine registers. All six LZS Engines are configured in the same manner, however only one register may be accessed at a time.

6.1.1 LZS Configuration Register

The LZS configuration register is shown below. The Host must write to this register to configure the LZS Engine managers during system initialization.

NOTE: Setting the software reset (bit 0) of this register will reset all logic associated with the specified LZS engine except for this register and its FIFO configuration register. When the reset is complete, the 9725 will clear the software reset bit.

Offset	x'0022'	LZS Engine #0
	x'00A2'	LZS Engine #1
	x'0122'	LZS Engine #2
	x'01A2'	LZS Engine #3
	x'0422'	LZS Engine #4
	x'04A2'	LZS Engine #5



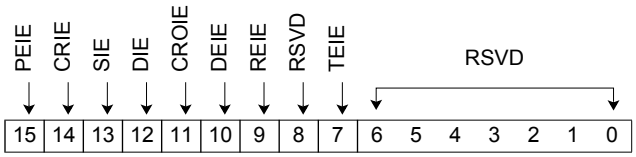
Field Name	Bits	Type	Reset	Description
RSVD	15:4	RW	0x00	Reserved.
FDBK_MODE	3	RW	1	Feedback Mode. Should be set to one if record CRC is enabled. This bit is meaningful only if the LZS_EN bit is set to 1 and the operation command is compression. If this bit is set, the compressed output stream from compression engine will be fed into the decompression engine. The decompression engine outputs the decompressed data and checks the record CRC. If the record CRC shows a mismatch, a CRC error flag will be generated. If this bit is not set, this data checking mechanism is disabled. If LZS_EN is set to 0 (eLZS mode) and RCRC is enabled, feedback is automatically enabled regardless of the state of this bit. 0 Feedback mode disabled 1 Output of compression engine fed back into decompression engine

Field Name	Bits	Type	Reset	Description
LZS_EN	2	RW	0	LZS / Enhanced LZS Enable. 0 eLZS enabled 1 LZS enabled
ENDIAN	1	RW	0	Endian format. 0 Little endian 1 Reserved
SW_RST	0	RW	0	Software reset. 0 Do not initiate a software reset 1 Initiate a software reset

6.1.2 LZS Interrupt Enable Register

The LZS Interrupt Enable register is shown below. The Host must write to this register during system initialization.

Offset	x'0024'	LZS Engine #0
	x'00A4'	LZS Engine #1
	x'0124'	LZS Engine #2
	x'01A4'	LZS Engine #3
	x'0424'	LZS Engine #4
	x'04A4'	LZS Engine #5



Field Name	Bits	Type	Reset	Description
PEIE	15	RW	0	Parity Error Interrupt Enable. This interrupt must be enabled by the host during initialization. 0 Parity error interrupt disabled 1 Parity error interrupt enabled
CRIE	14	RW	0	Command Ready Interrupt Enable. This interrupt must be disabled by the host during initialization. 0 Command ready interrupt disabled 1 Command ready interrupt enabled
SIE	13	RW	0	Source Data Request Interrupt Enable. This interrupt must be disabled by the host during initialization. 0 Source Data Request interrupt disabled 1 Source Data Request interrupt enabled
DIE	12	RW	0	Destination Data Request Interrupt Enable. This interrupt must be disabled by the host during initialization. 0 Destination Data Request interrupt disabled 1 Destination Data Request interrupt enabled

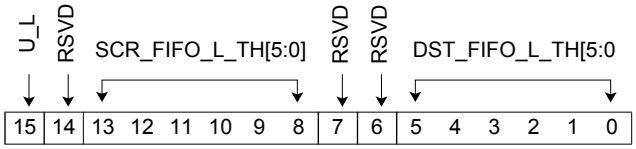
Field Name	Bits	Type	Reset	Description
CROIE	11	RW	0	Command/Result Overrun Interrupt Enable. This interrupt must be enabled by the host during initialization. 0 Command/Result Overrun interrupt disabled 1 Command/Result Overrun interrupt enabled
DEIE	10	RW	0	Data Error Interrupt Enable. This interrupt must be enabled by the host during initialization. 0 Data Error interrupt disabled 1 Data Error interrupt enabled
REIE	9	RW	0	Record CRC Error Interrupt Enable. This interrupt must be enabled by the host during initialization. 0 Record CRC Error interrupt disabled 1 Record CRC Error interrupt enabled
RSVD	8	RW	0	Reserved.
TEIE	7	RW	0	Token Error Interrupt Enable. This interrupt must be enabled by the host during initialization. 0 Token Error interrupt disabled 1 Token Error interrupt enabled
RSVD	6:0	RW	0000000	Reserved.

6.1.3 FIFO Configuration Register

The 9725 FIFO configuration register must be programmed by the host during initialization. A threshold value of 0x10is required for proper operation.

FIFO Configuration Word 1:

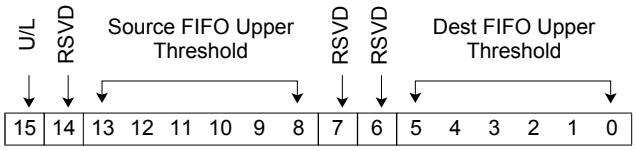
Offset	x‘002A’	LZS Engine #1
	x‘00AA’	LZS Engine #2
	x‘012A’	LZS Engine #3
	x‘01AA’	LZS Engine #4
	x‘042A’	LZS Engine #5
	x‘04AA’	LZS Engine #6



Field Name	Bits	Type	Reset	Description
U_L	15	RW	0	Upper/Lower Select. 0 Lower 1 Upper
RSVD	14	RW	0	Reserved.
SCR_FIFO_L_TH [5:0]	13:8	RW	0x02	Source FIFO Lower Threshold. The host must initialize this field to 0x10.
RSVD	7:6	RW	0	Reserved.
DST_FIFO_L_TH [5:0]	5:0	RW	0x02	Destination FIFO Lower Threshold. The host must initialize this field to 0x10.

FIFO Configuration Word 2:

Offset	x‘002C’	LZS Engine #1
	x‘00AC’	LZS Engine #2
	x‘012C’	LZS Engine #3
	x‘01AC’	LZS Engine #4
	x‘042C’	LZS Engine #5
	x‘04AC’	LZS Engine #6



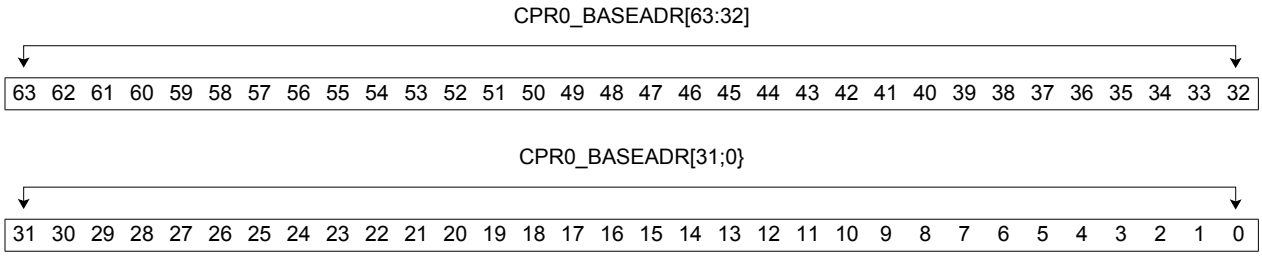
Field Name	Bits	Type	Reset	Description
U/L	15	RW	1	Upper/Lower Select. 0 = Lower 1 = Upper
RSVD	14	RW	0	Reserved.
SCR_FIFO_U_TH [11:6]	13:8	RW	0x02	Source FIFO Upper Threshold. The host must initialize this field to 0x10.
RSVD	7:6	RW	0	Reserved
DST_FIFO_L_TH [11:6]	5:0	RW	0x02	Destination FIFO Upper Threshold. The host must initialize this field to 0x10.

6.2 DMA Configuration Registers

6.2.1 Command Ring / Command Pointer Ring 0 Base Address Register

This is the base address of the Command Ring 0 in direct command addressing mode, and is the base address of the Command *Pointer* Ring 0 in indirect command addressing mode. The 9725 uses the base address plus an offset to fetch a command in direct command addressing mode, and uses the command address pointer to fetch a command in indirect addressing mode. The base address is aligned on the total command ring size boundary in direct command addressing mode, and is aligned on an 8 byte boundary in indirect command addressing mode.

Offset	x'0200'	CPR0_BASEADR[31:0]
	x'0204'	CPR0_BASEADR[63:32]



Field Name	Bits	Type	Reset Value	Description
CPR0_BASEADR [63:0]	63:0	RW	0x0000 0000 0000 0000	Command Pointer Ring 0 Base Address. This is the base address of Command Ring 0 in direct command addressing mode, and it is the base address of the Command *Pointer* Ring 0 in indirect command addressing mode.

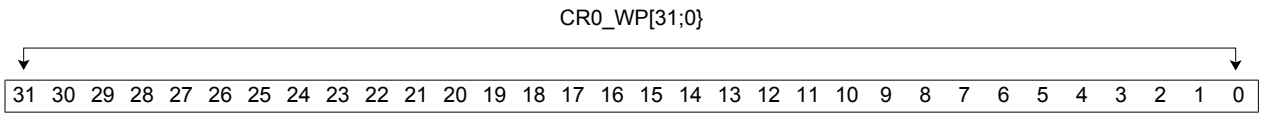
6.2.2 Command Ring 0 Write Pointer Register

The 9725 Command Ring 0 Write Pointer register must be updated by the host whenever a new command is submitted to Command Ring 0.

The 9725 Command Ring 0 Write Pointer register stores the index number (i.e., 1, 2, 3…) of the corresponding command ring write pointer in host memory. The host updates this register whenever a new command is submitted to Command Ring 0. The 9725 uses the stored index number, i.e., 1, 2, ,3, to refer to the Command ring/Command Pointer Ring 0 Base Address register (Section 6.2.1), and calculate the actual address of the newly submitted command structure in Command Ring 0.

Please refer to Section 3.5, "Command /Result Ring Example" for more information about this register.

Offset x'0224' CR0_WP[31:0]



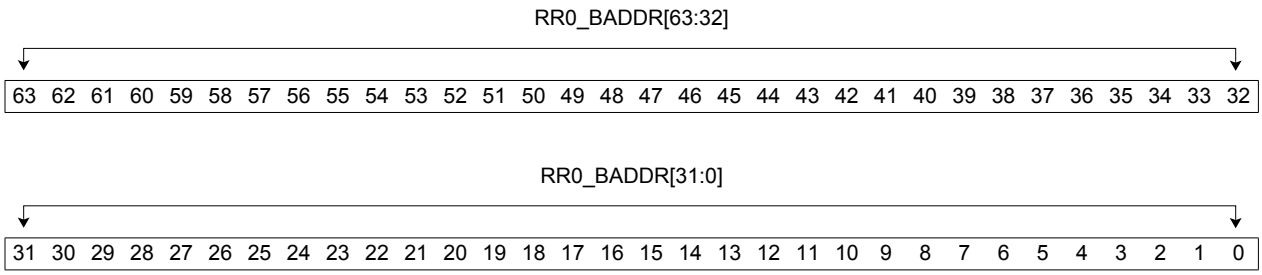
Field Name	Bits	Type	Reset Value	Description
CR0_WP[31:0]	31:0	RW	0x0000 0000	Command Ring 0 Write Pointer.

6.2.3 Result Ring 0 Base Address Register

The Result Ring 0 Base Address register is used to store the Result Ring base address in host memory. When the 9725 completes a command, it will write the command result into the result ring. The base address is aligned on the result ring size boundary.

The host must configure this register during system initialization.

Offset x‘0210’ RR0_BADDR[31:0]
x‘0214’ RR0_BADDR[63:32]

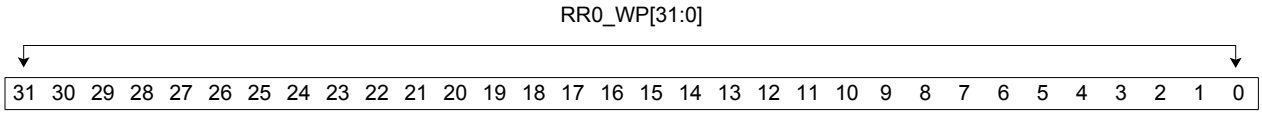


Field Name	Bits	Type	Reset	Description
RR0_BADDR [63:0]	63:0	RW	0x0000 0000 0000 0000	Result Ring 0 Base Address. The register is used to store the base address of Command Result Ring 0.

6.2.4 Result Ring 0 Write Pointer Register

The Result Ring 0 Write Pointer register may be used by the host to examine the Result Ring Write Pointer. This is a read-only register.

Offset x‘0218’

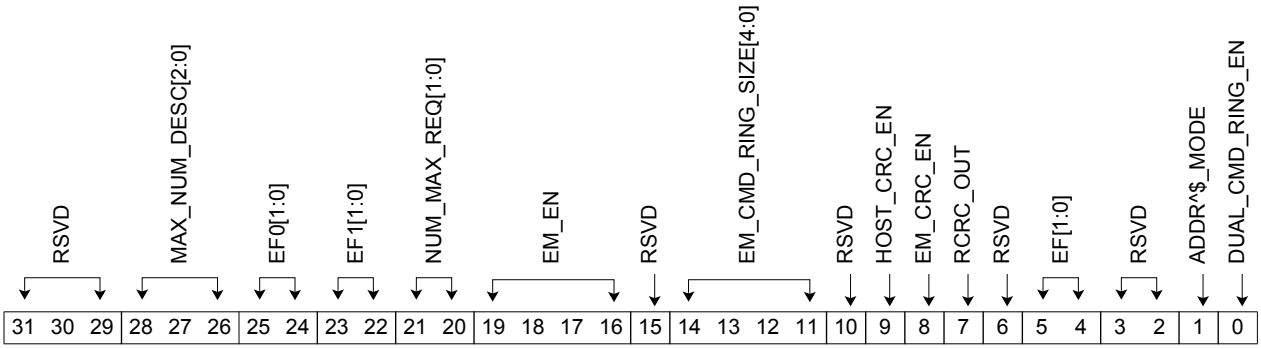


Field Name	Bits	Type	Reset	Description
RR0_WP [31:0]	31:0	RO	0x0000 0000 0000 0000	Result Ring 0 Write Pointer. This read only register may be read by the host to examine the Result Ring 0 Write Pointer.

6.2.5 DMA Configuration Register

This register is used to set the DMA configuration parameters. The host must configure this register during system initialization.

Offset x‘0220’



Field Name	Bits	Type	Reset Value	Description
RSVD	31:29		0	Reserved.
MAX_NUM_DESC [2:0]	28:26	RW	000	Maximum Number of Descriptors. The value for this field depends on the addressing mode. In direct command addressing mode, this field indicates the maximum number of descriptors in the command structure. 000 =8 001 =16 010 =32 011 =64 100 =128 101 =256 110 =512 111 =Reserved In indirect command addressing mode, the there is no limit on the number of descriptor; software must set the “last” flag of the last descriptor. Note: For 32-bit addressing, when the including write key command, MAX_NUM_DESC must be set >= 16. Note: The value of MAX_NUM_DESC applies to both command ring 0 and command ring 1.

Field Name	Bits	Type	Reset Value	Description
EF0[1:0]	25:24	RW	00	Descriptor Endian Format 0. If the endianess pointer equals 0 in Desc_src/Desc_dst (Section 3.3.3), EF0 specifies the 2-bit Endian Format value of the descriptor. See Endian Settings for more information. 00 No swap 01 Byte swap in word 10 Word swap, no byte swap in word 11 Word swap and byte swap in word
EF1[1:0]	23:22	RW	00	Descriptor Endian Format 1. If the endianess pointer equals 1 in Desc_src/Desc_dst, EF1 specifies the 2-bit Endian Format value of the descriptor. See Endian Settings for more information. 00 No swap 01 Byte swap in word 10 Word swap, no byte swap in word 11 Word swap and byte swap in word

Field Name	Bits	Type	Reset Value	Description
NUM_MAX_REQ[1:0]	21:20	RW	11	Number of Maximum Outstanding Requests for each Engine Manger. The meaning of this field depends on whether extended flags are supported on the host PCIe Root Complex. There is one tag for each descriptor fetch, the remaining tags are for data fetch. If the host does not support extended tags: 00 2 (supports max_read_request_size = 128, 256, 512) 01 3 (supports max_read_request_size = 128, 256, 512) 10 5 (supports max_read_request_size = 128, 256, 512) 11 5 (supports max_read_request_size = 128, 256, 512) If the host supports extended tags: 00 5 (supports max_read_request_size = 128, 256, 512) 01 9 (supports max_read_request_size = 128, 256, 512) 10 17 (supports max_read_request_size = 128, 256, 512) 11 17 (supports max_read_request_size = 128, 256) Note: when max_read_request_size = 512 bytes and max_req = 10, the Engine Manager will automatically use 8 tags (NOT 16) for data fetch because the source buffer is 4K bytes which can accommodate 8 packets with size 512 bytes).
EM_EN[3:0]	19:16	RW	0000	Engine Manager Enable. This field may be used to enable or disable the first four 9725 Engine Managers. 1000 =Engine Manager 3 enabled to fetch commands 0100 =Engine Manager 2 enabled to fetch commands 0010 =Engine Manager 1 enabled to fetch commands 0001 =Engine Manager 0 enabled to fetch commands All other values reserved.
RSVD	15		0	Reserved.

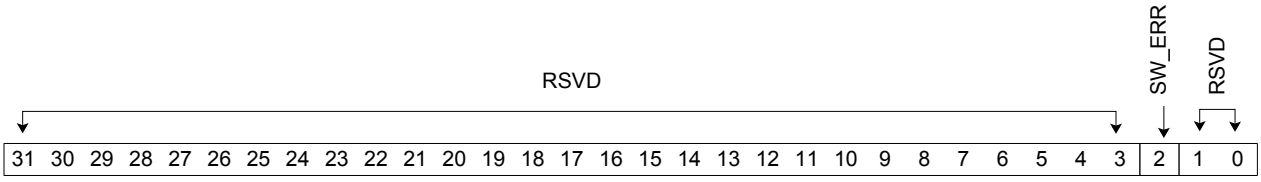
Field Name	Bits	Type	Reset Value	Description
EM_CMD_RING_ SIZE[4:0]	14:11	RW	00000	Engine Manager Command Ring Size. This field indicates the maximum number of command structures in each Engine Manager's command ring. 0000 =32 0001 =64 0010 =128 0011 =256 0111 =512 1000 =1K 1001 =2K 1010 =4K 1011 =8K 1100 =16K 1101 - 1111 Reserved In indirect command addressing mode, this field also indicates the command pointer ring size. Note: The value of for EM_CMD_RING_SIZE[4:0] is the same for command ring 0 and command ring 1.
RSVD	10		0	Reserved.
HOST_CRC_EN	9	RW	0	Host CRC Enable. This bit is mutually exclusive with Engine Manager CRC enable. 0 Host does not generate CRC 1 Host generates the CRC Refer to Section 2.3 for more information.
EM_CRC_EN	8	RW	0	Engine Manager CRC Enable. This bit is mutually exclusive with Host CRC enable. 0 9725 does not generate the CRC 1 9725 generates the CRC Refer to Section 2.3 for more information.
RCRC_OUT	7	RW	0	RCRC Output Select. This bit controls whether the 9725 strips the four CRC bytes from the data stream. This bit only applies to decode operations (decompress, decrypt) with CRC enabled (RAW + CRC). 0 9725 strips the four CRC bytes from the data stream; the output will be "RAW" data 1 9725 does not strip the four CRC bytes from the data stream; the output will be "RAW + CRC" Refer to Section 2.3 for more information.

Field Name	Bits	Type	Reset Value	Description
RSVD	6	RW	0	Reserved.
EF	5:4	RW	0	Command/Result Ring Endian Format. This field may be used to set the endian format of the host command ring and result ring. See Endian Settings for more information. 00 No swap 01 Byte swap in word 10 Word swap, no byte swap in word 11 Word swap and byte swap in word
RSVD	3:2	RW	0	Reserved.
ADDR64_MODE	1	RW	1	Address Mode. This field is used to set the DMA addressing mode/ 0 32-bit address mode 1 64-bit address mode
DUAL_CMD_RING_EN	0	RW	0	Dual Command Ring Enable. This field may be used to enable dual command ring mode. 0 Dual command mode disabled 1 Dual command mode enabled Note: If dual command ring mode is enabled, the host must set EM_RD_CRWP_EN to zero.

6.2.6 Software Control Register

The Software Control register may be used by a custom user application to control the external 9725 pin D13, CHIP_ERR.

Offset x‘0228’

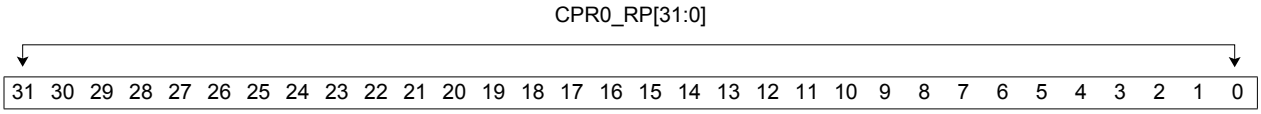


Field Name	Bits	Type	Reset	Description
RSVD	31:3	RW	0x0000 0000	Reserved.
SW_ERR	2	RW	0	Software Error. This bit may be used to control the external 9725 pin, CHIP_ERR. 0 Turn off the CHIP_ERR LED 1 Turn on the CHIP_ERR LED
RSVD	1:0	RW	00	Reserved.

6.2.7 Command Pointer Ring 0 Read Pointer Register

This register is the 9725 command pointer ring 0 read pointer. The host may read this register to determine how many entries in the command pointer ring 0 have been read by the 9725.

Offset x‘022C’

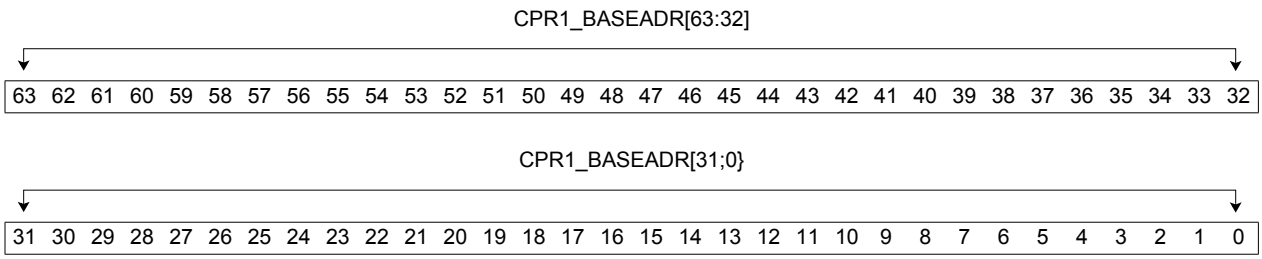


Field Name	Bits	Type	Reset	Description
CPR0_RP	31:0	RO	0x0000 0000	Command pointer ring 0 read pointer

6.2.8 Command Ring / Command Pointer Ring 1 Base Address Register

This is the base address of the Command Ring 1 in direct command addressing mode, and is the base address of the Command *Pointer* Ring 1 in indirect command addressing mode. The 9725 uses the base address plus an offset to fetch a command in direct command addressing mode, and uses the comm9725and address pointer to fetch a command in indirect addressing mode. The base address is aligned on the total command ring size boundary in direct command addressing mode, and is aligned on an 8 byte boundary in indirect command addressing mode.

Offset	x'03C8'	CPR1_BASEADR[31:0]
	x'03CC'	CPR1_BASEADR[63:32]



Field Name	Bits	Type	Reset Value	Description
CPR1_BASEADR [63:0]	63:0	RW	0x0000 0000 0000 0000	Command Pointer Ring 1 Base Address. This is the base address of Command Ring 1 in direct command addressing mode, and it is the base address of the Command *Pointer* Ring 1 in indirect command addressing mode.

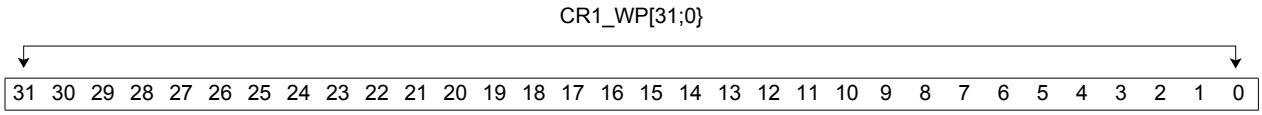
6.2.9 Command Ring 1 Write Pointer Register

The 9725 Command Ring 1 Write Pointer register must be updated by the host whenever a new command is submitted to Command Ring 1.

The 9725 Command Ring 1 Write Pointer register stores the index number (i.e., 1, 2, 3…) of the corresponding command ring write pointer in host memory. The host updates this register whenever a new command is submitted to Command Ring 1. The 9725 uses the stored index number, i.e., 1, 2, ,3, to refer to the Command ring/Command Pointer Ring 1 Base Address register (Section 6.2.8), and calculate the actual address of the newly submitted command structure in Command Ring 1.

Please refer to Section 3.5, "Command /Result Ring Example" for more information about this register.

Offset x'03E4'



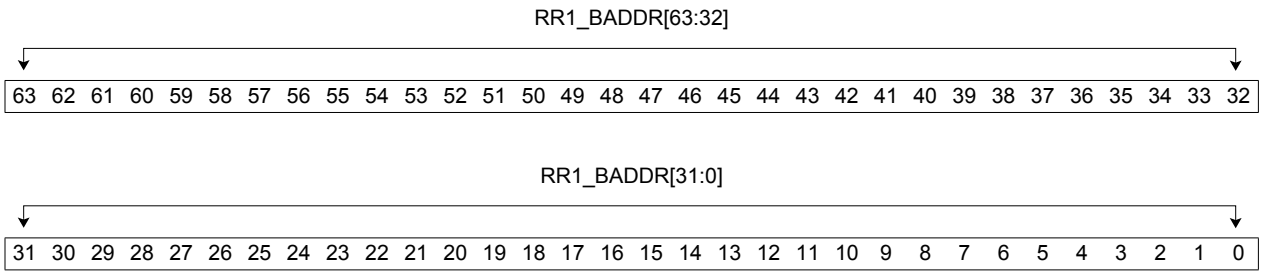
Field Name	Bits	Type	Reset Value	Description
CR1_WP[31:0]	31:0	RW	0x0000 0000	Command Ring 1 Write Pointer.

6.2.10 Result Ring 1 Base Address Register

The Result Ring 1 Base Address register is used to store the Result Ring base address in host memory. When the 9725 completes a command, it will write the command result into the result ring. The base address is aligned on the result ring size boundary.

The host must configure this register during system initialization.

Offset x‘03D8’ RR1_BADDR[31:0]
x‘03DC’ RR1_BADDR[63:32]

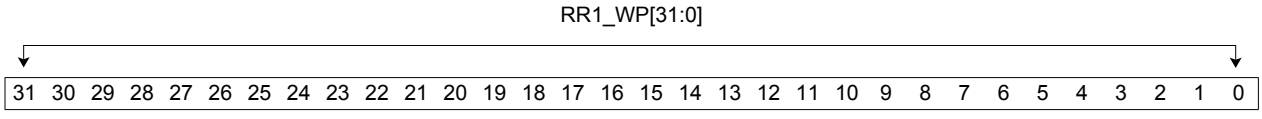


Field Name	Bits	Type	Reset	Description
RR1_BADDR [63:0]	63:0	RW	0x0000 0000 0000 0000	Result Ring 1 Base Address. The register is used to store the base address of Command Result Ring 1.

6.2.11 Result Ring 1 Write Pointer Register

The Result Ring 1 Write Pointer register may be used by the host to examine the Result Ring Write Pointer. This is a read-only register.

Offset x‘03E0’

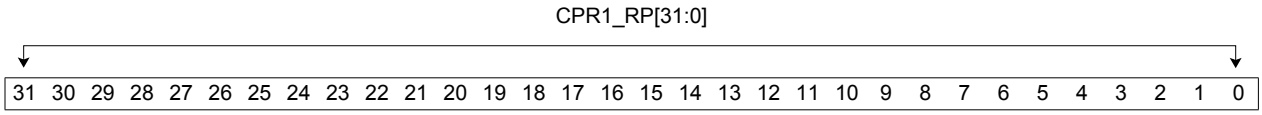


Field Name	Bits	Type	Reset	Description
RR1_WP [31:0]	31:0	RO	0x0000 0000	Result Ring 1 Write Pointer. This read only register may be read by the host to examine the Result Ring 1 Write Pointer.

6.2.12 Command Pointer Ring 1 Read Pointer Register

This register is the 9725 command pointer ring 1 read pointer. The host may read this register to determine how many entries in the command pointer ring 1 have been read by the 9725.

Offset x‘03E8’



Field Name	Bits	Type	Reset	Description
CPR1_RP	31:0	RO	0x0000 0000	Command pointer ring 1 read pointer

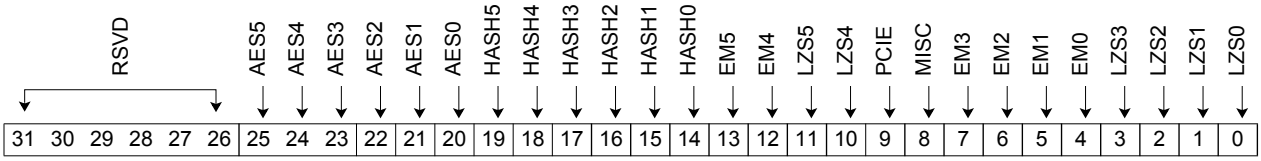
6.3 Status and Control Registers

6.3.1 Island Reset Register

The Island Reset register may be used to reset individual portions of the 9725 device.

The Engine manager and miscellaneous resets will clear the logic after one clock cycle; the 9725 resets will be automatically cleared after the 9725 reset is deactivated. After applying a reset, the host may poll this register to verify that the reset is complete. If the MISC reset bit is asserted, all registers and the PCIe Input, I/O Controller, Read Controller, and Write Controller modules will be reset.

Offset x'0240'



Field Name	Bits	Type	Reset Value	Description
RSVD	31:26	RW	000000	Reserved.
AES5	25	RW	0	AES Engine 5 Reset. 0 Do not reset AES Engine 5 1 Reset AES Engine 5
AES4	24	RW	0	AES Engine 4 Reset. 0 Do not reset AES Engine 4 1 Reset AES Engine 4
AES3	23	RW	0	AES Engine 3 Reset. 0 Do not reset AES Engine 3 1 Reset AES Engine 3
AES2	22	RW	0	AES Engine 2 Reset. 0 Do not reset AES Engine 2 1 Reset AES Engine 2
AES1	21	RW	0	AES Engine 1 Reset. 0 Do not reset AES Engine 1 1 Reset AES Engine 1
AES0	20	RW	0	AES Engine 0 Reset. 0 Do not reset AES Engine 0 1 Reset AES Engine 0

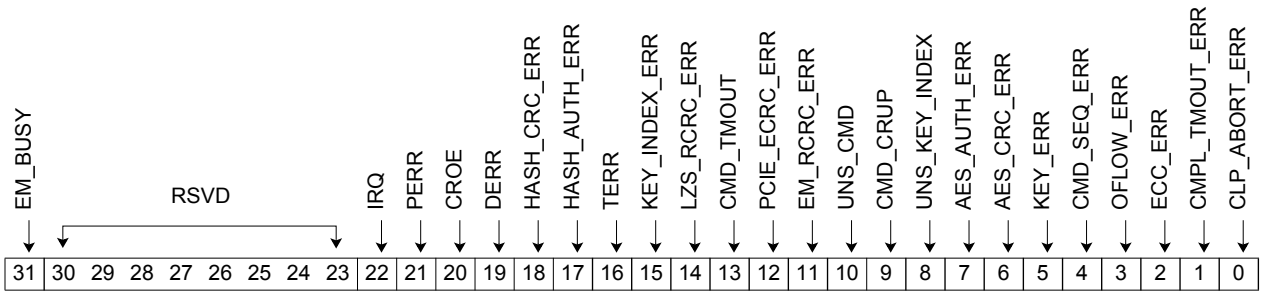
Field Name	Bits	Type	Reset Value	Description
HASH5	19	RW	0	Hash Engine 5 Reset. 0 Do not reset Hash Engine 5 1 Reset Hash Engine 5
HASH4	18	RW	0	Hash Engine 4 Reset. 0 Do not reset Hash Engine 4 1 Reset Hash Engine 4
HASH3	17	RW	0	Hash Engine 3 Reset. 0 Do not reset Hash Engine 3 1 Reset Hash Engine 3
HASH2	16	RW	0	Hash Engine 2 Reset. 0 Do not reset Hash Engine 2 1 Reset Hash Engine 2
HASH1	15	RW	0	Hash Engine 1 Reset. 0 Do not reset Hash Engine 1 1 Reset Hash Engine 1
HASH0	14	RW	0	Hash Engine 0 Reset. 0 Do not reset Hash Engine 0 1 Reset Hash Engine 0
EM5	13	RW	0	Engine Manager 5 Reset. 0 Do not reset Engine Manager 5 1 Reset Engine Manager 5
EM4	12	RW	0	Engine Manager 4 Reset. 0 Do not reset Engine Manager 4 1 Reset Engine Manager 4
LZS5	11	RW	0	LZS Engine 5 Reset. 0 Do not reset LZS Engine 5 1 Reset LZS Engine 5
LZS4	10	RW	0	LZS Engine 4 Reset. 0 Do not reset LZS Engine 4 1 Reset LZS Engine 4
PCIE	9	RW	0	PCIe Reset. 0 Do not reset PCIe interface 1 Reset PCIe interface
MISC	8	RW	0	Miscellaneous Reset. 0 Do not apply MISC reset 1 Reset all registers and the PCIe Input, I/O Controller, Read Controller, and Write Controller modules

Field Name	Bits	Type	Reset Value	Description
EM3	7	RW	0	Engine Manager 3 Reset. 0 Do not reset Engine Manager 3 1 Reset Engine Manager 3
EM2	6	RW	0	Engine Manager 2 Reset. 0 Do not reset Engine Manager 2 1 Reset Engine Manager 2
EM1	5	RW	0	Engine Manager 1 Reset. 0 Do not reset Engine Manager 1 1 Reset Engine Manager 1
EM0	4	RW	0	Engine Manager 0 Reset. 0 Do not reset Engine Manager 0 1 Reset Engine Manager 0
LZS3	3	RW	0	LZS Engine 3 Reset. 0 Do not reset LZS Engine 3 1 Reset LZS Engine 3
LZS2	2	RW	0	LZS Engine 2 Reset. 0 Do not reset LZS Engine 2 1 Reset LZS Engine 2
LZS1	1	RW	0	LZS Engine 1 Reset. 0 Do not reset LZS Engine 1 1 Reset LZS Engine 1
LZS0	0	RW	0	LZS Engine 0 Reset. 0 Do not reset LZS Engine 0 1 Reset LZS Engine 0

6.3.2 Engine Manager 0-5 Status Register

If a 9725 Engine Manager detects an error, the 9725 will set the appropriate bit in this register. The Host may read the Engine Manager Error Status register to determine the source of the error. This information may also be read from the entry in the result ring. Because the Engine Managers continue to process commands after an error, the error bits in this register may be overwritten. It is recommended that the host validates the error bit with the result ring error from the command.

Offset	x'0244'	Engine Manager 0 Status
	x'0248'	Engine Manager 1 Status
	x'024C'	Engine Manager 2 Status
	x'0250'	Engine Manager 3 Status
	x'03C0'	Engine Manager 4 Status
	x'03C4'	Engine Manager 5 Status



Field Name	Bits	Type	Reset Value	Description
EM_BUSY	31	RO	0	Engine Manager Busy. The host must use this bit to confirm that an Engine Manager is not busy before issuing a soft reset. 0 Engine Manager not busy 1 Engine Manager busy
RSVD	30:23	RW	0x00	Reserved.
IRQ	22	RO	0	Interrupt Request. This bit will be set if any of the errors defined in this register are set. 0 No 9725 errors detected 1 Indicates the 9725 detected an error and an interrupt was sent to the host
PERR	21	RO	0	9725 Parity Error. Indicates a parity error reported by the 9725 processor. This is a fatal error. 0 No parity error reported 1 Indicates a parity error reported by the 9725 processor

Field Name	Bits	Type	Reset Value	Description
CROE	20	RO	0	9725 Command/Result Overrun Error. Indicates a Command/result overrun reported by a 9725 processor. This is a fatal error. 0 No command/result overrun error reported 1 A command/result overrun error was reported by a 9725 processor
DERR	19	RO	0	Data Error. This is a fatal error. 0 No Data error reported 1 A Data error was reported by a 9725 processor
HASH_CRC_ERR	18	RO	0	Hash Engine CRC Error. 0 Hash engine did not detect CRC error 1 Hash engine detected CRC error
HASH_AUTH_ERR	17	RO	0	Hash Engine Authentication Error. 0 Hash engine did not detect authentication error 1 Hash engine detected authentication error
TERR	16	RO	0	9725 Token Error 0 9725 did not detect token error 1 9725 detected token error
KEY_INDEX_ERR	15	RO	0	Key Index Error. 0 Command key index did not exceed 255 1 Command key index exceeded 255
LZS_RCRC_ERR	14	RO	0	LZS Engine Record CRC Error. 0 LZS engine did not detect RCRC error in decode operation 1 LZS engine detected RCRC error in decode operation
CMD_TMOUT	13	RO	0	Command Timeout Error. Indicates a command was not able to complete. 0 No command timeout error detected 1 Indicates a command timeout occurred
PCIE_ECRC_ERR	12	RO	0	PCIe End-to-End CRC Error. 0 Command completed with no PCIe ECRC error 1 Command completed with PCIe ECRC error
EM_RCRC_ERR	11	RO	0	Engine Manager Record CRC Error. 0 Engine Manager did not detect a LZS RCRC error 1 Engine Manager detected a LZS RCRC error

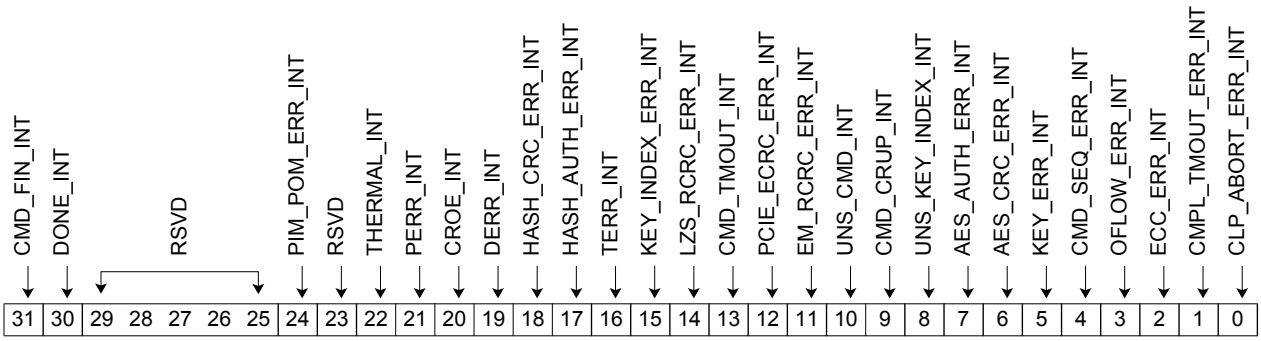
Field Name	Bits	Type	Reset Value	Description
UNS_CMD	10	RO	0	Unsupported Command Error. 0 Unsupported command not detected 1 Unsupported command detected
CMD_CRUP	9	RO	0	Command Corrupted Error. This bit may be set if a command never completed because the "last" bit was not set, or LZS engine did not assert an EOR (end of record) error or multiple record error. 0 Command Corrupted error did not occur 1 Command Corrupted error occurred
UNS_KEY_INDEX	8	RO	0	Unsupported Key Index Error. 0 Key index value within legal bounds 1 Host set an unsupported key index value
AES_AUTH_ERR	7	RO	0	AES Engine Authentication Error. 0 AES Engine did not detect AES-GCM mode authentication error 1 AES Engine detected AES-GCM mode authentication error
AES_CRC_ERR	6	RO	0	AES Engine CRC Error. 0 AES engine did not detect CRC or real time verification error 1 AES engine detected CRC or real time verification error
KEY_ERR	5	RO	0	Key Error. 0 Engine manger did not detect key write/read key CRC/ECC error 1 Engine manger detected key write/read key CRC/ECC error
CMD_SEQ_ERR	4	RO	0	Host Completion Sequence Error. This bit will be set if there are multiple completion packets for one outstanding read request, and the packets returned out of order. This is a non-fatal error. 0 No host completion sequence error detected 1 Indicates there are multiple completion packets for one outstanding read request, and the packets returned out of order
OFLOW_ERR	3	RO	0	Destination Data Overflow Error. This error occurs when the amount of destination data exceeded the total byte count of destination descriptors. This is a non-fatal error. 0 No destination data overflow detected 1 Indicates that the amount of destination data exceeded the total byte count of destination descriptors

Field Name	Bits	Type	Reset Value	Description
ECC_ERR	2	RO	0	Error Code Correction Error. 0 Engine Manager did not detect ECC error 1 Engine Manager detected ECC error
CMPL_ TMOUT_ERR	1	RO	0	Completion Timeout Error. Indicates a timeout occurred in the completion packet of source data. This is a non-fatal error. 0 No completion timeout error detected 1 Indicates a timeout occurred in the completion packet of source data
CLP_ABORT_ ERR	0	RO	0	Host Completion Abort Error. Indicates the host could not respond to a read request issued by the 9725. The 9725 will terminate the command associated with the completion packet marked with a completion abort error. This is a non-fatal error. 0 No host completion abort error detected 1 Indicates the host could not respond to a read request issued by the 9725. The 9725 will terminate the command associated with the completion packet marked with a completion abort error

6.3.3 Interrupt Status Register

The Interrupt Status register may be read by the host to determine the source of the interrupt. Interrupts may be removed by the host software by writing a one to clear that interrupt.

Offset x'0284'



Field Name	Bits	Type	Reset	Description
CMD_FIN_INT	31	RW1C	0	Command Finish Interrupt. 0 No Command Finish interrupt detected 1 Engine Manager has finished processing a command
DONE_INT	30	RW1C	0	Interrupt when Done. This bit when set indicates that the "final" command has completed processing (i.e., the chip is idle, with no commands pending). 0 Processing commands 1 No further commands to process
RSVD	29:25	RW	00000	Reserved.
PIM_POM_ERR_ INT	24	RW1C	0	PIM, POM, PCIe Error Interrupt. This bit when set indicates that the PIM/POM/ PCIe core detected an ECC or parity error. 0 No PIM/POM/PCIe error detected 1 PIM/POM/PCIe error detected
RSVD	23	RW	0	Reserved.
THERMAL_INT	22	RW	0	Thermal Interrupt. 0 Device temperature does not exceed threshold 1 Device temperature exceeded threshold

Field Name	Bits	Type	Reset	Description
PERR_INT	21	RW1C	0	Parity Error Interrupt. Indicates a parity error reported by the 9725 processor. This interrupt indicates a fatal error. 0 No 9725 parity interrupt detected 1 9725 parity interrupt detected
CROE_INT	20	RW1C	0	9725 Command/Result Overrun Error Interrupt. Indicates a Command/result overrun reported by a 9725 processor. This interrupt indicates a fatal error. 0 No Overrun interrupt detected 1 Overrun interrupt detected
DERR_INT	19	RW1C	0	9725 Data Error Interrupt. This interrupt indicates a fatal error. 0 No DERR interrupt detected 1 DERR interrupt detected
HASH_CRC_ERR_INT	18	RW1C	0	Hash Engine CRC Error Interrupt. 0 Hash engine did not detect CRC error 1 Hash engine detected CRC error
HASH_AUTH_ERR_INT	17	RW1C	0	Hash Engine Authentication Error Interrupt. 0 Hash engine did not detect authentication error 1 Hash engine detected authentication error
TERR_INT	16	RW1C	0	9725 Token Error Interrupt. This interrupt indicates a fatal error. 0 No TERR interrupt detected 1 TERR interrupt detected
KEY_INDEX_ERR_INT	15	RW1C	0	Key Index Error Interrupt. 0 Command key index did not exceed 255 1 Command key index exceed 255
LZS_RCRC_ERR_INT	14	RW1C	0	LZS Engine Record CRC Error Interrupt. 0 LZS engine did not detect RCRC error in decode operation 1 LZS engine detected RCRC error in decode operation
CMD_TMOUT_INT	13	RW1C	0	Command Timeout Error Interrupt. Indicates a command was not able to complete. 0 No command timeout error detected 1 Indicates a command timeout occurred
PCIE_ECRC_ERR_INT	12	RW1C	0	PCIe End-to-End CRC Error Interrupt. 0 Command completed with no PCIe ECRC error 1 Command completed with PCIe ECRC error

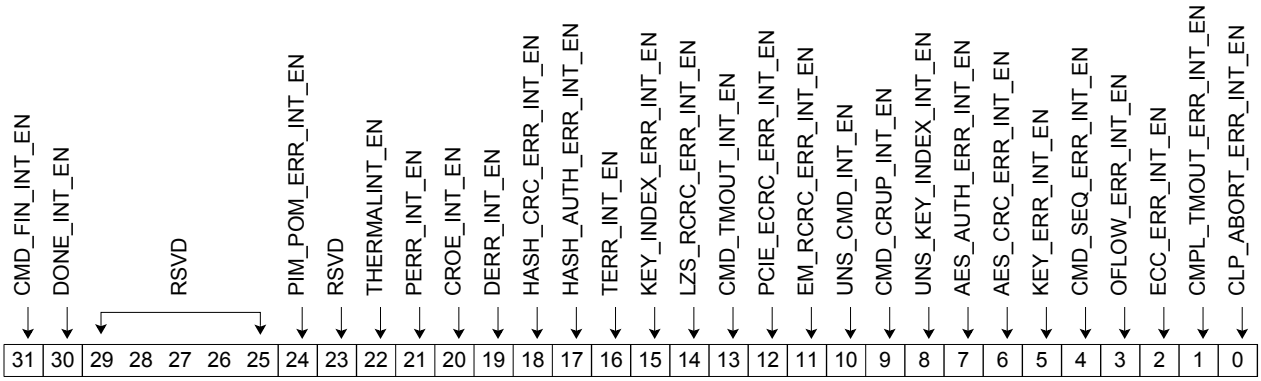
Field Name	Bits	Type	Reset	Description
EM_RCRC_ERR_INT	11	RW1C	0	Engine Manager Record CRC Error Interrupt. 0 Engine Manager did not detect a LZS RCRC error 1 Engine Manager detected a LZS RCRC error
UNS_CMD_INT	10	RW1C	0	Unsupported Command Error Interrupt. 0 Unsupported command not detected 1 Unsupported command detected
CMD_CRUP_INT	9	RW1C	0	Command Corrupted Error Interrupt. This bit may be set if a command never completed because the "last" bit was not set, or the LZS engine did not assert an EOR (end of record) error or multiple record error. 0 Command Corrupted error did not occur 1 Command Corrupted error occurred
UNS_KEY_INDEX_INT	8	RW1C	0	Unsupported Key Index Error Interrupt. 0 Key index value within legal bounds 1 Host set an unsupported key index value
AES_AUTH_ERR_INT	7	RW1C	0	AES Engine Authentication Error Interrupt. 0 AES Engine did not detect AES-GCM mode authentication error 1 AES Engine detected AES-GCM mode authentication error
AES_CRC_ERR_INT	6	RW1C	0	AES Engine CRC Error Interrupt. 0 AES engine did not detect CRC or real time verification error 1 AES engine detected CRC or real time verification error
KEY_ERR_INT	5	RW1C	0	Key Error Interrupt. 0 Engine manger did not detect key write/read key CRC/ECC error 1 Engine manger detected key write/read key CRC/ECC error
CMD_SEQ_ERR_INT	4	RW1C	0	Host Completion Sequence Error Interrupt. This bit will be set if there are multiple completion packets for one outstanding read request, and the packets returned out of order. This is a non-fatal error. 0 No host completion sequence error detected 1 Indicates there are multiple completion packets for one outstanding read request, and the packets returned out of order

Field Name	Bits	Type	Reset	Description
OFLOW_ERR_INT	3	RW1C	0	Destination Data Overflow Error Interrupt. This error occurs when the amount of destination data exceeded the total byte count of destination descriptors. This is a non-fatal error. 0 No destination data overflow detected 1 Indicates that the amount of destination data exceeded the total byte count of destination descriptors
ECC_ERR_INT	2	RW1C	0	Error Code Correction Error. Interrupt. 0 Engine Manager did not detect ECC error 1 Engine Manager detected ECC error
CMPL_TMOUT_ERR_INT	1	RW1C	0	Completion Timeout Error Interrupt. Indicates a timeout occurred in the completion packet of source data. This is a non-fatal error. 0 No completion timeout error detected 1 Indicates a timeout occurred in the completion packet of source data
CLP_ABORT_ERR_INT	0	RW1C	0	Host Completion Abort Error Interrupt. Indicates the host could not respond to a read request issued by the 9725. The 9725 will terminate the command associated with the completion packet marked with a completion abort error. This is a non-fatal error. 0 No host completion abort error detected 1 Indicates the host could not respond to a read request issued by the 9725. The 9725 will terminate the command associated with the completion packet marked with a completion abort error

6.3.4 Interrupt Enable Register

The Interrupt Enable register may be used by the host to mask interrupts. This register maps bit-by-bit to the Interrupt Status register. To enable an interrupt, the bit in this register must be set active to "1". The reset value for the Interrupt Enable register is all zeroes, in other words, all interrupts disabled.

Offset x'0288'



Field Name	Bits	Type	Reset Value	Description
CMD_FIN_INT_EN	31	RW	0	Command Finish Interrupt Enable. 0 Disable Command Finished interrupt 1 Enable Command Finished interrupt
DONE_INT_EN	30	RW	0	Done Interrupt Enable. 0 Disable Done interrupt 1 Enable Done interrupt
RSVD	29:25	RW	00000	Reserved.
PIM_POM_ERR_ INT_EN	24	RW	0	PIM, POM, PCIe Error Interrupt Enable. 0 Disable PIM/POM/PCIe error interrupt 1 Enable PIM/POM/PCIe error interrupt
RSVD	23	RW	0	Reserved.
THERMAL_INT_EN	22	RW	0	Thermal Interrupt Enable. 0 Disable Thermal interrupt 1 Enable Thermal interrupt
PERR_INT_EN	21	RW	0	Parity Error Interrupt Enable. 0 Disable Parity Error interrupt 1 Enable Parity Error interrupt
CROE_INT_EN	20	RW	0	Command/Result Overrun Error Interrupt Enable. 0 Disable Command/Result Overrun interrupt 1 Enable Command/Result Overrun interrupt

Field Name	Bits	Type	Reset Value	Description
DERR_INT_EN	19	RW	0	9725 Data Error Interrupt Enable. 0 Disable Data Error interrupt 1 Enable Data Error interrupt
HASH_CRC_ERR_INT_EN	18	RW	0	Hash Engine CRC Error Interrupt Enable. 0 Disable Hash Engine CRC Error interrupt 1 Enable Hash Engine CRC Error interrupt
HASH_AUTH_ERR_INT_EN	17	RW	0	Hash Engine Authentication Error Interrupt Enable. 0 Disable Hash Engine Authentication Error interrupt 1 Enable Hash Engine Authentication Error interrupt
TERR_INT_EN	16	RW	0	9725 Token Error Interrupt Enable. 0 Disable Token Error interrupt 1 Enable Token Error interrupt
KEY_INDEX_ERR_INT_EN	15	RW	0	Key Index Error Interrupt Enable. 0 Disable Key Index Error interrupt 1 Enable Key Index Error interrupt
LZS_RCRC_ERR_INT_EN	14	RW	0	LZS Engine Record CRC Error Interrupt Enable. 0 Disable LZS Engine Record CRC Error interrupt 1 Enable LZS Engine Record CRC Error interrupt
CMD_TMOUT_INT_EN	13	RW	0	Command Timeout Error Interrupt Enable. 0 Disable Command Timeout Error interrupt 1 Enable Command Timeout Error interrupt
PCIE_ECRC_ERR_INT_EN	12	RW	0	PCIe End-to-End CRC Error Interrupt Enable. 0 Disable PCIe ECRC Error interrupt 1 Enable PCIe ECRC Error interrupt
EM_RCRC_ERR_INT_EN	11	RW	0	Engine Manager Record CRC Error Interrupt Enable. 0 Disable Engine Manager RCRC Error interrupt 1 Enable Engine Manager RCRC Error interrupt
UNS_CMD_INT_EN	10	RW	0	Unsupported Command Error Interrupt Enable. 0 Disable Unsupported Command Error interrupt 1 Enable Unsupported Command Error interrupt
CMD_CRUP_INT_EN	9	RW	0	Command Corrupted Error Interrupt Enable. 0 Disable Command Corrupted Error interrupt 1 Enable Command Corrupted Error interrupt

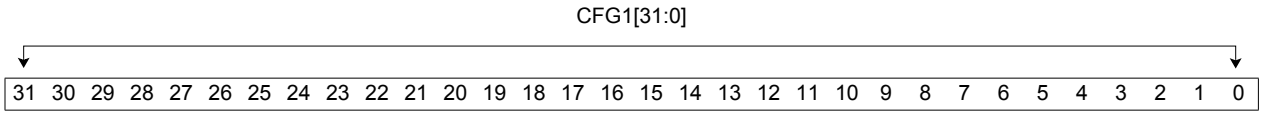
Field Name	Bits	Type	Reset Value	Description
UNS_KEY_INDEX_INT_EN	8	RW	0	Unsupported Key Index Error Interrupt Enable. 0 Disable Unsupported Key Index Error interrupt 1 Enable Unsupported Key Index Error interrupt
AES_AUTH_ERR_INT_EN	7	RW	0	AES Engine Authentication Error Interrupt Enable. 0 Disable AES Engine Authentication Error interrupt 1 Enable AES Engine Authentication Error interrupt
AES_CRC_ERR_INT_EN	6	RW	0	AES Engine CRC Error Interrupt Enable. 0 Disable AES Engine CRC Error interrupt 1 Enable AES Engine CRC Error interrupt
KEY_ERR_INT_EN	5	RW	0	Key Error Interrupt Enable. 0 Disable Key Error interrupt 1 Enable Key Error interrupt
CMD_SEQ_ERR_INT_EN	4	RW	0	Host Completion Sequence Error Interrupt Enable. 0 Disable Host Completion Sequence Error interrupt 1 Enable Host Completion Sequence Error interrupt
OFLOW_ERR_INT_EN	3	RW	0	Destination Data Overflow Error Interrupt Enable. 0 Disable Destination Data Overflow Error interrupt 1 Enable Destination Data Overflow Error interrupt
ECC_ERR_INT_EN	2	RW	0	Error Code Correction Error Interrupt Enable 0 Disable ECC Error interrupt 1 Enable ECC Error interrupt
CMPL_TMOUT_ERR_INT_EN	1	RW	0	Completion Timeout Error Interrupt Enable. 0 Disable Completion Timeout Error interrupt 1 Enable Completion Timeout Error interrupt
CLP_ABORT_ERR_INT_EN	0	RW	0	Host Completion Abort Error Interrupt Enable. 0 Disable Host Completion Abort Error interrupt 1 Enable Host Completion Abort Error interrupt

6.3.5 Config1 Register

This is a special register used to configure the 9725. The value for this register is set from the Flash and must not be modified.

NOTE: Modifying this register may cause unpredictable behavior from the 9725.

Offset x‘029C’



Field Name	Bits	Type	Reset Value	Description
CFG1[31:0]	31:0	RW	From Flash	Configuration 1. The value of this register must not be modified.

6.3.6 Flash Write Enable Register

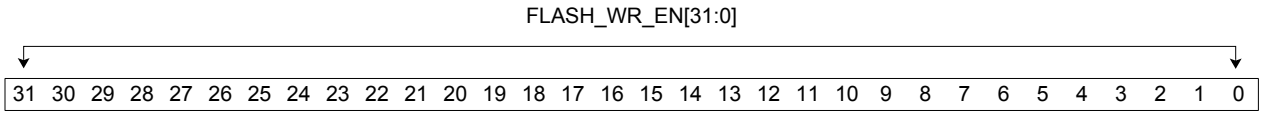
The Flash Write Enable register can be used to enable the host to write to the user region of the Flash using the Flash Address Register, Flash Data Register, and Flash Status Register. The host may read from the Flash at anytime.

The Flash User region is 64K bytes starting at offset 0x3e0000 (0x3E0000 - 0x3EFFFF).

NOTE: The Config2 register must be set to 0xFFFFFFFF before writing to this register.

NOTE: The host must only write to the user region of the Flash; writing to other regions of the Flash may cause unpredictable behavior from the 9725.

Offset x‘02A0’



Field Name	Bits	Type	Reset Value	Description
FLASH_WR_EN [31:0]	31:0	RW	0x0000 0000	Flash Write Enable. When the value of this register equals 0x7963_F5E4, the host can write to the user region of the 9725 using the Flash address, data, and status registers.

6.3.7 Flash Address Register

The Flash Address register can be used to set the Flash address for read, write and erase sector accesses to the Flash device. Writing the Flash address field of this register causes the 9725 to initiate the Flash access, therefore, the host software MUST follow the procedure defined below before writing to this register.

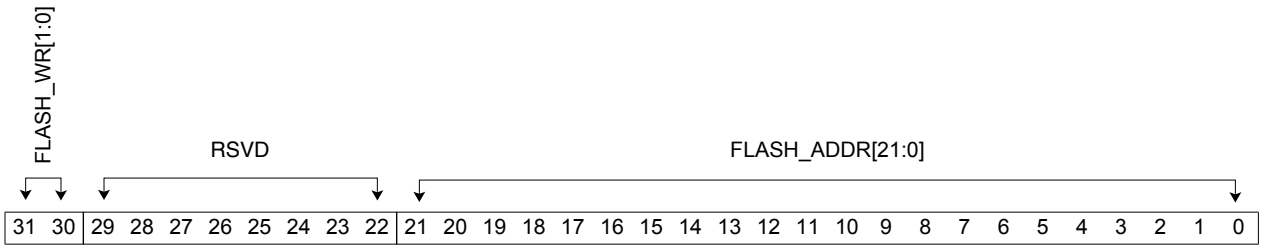
Procedure for Write or erase access to the Flash:

1. Write the data into the Flash Data Register.
2. Enable write access to the Flash using the Flash Write Enable Register.
3. Confirm that the previous Flash access is complete by reading the Flash Status Register.
4. Write the Flash address and set the Read/Write select to this register to initiate the Flash write.

Procedure for Read access to the Flash:

1. Confirm that the previous Flash access is complete by reading the Flash Status Register.
2. Write the Flash address and set the Read/Write select to this register to initiate the Flash read.
3. Read the data from the Flash Data Register.

Offset x'02A4'



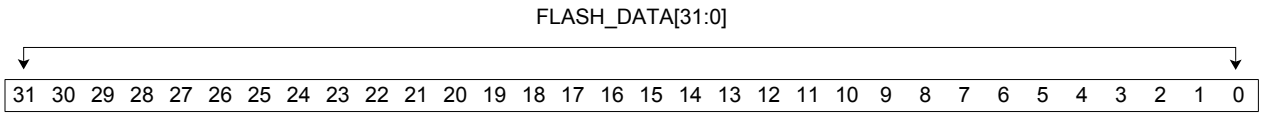
Field Name	Bits	Type	Reset Value	Description
FLASH_RW[1:0]	31:30	RW	00	Flash Read/Write Select. 00 Read Flash 01 Write Flash 10 Erase sector 11 Read Flash status register

Field Name	Bits	Type	Reset Value	Description
RSVD	29:22	RW	0x00	Reserved.
FLASH_ADDR[21:0]	21:0	RW	0x000000	Flash Physical Address. This field is used to set the Flash address for read, write and erase sector accesses. NOTE: the procedures described in the opening of this section MUST be followed to corrected access the Flash.

6.3.8 Flash Data Register

The Flash Data register contains the Flash read result or the data to be written to Flash.

Offset x‘02A8’



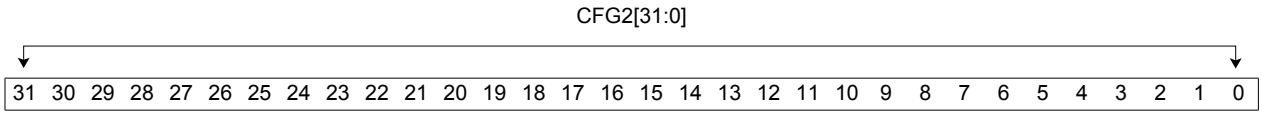
Field Name	Bits	Type	Reset Value	Description
FLASH_DATA[31:0]	31:0	RW	0x0000 0000	Flash Data. NOTE: the user must follow the read/write procedures outlined in Section 6.3.7 to access the Flash. For Flash write operations, the host should write the data to be written into the Flash into this field. For Flash read operations, this field will hold the Flash data after the access completes.

6.3.9 Config2 Register

This is a special register used to configure the 9725. The value for this register must be set to 0xFFFFFFFF.

NOTE: Modifying this register may cause unpredictable behavior from the 9725.

Offset x‘02AC’

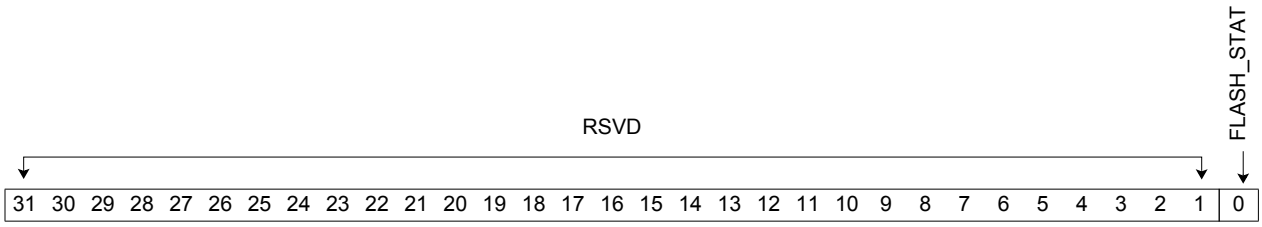


Field Name	Bits	Type	Reset Value	Description
CFG2[31:0]	31:0	RW	0x0000 0000	Configuration 2. The value of this register must be set to 0xFFFFFFFF.

6.3.10 Flash Status Register

The Flash Status register may be used by the host to distinguish when a Flash operation has completed.

Offset x‘02B0’



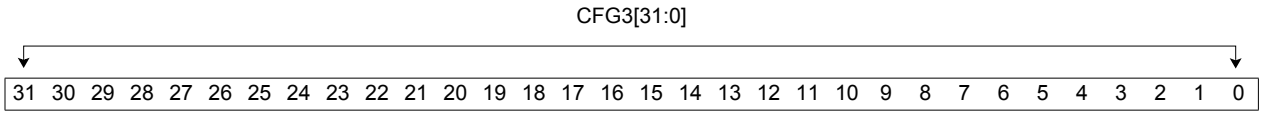
Field Name	Bits	Type	Reset Value	Description
RSVD	31:1	RW	0x0000	Reserved.
FLASH_STAT	0	RW	0	Flash Access Status. NOTE: the user must follow the read/ write procedures outlined in Section 6.3.7 to access the Flash. The 9725 will assert this bit when a Flash read/write operation has completed. The 9725 will automatically clear this bit when the host writes to the Flash Address Register. The host also can write a 1 to clear this bit. The host may force a status update by issuing a “rd_flash_status_reg” operation using the Flash Address Register.

6.3.11 Config3 Register

This is a special register used to configure the 9725. The value for this register is set from the Flash and must not be modified.

NOTE: Modifying this register may cause unpredictable behavior from the 9725.

Offset x‘02FC’

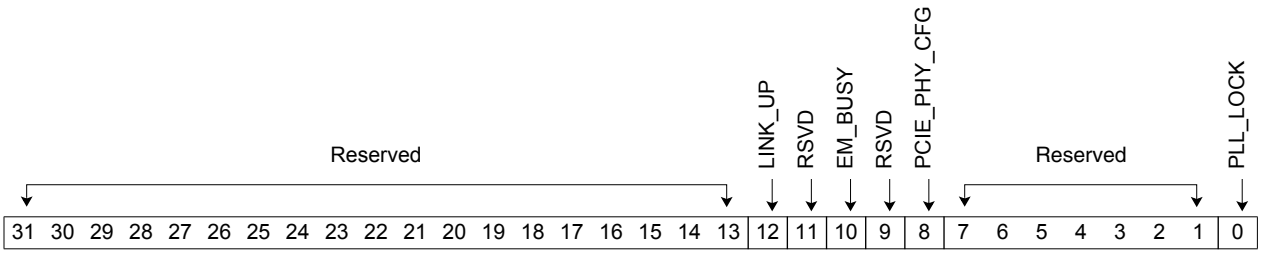


Field Name	Bits	Type	Reset Value	Description
CFG3[31:0]	31:0	RW	From Flash	Configuration 3. The value of this register must not be modified.

6.3.12 9725 Status Register

The Status register provides the 9725 device status to the host.

Offset x‘0344’



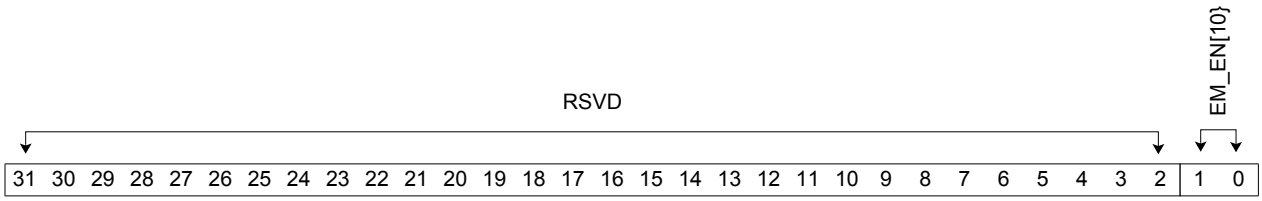
Field Name	Bits	Type	9725 Value	Description
RSVD	31:13	RW	0x0000	Reserved.
LINK_UP	12	RO	0	PCIe Link Status. This status bit is connected to the external pin N18, LINK_N, which can be used to drive a LED to indicate the PCIE link status. 0 PCIe link down (not operational) 1 PCIe link up (operational)
RSVD	11	RW	0	Reserved.
EM_BUSY	10	RO	0	9725 Engine Manager Busy. This status bit may be connected to a LED to indicate the processing state of all the 9725 Engine Managers. 0 Engine Managers not busy 1 Engine Managers busy
RSVD	9	RW	0	Reserved
PCIE_PHY_CFG	8	RO	0	PCIe PHY Configuration. This bit is used to configure the PCIe PHY interface. 0 PHY setting will be read from the 9725 pin L16, which should be pulled up externally. The PCIe PHY configurations will be determined by the values of the signals TXTERMADJ[1:0] through RXEQCTL[1:0] defined in Table 7-1. Refer to the *9725 Hardware Design User Guide*, UG-0196, for detailed information about these signals. 1 PHY setting will be read internally from the 9725

Field Name	Bits	Type	9725 Value	Description
RSVD	7:1	RW	0x00	Reserved.
PLL_LOCK	0	RO	0	PLL Status. This bit may be used to read the PLL lock status. 0 PLL not locked 1 PLL locked

6.3.13 Engine Manager 4/5 Enable Register

The Engine Manager 4/5 Control register may be used to enable or disable Engine Managers 4 or 5 from fetching commands from the DMA.

Offset x‘03F0’

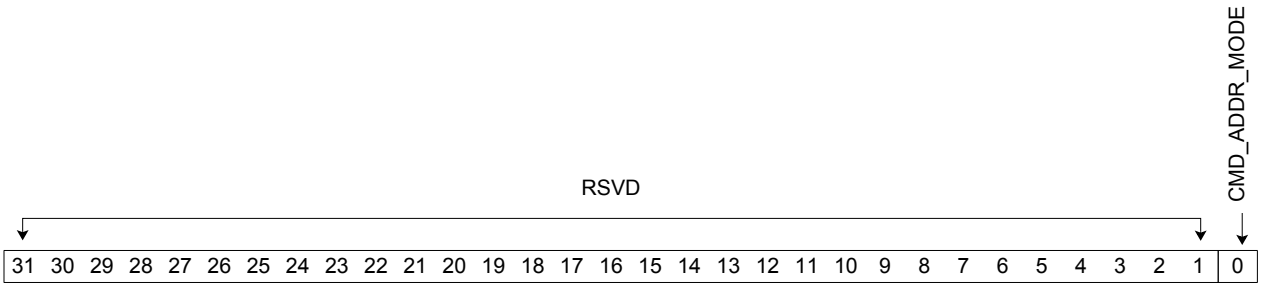


Field Name	Bits	Type	Reset Value	Description
RSVD	31:2	RW	0x000	Reserved.
EM_EN[1:0]	1:0	RW	00	Engine Manager Enable. This field is used to enable the Engine Managers 4 or 5 to fetch commands from the DMA. 10 Enable engine manager 5 11 Enable engine manager 4 All others values reserved.

6.3.14 Command Addressing Mode Register

The Command Addressing Mode register may be used to select between direct and indirect addressing mode. The command address mode should be set once during initialization.

Offset x‘03F4’

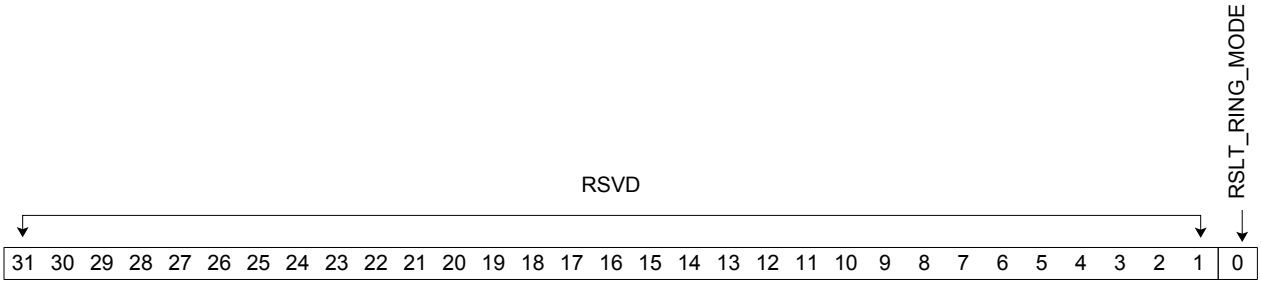


Field Name	Bits	Type	Reset Value	Description
RSVD	31:1	RW	0x0000	Reserved.
CMD_ADDR_MODE	0	RW	0	Command addressing mode. This bit is used to select the command addressing mode. 0 Direct addressing mode: Command address calculated as “base address + command offset” 1 Indirect addressing mode: Command address calculated as “command address pointer”

6.3.15 Result Ring Mode Register

The Result Ring Mode register may be used to select between direct and indirect addressing mode. The result ring mode should be set once during initialization.

Offset x‘03F8’



Field Name	Bits	Type	Reset Value	Description
RSVD	31:1	RW	0x0000	Reserved.
RSLT_RING_MODE	0	RW	0	Result Ring mode. This bit is used to select the command addressing mode. 0 32 bit result ring mode (compatible with legacy Exar VTL Express cards) 1 64 bit result ring mode

7 Interface Definition

7.1 PCI Express Interface

The 9725 provides a PCIe x8 interface for communicating with the host.

Table 7-1. PCIe Interface Description

Pin Name	Direction	Signal Type	Description
TXP7 - TXP0	Output	LVDS	Differential positive transmit output for lane 0/1/2/3/4/5/6/7
TXN7 - TXN0	Output	LVDS	Differential negative transmit output for lane 0/1/2/3/4/5/6/7
RXP7 - RXP0	Input	LVDS	Differential positive receive input for lane 0/1/2/3/4/5/6/7
RXN7 - RXN0	Input	LVDS	Differential negative receive input for lane 0/1/2/3/4/5/6/7
SREFCLK_P	Input	LVDS	Differential positive reference clock
SREFCLK_N	Input	LVDS	Differential negative reference clock
TXTERMADJ[1:0]	Input	LVTTL	Control bus to adjust transmit termination values
RXTERMADJ[1:0]	Input	LVTTL	Control bus to adjust receive termination values
HIDRV	Input	LVTTL	Mode bit to increase the nominal value of the lane's driver current value
LODRV	Input	LVTTL	Mode bit to decrease the nominal value of the lane's driver current value
DTX[3:0]	Input	LVTTL	4 bits digital word to control the driver current level
DEQ[3:0]	Input	LVTTL	4 bits digital word to control the equalization current level
RXEQCTL[1:0]	Input	LVTTL	Global mode select for the receiver equalizers of each lane

7.2 Flash Interface

The 9725 requires a Flash to store log information written by the SDK and to store user configuration data such as subsystem vendor ID, subsystem device ID and expansion ROM size.

Table 7-2. Flash Interface Description

Pin Name	Direction	Signal Type	Description
FLASH_SI	Input	LVTTL	Flash input data driven from the output of the Flash device
FLASH_CS	Output	LVTTL	Chip select output signal from the 9725 to the Flash device
FLASH_SCK	Output	LVTTL	Clock output to the Flash device
FLASH_SO	Output	LVTTL	Output data from the 9725 to the Flash device

7.3 Clock Interface

Table 7-3. Clock Interface Description

Pin Name	Direction	Signal Type	Description
DMA_PLL_REF_CLK	Input	LVTTL	DMA PLL reference clock input, 25MHz.

7.4 Miscellaneous Interface

The signals that comprise the Miscellaneous interface are listed in Table 7-4 below and further described below the table. Unless specified otherwise, all LVTTL signal types are 6mA.

Table 7-4. Miscellaneous Interface Description (Sheet 1 of 2)

Pin Name	Direction	Signal Type	Description
POR_N	Input	LVTTL (Schmitt)	Power on reset. 0 = Reset the entire 9725 1 = Normal operation
PERST_N	Input	LVTTL (Schmitt)	Connect to PCIe slot reset Unlike POR_N, PERST_N will not reset the PCIe sticky registers 0 = Reset the 9725 except the PCIe registers 1 = Normal operation
CHIP_ERR	Output	LVTTL (8mA)	9725 Error. This signal may be connected to a LED to indicate that 9725 has experienced a fatal error. This bit is under software control using the Software Control Register. This pin draws 8 mA of current. 0 = 9725 operational 1 = SDK reports that the 9725 has had a fatal error

Table 7-4. Miscellaneous Interface Description (Sheet 2 of 2)

Pin Name	Direction	Signal Type	Description
LINK_N	Output	LVTTL	PCIe Link Status. This active low signal may be connected to a LED to indicate the PCIe link status. This pin draws 6 mA of current. 0 = PCIe link operational 1 = PCIe link down
THERMAL_INT_N	Input	LVTTL	Thermal Interrupt. This pin may be connected to a thermal detection chip interrupt output pin. This pin will be asserted if the 9725 temperature exceeds the preset threshold set in the external thermal detection device. If no thermal detection chip is populated on the card, this signal should be connected to a pull-up resistor. 0 = Thermal interrupt has occurred 1 = Thermal conditions within thresholds
THERMAL_RST	Output	LVTTL (8mA)	Thermal Reset. This signal may be connected to an external thermal detection chip to reset the thermal detection device after an over-heated condition has occurred. Once asserted, it is expected that THERMAL_INT_N will be de-asserted.
PHY_REFCLK_SEL	Input	LVTTL	Selects the PCIe PHY reference clock 0 = 125M reference clock 1 = 100M reference clock
TDIODE_N	Analog	N/A	Thermal diode pin - negative. This pin is used to connect to an external thermal detection device and is used in conjunction with tdiode_p and thermal_int_n. Refer to Appendix A for more information.
TDIODE_P	Analog	N/A	Thermal diode pin - positive. This pin is used to connect to an external thermal detection device and is used in conjunction with tdiode_n and thermal_int_n. Refer to Appendix A for more information.
PULLUP	Input	LVTTL	Pull Up. Reserved input pin must be connected to a pull-up resistor to VDD33.
PULLDN	Input	LVTTL	Pull down. Reserved input pin must be connected to a pull-down resistor to ground.
NC	N/A	N/A	No Connection. Signals that must be left unconnected and require no termination.

7.5 JTAG Interface

Table 7-5. JTAG Interface Description

Pin Name	Direction	Signal Type	Description
JTAG_TCK	Input	LVTTL	JTAG Test clock This signal requires an external pull-down resistor.
JTAG_TDI	Input	LVTTL	JTAG Test data input This signal requires an external pull-up resistor.
JTAG_TDO	Output	LVTTL	JTAG Test data output
JTAG_TMS	Input	LVTTL	JTAG Test mode select This signal requires an external pull-up resistor.
JTAG_TRST_N	Input	LVTTL	JTAG Test reset This signal requires an external pull-down resistor.

7.6 Power and Ground Interface

Table 7-6. Power and Ground Interface Description

Pin Name	Description	Voltage Level
VDD	Core digital power supply	1.0V
VDD18	VDD Voltage power supply	1.8V
VDD33	3.3V Miscellaneous I/O power supply	3.3V
PCIE_VDD	CDR, TX and RX digital power supply	1.0V
PCIE_VDDA	PLL, Bias generator circuit analog power supply	1.03V
PCIE_VDDB	Analog power supply for all PCIE specific circuits	1.03V
PCIE_VTT	PCIe Termination voltage	1.5V
DMA_PLL_AHVDD	PLL analog supply	3.3V
DMA_PLL_DVDD	PLL digital power supply	1.0V
DMA_PLL_AHVSS	PLL analog ground	GND
DMA_PLL_DVSS	PLL digital ground	GND
VSS	Digital ground	GND

Refer to Chapter 8, "DC Specifications"" for detailed information on these power supplies.

8 DC Specifications

8.1 Absolute Maximum Ratings

Table 8-1. Absolute maximum ratings

Symbol	Description	Min	Max	Units
V_I	3.3V compatible IO voltage	VSS-0.5	VDD33+0.5, not to exceed 4V	V
V_O	LVDS compatible IO voltage	VSS-0.2	PCIE_VTT+0, not to exceed 2.2v	V
VDD33	DC Supply I/O Voltage	VSS-0.5	4.0	V
VDD, DMA_PLL_DVDD	DC Supply Core Voltage	VSS-0.2	1.25	V
DMA_PLL_AHVDD	DC Supply Analog Voltage	VSS-0.2	1.25	V
PCIE_VDD, PCIE_VDDA, PCIE_VDDB	DC Supply SERDES Analog Voltage	VSS-0.2	1.25	V
PCIE_VTT	DC Input Voltage	VSS-0.2	2.2	V
VDD18	DC Supply Voltage	VSS-0.3	2.2	V
T_{STG}	Storage Temperature	-10	130	°C
ESD_{HBM1}	Electrostatic Discharge Human Body Model Test 1		600[1]	V
ESD_{HBM2}	Electrostatic Discharge Human Body Model Test 2		1500[2]	V
ESD_{CDM}	Electrostatic Discharge Charge Device Model Test		500	V

Notes:

1. For balls C16, D16, D15, E15, D14, C14, C15, E14.
2. For all balls except C16, D16, D15, E15, D14, C14, C15, E14.

Caution

Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions may affect device reliability.

8.2 Power Supplies

The following subsections provide details concerning the digital and analog power supply connections on the 9725 device. Generally speaking, analog supplies can be derived from

the same power source as the digital supplies, but are more noise-sensitive and require additional filtering and careful layout/routing.

8.2.1 Digital Power Supplies

Table 8-2. VDD3 Power Supply Requirements

Parameter	Description	Min	Max	Units
Operation range	Voltage range for nominal operation conditions	3.15	3.45	V
Rise time	Time from 10% to 90% mark	0.1	10	mS
Overshoot	Maximum overshoot allowed		80	mV
Ripple	Maximum voltage ripple		60	mV

Table 8-3. VDD, DMA_PLL_DVDD Power Supply Requirements

Parameter	Description	Min	Max	Units
Operation range	Voltage range for nominal operation conditions	0.95	1.05	V
Rise time	Time from 10% to 90% mark	0.1	10	mS
Overshoot	Maximum overshoot allowed		50	mV
Ripple	Maximum voltage ripple		30	mV

Table 8-4. PCIE_VDD Power Supply Requirements

Parameter	Description	Min	Max	Units
Operation range	Voltage range for nominal operation conditions	0.98	1.08	V
Rise time	Time from 10% to 90% mark	0.1	10	mS
Overshoot	Maximum overshoot allowed		50	mV
Ripple	Maximum voltage ripple		30	mV

Table 8-5. VDD18 Power Supply Requirements

Parameter	Description	Min	Max	Units
Operation range	Voltage range for nominal operation conditions	1.71	1.89	V
Rise time	Time from 10% to 90% mark	0.1	10	mS
Overshoot	Maximum overshoot allowed		50	mV
Ripple	Maximum voltage ripple		50	mV

8.2.2 Analog Power Supplies

Table 8-6. DMA_PLL_AHVDD Power Supply Requirements

Parameter	Description	Min	Max	Units
Operation range	Voltage range for nominal operation conditions	3.15	3.45	V
Rise time	Time from 10% to 90% mark	0.1	10	mS
Overshoot	Maximum overshoot allowed		80	mV
Ripple	Maximum voltage ripple		60	mV

Table 8-7. PCIE_VDDA, PCIE_VDDB Power Supply Requirements

Parameter	Description	Min	Max	Units
Operation range	Voltage range for nominal operation conditions	0.98	1.08	V
Rise time	Time from 10% to 90% mark	0.1	10	mS
Overshoot	Maximum overshoot allowed		50	mV
Ripple	Maximum voltage ripple		30	mV

Table 8-8. PCIE_VTT Power Supply Requirements

Parameter	Description	Min	Max	Units
Operation range	Voltage range for nominal operation conditions	1.42	1.58	V
Rise time	Time from 10% to 90% mark	0.1	10	mS
Overshoot	Maximum overshoot allowed		50	mV
Ripple	Maximum voltage ripple		30	mV

8.3 Power Sequencing

The 9725 power supplies must be powered up in the sequence described in this section to ensure that the device does not latchup or have a shorted power supply condition after power-up.

For the 9725 I/O configuration, with the exception of the PCIE_VTT supply, different voltages are supplied to the core oxide and the second layer oxide, and there is the equivalent of a parasitic diode from the core power rail to the I/O power rail. Therefore, if the core is powered on before the I/O power, there will be current flowing through this parasitic diode which may damage the device or lead to a reduced operational life.

It is important not to exceed the power sequencing time limit to avoid bus conflicts. The time between power supplies power up should be between 0ms and 10ms (0ms<t<10ms), to avoid bus conflicts.

For PCIE power supplies, although PCIE_VTT is higher than PCIE_VDD, it must be powered after VDD for reliable operation.

The 9725 power supplies can be brought up simultaneously (with the exception of the PCIE_VTT) with the following constraints:

- The 1V supply can NEVER exceed the level of the 1.8V supply or the 3.3V supply.
- The 1.8V supply can NEVER exceed the level of the 3.3V supply.
- The PCIE_VTT supply must be power AFTER PCIE_VDD.

The figure below illustrates the power sequencing requirement.

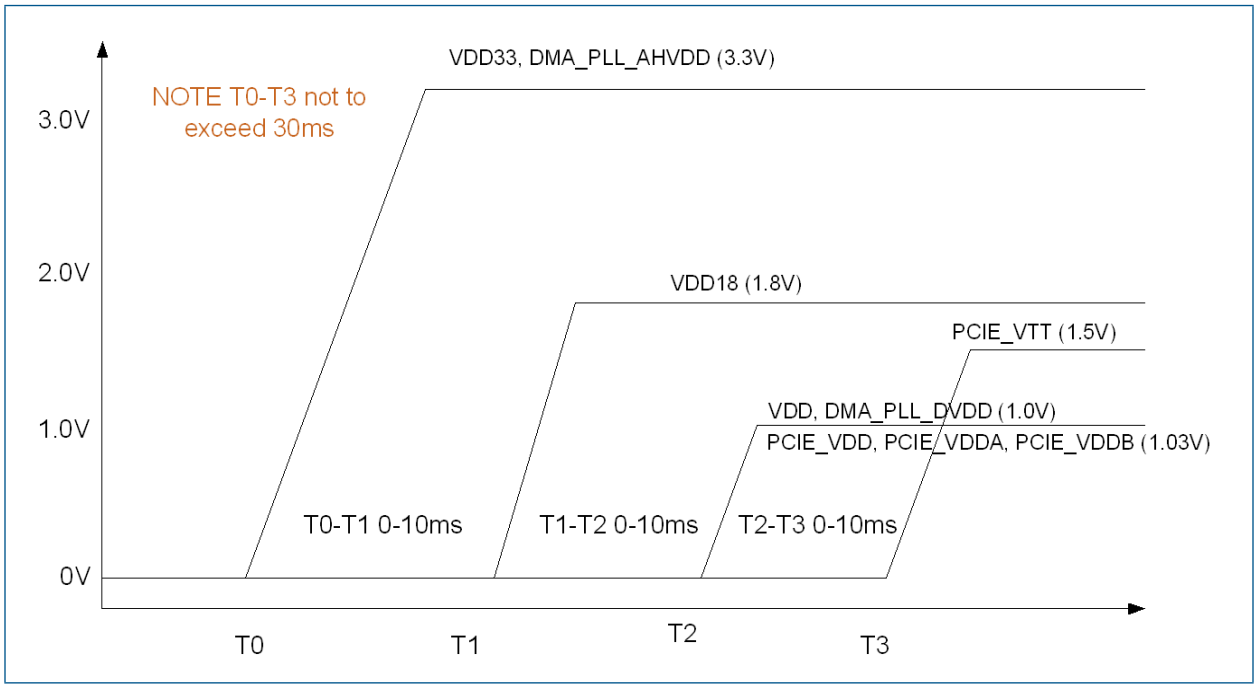


Figure 8-1. Power Supply Sequencing

The power down sequence is the opposite of the power-up sequence. Power down the lower core voltage first and the higher I/O supply second.

8.3.1 Power Consumption

Table 8-9. DC electrical characteristics, Receiver Logic Levels (Normal IO)

Power Supply	TYP Voltage (V)	TYP Current (mA)	TYP Power (W)	Max Voltage (V)	Max Current (mA)	Max Power (W)
VDD, DMA_PLL_DVDD	1.00	6880	6.68	1.05	10476	11.00
VDD33	3.30	5	0.02	3.47	9	0.03
VDD18	1.80	180	0.32	1.89	187	0.35
PCIE_VDD	1.03	200	0.21	1.08	210	0.23
PCIE_VDDA	1.03	74	0.08	1.08	167	0.18
PCIE_VDDB	1.03	56	0.06	1.08	65	0.07
PCIE_VTT	1.50	122	0.18	1.58	173	0.27
DMA_PLL_AHVDD	3.30	4	0.01	3.47	10	0.03
Total			7.56			12.17

8.4 I/O Characteristics

The I/O specifications for the 9725 depend on the type of buffer used for a particular signal.

Table 8-10. LVTTL DC Receiver Logic Levels (Normal IO)

Symbol	Description	Min	Typ	Max	Units
NOR_V_{IHT}	DC Input Logic Threshold High	2.0			V
NOR_V_{ILT}	DC Input Logic Threshold Low			0.8	V
NOR_V_{IH}	DC Input Voltage High			VDD33 + 0.5	V
NOR_V_{IL}	DC Input Voltage Low	-0.5			V
R_{pull}	External resistor value for Pull down or pull up pins	1K		10K	Ohm

Table 8-11. LVTTL DC Driver Logic Levels and Data (Normal IO)

Symbol	Description	Min	Typ	Max	Units
NOR_V_{OH}	DC Output Logic High[1]	VDD33-0.4		VDD33	V
NOR_V_{OL}	DC Output Logic Low	0		0.4	V
NOR_I_{OH}	DC Output High Current (PAD = VDD33-0.4V)	6 (8 if specified)			mA
NOR_I_{OL}	DC Output Low Current (PAD=0.4V)	6 (8 if specified)			mA

Table 8-11. LVTTL DC Driver Logic Levels and Data (Normal IO)

$NOR\_R_{drv\_33\_6}$	Driver Series Output Resistance (3.3V-6mA)	30		67	Ω
$NOR\_R_{drv\_33\_8}$	Driver Series Output Resistance (3.3V-8mA)	23		51	Ω
$NOR\_I_{leakage}$	Driver Pad Leakage[2]			100	nA
Notes: 1. Values are DC-only for unterminated drivers loaded with 1-pF capacitor, guaranteed to meet the PCI DC output current requirements over process, temperature, and voltage supply variations. 2. Pad Voltage driven to 3.6 V through a 0.010 Ohm load, VDD = Vdd(min), Fast Process, Low Temperature					

Table 8-12. PCIE LVDS Transmitter Logic Values

Symbol	Description	Min	Typ	Max	Units
$V_{TX-DIFFp}$	Peak-to-peak, single ended	400		600	mV
$V_{TX-DIFFpp}$	Peak-to-peak, differential	800		1200	mV
V_{OL}	Low-level output voltage		PCIE_VTT-1.5*$V_{TX-DIFF}$		V
V_{OH}	High-level output voltage		VTT - 0.5*$V_{TX-DIFFp}$		V

Table 8-13. PCIE LVDS Receiver Logic Values

Symbol	Description	Min	Typ	Max	Units
$V_{RX-DIFFp}$	Differential input voltage (peak-to-peak)	170		1200	mV

9 AC Specifications

9.1 Reset Timing

The 9725 has two reset pins that contain an internal Schmitt circuit.

PERST_N

This reset pin should be connected to the PCIe reset.

POR_N

This is an asynchronous power on reset.

Table 9-1. Reset Timing Requirements

Symbol	Description	Min	Max	Units	Notes
T_{PWR}	Time from power stable to assertion of hardware reset	0		ns	
T_{RST}	Reset width	240		ns	
T_{HLD}	Hold time after reset	100		ms	1
T_{DLY}	Delay between resets	600		us	1, 2

Notes:

1. PCI Express specification requires software to wait for at least 100ms from the end of reset before issuing configuration requests to the device. The 9725 chip requires 600 us to lock all PLLs after reset. T_{HLD} is not met until both POR_N and PERST_N have been de-asserted for a minimum of 100ms.
2. POR_N must be asserted once prior to the first PERST_N. If subsequent PERST_N assertions are required, POR_N does not need to be re-asserted.

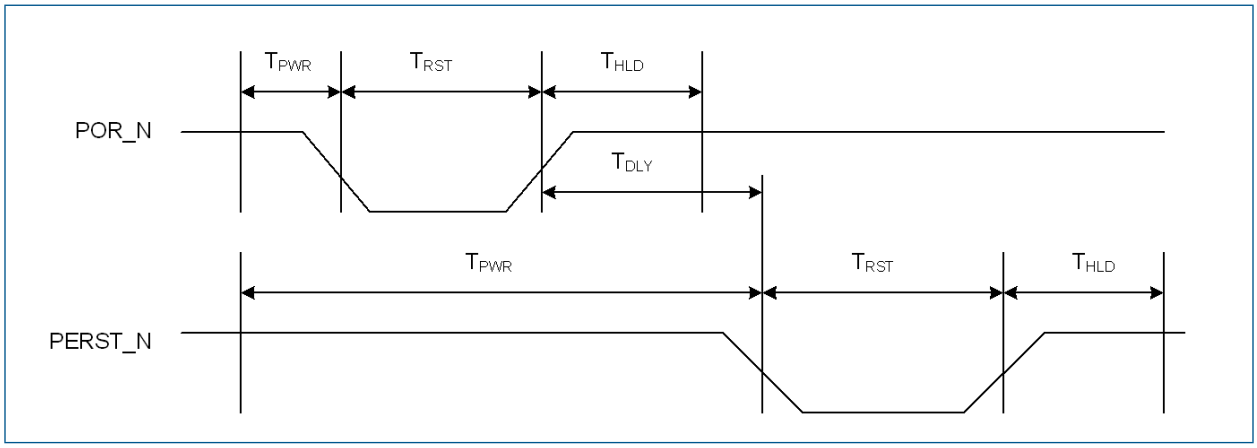


Figure 9-1. Reset Timing

9.2 PLL Clock Input

Table 9-2. PLL Reference Clock (dma_pll_ref_clk) Requirements

Symbol	Description	Min	Typ	Max	Units
F_{RefClk}	Input clock frequency		25		MHz
$D.C._{RefClk}$	Duty Cycle	45		55	%
$J_{CLK-REF}$	Input Jitter (peak-to-peak)			0.4	ns
Frequency tolerance	Input clock frequency deviation			150	ppm
Aging	Input clock long time deviation			+/- 5	ppm/ year
T_{rdy}	Lock time			0.5	ns
Note: Spread spectrum clocks are not supported.					

9.3 Flash Interface Timing

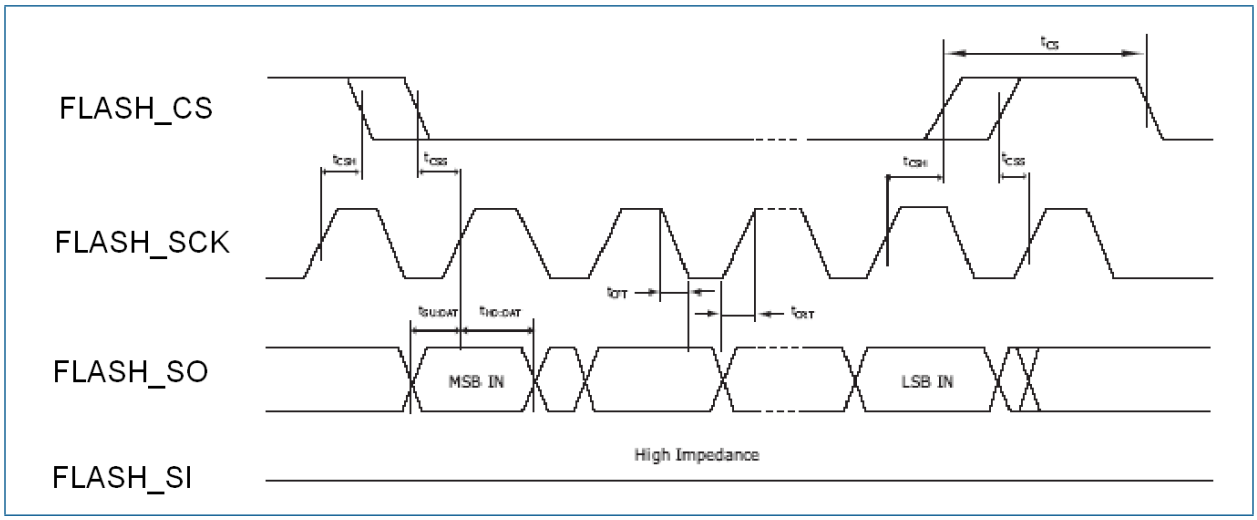


Figure 9-2. Flash Write Timing

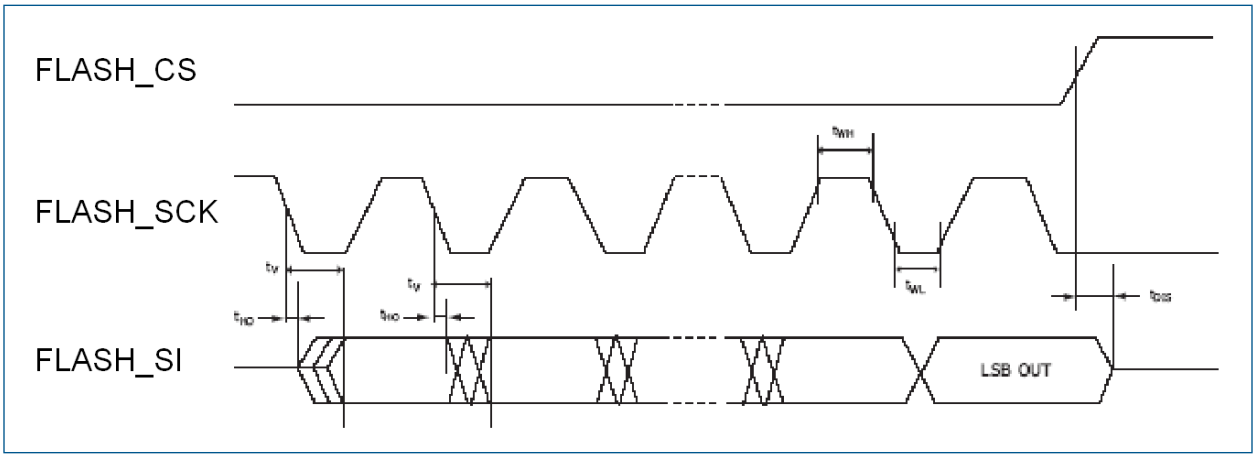


Figure 9-3. Flash Read Timing

Table 9-3. Flash Interface AC Characteristics

Symbol	Description	Min	Typ	Max	Units
t_{WH}	FLASH_SCK high time	9			ns
t_{WL}	FLASH_SCK low time	9			ns
t_{CSS}	FLASH_CS setup time	5			ns
t_{CSH}	FLASH_CS hold time	5			ns
t_V	FLASH_SI valid time	0		10	ns
t_{HO}	FLASH_SI hold time	0			ns
$t_{HD:DAT}$	FLASH_SO hold time	5			ns
$t_{SU:DAT}$	FLASH_SO setup time	5			ns
Note: The parameters in this table were measured with a 30pf load.					

9.4 Thermal Interface Timing (Optional)

The thermal interface is optional. The pins THERMAL_RST and THERMAL_INT_N are asynchronous signals and have no timing requirements.

The table below defines the thermal diode specifications.

Table 9-4. Thermal Diode Specifications

Symbol	Min	Typ	Max	Units	Notes
I_forward_bias	5		500	uA	1, 2
n_ideality	0.998	0.998	1.013		3, 4

Notes:

1. The thermal diode must not be used in reverse bias.
2. Typically, thermal detection devices use two different currents to solve two equations (for example, 10 and ~180uA).
3. The thermal diode has a forward bias of 630mV at room temperature.
4. The diode ideality factor, n_ideality, is represented by the diode equation:

I-Io(e (Vd*q) / (nkT) -1).

Figure 9-4 shows the thermal diode characteristics. The top graph gives the diode characteristics when operating at 10uA and 180uA; the slopes are -2.0833mV/C and -1.8125 mV/C. The bottom graph shows the voltage difference of the diode when the current I is 180uA and 10uA; the slope is 249uV/C.

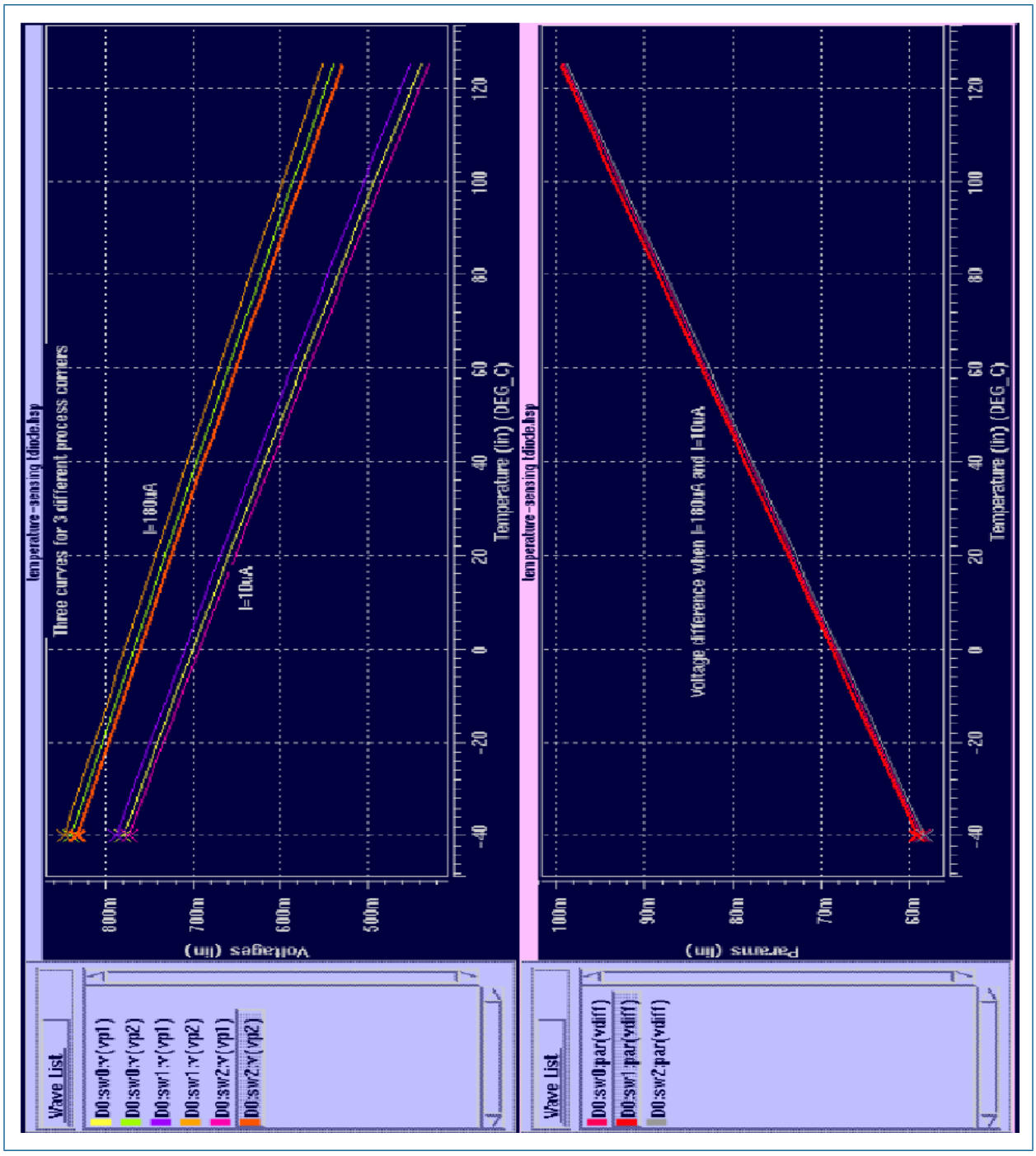


Figure 9-4. Thermal Diode Characteristics

9.5 JTAG Interface Timing

The 9725 is designed to support the IEEE 1149.1 JTAG standard.

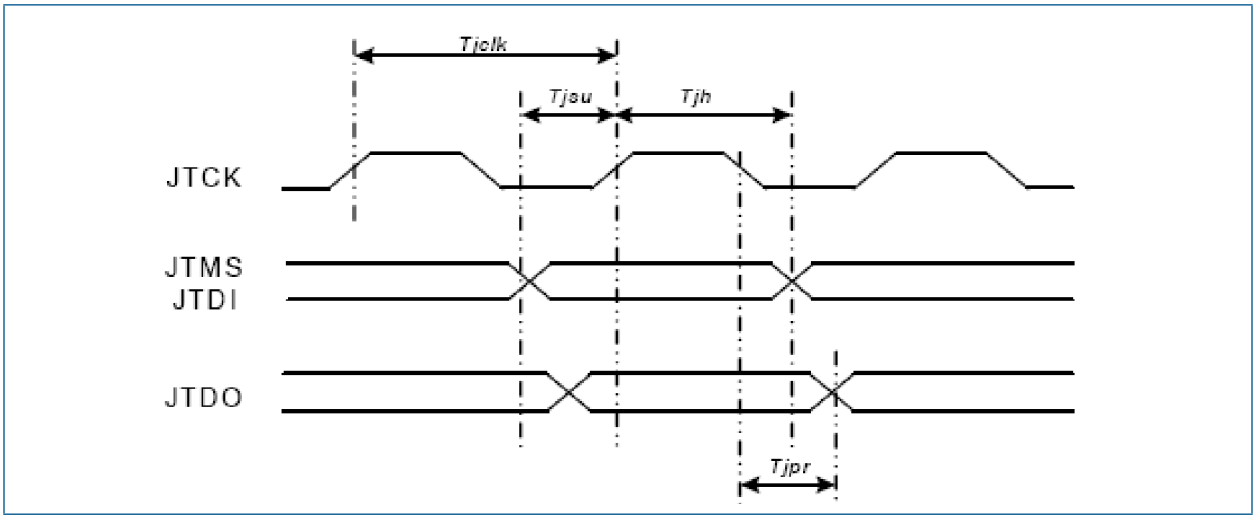


Figure 9-5. JTAG Timing

Table 9-5. JTAG Interface AC Characteristics

Symbol	Description	Min	Max	Units
T_{jclk}	JTAG clock frequency		20	ns
T_{th}	JTAG_TMS and JTAG_TDI hold time	10		ns
T_{jsu}	JTAG_TMS and JTAG_TDI setup time	10		ns
T_{jpr}	JTAG_TDO propagation delay		15	ns

9.6 PCIe Interface Timing

The 9725 PCIe interface is fully compliant to the PCI Express Electromechanical Specification Revision 1.1 standard.

10 Package Information

This chapter provides general and mechanical package information, as well as ball assignment drawings.

10.1 General Information

Table 10-1. 9725 General Package Information

Package Information	Description
Package type	FCBGA
Ball count	396
Package size	21 x 21 mm
Pitch	1 mm

10.2 Thermal Specifications

Each system application will require thermal analysis based on the application's environment.

Table 10-2. Thermal operating conditions

Symbol	Description	Min	Max	Units
Tj	Junction Temperature	0	125	°C

Notes:

1. For normal device operation, adhere to the limits in this table. Sustained operations of a device at conditions exceeding these values, even if they are within the absolute maximum rating limits, may result in permanent device damage or impaired device reliability. Device functionality to stated DC and AC limits is not guaranteed if conditions exceed recommended operating conditions.

2. Recommended operation conditions require accuracy of the power supplies as described in Section 8.2.

Table 10-3. Thermal resistance

	Max
Internal thermal resistance, (θ_{jc})	0.06 °C/W
Thermal resistance, (θ_{ja}) at 0 m/s airflow	16.2 °C/W
Thermal resistance, (θ_{ja}) at 1 m/s airflow	14.2 °C/W
Thermal resistance, (θ_{ja}) at 2 m/s airflow	13.2 °C/W
Temperature correlation (Ψ-jt) at 0 m/s airflow	0.01 °C/W

10.3 Mechanical Information

Figure 10-1 shows the mechanical specifications for the 9725 device.

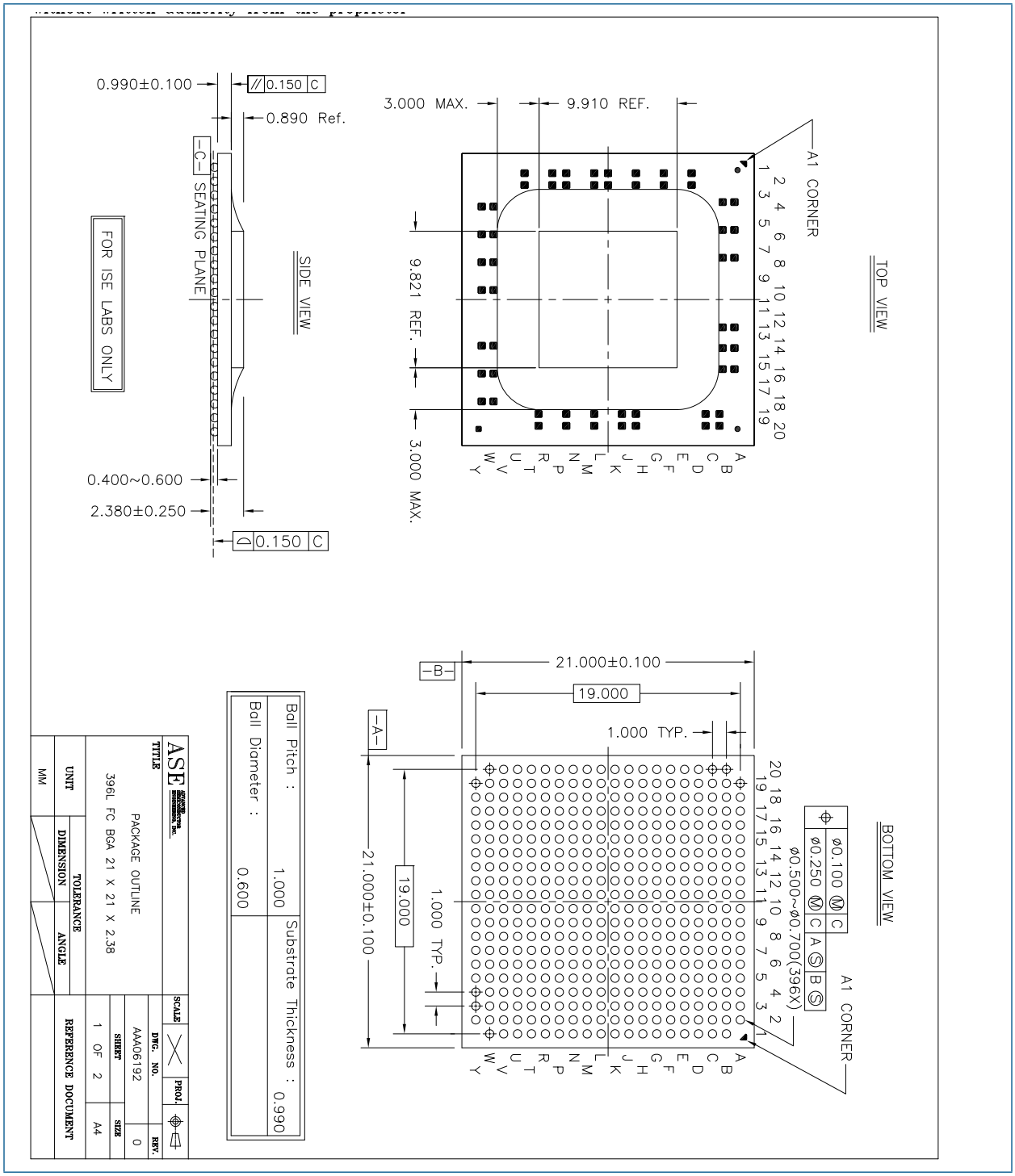


Figure 10-1. 9725 Mechanical Specification

10.4 Ball Assignment

The figures in this section illustrate the ball assignment from the top view.

	1	2	3	4	5	6	7	8	9	10
A		NC	NC	NC	NC	NC	VDD	NC	NC	PULLDN
B	NC	VDD18	NC	NC	NC	NC	NC	NC	VSS	PULLDN
C	NC	NC	VSS	NC	NC	VSS	NC	NC	NC	PULLDN
D	NC	NC	NC	NC	NC	NC	NC	VDD18	NC	PULLDN
E	NC	NC	NC	NC	VDD18	PULLDN	NC	NC	NC	PULLDN
F	NC	NC	VSS	NC	NC	VSS	NC	VSS	VDD	VSS
G	NC	NC	NC	NC	NC	VDD	VSS	VDD	VSS	VDD
H	NC	NC	NC	VDD18	NC	VSS	VDD	VSS	VDD	VSS
J	NC	NC	VSS	NC	NC	VDD	VSS	VDD	VSS	VDD
K	VDD	NC	NC	NC	NC	NC	VDD	VSS	VDD	VSS

Figure 10-2. 9725 Ball Assignment - Quadrant 1, Balls A1:K10

11	12	13	14	15	16	17	18	19	20	
PULLDN	PULLDN	PULLDN	JTAG_TCK	JTAG_TDI	JTAG_TDO	JTAG_TMS	VSS	VDD33		A
PULLDN	PULLDN	thermal_rst	VDD33	VSS	VSS	JTAG_TRST_N	PULLDN	PULLDN	PULLDN	B
PULLDN	VSS	NC	VSS	VSS	dma_pll_dvdd	VSS	PULLDN	PULLDN	PULLDN	C
VDD33	PULLDN	chip_err	VSS	dma_pll_ahvdd	dma_pll_dvss	VSS	VDD33	NC	NC	D
PULLDN	PULLDN	PULLDN	VSS	dma_pll_ahvss	dma_pll_ref_clk	VSS	NC	NC	NC	E
VDD	VSS	VDD	VSS	VDD	VSS	PULLDN	NC	flash_si	PULLDN	F
VSS	VDD	VSS	VDD	VSS	flash_so	flash_cs	VDD33	PULLDN	PULLUP	G
VDD	VSS	VDD	VSS	VDD	NC	flash_sck	PULLDN	PULLDN	PULLDN	H
VSS	VDD	VSS	VDD	VSS	PULLDN	PULLDN	VSS	VDD33	PULLDN	J
VDD	VSS	VDD	VSS	VDD	PULLDN	VDD33	PULLUP	PULLUP	PULLUP	K

Figure 10-3. 9725 Ball Assignment - Quadrant 2, Balls A11:K20

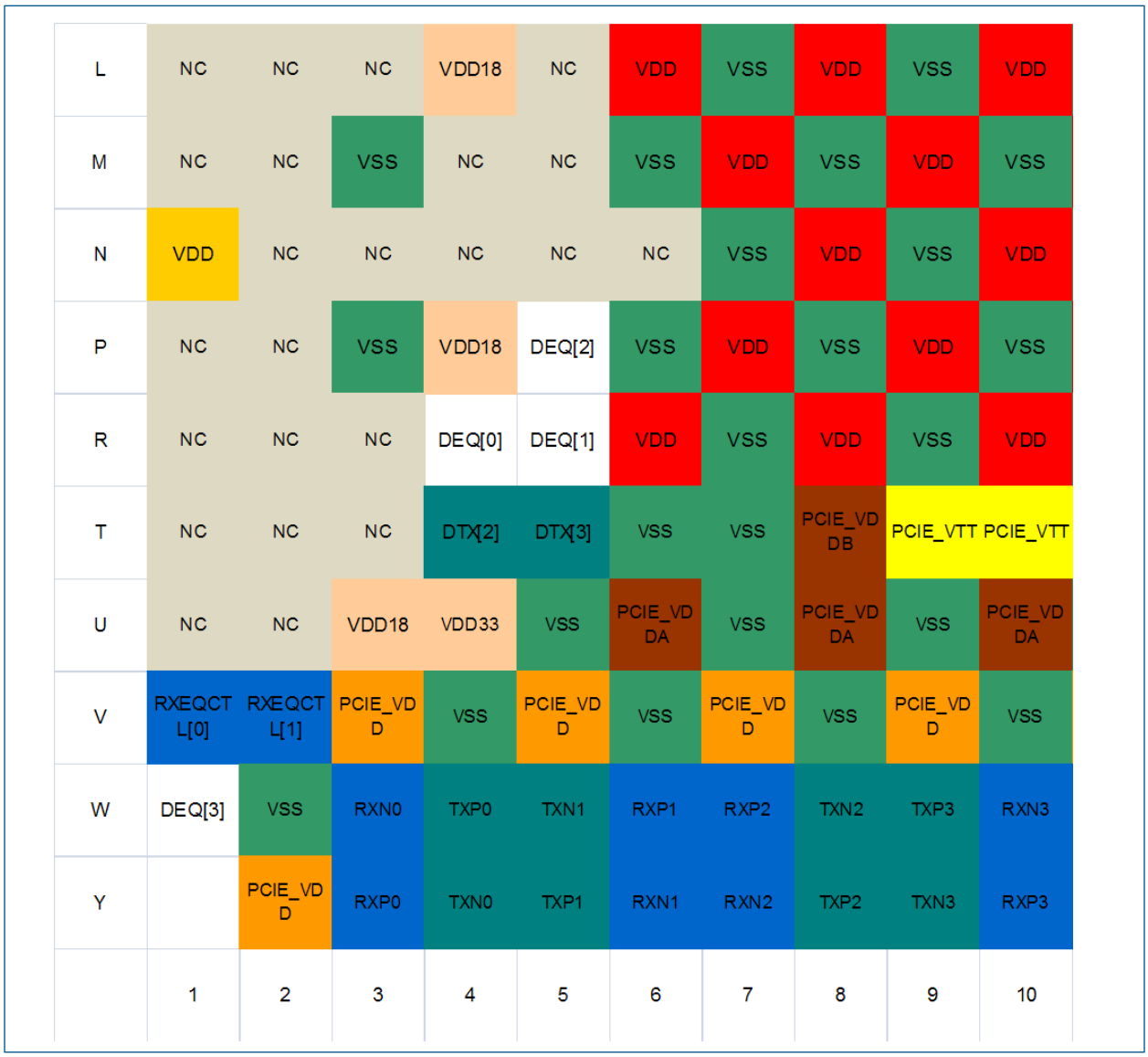


Figure 10-4. 9725 Ball Assignment - Quadrant 3, Balls L1:Y10

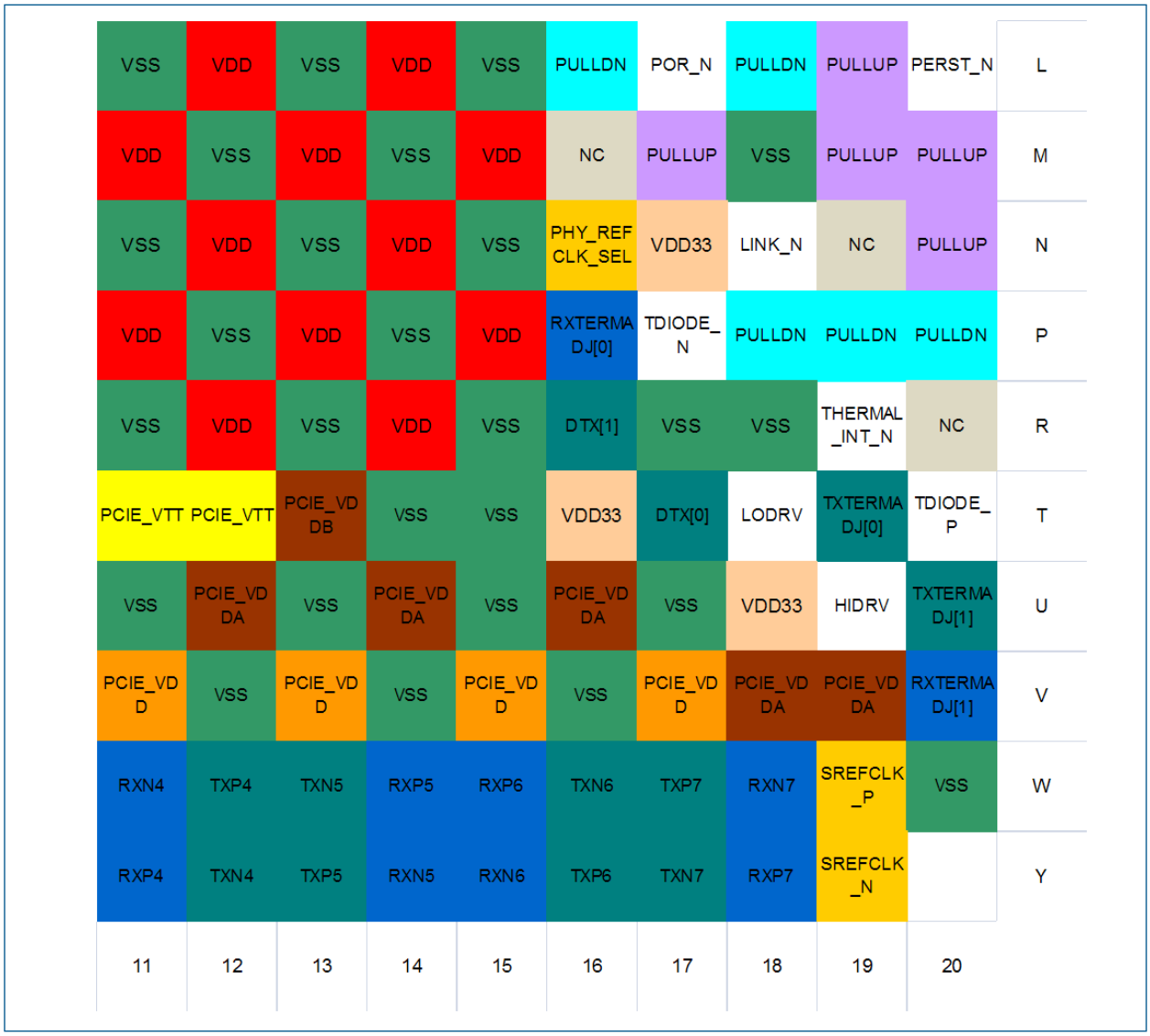


Figure 10-5. 9725 Ball Assignment - Quadrant 4, Balls L11:Y20

10.5 Ball List

10.5.1 Numeric Ball List

Ball num	Ball name
A2	NC
A3	NC
A4	NC
A5	NC
A6	NC
A7	VDD
A8	NC
A9	NC
A10	PULLDN
A11	PULLDN
A12	PULLDN
A13	PULLDN
A14	JTAG_TCK
A15	JTAG_TDI
A16	JTAG_TDO
A17	JTAG_TMS
A18	VSS
A19	VDD33
B1	NC
B2	VDD18

Ball num	Ball name
B3	NC
B4	NC
B5	NC
B6	NC
B7	NC
B8	NC
B9	VSS
B10	PULLDN
B11	PULLDN
B12	PULLDN
B13	THERMAL_RST
B14	VDD33
B15	VSS
B16	VSS
B17	JTAG_TRST_N
B18	PULLDN
B19	PULLDN
B20	PULLDN
C1	NC
C2	NC

Ball num	Ball name
C3	VSS
C4	NC
C5	NC
C6	VSS
C7	NC
C8	NC
C9	NC
C10	PULLDN
C11	PULLDN
C12	VSS
C13	NC
C14	VSS
C15	VSS
C16	DMA_PLL_DVDD
C17	VSS
C18	PULLDN
C19	PULLDN
C20	PULLDN
D1	NC
D2	NC

Ball num	Ball name
D3	NC
D4	NC
D5	NC
D6	NC
D7	NC
D8	VDD18
D9	NC
D10	PULLDN
D11	VDD33
D12	PULLDN
D13	CHIP_ERR
D14	VSS
D15	DMA_PLL_AHVDD
D16	DMA_PLL_DVSS
D17	VSS
D18	VDD33
D19	NC
D20	NC
E1	NC
E2	NC

Ball num	Ball name
E3	NC
E4	NC
E5	VDD18
E6	PULLDN
E7	NC
E8	NC
E9	NC
E10	PULLDN
E11	PULLDN
E12	PULLDN
E13	PULLDN
E14	VSS
E15	DMA_PLL_AHVSS
E16	DMA_PLL_REF_CLK
E17	VSS
E18	NC
E19	NC
E20	NC
F1	NC
F2	NC

Ball num	Ball name
F3	VSS
F4	NC
F5	NC
F6	VSS
F7	NC
F8	VSS
F9	VDD
F10	VSS
F11	VDD
F12	VSS
F13	VDD
F14	VSS
F15	VDD
F16	VSS
F17	PULLDN
F18	NC
F19	FLASH_SI
F20	PULLDN
G1	NC
G2	NC

Ball num	Ball name
G3	NC
G4	NC
G5	NC
G6	VDD
G7	VSS
G8	VDD
G9	VSS
G10	VDD
G11	VSS
G12	VDD
G13	VSS
G14	VDD
G15	VSS
G16	FLASH_SO
G17	FLASH_CS
G18	VDD33
G19	PULLDN
G20	PULLUP
H1	NC
H2	NC

Ball num	Ball name
H3	NC
H4	VDD18
H5	NC
H6	VSS
H7	VDD
H8	VSS
H9	VDD
H10	VSS
H11	VDD
H12	VSS
H13	VDD
H14	VSS
H15	VDD
H16	NC
H17	FLASH_SCK
H18	PULLDN
H19	PULLDN
H20	PULLDN
J1	NC
J2	NC

Ball num	Ball name
J3	VSS
J4	NC
J5	NC
J6	VDD
J7	VSS
J8	VDD
J9	VSS
J10	VDD
J11	VSS
J12	VDD
J13	VSS
J14	VDD
J15	VSS
J16	PULLDN
J17	PULLDN
J18	VSS
J19	VDD33
J20	PULLDN
K1	VDD
K2	NC

Ball num	Ball name
K3	NC
K4	NC
K5	NC
K6	NC
K7	VDD
K8	VSS
K9	VDD
K10	VSS
K11	VDD
K12	VSS
K13	VDD
K14	VSS
K15	VDD
K16	PULLDN
K17	VDD33
K18	PULLUP
K19	PULLUP
K20	PULLUP
L1	NC
L2	NC

Ball num	Ball name
L3	NC
L4	VDD18
L5	NC
L6	VDD
L7	VSS
L8	VDD
L9	VSS
L10	VDD
L11	VSS
L12	VDD
L13	VSS
L14	VDD
L15	VSS
L16	PULLDN
L17	POR_N
L18	PULLDN
L19	PULLUP
L20	PERST_N
M1	NC
M2	NC

Ball num	Ball name
M3	VSS
M4	NC
M5	NC
M6	VSS
M7	VDD
M8	VSS
M9	VDD
M10	VSS
M11	VDD
M12	VSS
M13	VDD
M14	VSS
M15	VDD
M16	NC
M17	PULLUP
M18	VSS
M19	PULLUP
M20	PULLUP
N1	VDD
N2	NC

Ball num	Ball name
N3	NC
N4	NC
N5	NC
N6	NC
N7	VSS
N8	VDD
N9	VSS
N10	VDD
N11	VSS
N12	VDD
N13	VSS
N14	VDD
N15	VSS
N16	PHY_REFCLK_SEL
N17	VDD33
N18	LINK_N
N19	NC
N20	PULLUP
P1	NC
P2	NC

Ball num	Ball name
P3	VSS
P4	VDD18
P5	DEQ[2]
P6	VSS
P7	VDD
P8	VSS
P9	VDD
P10	VSS
P11	VDD
P12	VSS
P13	VDD
P14	VSS
P15	VDD
P16	RXTERMADJ[0]
P17	TDIODE_N
P18	PULLDN
P19	PULLDN
P20	PULLDN
R1	NC
R2	NC

Ball num	Ball name
R3	NC
R4	DEQ[0]
R5	DEQ[1]
R6	VDD
R7	VSS
R8	VDD
R9	VSS
R10	VDD
R11	VSS
R12	VDD
R13	VSS
R14	VDD
R15	VSS
R16	DTX[1]
R17	VSS
R18	VSS
R19	THERMAL_INT_N
R20	NC
T1	NC
T2	NC

Ball num	Ball name
T3	NC
T4	DTX[2]
T5	DTX[3]
T6	VSS
T7	VSS
T8	PCIE_VDDB
T9	PCIE_VTT
T10	PCIE_VTT
T11	PCIE_VTT
T12	PCIE_VTT
T13	PCIE_VDDB
T14	VSS
T15	VSS
T16	VDD33
T17	DTX[0]
T18	LODRV
T19	TXTERMADJ[0]
T20	TDIODE_P
U1	NC
U2	NC

Ball num	Ball name
U3	VDD18
U4	VDD33
U5	VSS
U6	PCIE_VDDA
U7	VSS
U8	PCIE_VDDA
U9	VSS
U10	PCIE_VDDA
U11	VSS
U12	PCIE_VDDA
U13	VSS
U14	PCIE_VDDA
U15	VSS
U16	PCIE_VDDA
U17	VSS
U18	VDD33
U19	HIDRV
U20	TXTERMADJ[1]
V1	RXEQCTL[0]
V2	RXEQCTL[1]

Ball num	Ball name
V3	PCIE_VDD
V4	VSS
V5	PCIE_VDD
V6	VSS
V7	PCIE_VDD
V8	VSS
V9	PCIE_VDD
V10	VSS
V11	PCIE_VDD
V12	VSS
V13	PCIE_VDD
V14	VSS
V15	PCIE_VDD
V16	VSS
V17	PCIE_VDD
V18	PCIE_VDDA
V19	PCIE_VDDA
V20	RXTERMADJ[1]
W1	DEQ[3]
W2	VSS

Ball num	Ball name
W3	RXN0
W4	TXP0
W5	TXN1
W6	RXP1
W7	RXP2
W8	TXN2
W9	TXP3
W10	RXN3
W11	RXN4
W12	TXP4
W13	TXN5
W14	RXP5
W15	RXP6
W16	TXN6
W17	TXP7
W18	RXN7
W19	SREFCLK_P
W20	VSS
Y2	PCIE_VDD
Y3	RXP0

Ball num	Ball name
Y4	TXN0
Y5	TXP1
Y6	RXN1
Y7	RXN2
Y8	TXP2
Y9	TXN3
Y10	RXP3
Y11	RXP4
Y12	TXN4
Y13	TXP5
Y14	RXN5
Y15	RXN6
Y16	TXP6
Y17	TXN7
Y18	RXP7
Y19	SREFCLK_N

10.5.2 Alphabetic List

Ball num	Ball name
D13	CHIP_ERR
R4	DEQ[0]
R5	DEQ[1]
P5	DEQ[2]
W1	DEQ[3]
D15	DMA_PLL_AHVDD
E15	DMA_PLL_AHVSS
C16	DMA_PLL_DVDD
D16	DMA_PLL_DVSS
E16	DMA_PLL_REF_CLK
T17	DTX[0]
R16	DTX[1]
T4	DTX[2]
T5	DTX[3]
G17	FLASH_CS
H17	FLASH_SCK
F19	FLASH_SI
G16	FLASH_SO
U19	HIDRV
A14	JTAG_TCK

Ball num	Ball name
A15	JTAG_TDI
A16	JTAG_TDO
A17	JTAG_TMS
B17	JTAG_TRST_N
N18	LINK_N
T18	LODRV
A2	NC
A3	NC
A4	NC
A5	NC
A6	NC
A8	NC
A9	NC
B1	NC
B3	NC
B4	NC
B5	NC
B6	NC
B7	NC
B8	NC

Ball num	Ball name
C1	NC
C13	NC
C2	NC
C4	NC
C5	NC
C7	NC
C8	NC
C9	NC
D1	NC
D19	NC
D2	NC
D20	NC
D3	NC
D4	NC
D5	NC
D6	NC
D7	NC
D9	NC
E1	NC
E18	NC

Ball num	Ball name
E19	NC
E2	NC
E20	NC
E3	NC
E4	NC
E7	NC
E8	NC
E9	NC
F1	NC
F18	NC
F2	NC
F4	NC
F5	NC
F7	NC
G1	NC
G2	NC
G3	NC
G4	NC
G5	NC
H1	NC

Ball num	Ball name
H16	NC
H2	NC
H3	NC
H5	NC
J1	NC
J2	NC
J4	NC
J5	NC
K2	NC
K3	NC
K4	NC
K5	NC
K6	NC
L1	NC
L2	NC
L3	NC
L5	NC
M1	NC
M16	NC
M2	NC

Ball num	Ball name
M4	NC
M5	NC
N19	NC
N2	NC
N3	NC
N4	NC
N5	NC
N6	NC
P1	NC
P2	NC
R1	NC
R2	NC
R20	NC
R3	NC
T1	NC
T2	NC
T3	NC
U1	NC
U2	NC
V11	PCIE_VDD

Ball num	Ball name
V13	PCIE_VDD
V15	PCIE_VDD
V17	PCIE_VDD
V3	PCIE_VDD
V5	PCIE_VDD
V7	PCIE_VDD
V9	PCIE_VDD
Y2	PCIE_VDD
U10	PCIE_VDDA
U12	PCIE_VDDA
U14	PCIE_VDDA
U16	PCIE_VDDA
U6	PCIE_VDDA
U8	PCIE_VDDA
V18	PCIE_VDDA
V19	PCIE_VDDA
T13	PCIE_VDDB
T8	PCIE_VDDB
T10	PCIE_VTT
T11	PCIE_VTT

Ball num	Ball name
T12	PCIE_VTT
T9	PCIE_VTT
L20	PERST_N
N16	PHY_REFCLK_SEL
L17	POR_N
A10	PULLDN
A11	PULLDN
A12	PULLDN
A13	PULLDN
B10	PULLDN
B11	PULLDN
B12	PULLDN
B18	PULLDN
B19	PULLDN
B20	PULLDN
C10	PULLDN
C11	PULLDN
C18	PULLDN
C19	PULLDN
C20	PULLDN

Ball num	Ball name
D10	PULLDN
D12	PULLDN
E10	PULLDN
E11	PULLDN
E12	PULLDN
E13	PULLDN
E6	PULLDN
F17	PULLDN
F20	PULLDN
G19	PULLDN
H18	PULLDN
H19	PULLDN
H20	PULLDN
J16	PULLDN
J17	PULLDN
J20	PULLDN
K16	PULLDN
L16	PULLDN
L18	PULLDN
P18	PULLDN

Ball num	Ball name
P19	PULLDN
P20	PULLDN
G20	PULLUP
K18	PULLUP
K19	PULLUP
K20	PULLUP
L19	PULLUP
M17	PULLUP
M19	PULLUP
M20	PULLUP
N20	PULLUP
V1	RXEQCTL[0]
V2	RXEQCTL[1]
W3	RXN0
Y6	RXN1
Y7	RXN2
W10	RXN3
W11	RXN4
Y14	RXN5
Y15	RXN6

Ball num	Ball name
W18	RXN7
Y3	RXP0
W6	RXP1
W7	RXP2
Y10	RXP3
Y11	RXP4
W14	RXP5
W15	RXP6
Y18	RXP7
P16	RXTERMADJ[0]
V20	RXTERMADJ[1]
Y19	SREFCLK_N
W19	SREFCLK_P
P17	TDIODE_N
T20	TDIODE_P
R19	THERMAL_INT_N
B13	THERMAL_RST
Y4	TXN0
W5	TXN1
W8	TXN2

Ball num	Ball name
Y9	TXN3
Y12	TXN4
W13	TXN5
W16	TXN6
Y17	TXN7
W4	TXP0
Y5	TXP1
Y8	TXP2
W9	TXP3
W12	TXP4
Y13	TXP5
Y16	TXP6
W17	TXP7
T19	TXTERMADJ[0]
U20	TXTERMADJ[1]
A7	VDD
F11	VDD
F13	VDD
F15	VDD
F9	VDD

Ball num	Ball name
G10	VDD
G12	VDD
G14	VDD
G6	VDD
G8	VDD
H11	VDD
H13	VDD
H15	VDD
H7	VDD
H9	VDD
J10	VDD
J12	VDD
J14	VDD
J6	VDD
J8	VDD
K1	VDD
K11	VDD
K13	VDD
K15	VDD
K7	VDD

Ball num	Ball name
K9	VDD
L10	VDD
L12	VDD
L14	VDD
L6	VDD
L8	VDD
M11	VDD
M13	VDD
M15	VDD
M7	VDD
M9	VDD
N1	VDD
N10	VDD
N12	VDD
N14	VDD
N8	VDD
P11	VDD
P13	VDD
P15	VDD
P7	VDD

Ball num	Ball name
P9	VDD
R10	VDD
R12	VDD
R14	VDD
R6	VDD
R8	VDD
B2	VDD18
D8	VDD18
E5	VDD18
H4	VDD18
L4	VDD18
P4	VDD18
U3	VDD18
A19	VDD33
B14	VDD33
D11	VDD33
D18	VDD33
G18	VDD33
J19	VDD33
K17	VDD33

Ball num	Ball name
N17	VDD33
T16	VDD33
U18	VDD33
U4	VDD33
A18	VSS
B15	VSS
B16	VSS
B9	VSS
C12	VSS
C14	VSS
C15	VSS
C17	VSS
C3	VSS
C6	VSS
D14	VSS
D17	VSS
E14	VSS
E17	VSS
F10	VSS
F12	VSS

Ball num	Ball name
F14	VSS
F16	VSS
F3	VSS
F6	VSS
F8	VSS
G11	VSS
G13	VSS
G15	VSS
G7	VSS
G9	VSS
H10	VSS
H12	VSS
H14	VSS
H6	VSS
H8	VSS
J11	VSS
J13	VSS
J15	VSS
J18	VSS
J3	VSS

Ball num	Ball name
J7	VSS
J9	VSS
K10	VSS
K12	VSS
K14	VSS
K8	VSS
L11	VSS
L13	VSS
L15	VSS
L7	VSS
L9	VSS
M10	VSS
M12	VSS
M14	VSS
M18	VSS
M3	VSS
M6	VSS
M8	VSS
N11	VSS
N13	VSS

Ball num	Ball name
N15	VSS
N7	VSS
N9	VSS
P10	VSS
P12	VSS
P14	VSS
P3	VSS
P6	VSS
P8	VSS
R11	VSS
R13	VSS
R15	VSS
R17	VSS
R18	VSS
R7	VSS
R9	VSS
T14	VSS
T15	VSS
T6	VSS
T7	VSS

Ball num	Ball name
U11	VSS
U13	VSS
U15	VSS
U17	VSS
U5	VSS
U7	VSS
U9	VSS
V10	VSS
V12	VSS
V14	VSS
V16	VSS
V4	VSS
V6	VSS
V8	VSS
W2	VSS
W20	VSS

11 Errata

Errata are design defects or errors. Hardware Errata may cause the 9725 behavior to deviate from published specifications.

Errata 1. If AES-CBC/HMAC-SHA1 commands and masked HASH commands are interleaved in the 9725, the MAC value calculated for the AES-CBC/HMAC-SHA1 command may be incorrect.

Description:

If an AES-CBC/HMAC-SHA1 command executes and is followed by a masked HASH command, the MAC value calculated for the AES-CBC/HMAC-SHA1 command in any of the six algorithm engines may be incorrect.

Solution:

Do not mix HASH mask commands with AES-CBC/HMAC-SHA1 commands.

A Temperature Monitoring

The 9725 may be optionally connected to a thermal detection device to monitor the 9725 junction temperature using the signal pins TDIODE_N and TDIODE_P.

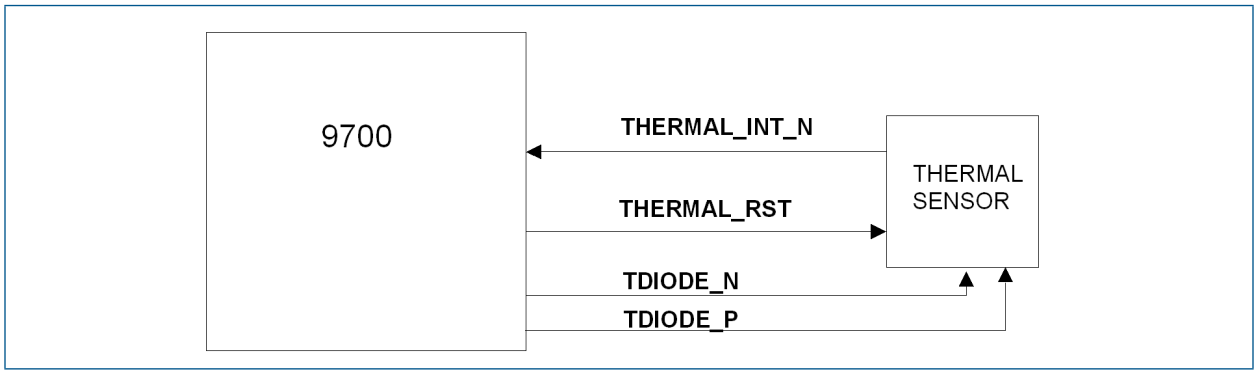


Figure A-1. Thermal Detection Device Circuit Example

The 9725 SDK will determine if the 9725 device junction temperature exceeds the maximum temperature threshold, and if an over-heated condition occurs, send an over-heat interrupt to host. The host should then cease submitting commands to the 9725 and reset the thermal monitoring device. Once the 9725 junction temperature is less than the maximum temperature threshold, normal operation may resume. However, if the 9725 junction temperature remains above the threshold, the host should report a fatal error.

For further information, refer to the *9725 Hardware Design User Guide*, UG-0196.

I Document Revision History

This section lists the additions, deletions, and modifications made to this document for each release of this document.

I.1 Document Revision A

Update 1. Initial release.

I.2 Document Revision B

Update 1. Chapter 1 Product Description: added section 1.1 Features, added section 1.2 Applications, section 1.3 Ordering Information.

Update 2. Removed all references to DD2 SDRAM.

Update 3. Removed previous Ch 2 Device Characterization.

Update 4. Added Ch 2 Operation, Ch 3 Data Structures, Ch 4 Modules, Ch 5 Registers, Ch 7 AC Specifications, Ch 8 DC Specifications, Appendix A.

Update 5. Ch 9 Package Information: updated layout drawing, added pin lists.

I.3 Document Revision 00

Update 1. Renamed 9700 as 9725 throughout.

Update 2. Corrected ball count from 400 to 396.

Update 3. Chapter 5: completed all PCIe register definitions.

Update 4. Section 7.1 Absolute Maximum Ratings: added ESD specs.

Update 5. Section 7.4.1 Power Consumption: updated values in table.

Update 6. Table 9-1 9725 General Package Information: corrected number of pins.

Update 7. Figure 9-1 9725 Mechanical Specification: updated for RoHS package.

I.4 Document Revision 01

Update 1. Replaced Table 9-2 with Table 7-2.

Update 2. Section 7.3.1 Power Consumption: corrected values in this table.

Update 3. Section 8.1 Reset Timing: added timing diagram for this interface.

I.5 Document Revision 02

Update 1. Section 2.3 CRC Protection: added description of RCRC control signals and Table 2-1.

Update 2. Section 2.6 Endian Settings: added this new section.

Update 3. Section 3.1 Definitions: added this new section.

Update 4. Section 3.3 Desc_src, Desc_dst: added bit 38, ERR_EP.

Update 5. Section 5.1.11 BAR Register 0/1: 4KB memory reserved for BAR registers.

Update 6. Added Chapter 6, 9725 Register Definition.

I.6 Document Revision 03

Update 1. Section 6.2.2 Command Ring 0 Write Pointer Register: corrected address to 0x224 - 0x227.

Update 2. Section 6.2.9 Command Ring1 Write Pointer Register: corrected address to 0x3E0 - 0x3E3.

Update 3. Section 6.3.3 Interrupt Status Register: changed bit 22 to THERMAL_INT.

Update 4. Section 6.3.4 Interrupt Enable Register: changed bit 22 to THERMAL_INT_EN.

I.7 Document Revision 04

Update 1. Section 3.3 Desc_src and Desc_dst: added text describing the hash buffer.

Update 2. Table 6-1 Register List: corrected CR0_WP to address 0x224, address 0x208 is now reserved. Corrected CR1_WP to address 0x3E4, address 0x3D0 is now reserved.

Update 3. Section 6.2.2 Command Ring 0 Write Pointer Reg: added additional description, corrected width from 62-bit to 32-bit.

Update 4. Section 6.2.9 Command Ring 1 Write Pointer Reg: added additional description, corrected address to 0x3E4, corrected width from 62-bit to 32-bit.

Update 5. Section 6.3.7 Flash Address Register: corrected instructions for reading the Flash.

Update 6. Section 10.4 Ball Assignment: corrected ball G20 to PULLUP.

Update 7. Section 10.5 Ball List: corrected ball G20 to PULLUP.

I.8 Document Revision 05

Update 1. Sections 3.3.3.1 through 3.3.3.4: added Descriptors with Encryption/Decryption commands, Desc_src0 for Write Key commands, and Desc_src0/ Desc_src1 for Hash commands.

Update 2. Errata 1 added.

48720 Kato Road
Fremont, CA 94538
p: 408.399.3500
www.exar.com