

7902

Network Security Processor Data Sheet



Hifn

Intelligent Secure Networking

Hifn supplies the Internet's most important raw materials for the creation of intelligent and secure networks: compression, encryption, and flow classification. This is central to the growth of the Internet, helping to make electronic mail, web browsing, Internet shopping and multimedia communications better, faster and more secure.

Hi/fn, Inc.
750 University Avenue
Los Gatos, CA 95032
info@hifn.com
http://www.hifn.com
Tel: 408-399-3500
Fax: 408-399-3501

For technical support, please contact your local Hifn sales office, representative or distributor. For locations check: www.hifn.com

Disclaimer

Hi/fn reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

Hi/fn warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with Hi/fn's standard warranty. Testing and other quality control techniques are utilized to the extent Hi/fn deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

HI/FN SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of Hi/fn products in such critical applications is understood to be fully at the risk of the customer. Questions concerning potential risk applications should be directed to Hi/fn through a local sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

Hi/fn does not warrant that its products are free from infringement of any patents, copyrights or other proprietary rights of third parties. In no event shall Hi/fn be liable for any special, incidental or consequential damages arising from infringement or alleged infringement of any patents, copyrights or other third party intellectual property rights.

"Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals," must be validated for each customer application by customer's technical experts.

The use of this product may require a license from Motorola. A license agreement for the right to use Motorola patents may be obtained through Hi/fn or directly from Motorola.

DS-0040-00 (1/01) © 1997-2001 by Hi/fn, Inc., including one or more U.S. patents No.: 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, and 5,532,694. Other patents pending. Hi/fn and LZS are registered trademarks of Hi/fn, Inc. Hifn is a trademark of Hi/fn, Inc. All other trademarks are the property of their respective holders.

This product must be exported from the United States in accordance with the Export Administration Regulations. Diversion contrary to U.S. law prohibited.

Table of Contents

1	Product Description	7
2	Performance Summary.....	8
2.1	Symmetric Key Processing Units.....	8
2.2	Protocols.....	8
2.3	Public Key	9
3	Product Overview	11
3.1	Operation	11
3.2	Functional Blocks	11
4	Signal Description.....	13
4.1	Signal List.....	13
4.2	Clocks	15
4.3	Phase Lock Loop (PLL).....	15
4.4	Reset	15
5	Memory Map	16
5.1	Byte and Bit Ordering.....	16
5.2	Registers	17
5.3	Internal Memory	17
6	External RAM.....	19
7	Processor Bus Interface	20
7.1	MPC860/8260 Bus Interface	20
7.2	Bus Transfer Overview – MPC860 Mode	22
7.3	Processor Bus Interface Signal Descriptions	28
8	General Registers.....	31
8.1	General Configuration Register	31
8.2	Chip ID Register.....	31
9	Public Key Engine & Random Number Generator	32
9.1	Overview	32
9.2	Random Number Generator.....	32
9.3	Public Key Data Movement.....	32
9.4	Operations.....	33
9.5	Operands.....	33
9.6	Reciprocal Calculations	38
9.7	Public Base Address Register	40
9.8	Public Operand Length Register.....	40
9.9	Public Operation Register.....	41
9.10	Public Status Register	42
9.11	Public Interrupt Enable Register.....	42
9.12	RNG Configuration Register	43
9.13	RNG Data Register	43
10	Packet Engine	44
10.1	Overview	44
10.2	Data Movement	45
10.3	Descriptors.....	46
10.4	Command Structure	48
10.5	Read RAM/Write RAM Command Structures	56
10.6	Source Structures.....	57
10.7	Dest Structures.....	57
10.8	Result Structures.....	57
10.9	External RAM Usage.....	61
10.10	Host Command Index Register	62
10.11	Host Source Index Register	63

10.12	Host Result Index Register	63
10.13	Host Destination Index Register	64
10.14	Packet Status Register.....	64
10.15	Packet Interrupt Enable Register	67
10.16	Packet Configuration Register	68
11	DC Specifications	69
11.1	Power Sequencing	69
11.2	Recommended Operating Conditions	69
11.3	DC Characteristics	70
12	AC Specifications	71
13	Thermal Specifications	76
14	Pin List.....	77
15	Physical Specifications	79

Figures

Figure 1.	Typical VPN Router Example	7
Figure 2.	Processing Unit Performance	8
Figure 3.	Protocol Performance	8
Figure 4.	IKE Performance.....	9
Figure 5.	Processing Unit Performance (50MHz Operation).....	9
Figure 6.	Processing Unit Performance (66MHz Operation).....	10
Figure 7.	7902 Internal Block Diagram	11
Figure 8.	CPU Interface Signal List.....	13
Figure 9.	External RAM Interface Signal List	14
Figure 10.	Miscellaneous Signal List.....	14
Figure 11.	Internal Clock and PLL Circuit	15
Figure 12.	7902 Operating Frequency Range	15
Figure 13.	7902 Memory Map.....	16
Figure 14.	Byte Order in Memory	16
Figure 15.	7902 Register Summary	17
Figure 16.	Internal Memory Map.....	18
Figure 17.	MPC860 / 7902 Interface	21
Figure 18.	Input Sample Window	22
Figure 19.	MPC860 Single-Beat Write and Read Timing	24
Figure 20.	MPC860 Burst Transfer Read and Write Timing.....	26
Figure 21.	Transfer Error Acknowledge (TEA#) Timing	27
Figure 22.	Transfer Sizes Supported (MPC860 Mode)	29
Figure 23.	Public Key Operations.....	33
Figure 24.	Public Key Operand Storage in Internal Memory	34
Figure 25.	Operand Pointers and Offsets	34
Figure 26.	Field Values and Memory Locations.....	35
Figure 27.	Operand Use for Each Operation.....	35
Figure 28.	Public Key Memory Use General Purpose Example	38
Figure 29.	Packet Engine Block Diagram.....	44
Figure 30.	Command Structures	48
Figure 31.	Typical Use of Descriptors For a Command That Requires Encryption Context	56
Figure 32.	Result Structures.....	58
Figure 33.	Context Memory Modes.....	61
Figure 34.	External RAM Memory Usage	62
Figure 35.	Absolute Maximum Ratings	69
Figure 36.	Recommended Operating Conditions	69

Figure 37. DC Electrical Characteristics.....	70
Figure 38. AC Specification Definition	71
Figure 39. AC Specification Derating.....	71
Figure 40. Reset Timing	71
Figure 41. External Clock.....	72
Figure 42. Read/Write CPU Timing (Single Beat)	73
Figure 43. External SRAM Read Timing.....	74
Figure 44. External SRAM Write Timing.....	75
Figure 45. Thermal Specifications	76
Figure 46. Pin List (Numeric)	77
Figure 47. Pin List (Alphabetical).....	78
Figure 48. 144-Pin TQFP Package	79

THIS PAGE INTENTIONALLY BLANK

1 Product Description

The 7902 Network Security Processor implements symmetric key encryption, public key encryption, authentication, and data compression in hardware. Its pipelined architecture allows many of these functions to be performed in a single pass. The integrated algorithms support standard network security protocols including IPSec, PPTP, L2TP, PPP, and others. The algorithms implemented by the processor include DES, Triple-DES, and RC4 encryption, SHA-1 and MD5 hash algorithms, HMAC functions, and LZS and MPPC data compression.

The 7902 also includes a math processor and a true hardware random number generator. These features are provided to support the public key cryptography required for key generation, exchange, and authentication such as used by X.509.

The processing speed of the 7902 supports the equivalent of two full duplex T1/E1 data communication lines. The 7902 interfaces directly with the MPC860 bus; no external interface logic is required with this bus. The 7902 also interfaces with the MPC8260 with a minimal amount of external logic.

Figure 1 shows the placement of a 7902 in a typical VPN Router.

Features

- Pipelined security processor supporting major security protocols including IPSec, PPTP, L2TP, PPP, and IKE
- Symmetric key encryption (DES, Triple-DES, and RC4)
- Authentication (SHA-1 and MD5)
- Compression (LZS and MPPC)
- Public Key processing unit (2048-bit modular arithmetic and exponentiation)
- Random Number Generator
- Internal memory buffers data packets
- Direct interface to 50 MHz MPC860 bus
- Minimal external logic interface to MPC8260 bus
- Maximum frequency of 66 MHz

Part Number	Package
7902PT6	144-Pin TQFP

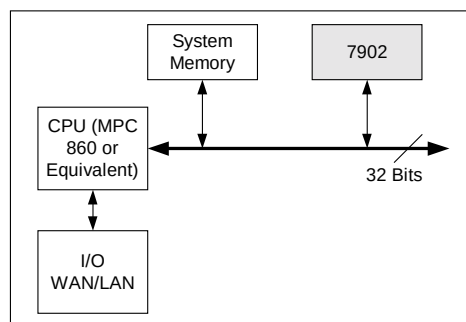


Figure 1. Typical VPN Router Example

2 Performance Summary

The figures in this section summarize the performance of the 7902 functional units. Performance of the 7902 when multiple engines are used (for example, the compression, MAC and encryption engines are all engaged) can be approximated by considering the 7902 throughput of the slowest engine.

The MAC and encryption engine speeds are accelerated (effectively multiplied) by the actual compression ratio achieved by the compression engine. For example, if the achieved compression ratio is 2:1, then the MAC and encryption engine speeds are effectively doubled and the compression engine would be the slowest engine. This performance data reflects the following conditions:

1. 1500-byte packets.
2. Single session or security association.

2.1 Symmetric Key Processing Units

Protocol	50 MHz Performance	66 MHz Performance
DES	143 Mbps	188 Mbps
3DES	53 Mbps	70 Mbps
RC4	78 Mbps	103 Mbps
SHA-1	50 Mbps	66 Mbps
MD5	60 Mbps	79 Mbps
LZS Compression	40 Mbps	52 Mbps
LZS Decompression	75 Mbps	99 Mbps
MPPC Compression	33 Mbps	43 Mbps
MPPC Decompression	67 Mbps	88 Mbps

Note: The performance numbers are based on simulation results. Text from the United States Constitution was arbitrarily selected for the compression and decompression simulations. The compression ratio is approximately 2:1.

Figure 2. Processing Unit Performance

2.2 Protocols

Protocol	50 MHz		66 MHz	
	Performance	No. Packets/sec	Performance	No. Packets/sec
IPSec (3-DES, SHA-1, LZS)	37 Mbps	25 K	48 Mbps	33 K
PPTP (RC4, MPPC)	32 Mbps	40 K	42 Mbps	52 K

Note: The performance and packets/sec numbers are based on simulation results. The performance numbers are based on the rate of the uncompressed data; the data was arbitrarily selected from the United States Constitution. The compression ratio is approximately 2:1.

Figure 3. Protocol Performance

2.3 Public Key

IKE Handshake	No. Connections/sec 50 MHz	No. Connections/sec 66 MHz
Two 1024-bit Diffie-Hellman operations (Quick Mode)	6	8
Two 1024-bit Diffie-Hellman operations, 1 RSA sign, 2 RSA verifies (Main Mode)	3	4
Four 1024-bit Diffie-Hellman operations, 1 RSA sign, 2 RSA verifies (Main Mode + Quick Mode)	2	2.7

Notes: 180-bit exponent. The number of connections/sec is based on simulation results.

Figure 4. IKE Performance

Operation – 50 MHz	Time to complete			
	2048-bit key	1024-bit key	768-bit key	512-bit key
RSA private key	873.9 ms	125.4 ms	49.5 ms	18.5 ms
RSA public key (3-bit exponent)	4.8 ms	1.3 ms	0.8 ms	0.4 ms
Diffie-Hellman (180-bit exponent)	286.5 ms	76.6 ms	50.9 ms	21.8 ms
Diffie-Hellman (exponent = key size)	3260.4 ms	435.6 ms	216.5 ms	62.1 ms
DSA sign		136.7 ms	82.2 ms	39.6 ms
DSA verify		205 ms	123.2 ms	59.4 ms

Notes: Performance assumes a uniform distribution of ones and zeros in the exponent. The performance numbers are based on simulation results.

Figure 5. Processing Unit Performance (50MHz Operation)

Operation – 66 MHz	Time to complete			
	2048-bit key	1024-bit key	768-bit key	512-bit key
RSA private key	662 ms	95.0 ms	37.5 ms	14.0 ms
RSA public key (3-bit exponent)	3.6 ms	0.95 ms	0.58 ms	0.27 ms
Diffie-Hellman (180-bit exponent)	217 ms	58.0 ms	38.5 ms	16.5 ms
Diffie-Hellman (exponent = key size)	2,470 ms	330 ms	164 ms	47.0 ms
DSA sign		103.5 ms	62.2 ms	30.0 ms
DSA verify		155.3 ms	93.3 ms	45.0 ms

Note: Performance assumes a uniform distribution of ones and zeros in the exponent. The performance numbers are based on simulation results.

Figure 6. Processing Unit Performance (66MHz Operation)

3 Product Overview

3.1 Operation

The 7902 contains several processing units—Random Number Generator, Public Key, symmetric key encryption, compression, padding, and authentication. The symmetric key encryption, compression, padding, and authentication units are combined into a single functional block labeled *Packet Engine*.

In normal operation, the host buffers data packets (or fragments) and commands into the internal memory of the 7902. Since internal memory is memory mapped, there is no need for sophisticated external DMA hardware. Simple memory-to-memory DMA transfers or CPU I/O transfers may be used. As the 7902 executes each command, the resulting data packets are stored back into internal memory. The host then transfers the resulting data out.

The 7902 internal block diagram is shown in Figure 7.

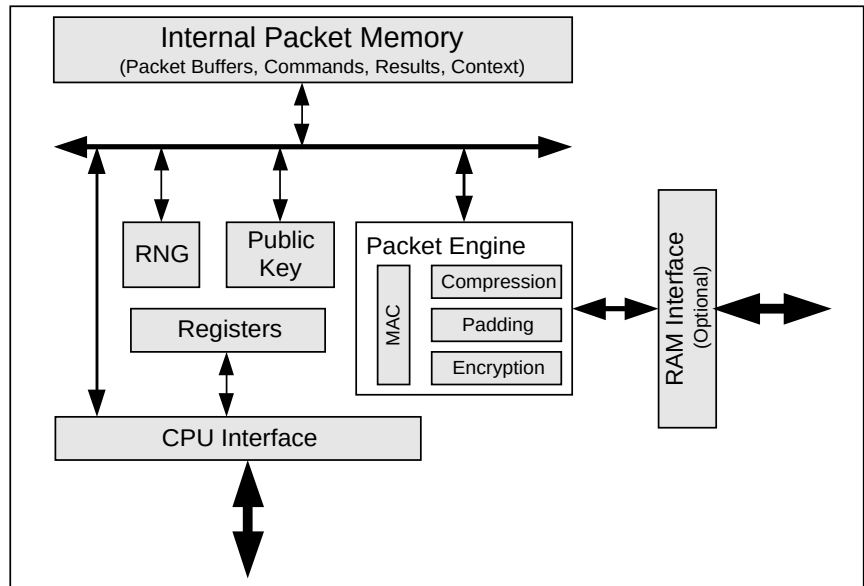


Figure 7. 7902 Internal Block Diagram

3.2 Functional Blocks

3.2.1 Internal Packet Memory

The internal packet memory (IPM) is 8K bytes in size and is used to buffer data packets, command structures, context information, context information, and descriptors.

The IPM is mapped into the host memory space and is accessed as host memory. See section 5 Memory Map for more information and the memory map.

3.2.2 RAM Interface

The RAM interface controls an external 16-bit memory device. Up to 256Kx16 of SRAM is supported. External memory is required only if the data compression engine is used. External memory has space available for keys and Initialization Vectors (IVs) to be stored. Therefore, they don't need to be supplied for every encryption/authentication operation.

This memory is not directly accessible by the host CPU as it is dedicated for use by the compression engine.

3.2.3 Registers

Programmable registers are used to store configuration information for the functional units. The registers are all memory-mapped.

3.2.4 Processor Bus Interface

The synchronous Processor Bus Interface gluelessly supports the MPC860 bus. With a minimal amount of external logic, the MPC8260 bus interface is supported. Since all registers and internal memory are accessed as system memory, there is no need for DMA handshaking signals. Data transfers are performed with memory-to-memory DMA transfers or CPU I/O transfers.

3.2.5 Packet Engine

The Packet Engine contains the symmetric encryption, compression, padding, and authentication processing units. The Packet Engine accesses data packets and commands directly from the IPM. The resulting data and status information are stored back into IPM.

3.2.6 Random Number Generator

The Random Number Generator (RNG) is based on internal free-running oscillators whose frequencies drift relative to each other and to the 7902's internal clock. The phase relation of these signals is unpredictable, and this is used to provide a random bit stream. The host must further process this random bit stream to create cryptographic-quality random numbers.

3.2.7 Public Key

The Public Key processing unit implements modular arithmetic functions. The Public Key processing unit accesses operands and commands directly from the IPM. The arithmetic results are stored back into IPM.

4 Signal Description

4.1 Signal List

4.1.1 CPU Interface

Name	Type	Description																					
A[13-0]/MA[18-31]	I	Address Bus. Important: The address bus lines have dual names since the signal names and signal ordering differ with different processors. MA[18-31] signal names are for use with MPC860/8260 processors. A[13-0] signal names are for use with other processors.																					
D[31-0]/MD[0-31]	I/O8	Data bus. Important: The data bus lines have dual names since the signal names and signal ordering differ with different processors. MD[0-31] signal names are for use with MPC860/8260 processors. D[31-0] signal names are for use with other processors.																					
TSIZ[0-3]	I	Transfer Size. Important: A subset of the transfer sizes available with the MPC860 and MPC8260 processors is supported. The transfer sizes supported are shown in the table below: <table> <tr> <th>TSIZ[0-1]</th><th>BURST#</th><th>MPC860</th></tr> <tr> <td>00</td><td>1</td><td>4 bytes</td></tr> <tr> <td>00</td><td>0</td><td>16 bytes</td></tr> </table> Note: Other transfer sizes are not supported. TSIZ[0-1] must be connected for proper operation of TEA#. TSIZ[2-3] should be tied low. <table> <tr> <th>TSIZ[0-3]</th><th>BURST#</th><th>MPC8260</th></tr> <tr> <td>0000</td><td>1</td><td>8 bytes</td></tr> <tr> <td>0010</td><td>0</td><td>32 bytes</td></tr> <tr> <td>0100</td><td>1</td><td>4 bytes</td></tr> </table> Note: Other transfer sizes are not supported. TSIZ[0-3] must be connected for proper operation of TEA#.	TSIZ[0-1]	BURST#	MPC860	00	1	4 bytes	00	0	16 bytes	TSIZ[0-3]	BURST#	MPC8260	0000	1	8 bytes	0010	0	32 bytes	0100	1	4 bytes
TSIZ[0-1]	BURST#	MPC860																					
00	1	4 bytes																					
00	0	16 bytes																					
TSIZ[0-3]	BURST#	MPC8260																					
0000	1	8 bytes																					
0010	0	32 bytes																					
0100	1	4 bytes																					
IRQ#	3S	Interrupt Request. Important: When this signal is not being driven low by the 7902 for an interrupt request, it is in a high-z state. An external pull-up resistor should be tied to this signal.																					
CS#	I	Chip Select.																					
BURST#	I	Burst Transfer.																					
TS#	I	Transfer Start.																					
TA#	3S	Transfer Acknowledge.																					
AACK#	3S	Address Acknowledge. (Used with 8260 only.)																					
DBB#	I	Data Bus Busy. (Used with 8260 only.)																					
RW#	I	Read / Write. RW# is not used with the MPC8260. In the MPC8260 mode, RW# functions as TT1.																					
TEA#	3S	Transfer Error Acknowledge.																					
BDIP#/PSDVAL#	I/3S	BDIP# - 860 mode (input) - Burst data in progress. PSDVAL# - 8260 mode (tri-state) - Partial Data Valid.																					
CLK	I	Bus Clock.																					
BMODE	I	Bus Mode. Low = MPC8260 mode. High = MPC860 mode.																					

Notes: I-input, O-output, I/O-Input/Output, 3S-TriState. Refer to the DC Specifications for details regarding the output drive capabilities.

Figure 8. CPU Interface Signal List

4.1.2 External RAM Interface

Name	Type	Description
CA[17-0]	O8	SRAM Address. Important: Bit 0 is least-significant bit.
COE#	O8	SRAM Output Enable.
CD[15-0]	I/O8	SRAM Data. Important: Bit 0 is least-significant bit.
CUB#	O4	SRAM Upper Byte Enable.
CLB#	O4	SRAM Lower Byte Enable.
CWE#	O8	SRAM Write Enable.

Notes: O-output, I/O-Input/Output. Refer to the DC Specifications for details regarding the output drive capabilities.

Figure 9. External RAM Interface Signal List

4.1.3 Miscellaneous

Name	Type	Description
RESET#	I	Reset.
PLLE#	I	PLL enable input. High=PLL disabled, low=PLL enabled.
NDTEST#	I	NAND Tree Test Enable input. High = normal mode, low = NAND tree test mode.
NDPI	I	NAND Tree input. This input should be low in normal mode.
NDPO	O4	NAND Tree output. This pin should be left open in normal mode.
VDD		+3.3 volts.
VDD2		+2.5 volts.
VSS		Ground.
AVDD2		PLL VDD. Connect to analog +2.5 volts.
AVSS		PLL VSS. Connect to analog ground.
NC		No connection. Important: Pins labeled as NC must not have anything connected to them.

Notes: O-output, I/O-Input/Output. Refer to the DC Specifications for details regarding the output drive capabilities.

Figure 10. Miscellaneous Signal List

4.2 Clocks

The 7902 has a single clock input signal (CLK). This clock signal drives I/O timing as well as the operation of the functional units (the Random Number Generator, Public Key processor, and Packet Engine).

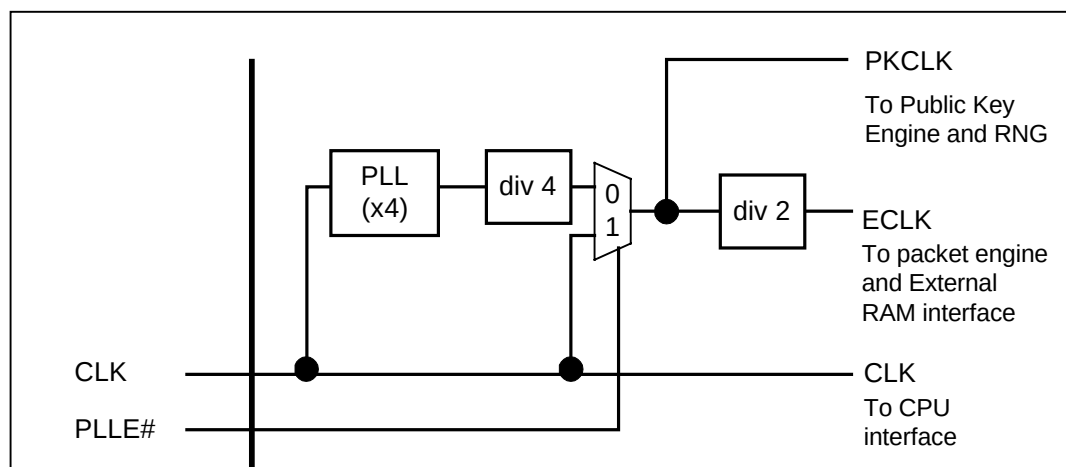


Figure 11. Internal Clock and PLL Circuit

4.3 Phase Lock Loop (PLL)

The 7902 has an integrated PLL that does not require external components. The PLL should be enabled or disabled based on the input clock frequency (see Figure 11 and Figure 12).

PLL	CLK
Disabled	< 40MHz
Enabled	40-66MHz

Figure 12. 7902 Operating Frequency Range

4.4 Reset

Reset halts all the internal functional blocks and resets the registers. A reset may be initiated with the RESET# signal, or the RESET bit in the Configuration Register. All registers return to their reset value as defined in each register description.

5 Memory Map

All registers and internal memory is accessed by the system with normal memory transfers. The 7902 occupies 16 Kbytes of the system memory map. The memory map is shown in Figure 13.

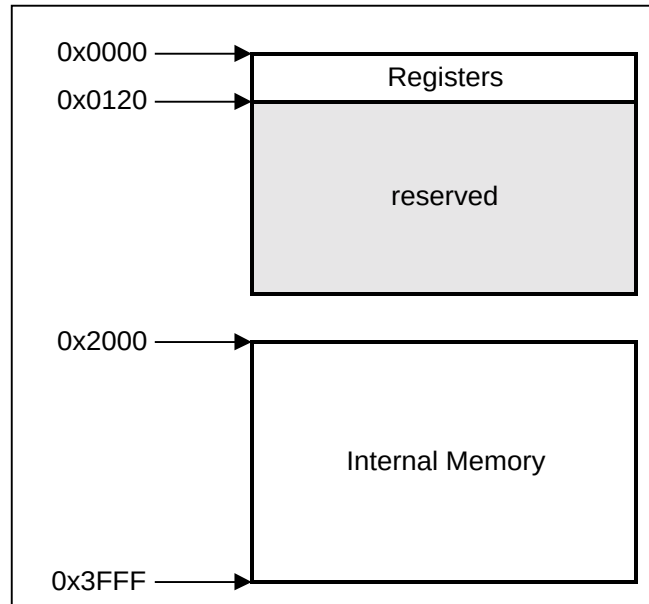


Figure 13. 7902 Memory Map

5.1 Byte and Bit Ordering

5.1.1 Byte Ordering

All addresses are presented as byte addresses. Multi-byte transfers are assumed to be ordered in big-endian style. For example, the byte string 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27 would appear in memory as shown in Figure 14.

Address	00	01	02	03	04	05	06	07
Contents	0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27

Figure 14. Byte Order in Memory

This byte string would be transferred over the 32-bit bus as 0x20212223, 0x24252627.

5.1.2 Bit Ordering

The convention used for bus signal labels with the MPC860/8260 is the opposite order from other processor buses. The MPC860/8260 convention uses bit zero as the most-significant bit while other processors use bit zero as the least-significant bit. This convention applies to address signals, data signals, and register bits.

This datasheet adopts the convention that the least-significant bit is always labeled as bit zero; however, the address and data bus pins have been given dual names to make it easier when using processors supporting the MPC860/8260 bit

ordering convention. The address bus pin names are A[13-0]/MA[18-31] and the MA[18-31] signals should be used with processors supporting the MPC860/8260 convention. Similarly, the data bus pin names are D[31-0]/MD[0-31] and the MD[0-31] signals should be used with processors supporting the MPC860/8260 convention.

5.2 Registers

The 32-bit programmable registers are mapped into the first 8 Kbytes of the memory address space (0x0000 to 0x1FFF), although only a small subset of this address space is actually used. A register summary is shown in Figure 15.

Category	Address	Register
General	0x0004	General Configuration
	0x0008	Chip ID
Symmetric Key and Compression	0x0080	Host Command Index
	0x0084	Host Source Index
	0x0088	Host Result Index
	0x008C	Host Destination Index
	0x0090	Packet Status
	0x0094	Packet Interrupt Enable
	0x0098	Packet Configuration
Public Key and RNG	0x0100	Public Base Address
	0x0104	Public Operand Length
	0x0108	Public Operation
	0x010C	Public Status
	0x0110	Public Interrupt Enable
	0x0114	RNG Configuration
	0x0118	RNG Data

Note: Unused addresses are reserved and must not be accessed.

Figure 15. 7902 Register Summary

5.3 Internal Memory

Internal memory is mapped into the address range 0x2000 to 0x3FFF. The size of this memory is 8 Kbytes. Data may be read and written to this memory in the same manner as system memory.

Certain regions of internal memory are reserved for special functions. These are identified in the internal memory map shown in Figure 16. The base address associated with descriptor rings is fixed. The base address associated with public key operations may be configured in the Public Base Address Register.

Warning: The Public Key memory space should not be accessed while a Public Key operation is taking place. The PUBLIC DONE bit in the Public Status Register may be used to determine if a Public Key operation is in progress.

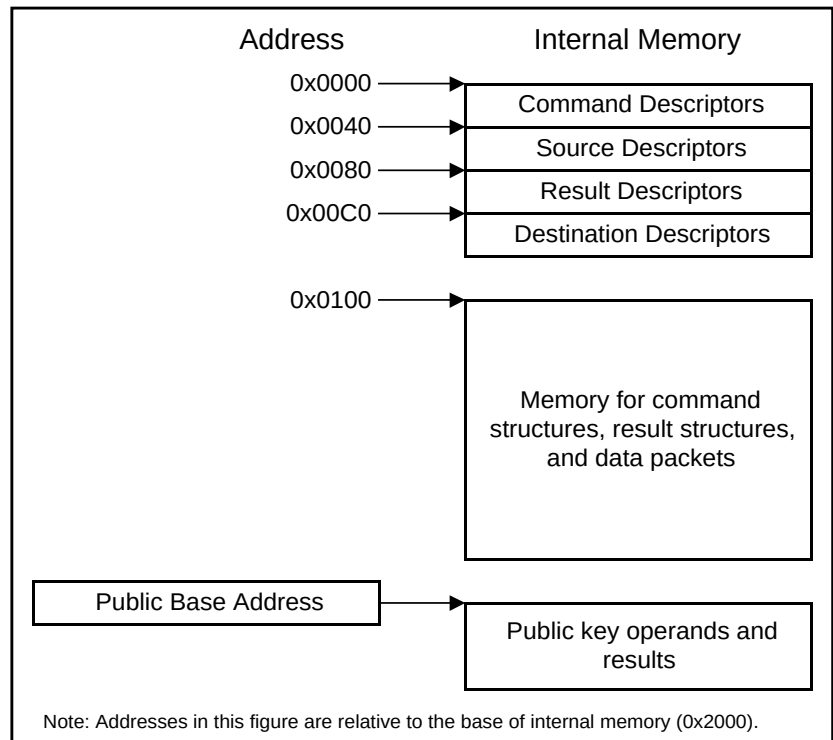


Figure 16. Internal Memory Map

6**External RAM**

The RAM interface controls an external 16-bit memory device. Use of external memory is required only if data compression is used; otherwise, external memory is not required. This memory is dedicated to the compression engine and is not directly accessible from the system.

The 7902 supports up to 256Kx16 external memory. The RAM interface signals connect directly to the 7902 without any additional glue logic.

Each full-duplex LZS session requires 16 Kbytes of external RAM. Each full-duplex MPPC session requires 32 Kbytes. This allows support of up to 32 full-duplex LZS sessions or 16 full-duplex MPPC sessions.

The external RAM address used by the packet processor command is set by the `SESSION #` field in the Base Command Structure. The `SESSION #` field is an index into the external RAM. Each unit is an offset of 16 Kbytes starting at address location zero.

Separate encode and decode commands while using LZS compression may use the same `SESSION #` value. The memory use of compression and decompression do not overlap within the 16 Kbyte memory range.

However, MPPC requires separate `SESSION #` values for encode and decode commands. Each half-duplex MPPC session requires a full 16 Kbytes.

7

Processor Bus Interface

The 7902 CPU interfaces gluelessly to the MPC860 bus and with minimal external logic to the MPC8260 bus. Because there are some differences between the MPC860 and the MPC8260 bus interface, the bus mode interface is selected using the BMODE signal.

The MPC860 bus is a synchronous, burstable bus. The 7902 supports a subset of the transfer sizes available with the MPC860 controlled by the $\tau_{SIZ[0-1]}$ inputs.

The 7902 interface is similar in many respects to an SRAM. Data can be transferred between the CPU and the 7902 using standard load and store CPU operations or memory to memory DMA operations. It also supports a burst mode for faster read/write operations.

Other CPUs can be used with the 7902 provided the interface meets function and timing requirements (see Figure 42).

7.1 MPC860/8260 Bus Interface

Since there are some differences between the MPC860 and MPC8260 buses, the BMODE input signal is provided to set the bus interface logic appropriately. When BMODE is active (high), the MPC860 mode is enabled.

7.1.1 MPC860 Bus Interface

The 7902 incorporates the necessary logic to gluelessly interface to the MPC860. The 7902's interface logic supports a minimal subset of the MPC860's bus signals. These are the signals necessary to support both single-beat and burst data transfers.

Interfacing to the 7902 requires the use of one of the General Purpose Chip-select Machines (GPCMs) within the memory controller of the MPC860 to generate the chip select signal. The GPCM should be initialized for a 32-bit port size, no parity checking, burst, and external TA.

Figure 17 shows the interface connections between the MPC860 and the 7902.

7.1.2 MPC8260 Bus Interface

With minimal external logic, the 7902 interfaces with the MPC8260 bus. For additional information, refer to the 7902 Application Note (AN-0023).

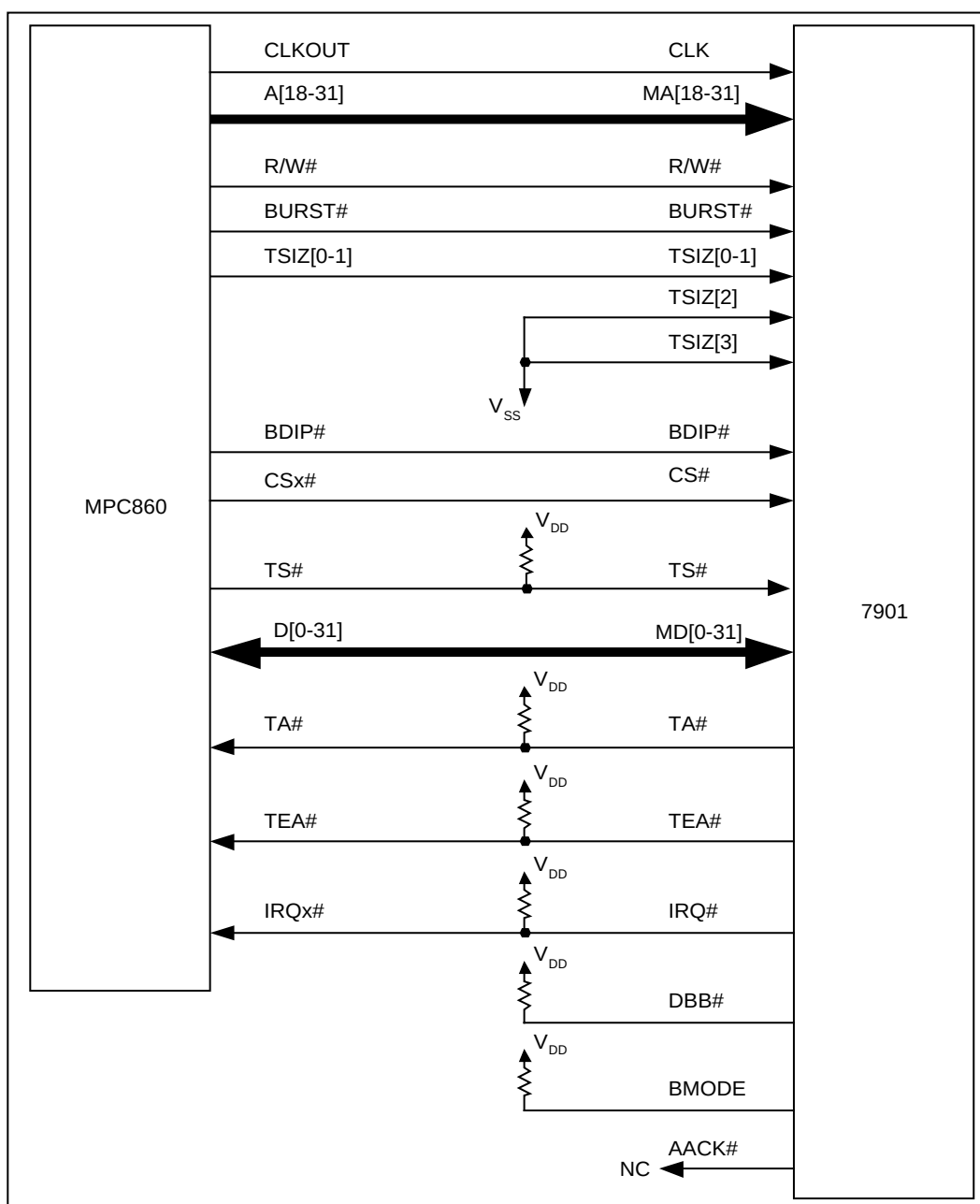


Figure 17. MPC860 / 7902 Interface

7.2 Bus Transfer Overview – MPC860 Mode

The data bus transfers information between the CPU and the 7902. Transfers are always performed using the full 32-bit width of the data bus.

The 7902's address bus specifies the address for the transfer. Control signals indicate the beginning of the cycle and the type of cycle, as well as the address space and size of the transfer. The 7902 controls the cycle length with the $\tau_{A\#}$ signal, which terminates the cycle. The strobe signal ($\tau_{S\#}$) indicates the validity of the address as well as the direction, type, and size of the transfer and provides timing information. Because the 7902's bus is synchronous, the bus and control input signals must be timed to set-up and hold times relative to the rising edge of the clock. The 7902 requires two wait states, so single-beat bus cycles are fixed at four clock cycles.

Furthermore, for all inputs, the 7902 latches the input's level during a sample window, shown in Figure 18, around the rising clock edge. To ensure that an input signal is recognized on a specific rising clock edge, that input must be stable during the sample window. If an input changes during the window, the level recognized by the 7902 is unpredictable; however, the 7902 always resolves the latched level to either a logical high or low before using it. For deterministic operation, all input signals must obey the protocols described in this chapter in addition to meeting input set up and hold times.

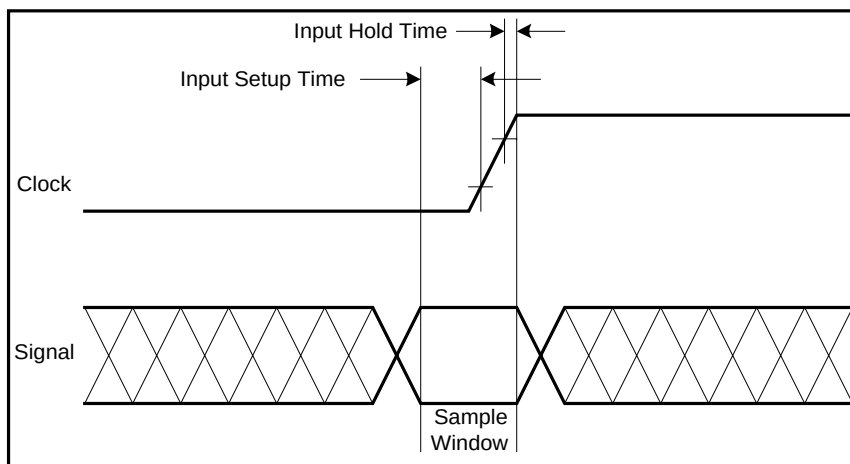


Figure 18. Input Sample Window

$\tau_{SIZ}[0:1]$ indicates the number of bytes to be transferred during an operand cycle (consisting of one or more bus cycles). These signals should be valid at the rising edge of the clock in which the transfer start signal ($\tau_{S\#}$) is asserted.

7.2.1 Single-Beat Transfer

The 7902 supports the read and write of 4-byte single-beat transfers. The transfer must be word aligned (the two LSBs of the address bus must be 0) and $\tau_{SIZ}[0:1]$ driven appropriately, otherwise the $\tau_{EA\#}$ signal is asserted to indicate a transfer error. The 7902 also has a fixed two-wait state requirement for read and write transfers regardless of the clock frequency.

The address transfer phase starts with the CPU asserting $\tau S\#$. At the same time that $\tau S\#$ is asserted, $CS\#$ is also asserted and the address is placed on the address bus. $\tau SIZ[0-1]$ is set appropriately for the transfer size and $R/W\#$ is driven to indicate the direction of the data transfer. $\tau S\#$ is only asserted for one cycle while all other signals are held valid (except $CS\#$) by the CPU until the transfer is terminated. It is not required that $CS\#$ be held valid until the transfer is terminated although it is acceptable.

The data transfer starts one clock cycle after $\tau S\#$ has been asserted. At this time, the CPU drives the data bus with the data to be written if the transfer is a write; otherwise, the 7902 drives the data bus with the data to be read by the CPU. The 7902 has a fixed two-wait state requirement before the data can be latched for write operations or read for read operations. After two wait states $\tau A\#$ is asserted and the 7902 latches the data (write transfer) or indicates to the CPU that the data can be read (read transfer). $\tau A\#$ is then driven high for one clock and then released (tri-stated). Figure 19 shows the basic timing for a single-beat read/write transfer.

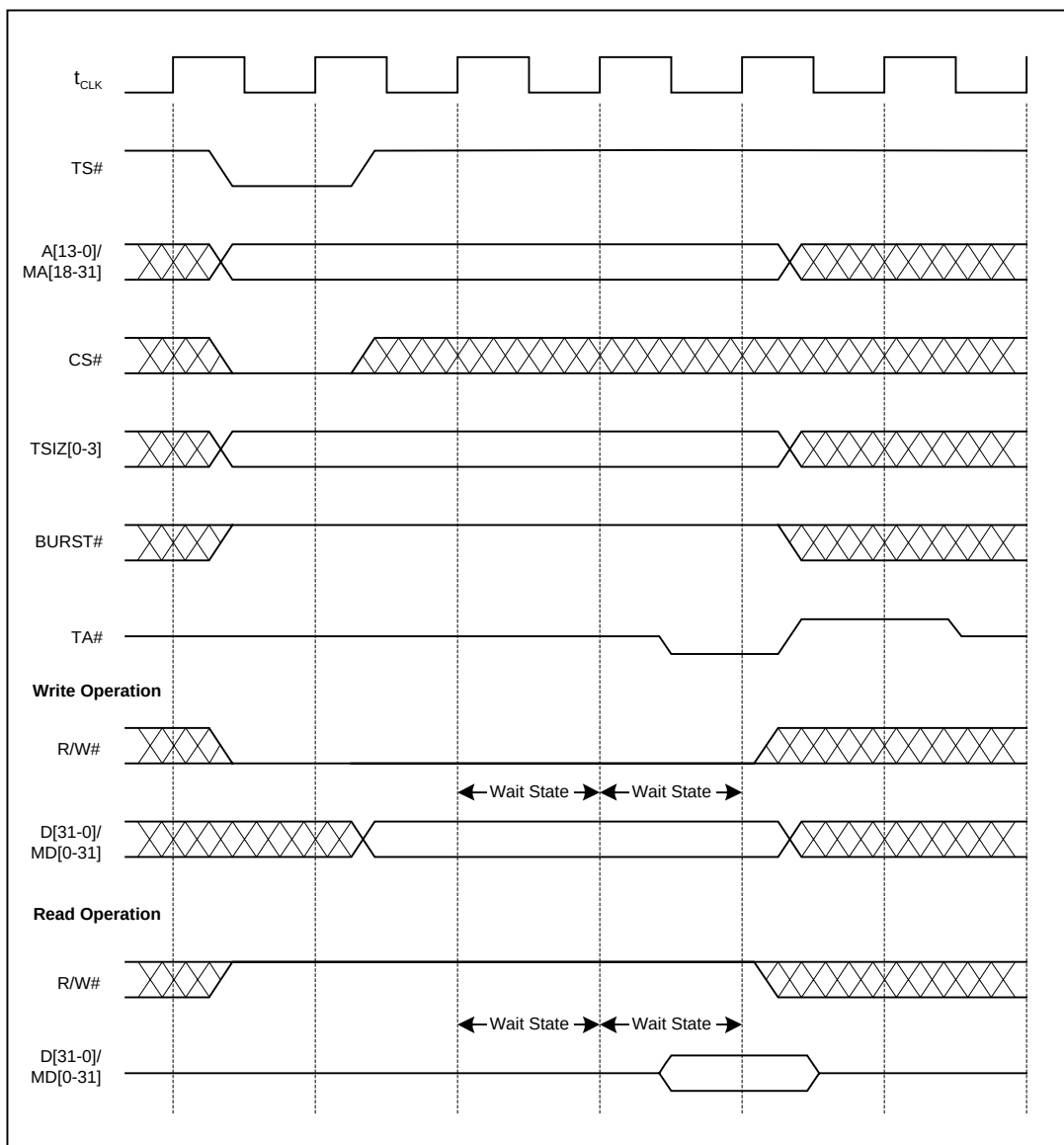


Figure 19. MPC860 Single-Beat Write and Read Timing

7.2.2 Burst Transfers

Burst transfers are similar in operation to a single-beat transfer. Single-beat transfers only support the read/write of 4-bytes while a burst transfer supports 16-bytes. Burst transfers must be aligned to a 16-byte memory boundary.

The CPU supplies the starting address that points to the first word and the 7902 samples/drives each word on the data bus until all four words are transferred.

Address and transfer attributes supplied by the CPU must remain stable (except $CS\#$) during the entire transfer. Each word transfer on the data bus is terminated with the assertion of $TA\#$; therefore, $TA\#$ is asserted for four clocks.

Burst Write

During the data transfer of a burst write cycle, the CPU sends data to the 7902. $BDIP\#$ is asserted to indicate that the CPU intends to continue the data transfer beyond the first word. When the 7902 receives the data word, it asserts $TA\#$ to indicate to the CPU that it is ready for the next word transfer. The CPU again drives the next data word and $BDIP\#$ remains asserted. If there is no more data to write, $BDIP\#$ is deasserted to indicate to the 7902 that the next data word is the last one in the burst write.

Burst Read

During the data transfer of a burst read cycle, the CPU receives data from the 7902. If the CPU needs more than one data word, it asserts $\text{BDIP\#}$. The CPU deasserts $\text{BDIP\#}$ prior to the last word that is transferred. Thus, the 7902 stops driving new data at the rising clock edge after $\text{BDIP\#}$ goes high.

Since the 7902 only supports 32-bit transfers and 16-byte bursts, the burst must be 4 beats. See Figure 20 for the basic timing of read and write burst transfers.

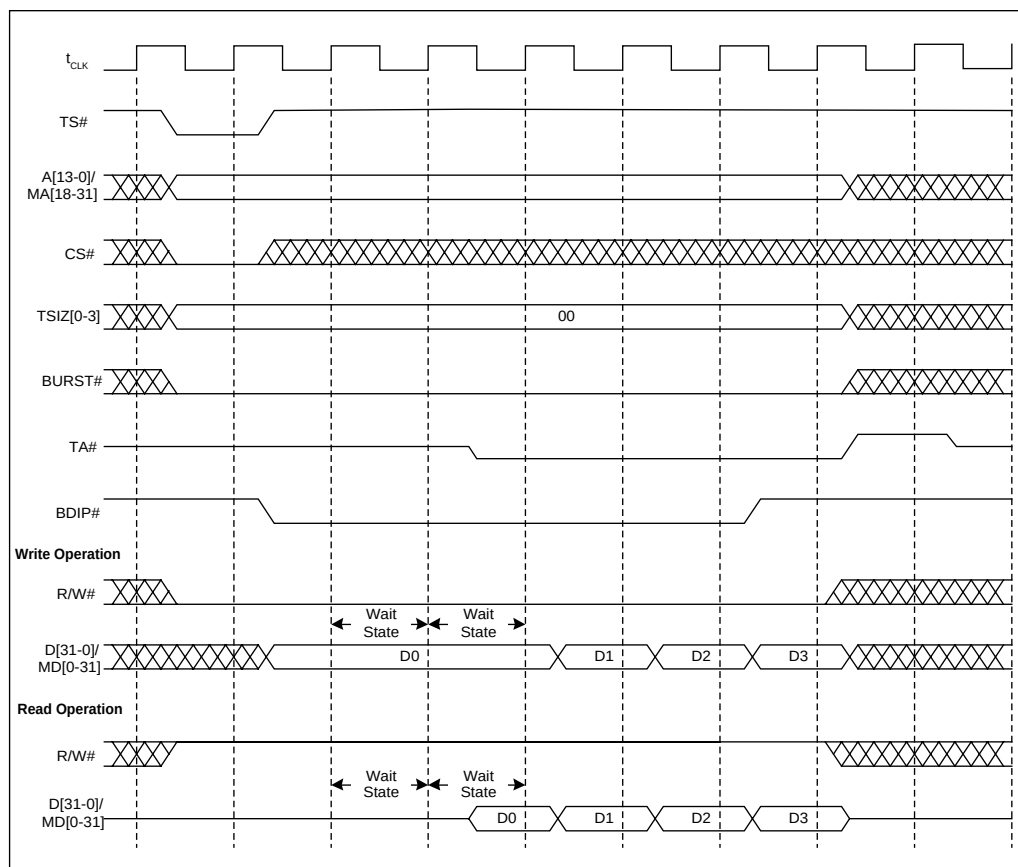


Figure 20. MPC860 Burst Transfer Read and Write Timing

7.2.3 Transfer Errors

The 7902 supports a subset of the transfer sizes and alignment available with the MPC860. See Figure 21 for the basic timing of $TEA\#$. The transfer error acknowledge ($TEA\#$) signal is used to indicate to the CPU when an unsupported transfer size is being attempted. If the transfer size being requested at the beginning of transfer (as set on the $TSIZ[0-1]$ signals) is not a size supported (see Figure 22 for the sizes supported) then $TEA\#$ will be asserted. If the CPU attempts to do a data transfer that is not word aligned ($A[0-1]/MA[31-30]$ are low) then $TEA\#$ will be asserted.

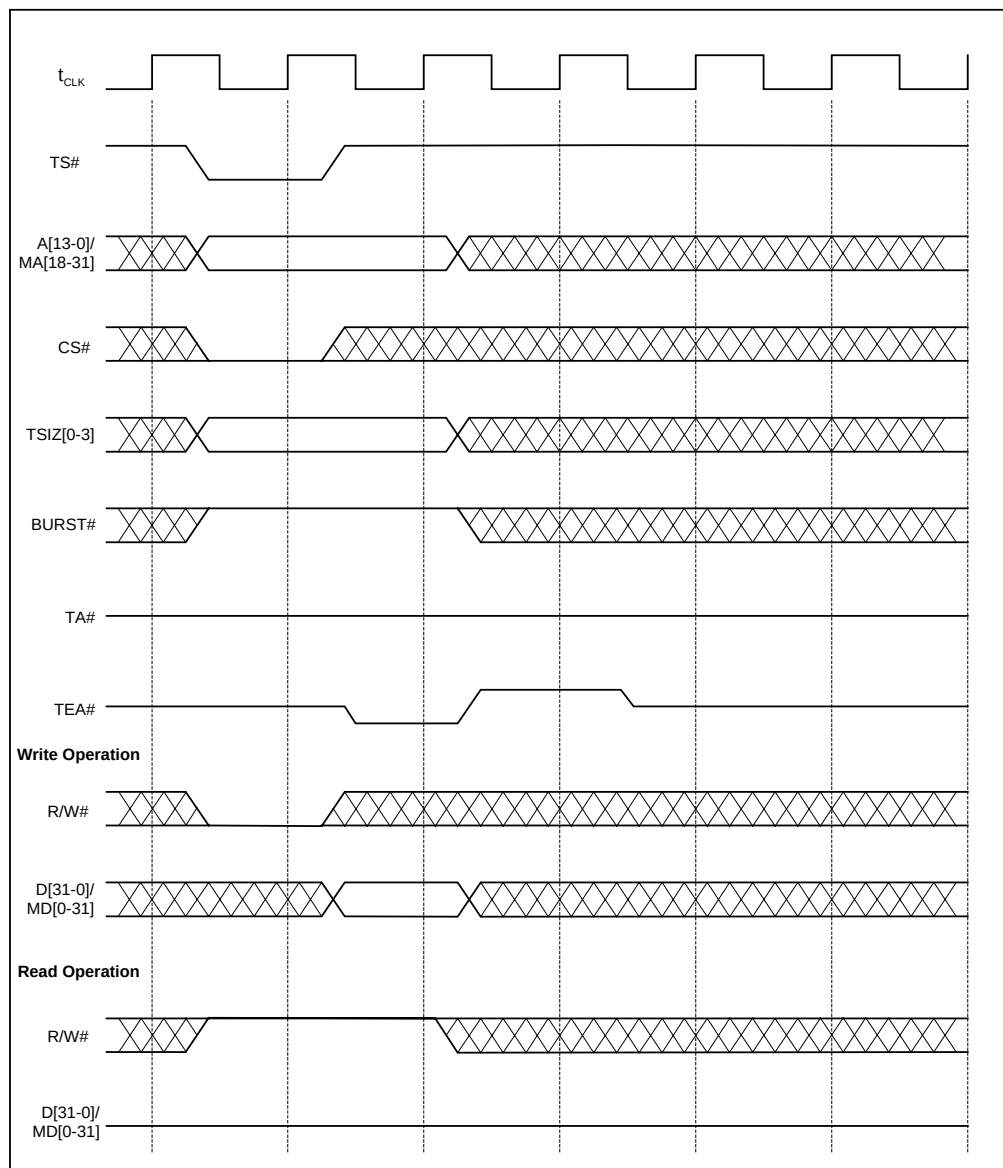


Figure 21. Transfer Error Acknowledge ($TEA\#$) Timing

7.3 Processor Bus Interface Signal Descriptions

7.3.1 Bus Interface Clock

The CLK input signal drives the Random Number Generator, Public Key, and Packet Engine as well as the synchronous bus interface. The MPC860/8260 processors' bus clock output must be tied to the 7902's CLK input for proper operation of the Processor Bus Interface.

7.3.2 Address Bus (A[13-0]/MA[18-31])

The address bus signals have been given dual names to clearly indicate the correct connection to the CPU being used. The MA[18-31] signals are used with processors supporting the MPC860/8260 bit numbering conventions and the A[13-0] signals are used with other processors.

The address bus consists of 14 signals (A[13-0]/MA[18-31]) that are inputs. A[0]/MA[31] is the LSB and A[13]/MA[18] is the MSB of the address. The value on the bus specifies the physical address of the register or internal memory location being accessed. The address bus is sampled when TS# and CS# are asserted. The address must be held on the bus until the cycle is complete as indicated by the 7902 asserting TA#.

The 7902 only supports word (32-bit) aligned data transfers so A[0-1]/MA[31-30] must always be low during a data transfer. If both bits are not low, then TEA# is asserted to indicate a transfer error.

7.3.3 Data Bus (D[31-0]/DM[31-0])

The data bus signals have been given dual names to clearly indicate the correct connection to the CPU being used. The MD[0-31] signals are used with processors supporting the MPC860/8260 bit numbering conventions and the D[31-0] signals are used with all other processors.

The data bus consists of 32 signals (D[31-0]/MD[0-31]) that are both inputs and outputs. D[0]/MD[31] is the LSB and D[31]/MD[0] is the MSB of the data. The data bus is normally in the input mode except during a read operation.

During a write operation the data is latched on the positive edge of the clock when TA# is asserted.

During a read operation, two clock cycles after TS# is asserted, the data bus is put in the output mode and the data is placed on the bus. The data is held valid while TA# is asserted and then the data bus is placed back in the input mode.

If a transfer error is detected the TEA# signal is asserted and the transfer cycle is terminated. If a transfer error occurs during a read operation, the data bus stays in the input mode, since no data is output.

7.3.4 Burst Transfer (BURST#)

The burst transfer signal (BURST#) is an input. This signal is asserted at the beginning of a bus cycle (along with the address) to indicate that the data transfer is a burst transfer. In the MPC860 mode, a 16-byte burst is initiated when BURST# is low. The TSIZ[1-0] signals must be driven according to Figure 22 for proper burst transfer operation.

7.3.5 Transfer Size (TSIZ[0-3])

The transfer size signals (TSIZ[0-3]) are inputs and indicate the size of the data transfer for the current transaction. A subset of the various transfer sizes available with the MPC860 are supported as shown in Figure 22. TEA# is asserted when the transfer size for the current transaction is not supported.

BURST#	TSIZ[0-1]	Transfer Size
1	00	Word (4 bytes)
0	00	Burst (16 bytes)

Note: In the MPC860 mode, TSIZ[3-2] are not used and should be tied low to prevent the input buffers from floating.

Figure 22. Transfer Sizes Supported (MPC860 Mode)

7.3.6 Chip Select (CS#)

The chip select signal (CS#) is an input and indicates when the CPU is requesting a transfer. CS# is sampled on the positive edge of the clock when TS# is asserted. If CS# is asserted, a data transfer is initiated with the direction based on the state of the read/write (R/W#) signal.

7.3.7 Transfer Start (TS#)

The transfer start signal (TS#) is an input and indicates the beginning of a transfer cycle. The TS# signal is a shared line among other devices on the bus and must be pulled high with a resistor. TS# is sampled on the positive edge of each clock. When TS# is asserted, CS# is sampled to determine if a transfer is being requested by the CPU. If CS# is asserted, then a data transfer is initiated with the direction based on the state of the read/write (R/W#) signal.

7.3.8 Transfer Acknowledge (TA#)

The transfer acknowledge signal (TA#) is a tri-stateable output and indicates the end of a data transfer. The TA# signal is a shared line among other devices on the bus and must be pulled high with a resistor. It is in a tri-state mode unless it is being driven at the end of the data transfer. At the end of a data transfer, the TA# line is driven low for one clock to indicate to the CPU that the transfer is complete. After TA# has been driven low for one clock, it is driven high for one clock and then it is released (tri-stated). If a transfer error occurred and TEA# was asserted, TA# is not asserted.

7.3.9 Address Acknowledge (AACK#)

The address acknowledge signal (AACK#) is only used in the MPC8260 mode and is a tri-stateable output. It should be left open when the MPC860 mode is used.

7.3.10 Data Bus Busy (DBB#)

The data bus busy signal (DBB#) is an input and is only used in the MPC8260 mode. It must be tied high when the MPC860 mode is used.

7.3.11 Read/Write (R/W#)

The read/write signal (R/W#) is an input and indicates the direction of the data flow for data transfers. R/W# is sampled on the positive edge of the clock when TS# is asserted and must be held in that state during the entire transfer. If R/W# is driven high, a read data transfer is in progress; if it is low, a write transfer is in progress.

7.3.12 Transfer Error Acknowledge (TEA#)

The transfer error acknowledge signal (TEA#) is an output that indicates to the CPU when an unsupported data transfer is requested (see Figure 22 for the transfer sizes supported). TEA# is asserted if an unsupported transfer size is attempted or if a non-word aligned data transfer is attempted (A[0-1]/MA[31-30] are not low). If a transfer error occurs, TEA# is asserted for one clock starting on the positive edge of the bus clock after TS# is sampled low.

7.3.13 Burst Data In Progress/Partial Data Valid (BDIP#/PSDVAL#)

In the MPC860 mode, this pin is the BDIP# signal and is an input indicating that a burst data transfer is in progress. BDIP# is sampled on the positive edge of the clock when TS# and BURST# are asserted. It must remain asserted for the duration of the burst data, and then be deasserted prior to the last data word.

7.3.14 Bus Mode (BMODE)

The bus mode signal (BMODE) is an input and is used to set the bus interface mode. When BMODE is high, the MPC860 bus interface is used. When BMODE is low, the MPC8260 bus interface is used.

Note: If another processor is used, it is recommended that BMODE be tied high and the MPC860 bus used.

7.3.15 Interrupt Request (IRQ#)

The interrupt request signal (IRQ#) is a tri-stateable output and indicates an interrupt request. It should be pulled high with a resistor. It is in a tri-state mode unless it is being driven low indicating that an enabled interrupt condition has been met. The IRQ# signal remains low until the status bit for the event that caused the interrupt is cleared.

8 General Registers

8.1 General Configuration Register

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
reserved																															Reset
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0004-0x0007																Use this register to configure general 7902 functions.															
Field																Description															
RESET																Set this bit to 1 to reset the device. When read, this bit is a 1 while the device is performing a reset operation (either a hardware or software reset). The device should not be accessed while this bit is a 1 (except to read the General Configuration Register). After the reset operation is complete, this bit returns to 0.															

8.2 Chip ID Register

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
Product ID																Revision ID															
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0008-0x000B								This register reports the ChipID. The ChipID uniquely identifies this device. All fields are read-only.																							
Field								Description																							
PRODUCTID								The Product ID value represents the identification number for the 7902 device and will not change. The Product ID of this device is 0x0004.																							
REVISIONID								The Revision ID value represents the actual revision number for the 7902. The Revision ID of this device is 0x0000. For this particular device, 0x0000 represents the first version of this device.																							

9 Public Key Engine & Random Number Generator

9.1 Overview

Public key operations require CPU-intensive math calculations. The Public Key engine reduces the load on the CPU by performing the arithmetic in hardware.

Public key operations also require a cryptographic-quality random number for a high level of security. The random number generator provides an unpredictable number, and with minimal CPU processing a cryptographic-quality random number can be generated.

9.2 Random Number Generator

The Random Number Generator is based on internal free-running oscillators whose frequencies drift relative to each other and to the internal clock of the 7902. The phase relation of these signals is unpredictable, and this is used to create entropy to generate a random bit stream.

The output of the random number generator only serves as the raw input to a cryptographic process that generates cryptographically secure random numbers. The process required is beyond the scope of this datasheet. Source code of the software required to perform this process is available from Hi/fn.

The Random Number Generator must be enabled after each reset by setting the `RNG_ENABLE` bit in the Public Configuration register. Once enabled, the generator begins producing a random bit stream. It is important to note that the data obtained from the first read of the RNG data register after a reset should be discarded.

9.3 Public Key Data Movement

The Public Key engine assists in performing CPU-intensive modular arithmetic, including modular exponentiation. The user first sets configuration registers, and then issues a starting command and opcode. The CPU determines whether the unit is idle or busy by checking the Public Status register.

Public Key data movement is outlined below:

Set up Source data

1. Place source operands in memory.
2. Set the public key base address pointer.
3. Set the public key operand length register.

Issue a Command

4. Set the Public Operation Register. Setting the Public Operation Register starts the Public Key engine.

Read Out Result and Destination Data

5. When the Public Key engine has completed, the `PUBLIC DONE` bit in the Public Status Register is set and an interrupt is generated if the interrupt enable bit is set.
6. Read the result data from the internal memory.
7. Repeat (go to step 1).

9.4 Operations

The following table illustrates the operations supported by the Public Key engine. All operands and results must be of length `MODLEN` unless otherwise specified.

Operation	Description
Add	$A + B$
Add w / Carry	$A + B + \text{Carry}$
Subtract	$A - B$
Subtract w / Carry	$A - B - \text{Carry}$
Mod Add	$(A + B) \bmod M$
Mod Sub	$(A - B) \bmod M$
Inc A	$A + 1$
Dec A	$A - 1$
Mult	$A * B$
Mod Mult	$(A * B) \bmod M$
Mod Red	$A \bmod M$
Mod Exp	$(A ^ B) \bmod M$

Figure 23. Public Key Operations

9.5 Operands

All operations use one, two, or three operands. These are labeled A, B, and M for operand A, operand B, and the modulus. Each of these operands are actually an array of elements in internal memory. The host sets the base address of each array by configuring the `A`, `B`, and `M` fields in the Public Operation Register. The values written to the `A`, `B`, and `M` fields are offsets into internal memory, relative to the `PUBLIC BASE ADDRESS` field set in the Public Base Address Register.

9.5.1 Public Key Operand Storage

Operands are stored in the internal memory. Each operand is broken up into a series of 32-bit values. The least significant 32-bit value is stored in the lowest address location, and the most significant 32-bit value is stored in the highest address location.

Because data is always transferred as 32-bit values, the byte-endianness of system memory does not affect 7902 public key operands. 32-bit values are always presented on the CPU bus in proper byte order.

For example, if the Public Base Address is set to 0x3000, and the A Offset is set to 0x0000, operand A with a value of 0x0001020304050607 would be stored in internal memory as shown in Figure 24.

Internal Address	32-Bit Value
0x3000	0x04050607
0x3004	0x00010203

Figure 24. Public Key Operand Storage in Internal Memory

Field	Description
PUBLIC BASE ADDRESS	Base address of Public Key engine. Points anywhere in the internal memory. It must be 512-byte aligned. All other public key offsets are relative to this base address.
A OFFSET	Offset of operand A from PUBLIC BASE ADDRESS. (Units of 64 bytes).
B OFFSET	Offset of operand B from PUBLIC BASE ADDRESS. (Units of 64 bytes).
M OFFSET	Offset of operand M from PUBLIC BASE ADDRESS. (Units of 64 bytes).
MODLEN	Length of modulus (MODLEN=2 to 64). Actual modulus length in bits is MODLEN*32. Actual modulus length must be between 64 and 2,048 bits inclusive. The most significant bit of the <i>value</i> of the modulus (not the length) must be a 1.
REDLEN	Length of reducend. Actual reducend length in bits is (REDLEN+2)*MODLEN*32 (REDLEN = 0 to 13). Actual reducend length must be less than or equal to 2,048 bits or 15*MODLEN*32, (REDLEN=13), whichever is smaller. The minimum reducend length is 2*MODLEN*32 (REDLEN=0). There are no restrictions regarding the most significant bit of the <i>value</i> (not the length) of the reducend.
EXPLEN	Length of exponent. Actual exponent length in bits is EXPLEN+1. The minimum actual exponent length must be 2 (EXPLEN=1). The maximum actual exponent length must be less than or equal to 2,048 bits (EXPLEN=2,047) or MODLEN*32 (EXPLEN=MODLEN*32-1), whichever is smaller. The most-significant bit of the <i>value</i> of the exponent (not the length) must be a 1.

Figure 25. Operand Pointers and Offsets

Some example values for pointers and offsets are shown in Figure 26.

Field	Value of Field	Actual Bas Address in Internal Memory
PUBLIC BASE ADDRESS	0x3000	0x3000
A OFFSET	0x0	0x3000
B OFFSET	0x4	0x3100
M OFFSET	0x8	0x3200

Figure 26. Field Values and Memory Locations

Each operand is an array of elements in internal memory. Each element has a size of MODLEN, as configured by the host in the Public Configuration Register. The number of elements used in each array varies by operation. This is shown in Figure 27.

Notes:

1. The results of each operation are stored after operand B, in B(1) and in the case of the multiply operation, B(2).
2. Mod Red operations destroy the contents of operand A.
3. Operands, modulus, result RAM, or scratch pad should not be modified while the Public Key engine is running.

Operation	Operand A	Operand B	Operand M ²	Result	Scratch
Add	A(0)	B(0)		B(1)	
Add w / Carry	A(0)	B(0)		B(1)	
Subtract	A(0)	B(0)		B(1)	
Subtract w / Carry	A(0)	B(0)		B(1)	
Mod Add	A(0)	B(0)	M(0)	B(1)	
Mod Sub	A(0)	B(0)	M(0)	B(1)	
Inc A	A(0)			B(1)	
Dec A	A(0)			B(1)	
Mult	A(0)	B(0)		B(2-1)	
Mod Mult	A(0)	B(0)	M(1-0)	B(1)	B(2), M(4-2)
Mod Red	A(14-0) ^{3, 4}		M(1-0)	B(1)	B(2), M(4-2)
Mod Exp	A(0)	B(0)	M(1-0)	B(1)	B(2), M(4-2) ¹

Note 1: Host must clear M(2) prior to starting Mod Exp operation.

Note 2: M(1) is used to store the reciprocal of M(0). The MSB of M(0) must be a 1.

Note 3: Operand A is of size, in bytes, [RedLen+2]*[ModLen]*4, up to a maximum of 2048 bits. The value of RedLen is 0 to 13.

Note 4: Mod Red operations destroy the contents of operand A.

General Notes:

1. Blank entries are free for system use.
2. Byte address of references relative to internal memory: A(n)=[A Offset]*64+n*[ModLen]*4, B(n)=[B Offset]*64+n*[ModLen]*4, M(n)=[M Offset]*64+n*[ModLen]*4.

Figure 27. Operand Use for Each Operation

9.5.2 Public Key Memory Usage

Individual memory chunk addresses can be calculated using the following equations:

$$\begin{aligned}A(n) &= BA + A + n * \text{chunksize} \\B(n) &= BA + B + n * \text{chunksize} \\M(n) &= BA + M + n * \text{chunksize}\end{aligned}$$

Note: All quantities are in bytes.

Where:

chunksize = modulus size for current operation, 4-byte granular
BA = Base Address (512-byte granular)
A = A Offset (64-byte granular)
B = B Offset (64-byte granular)
M = M Offset (64-byte granular)
A(n) = A operand memory chunk #n address
B(n) = B operand memory chunk #n address
M(n) = M operand memory chunk #n address

Note: All quantities are in bytes.

The maximum amount of memory used can be calculated using the following equation:

$$\text{mem_used} = 256 + \text{max_chunksize} * 8$$

Where:

mem_used = size of memory space reserved for public key operations
max_chunksize = maximum supported modulus size, rounded up to a multiple of 64-bytes

General Purpose Example

This example sets the pointers to use the upper address space of the internal memory. In this example, it is assumed that the maximum key length to be supported is 1024 bits, so the maximum operand length is 1024 bits. A max_chunksize of 1024 bits is used to minimize the amount of internal memory allocated for public key operations. Max_chunksize can be larger than the maximum key length to be supported. Since internal memory is limited and a shared resource with the packet engine, the most efficient use is realized when max_chunksize equals the maximum key length.

The equations to be used to determine the addresses are:

$$\begin{aligned}\text{start_addr} &= \text{max_addr} - \text{mem_used} \\BA &= \text{start_addr} \text{ (rounded down to the nearest 512-byte)} \\A &= \text{start_addr} - BA \\B &= A + 256 \\M &= A + 256 + 3 * \text{max_chunksize}\end{aligned}$$

Where:

$\text{max_addr} = 16384$

start_addr = lowest address that the public key engine will use

mem_used = size of memory space reserved for Public Key engine use

BA = Base Address (512-byte granular)

A = A Offset (64-byte granular)

B = B Offset (64-byte granular)

M = M Offset (64-byte granular)

With substitution we get:

$\text{mem_used} = 256 + \text{max_chunksize} * 8$ (**Note:** all units in bytes)

$\text{mem_used} = 256 + (1024 \text{ bits} / 8 \text{ bits/byte}) * 8$

$\text{mem_used} = 1280$ or 0x500 bytes

$\text{start_addr} = \text{max_addr} - \text{mem_used}$

$\text{start_addr} = 0x4000 - 0x500$

$\text{start_addr} = 0x3B00$

BA = start_addr (rounded down to the nearest 512-byte)

BA = 0x3A00

A = $\text{start_addr} - \text{BA}$

A = 0x3B00 – 0x3A00

A = 256 or 0x100

B = A + 256

B = 256 + 256

B = 512 or 0x200

M = A + 256 + 3*max_chunksize

M = 256 + 256 + 3*(1024 bits / 8 bits/byte)

M = 896 or 0x380

Figure 28 shows the memory map of the general-purpose example above.

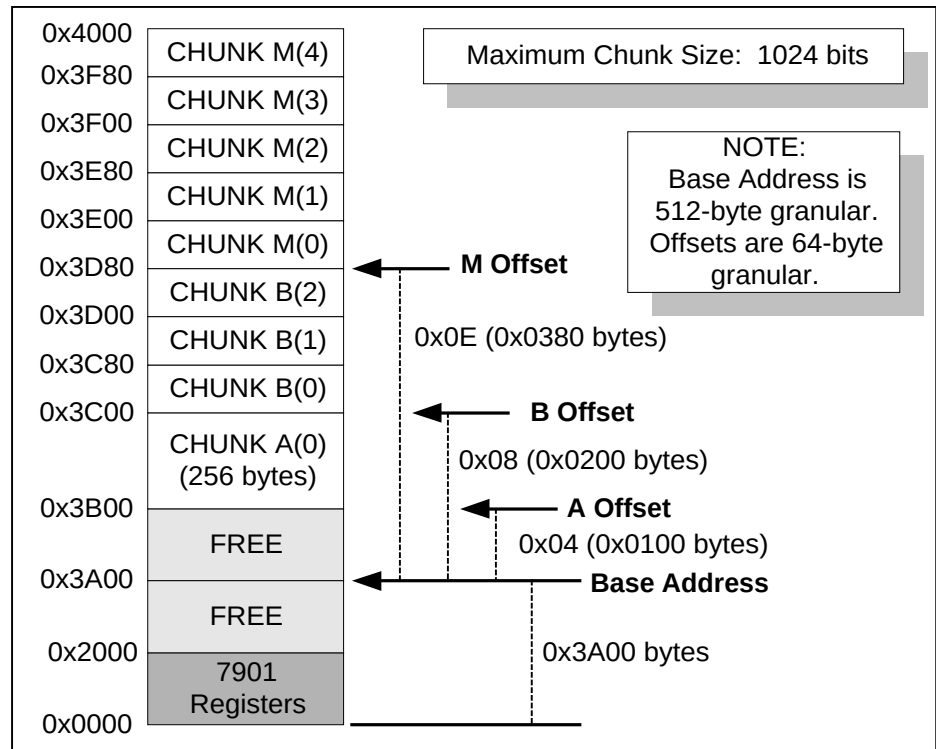


Figure 28. Public Key Memory Use General Purpose Example

9.6 Reciprocal Calculations

The reciprocal is the key to performing modular reduction with a hardware multiplier (otherwise a divide function is required). Normally a reciprocal is thought of as the multiplicative inverse of a value:

$$\text{if } a \text{ is the reciprocal of } x, \text{ then } a * x = 1$$

where a is any rational number.

In the integer realm of public key cryptography, a reciprocal has a slightly different definition—both the modulus and the reciprocal must be integers. This is done by choosing an appropriate value for the '1' normally associated with multiplicative inverses. For example, if 16-bit values are being dealt with:

$$\begin{aligned} x &= 0xABCD \\ a &= ? \end{aligned}$$

and a 16-bit reciprocal is wanted, the '1' is represented by a one followed by 32 zeros. The reciprocal is computed by:

$$a = 0x100000000/x = 0x100000000/0xABCD = 0x17D77$$

The result of integer division is just the integer quotient—any remainder is ignored. This reciprocal is an approximation, since $a * x$ is somewhat less than $0x100000000$.

Note that the 16-bit reciprocal is really 17 bits long. The choice of a '1' value determined this, since 33 bits divided by 16 bits gives at least a 17-bit result. If the '1' is defined to be 32 bits instead of 33 and it is assumed that the top bit of the modulus is always set, then the reciprocal is fixed at 16 bits:

$$a = 0x80000000/0xABCD = 0xBEBB$$

For other sized reciprocals, a '1' is defined that has only the top bit set of a value of bit length:

$$\text{modulus_bits} + \text{reciprocal_bits}$$

Note the reciprocal, while being well behaved, is smaller than the true reciprocal by a factor of two. The Public Key engine automatically restores this factor.

To summarize the steps involved in generating a reciprocal for use in the 7902:

1. N = a single 1, followed by $((\text{Modulus Length} * 2) - 1)$ zeros. For example, for a 512 bit modulus, $N = 1$ followed by 1023 zeros (binary number representation).
2. $\text{RECIPROCAL} = N$ divided by MODULUS .

The reciprocal can be precomputed once for a given modulus, then stored with the modulus. The precomputation can be done on a host, which has a divide function.

Note: The modulus reciprocal is necessary for Mod Mult, Mod Red, and Mod Exp operations. If an inappropriate value for the modulus reciprocal is provided, the Public Key engine may never complete the operation, the PUBLIC DONE bit will never be set and the public done interrupt, if enabled, will not be generated. Starting a NOP operation will stop the Public Key engine. It should also be noted that starting any operation stops the Public Key engine and restarts it with the new operation.

9.7 Public Base Address Register

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 reserved Public Base Address																															
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0100-0x0103																This register is used to set the Public Base Address. All fields are R/W.															
Reset Value																xxxx_xxxx_xxxx_xxxx_xx00_0000_0000_0000															
Field																Description															
PUBLIC BASE ADDRESS																Base address of Public Key engine. Points anywhere in the internal memory. It must be 512-byte aligned (bits 8-0 must be 0). Valid values are from 0x2000 to 0x3E00. All other public key offsets are relative to this base address.															

9.8 Public Operand Length Register

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 reserved RedLen ExpLen ModLen																															
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0104-0x0107																This register is used to configure the size of the operands used for public key operations. All fields are R/W.															
Reset Value																xxxx_xxxx_xx00_0000_0000_0000_0000															
Field																Description															
REDLEN																Length of reducend. Actual reducend length in bits is $(REDLEN+2)*MODLEN*32$ ($REDLEN = 0$ to 13). Actual reducend length must be less than or equal to 2,048 bits or $15*MODLEN*32$, ($REDLEN$ of 13), whichever is smaller. The minimum reducend length is $2*MODLEN*32$ ($REDLEN$ of 0).															
EXPLEN																Length of exponent. Actual exponent length in bits is $EXPLEN+1$. Actual exponent length must be between 2 ($EXPLEN$ of 1) and the actual modulus length ($MODLEN*32$), inclusive. The most-significant bit of the value of the exponent (not the length) must be a 1.															
MODLEN																Length of modulus. Actual modulus length in bits is $MODLEN*32$. Actual modulus length must be between 64 and 2,048 bits inclusive ($MODLEN$ of 2 to 64). The most-significant bit of the value of the modulus (not the length) must be a 1.															

9.9 Public Operation Register

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																																															
reserved																Opcode								M Offset								B Offset								A Offset							
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																																															
Addr: 0x0108-0x010B																This register is used to identify the location of the public key operands and to start an operation defined by the Opcode. All fields are R/W.																															
Reset Value																xxxx_xxxx_xx00_0000_0000_0000_0000																															
Field																Description																															
OPCODE																Opcode		Operation		Opcode		Operation																									
																0000		NOP		1000		Dec A																									
																0001		Add		1001		Mult																									
																0010		Add w / C		1010		Mod Mult																									
																0011		Sub		1011		Mod Red																									
																0100		Sub w / C		1100		Mod Exp																									
																0101		Mod Add		1101		reserved																									
																0110		Mod Sub		1110		reserved																									
M OFFSET																0111		Inc A		1111		reserved																									
																Sets the offset from the public base address of the Modulus operand. Actual Modulus offset is [M OFFSET]*64.																															
B OFFSET																Sets the offset from the public base address of the B operand. Actual B offset is [B OFFSET]*64.																															
A OFFSET																Sets the offset from the public base address of the A operand. Actual A offset is [A OFFSET]*64.																															

Note: If the Public Key engine is performing an operation and another operation is started, the current operation is stopped and the engine is restarted with the new operation.

9.10 Public Status Register

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
reserved																												Carry	Public Done		
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x010C-0x010F										This register reports the status of the Public Key engine. Some fields are constantly updated. Other fields, once set to 1, remain set to 1 until cleared by the host. See the individual field descriptions for details.																					
Reset Value										xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xx00																					
Field										Description																					
CARRY										Reports the carry flag for the previous public key operation (ADD, SUB, ADD w/C, SUB w/C, INC, or DEC); otherwise, any other operation results in an invalid state. For ADD and INC operations, the carry flag is set if there is a carry out of the MSB; otherwise, it is cleared. For SUB and DEC operations, the carry flag is set if the result is equal to or greater than zero. If the result is less than zero, the carry flag is cleared.																					
PUBLIC DONE										Reports that the Public Key engine is idle. Do not access public key operands in internal memory unless the Public Key engine is idle. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position or by writing an opcode to the Public Operation Register.																					

9.11 Public Interrupt Enable Register

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																																
reserved																															Public Done	
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																																
Addr: 0x0110-0x0113																This register enables the Public Key Interrupt signal based on the status reported by the Public Status register. All fields are R/W, returning the last value written.																
Reset Value																xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxx0																
Field																Description																
PUBLIC DONE																If set to 1, an interrupt is generated when the PUBLIC DONE bit in the Public Status register is set to 1.																

9.12 RNG Configuration Register

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 reserved RNG Enable																															
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0114-0x0117																This register configures the Random Number Generator. All fields are R/W, returning the last value written.															
Reset Value																xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxx0															
Field																Description															
RNG ENABLE																If set to 1, the random number generator is enabled. Setting this bit to 0 reduces the power consumption of the device.															

9.13 RNG Data Register

313029282726252423222120191817161514131211109876543210 Data																															
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0118-0x011B																Use this register to access random data from the Random Number Generator.															
Field																Description															
DATA																Random data. The host processes this data to create cryptographic-quality random numbers. See section 9.2 for details.															

10 Packet Engine

10.1 Overview

The Packet Engine is a compression/authentication/encryption unit with its own Source and Destination DMA units to access internal memory. Most of the operation of the Packet Engine is controlled on a per-command basis through the command stream that is read by the Source DMA unit.

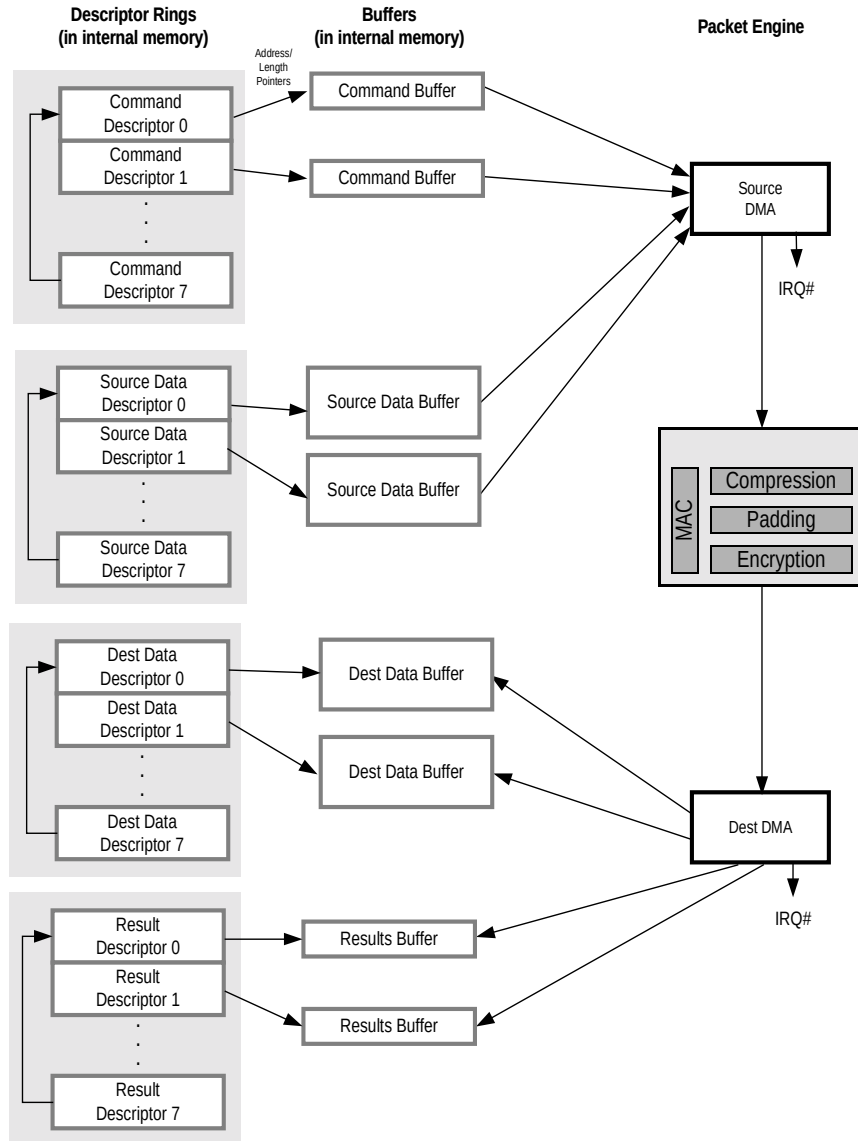


Figure 29. Packet Engine Block Diagram

Command and data flow is controlled by four data structures called *descriptors*. Descriptors are arranged into descriptor rings. There are four descriptor rings: Command, Source Data, Destination Data, and Results. Each has its own data format. The rings in turn point to memory buffers containing the actual commands, source data, destination data, and results.

10.2 Data Movement

Most of the Packet Engine's operation is controlled by the descriptors.

The host creates descriptors and data buffers. Once a descriptor is written, the Host must update the Host Index of the descriptor ring. This is used by the internal DMA to determine which descriptors are valid.

Command setup and execution works as follows:

1. The host writes source commands to command buffers and source data to data buffers. Fragmentation is allowed.
2. The host initializes command and source data descriptors that point to the command and source data buffers. Fragmentation is handled using multiple command and source descriptors (one per fragment). The *LAST* bit in each descriptor indicates the fragmentation status. If the *LAST* bit is 0, there is an additional fragment following the current descriptor. If the *LAST* bit is 1, this is the final fragment. The host updates the Host Index of each descriptor ring as the last step of initialization.
3. The host creates "empty" dest data and results descriptors. An empty descriptor has its pointers initialized to point to available buffers and its *BUFFER LENGTH* field set to the length of each buffer. The host updates the Host Index of each descriptor ring as the last step in each descriptor's initialization.

Once an operation has begun, the Packet Engine uses descriptors automatically, while continuously updating the Device Index fields in the Packet Status Register.

Interrupts are enabled on a variety of conditions using the Packet Interrupt Enable Register.

On the input side, the Packet Engine consumes a source data descriptor for every command, even if the command has no associated source data. On the output side, Dest Data and Result descriptors are used only as needed.

Important Note: During the first command after a reset, if a packet engine command generates data prior to the availability of a valid destination descriptor, an extra byte of data may be included in the first destination buffer. After data is generated for the first command after a reset, the extra byte is not included in the destination buffer. To avoid this condition, validate destination descriptors prior to the first command descriptor after a reset.

10.3 Descriptors

10.3.1 Command Descriptor

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
reserved		Last	reserved			Mask Done	reserved									Buffer Length															
Command Ptr																															

Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.

	Command Descriptor points to a command block or fragment.
Field	Description
LAST	Set this bit to 1 to indicate this descriptor points to the last (or only) fragment of a block.
MASK DONE	Set this bit to 1 to inhibit the setting of the COMMAND DONE bit in the Packet Status Register.
BUFFER LENGTH	Length of the block pointed to by the command pointer, in bytes. BUFFER LENGTH must not be zero. Also, the value must be a multiple of four. Bits 1-0 must be 0.
COMMAND PTR	Pointer to block. Pointer is relative to internal memory. Pointer must be 32-bit aligned. Bits 1-0 must be 0.

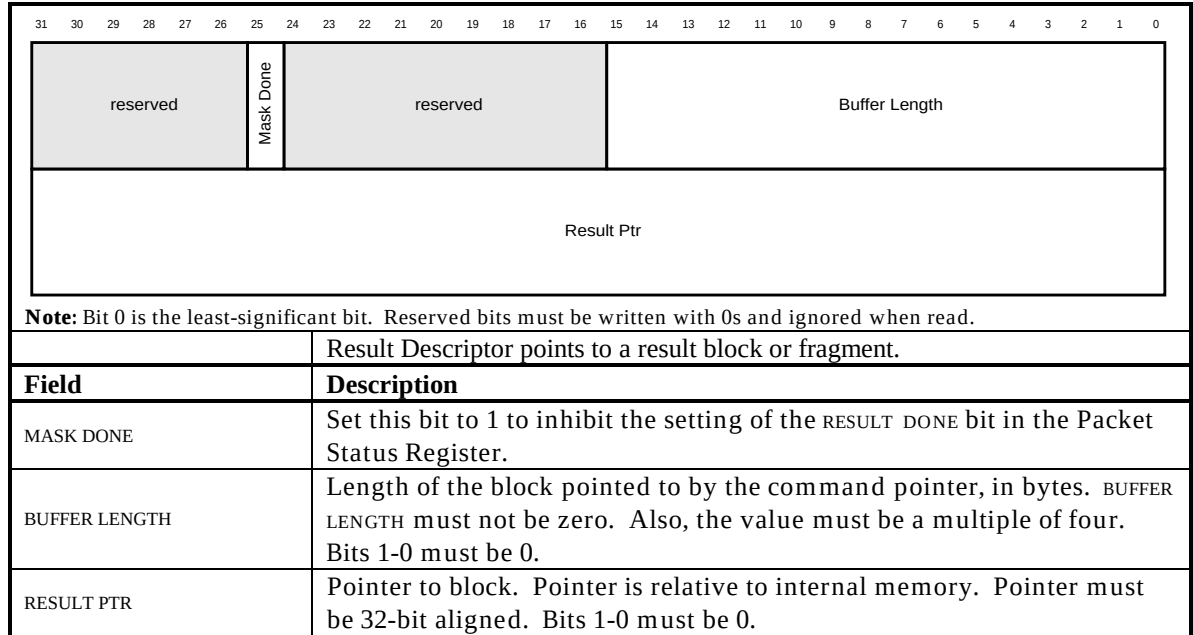
10.3.2 Source Descriptor

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
reserved		Last	reserved			Mask Done	reserved									Buffer Length															
Source Ptr																															

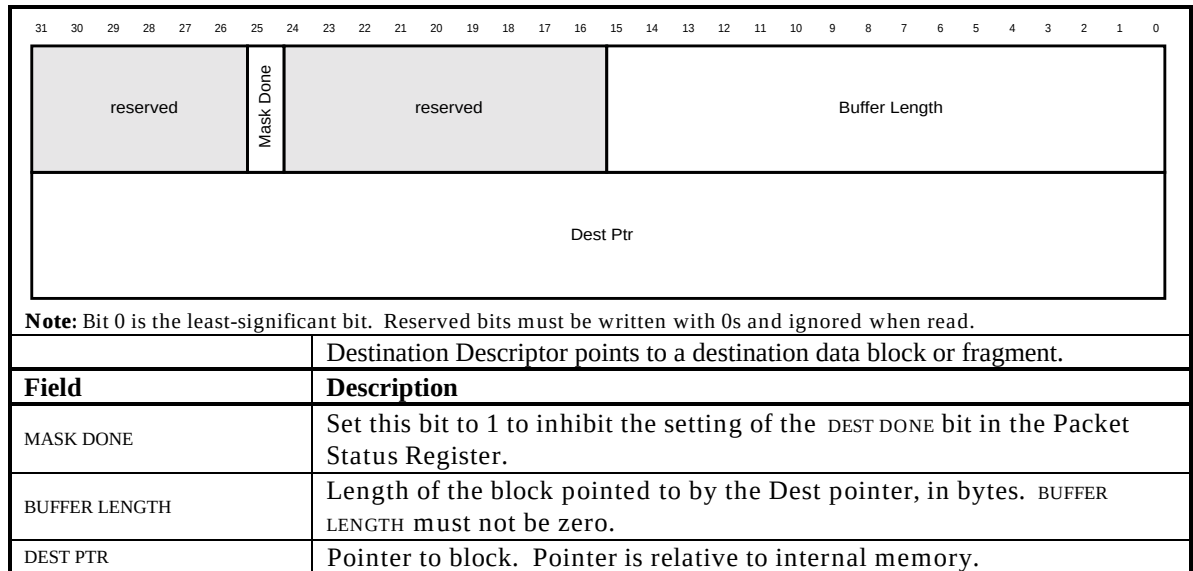
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.

	Source Descriptor points to a source data block or fragment.
Field	Description
LAST	Set this bit to 1 to indicate this descriptor points to the last (or only) fragment of a block.
MASK DONE	Set this bit to 1 to inhibit the setting of the SOURCE DONE bit in the Packet Status Register.
BUFFER LENGTH	Length of the block pointed to by the source pointer, in bytes. BUFFER LENGTH must not be zero.
SOURCE PTR	Pointer to block. Pointer is relative to internal memory.

10.3.3 Result Descriptor



10.3.4 Destination Descriptor



10.4 Command Structure

The Encode command performs encryption, compression, or both. The decode command performs decryption, decompression, or both. In addition, authentication (MAC) and padding can be performed.

Each of the four steps (encryption/decryption, compression/decompression, padding, and MAC) are specified in separate data structures, which specify the command completely. These structures follow a *base command structure* in the following order: compression, pad, MAC, encrypt. If a stage is not used, its command structure must be omitted from the command. In addition, an Encryption Context Structure may follow the rest if the MAC or encryption command structures called for new keys or a new IV.

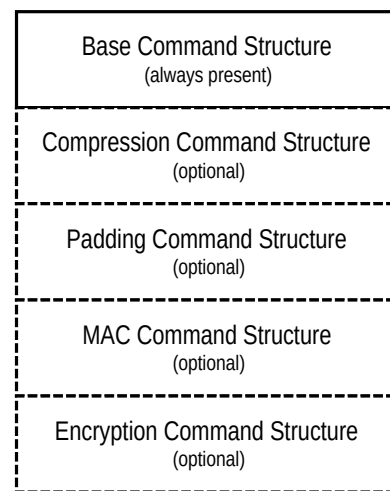
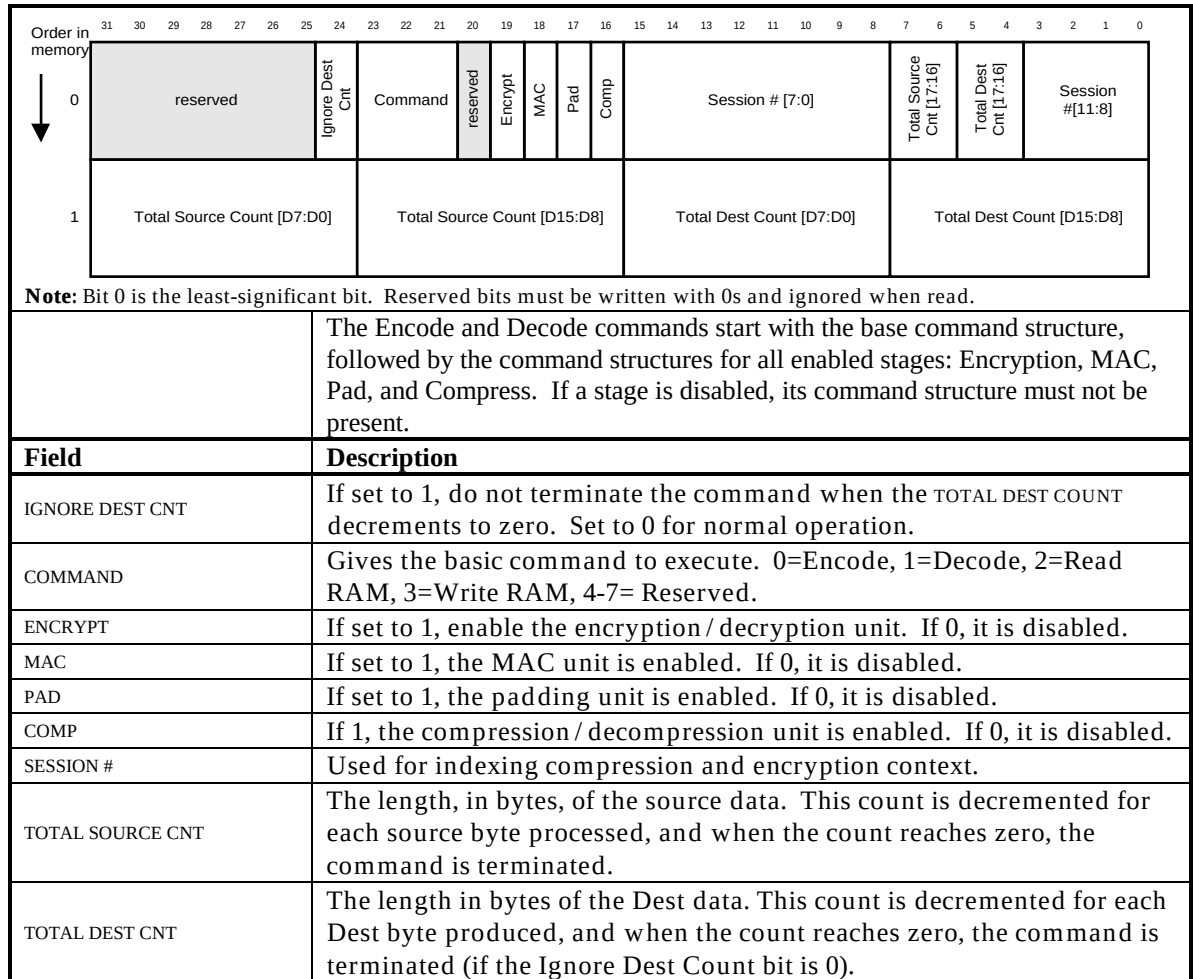


Figure 30. Command Structures

The data used for the command is specified separately in the Source Data descriptor. The compression history, session keys, and authentication data are stored in the external RAM, which is indexed by the *session number* parameter in the base command structure.

10.4.1 Base Command Structure



10.4.2 Compress Command Structure

Order in memory ↓	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
	reserved				Clear Hist	Update Hist	Strip 0 / Restart	reserved	MPPC	Comp Source Cnt [D17:16]	reserved				Must be '1'	Comp Header Cnt [D7:0]				Comp Header Cnt [D15:8]												
	Comp Source Count [D7:0]								Comp Source Cnt [D15:8]								reserved															

Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.

	The Compress command structure tells the compression/decompression unit which operation to perform, and which portion of the data stream to process. When enabled, the unit must process at least one byte. In decompression, compression history may become invalid if only a fragment of the original data is decompressed. In MPPC mode, the decompressor must decompress the same number of bytes that were compressed in the original operation. In LZS mode, decompression must end on an End Marker. When an End Marker is encountered during LZS decompression, the remainder of the source bytes (up to the end of the Comp Source Counter) is discarded.
Field	Description
CLEAR HIST	If set to 1, clears the session’s compression history before compression or decompression begins. If 0, uses the history stored as part of the session context. This bit has to be set on all stateless compression and decompression commands. If it is not set, there is the possibility of a corrupt or malicious packet exposing data from a prior history. This bit must be set on the first packet in stateful compression and decompression sessions.
UPDATE HIST	Only valid during decompression. If 1, do not decompress the data, but pass it through the decompressor unaltered. However, the compression history is updated as if the input data had been the output of the decompressor.
STRIP 0 / RESTART	See the text below this table.
MPPC	Selects the compression algorithm. If 1, MPPC is used. If 0, LZS is used.
COMP HEADER CNT	Selects the number of bytes of header data to pass through the unit unmodified.
COMP SOURCE CNT	After passing through the bytes specified in the COMP HEADER CNT, the compression / decompression unit processes the number of bytes given in the COMP SOURCE CNT. After compressing or decompressing the specified number of source bytes, the unit returns to pass-through mode. The TOTAL SOURCE CNT (in the Base Command structure) also ends processing. When either counter has expired, the compression / decompression unit flushes out its internal data and (if compressing in LZS mode) appends an End Marker to the compressed data stream.

Strip 0/Restart

This bit has two functions, depending on the compression algorithm selected.

LZS mode. In LZS mode, setting this bit enables the "Strip 0" mode of the LZS compression format.

Enabling the Strip 0 feature generally reduces the size of the compressed data stream by one byte. If this bit is set to 1 on a Compress operation, the last byte of compressed data is eliminated if the value of this last byte is zero. Based on the LZS format, this last byte is always part of the End Marker and is zero approximately 88% of the time. If the last byte is eliminated, the Dest Counter does not count it. If padding is enabled, then Strip 0 should not be used.

On a Decompress operation, a byte with a value of zero is inserted in the source compressed data stream just before the check field, or at the end of the compressed data stream if there is no check field. The Source Counter does not count the inserted byte.

If the Strip 0 mode is enabled during a Decompress operation, the 7902 must know the exact number of source bytes so that it can insert the zero value byte at the correct location in the data stream. The pad length must contain the exact number of padding bytes.

Note: Many data communication standards define the Strip 0 feature as an option. However, this mode is incompatible with the ANSI X3.241-1994 compression format standard.

The Strip 0 bit cannot be set if decompressing with the padding processing unit disabled.

MPPC Mode. In MPPC mode (the MPPC bit is set to one), the STRIP 0/RESTART bit is known as the RESTART bit, and is used to implement the "restart" function of the MPPC protocol.

In MPPC mode, this bit, if set to one, tells the decompression engine to move the data to the front of the compression history, as specified in the MPPC protocol.

During a compression operation, the processing unit automatically moves the data to the beginning of the history buffer as required, so the RESTART bit must be set to 0.

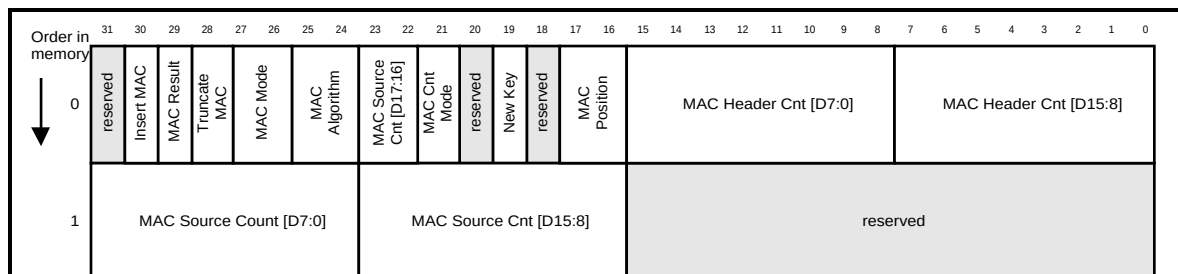
10.4.3 Pad Command Structure

Order in memory ↓ 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
	reserved						Pad Algorithm	Pad Source Cnt [D17:D16]		Pad Count Mode	reserved	Pad Length			Pad Source Cnt [D7:0]					Pad Source Cnt [D15:8]												

Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.

	During encoding, the padding unit rounds out data fields to a modulo eight length, which is required by many protocols.
Field	Description
PAD ALGORITHM	During decoding, this field is ignored. For encoding, this field specifies the padding mode, as defined below: Mode 0. The padding unit inserts 1-8 bytes to make the total length a multiple of 8. The value of the bytes will be equal to the number of bytes added minus one. For example, if six bytes were added, each would have a value of five. Mode 1. The padding unit inserts 1-8 bytes to make the total length a multiple of 8. The first padding byte is set to one, and the value of each subsequent byte is incremented. Mode 2. The padding unit inserts 0-7 bytes to make the total length a multiple of 8. If bytes are inserted, the first will have a value of one, and subsequent bytes are incremented. Mode 3. The padding unit inserts two fields. The first is 0-7 bytes, as in Mode 2. The second is a single byte containing the size, in bytes, of the first field.
PAD COUNT MODE	This bit is valid only for an encode operation. For a decode operation, this bit should be set to 0. For encode operations, if set to 1, the Pad Source Counter starts decrementing after the last byte processed by the previous processing unit, as determined by the source counter of the previous unit. If 0, the Pad Source Counter starts decrementing after the last header byte, as set by the PAD HEADER CNT field, has been passed through. The ENCRYPT HEADER CNT field in the Encryption command structure initializes the Pad Header Counter.
PAD LENGTH	Adjusts the number of padding bytes by the specified number of bytes. When encoding, padding is increased. When decoding, padding is decreased.
PAD SOURCE CNT	The number of source bytes to process. The padding unit passes through header bytes in a way determined by the PAD COUNT MODE bit, then processes bytes equal to PAD SOURCE CNT. Once PAD SOURCE CNT bytes have been processed, or the last byte as defined by the TOTAL SOURCE CNT has been processed, the padding processing unit completes, and data is once more passed through unmodified.

10.4.4 MAC Command Structure



Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.

	The MAC unit calculates authentication codes on the data stream. During an encode operation, the calculated MAC can be emitted as part of the output. During a decode operation, the calculated MAC is compared to the MAC in the source data stream.
Field	Description
INSERT MAC	For encode operations, this bit, if set, indicates that a MAC is to be inserted into the data stream. If clear, no MAC is inserted. On decode, this bit, if set, indicates that the MAC is to be stripped from the data stream and compared to the calculated MAC. The result is reflected in the MAC MISCOMPARE bit in the MAC Result structure.
MAC RESULT	If this bit is set to 1, the calculated MAC is written into the MAC Result structure. If 0, the calculated MAC is not written into the Result structure.
TRUNCATE MAC	If this bit is set to 1, the MAC is truncated to 12 bytes. The most-significant bytes are stripped. If 0, the MAC is full-length, which is 20 bytes for SHA and 16 bytes for MD5.
MAC MODE	Determines the MAC operation, as follows: 00: HMAC 01: SSL MAC. (Valid only if the MD5 MAC algorithm is selected. The SHA algorithm may not be used in the SSL MAC mode. 10: Hash only 11: Reserved.
MAC ALGORITHM	Determines the MAC hashing algorithm, as follows: 00: SHA 01: MD5 others: Reserved
MAC CNT MODE	If set to 1, the MAC Source Counter begins after the last byte processed by the previous processing unit, as determined by the source counter of that unit. If 0, the MAC Source Counter begins after the last header byte has been passed through, as determined by the MAC HEADER CNT field.
NEW KEY	If set to 1, a new key is to be supplied in the Encryption Context structure.
MAC POSITION	Places the MAC in relation to the other processing units, as follows: 00: Between compression and padding 01: Between padding and encryption 10: After encryption on encode, before decryption on decode 11: Reserved
MAC HEADER CNT	The number of Source bytes to skip before MAC processing begins. This field is ignored if the MAC Count Mode bit is set to 1.
MAC SOURCE CNT	This gives the number of source bytes to process before resuming pass-through operation.

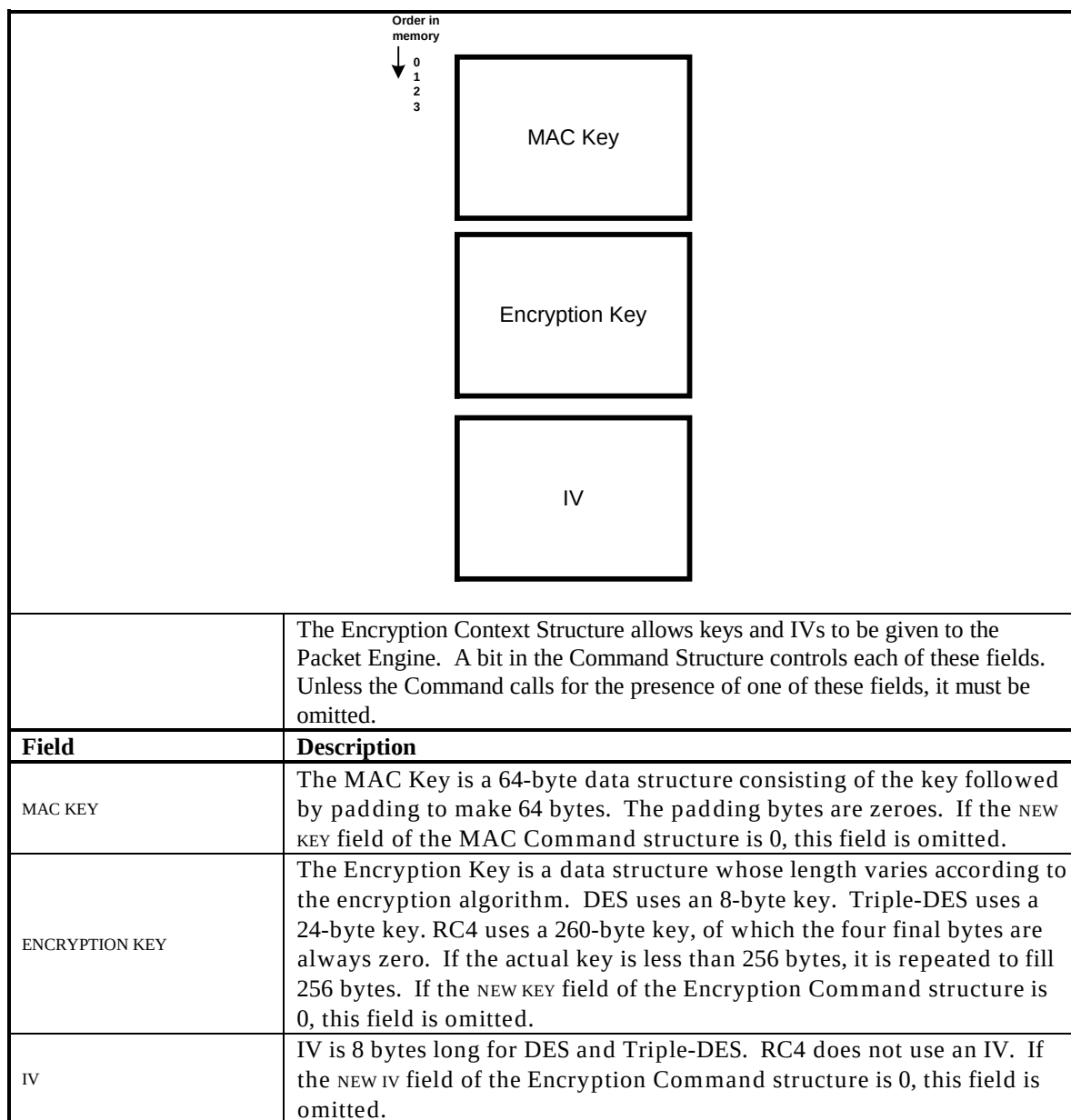
10.4.5 Encryption Command Structure

Order in memory ↓	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
	reserved	Clear Encrypt Cntxt	reserved	Encrypt Mode	reserved	Encrypt Algorithm	Encrypt Source Cnt [D17:16]	Encrypt Cnt Mode	New IV	New Key	reserved	Encrypt Header Cnt [D7:0]						Encrypt Header Cnt [D15:8]														
	Encrypt Source Count [D7:0]						Encrypt Source Cnt [D15:8]						reserved																			

Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.

Field	Description
CLEAR ENCRYPT CONTEXT	External RAM not used: Since encryption context can only be saved to external RAM, this bit should always be set to 1. External RAM used: If set to 1, the encryption context is not saved to external memory. This may improve performance if it is known that the context will not be used in the future. If this bit is set to 0, context is updated normally.
ENCRYPT MODE	This field is valid only for DES and 3DES. It must be set to 0b00 otherwise. The encoding is as follows: 00: ECB 01: CBC 10: CFB-64 11: OFB
ENCRYPT ALGORITHM	Determines the encryption algorithm, as follows: 00: DES 01: 3DES 10: RC4 11: reserved
ENCRYPT CNT MODE	If set to 1, the Encrypt Source Counter begins after the last byte processed by the previous processing unit, as determined by the source counter of that unit. If 0, the Encrypt Source Counter begins after the last header byte has been passed through, as determined by the ENCRYPT HEADER CNT field.
NEW IV	If set to 1, a new DES /3DES encryption initialization vector (IV) is supplied in the Encryption Context structure. Only valid for DES and 3DES, and then only when the encryption mode is CBC, CFB, or OFB. In all other cases, this field must be 0.
NEW KEY	If set to 1, a new key is supplied in the Encryption Context structure.
ENCRYPT HEADER CNT	The number of Source bytes to skip before encryption processing begins. This field is ignored if the ENCRYPT CNT MODE bit is set to 1.
ENCRYPT SOURCE CNT	This gives the number of source bytes to process before resuming pass-through operation.

10.4.6 Encryption Context Structure



If the encryption command calls for a new key, MAC key, or IV, this must be provided to the Packet Engine. The data takes the form of an *Encryption Context Structure*, with the format described below.

The Encryption Context structure can be considered another data structure, one that has a form similar to the usual data structures. It can be physically added in front of the data in the same data buffer or (more typically), it is placed in its own data buffer. See Figure 31 for a typical use of descriptors for commands that require encryption context.

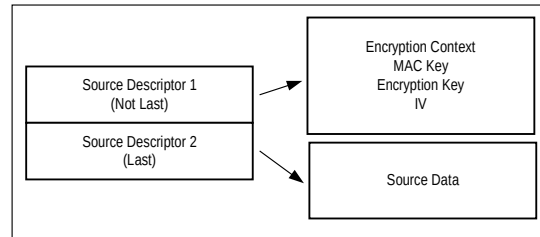
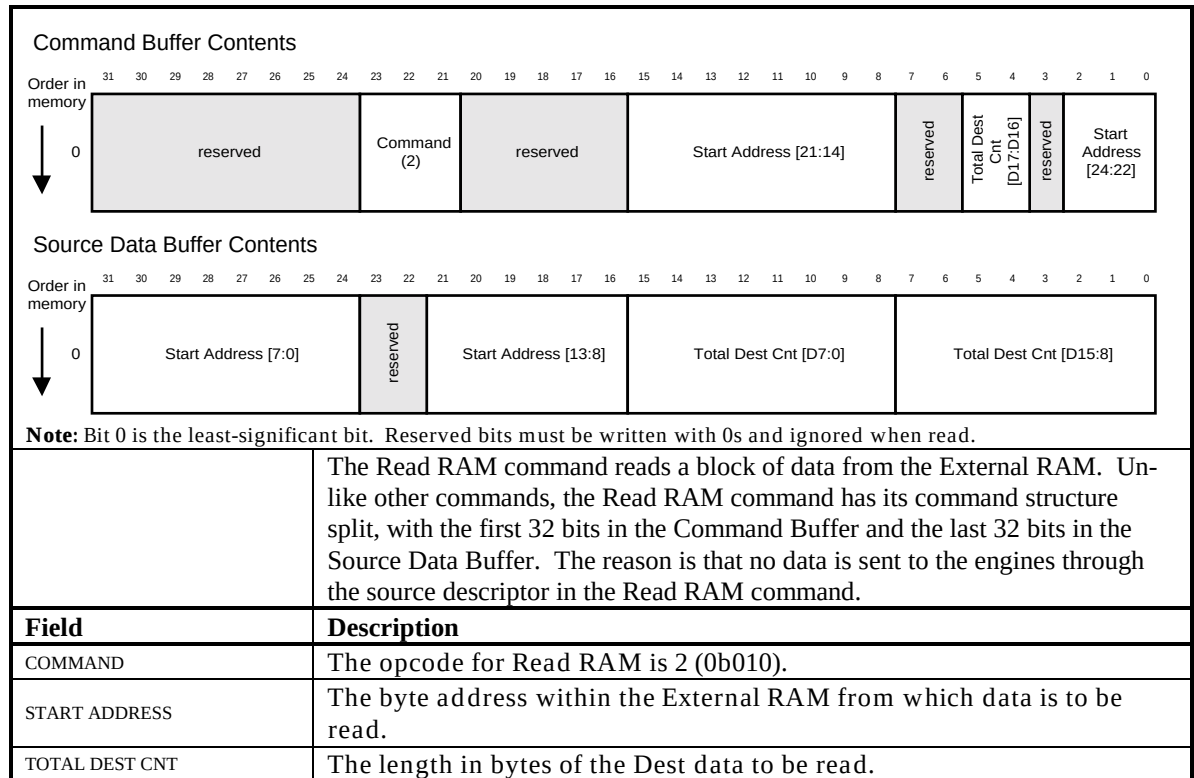


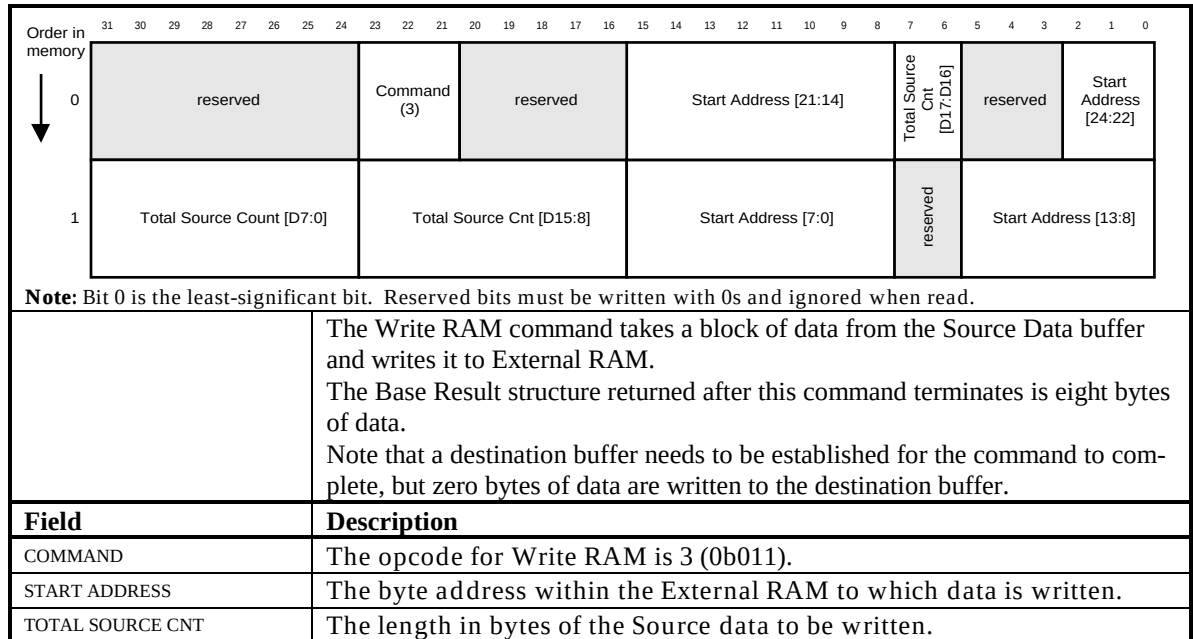
Figure 31. Typical Use of Descriptors For a Command That Requires Encryption Context

10.5 Read RAM/Write RAM Command Structures

10.5.1 Read RAM Command Structure



10.5.2 Write RAM Command Structure



10.6 Source Structures

The Source Data structure is a byte stream containing source data for the Packet Engine.

10.7 Dest Structures

The Destination Data structure is a byte stream containing the output of the Packet Engine. The number of valid bytes is given in the Total Dest Count field of the Base Result structure.

10.8 Result Structures

Like the Command structure, the Result structure starts with a Base structure and is followed by structures for the enabled units. The order of appearance is Base, Compression, MAC, and Encryption. If the unit is not enabled, its Result structure is not present. The Base Result structure is present even if no units were enabled. Figure 32 shows the Result structures.

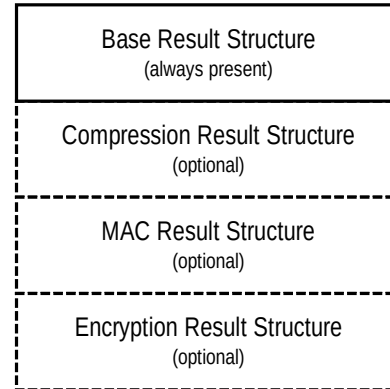


Figure 32. Result Structures

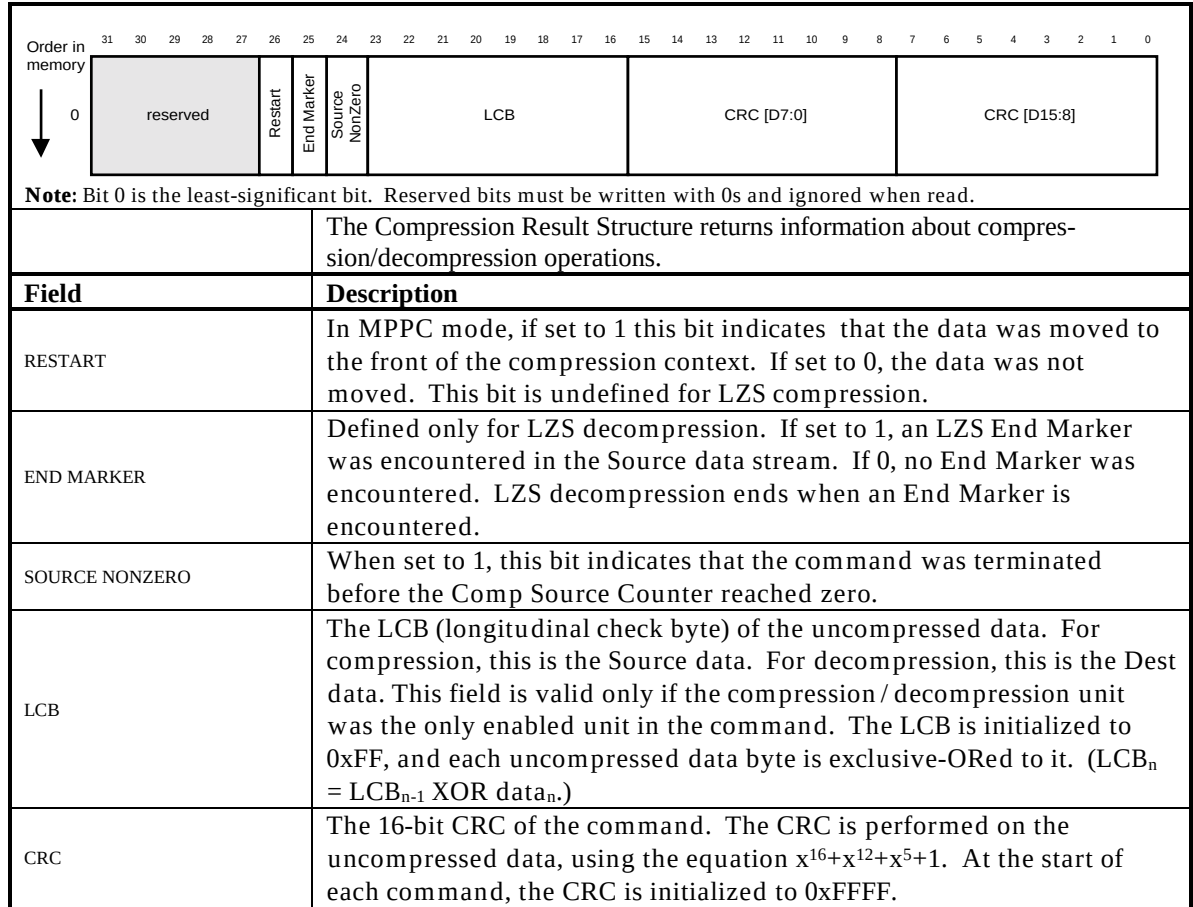
10.8.1 Base Result Structure

Order in memory ↓	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																															
	reserved																Dest Overrun	reserved	Session # [D7:D0]						Total Source Cnt [D17:D16]	Total Dest Cnt [D17:D16]	reserved	Session # [D10-D8]				
	Total Source Count [D7:0]								Total Source Cnt [D15:8]								Total Dest Count [D7:0]								Total Dest Cnt [D15:8]							

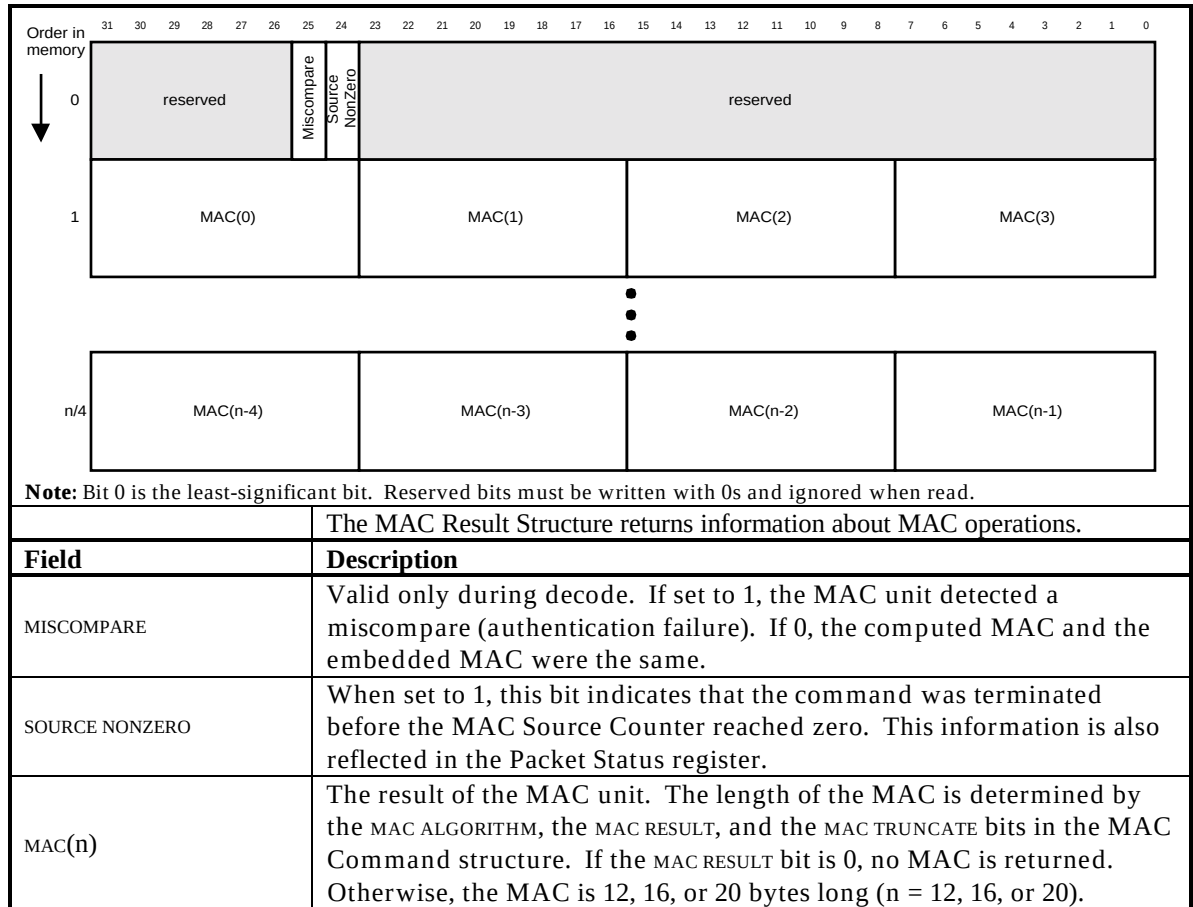
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.

	The Base Result Structure, like the Base Command Structure, is the first of as many as five appended result structures. In order of appearance, these are: Base, Compression, MAC, and Encryption. If a given stage is disabled in the command, its result structure is omitted.
Field	Description
DEST OVERRUN	Set to 1 when the command produces more data than specified in the TOTAL DEST CNT field of the Base Command structure, and the IGNORE DEST CNT bit was set to 0.
SESSION #	Contains the value of the SESSION # field of the corresponding Base Command structure. This field is undefined for a Read RAM or Write RAM command. Bit 11 of the SESSION # field is not returned.
TOTAL SOURCE CNT	Final value of the Total Source counter at command termination. The counter is decremented for each source data byte used, and should be zero if the command terminated cleanly. This field is undefined for a Read RAM command.
TOTAL DEST CNT	Final value of the Total Dest counter at command termination. The counter is decremented for each destination data byte produced. This field is undefined for a Write RAM command.

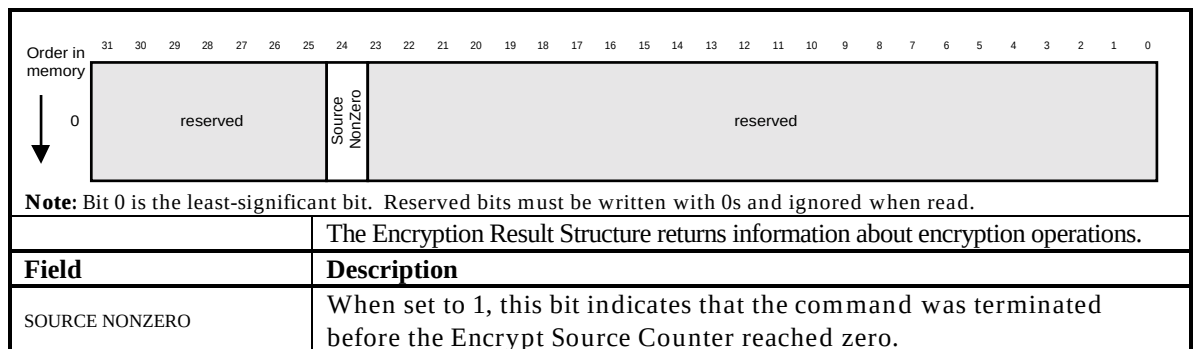
10.8.2 Compression Result Structure



10.8.3 MAC Result Structure



10.8.4 Encryption Result Structure



10.9 External RAM Usage

Use of external memory is required only if the data compression engine is used. When external memory is used there is some available space for keys and IVs to be stored so they don't need to be supplied for every encryption/authentication operation.

The 7902 has several external memory maps. These are set according to Packet Configuration register bits, as shown in Figure 33.

Compression Config	Encryption Config	Description
0	X	Multi-History mode. Sessions are each 16KB, starting at address 0 and building up.
1	0	Single-History mode. Each session history is 512 bytes. One 32 KB compression history is shared by all sessions
1	1	As above, but session history is 128 bytes.

Figure 33. Context Memory Modes

External RAM can be configured in one of two modes, depending on the setting of the COMPRESSION CONFIGURATION bit in the Packet Configuration register. The choices are between a compression history that is shared between all sessions, and an independent compression history for each session.

The multiple-history option combines encryption and MAC context with compression history in a single 16KB block. An LZS session uses 16 KB per full-duplex session, while MPPC uses 16 KB per half-duplex session. Session 0 starts at address 0 in External RAM. As all sessions are 16 KB in size, the address of session n is $16384 * n$. 32 sessions can fit into the 128KB external RAM.

The single-history option is used when all sessions are guaranteed to use only stateless compression algorithms. All sessions share a single 32KB compression history. The incremental per-session context is either 128 or 512 bytes, depending on the setting of the ENCRYPTION CONFIGURATION bit in the Packet Configuration register. The 32KB context area starts at address 0 in External RAM, and individual sessions start at 32 KB and work upwards in increments of 128 or 512 bytes. Single-history mode supports 3840x128 byte and 960x512 byte sessions.

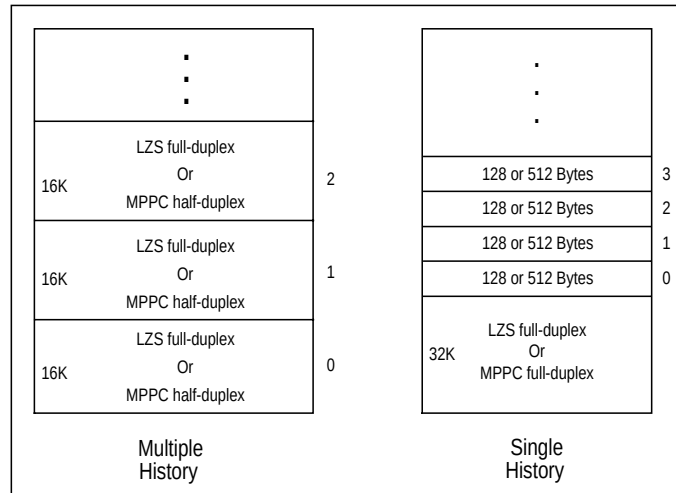


Figure 34. External RAM Memory Usage

10.10 Host Command Index Register

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
reserved																															
Host Command Index																															
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0080-0x0083																This register is used to update the Host Command Index. All fields are R/W.															
Reset Value																xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_0000															
Field																Description															
HOST COMMAND INDEX																The Host Command Index is a pointer to one of eight locations in internal memory where the next Command Descriptor is to be written by the host. This field must be updated after a command structure and descriptor have been written to internal memory by the host. Due to an ambiguity of the empty / full command descriptor status, the Host Command Index cannot contain the same value as the Device Command Index. To overcome this ambiguity, index values must be incremented from 0 through 15. Since there are only eight locations in memory where command descriptors are located, the index values of 8 through 15 actually point to command descriptors in locations 0 through 7. The actual byte offset relative to internal memory is $0x0000 + [(Index\ Value) \bmod 8] * 8$.															

10.11 Host Source Index Register

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				
reserved																												Host Source Index							
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																																			
Addr: 0x0084-0x0087								This register accesses the Host Source Index. All fields are R/W.																											
Reset Value								xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_0000																											
Field								Description																											
HOST SOURCE INDEX								The Host Source Index is a pointer to one of eight locations in internal memory where the next Source Descriptor is to be written by the host. This field must be updated after a source structure and descriptor have been written to internal memory by the host. Due to an ambiguity of the empty / full source descriptor status, the Host Source Index cannot contain the same value as the Device Source Index. To overcome this ambiguity, index values must be incremented from 0 through 15. Since there are only eight locations in memory where source descriptors are located, the index values of 8 through 15 actually point to source descriptors in locations 0 through 7. The actual byte offset relative to internal memory is $0x0040 + [(Index\ Value) \bmod 8] * 8$.																											

10.12 Host Result Index Register

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				
reserved																												Host Result Index							
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																																			
Addr: 0x0088-0x008B								This register accesses the Host Result Index. All fields are R/W.																											
Reset Value								xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_0000																											
Field								Description																											
HOST RESULT INDEX								The Host Result Index is a pointer to one of eight locations in internal memory where the next Result Descriptor is to be written by the host. This field must be updated after a result structure buffer has been allocated and the result descriptor has been written to internal memory by the host.																											
								Due to an ambiguity of the empty/full result descriptor status, the Host Result Index cannot contain the same value as the Device Result Index. To overcome this ambiguity, index values must be incremented from 0 through 15. Since there are only eight locations in memory where result descriptors are located, the index values of 8 through 15 actually point to result descriptors in locations 0 through 7. The actual byte offset relative to internal memory is 0x0080+[(Index Value) mod 8]*8.																											

10.13 Host Destination Index Register

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
reserved																												Host Destination Index			
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x008C-0x008F		This register accesses the Host Destination Index. All fields are R/W.																													
Reset Value		xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_0000																													
Field		Description																													
HOST DESTINATION INDEX		The Host Destination Index is a pointer to one of eight locations in internal memory where the next Destination Descriptor is to be written by the host. This field must be updated after a destination structure buffer has been allocated and the destination descriptor has been written to internal memory by the host. Due to an ambiguity of the empty / full destination descriptor status, the Host Destination Index cannot contain the same value as the Device Destination Index. To overcome this ambiguity, index values must be incremented from 0 through 15. Since there are only eight locations in memory where destination descriptors are located, the index values of 8 through 15 actually point to destination descriptors in locations 0 through 7. The actual byte offset relative to internal memory is $0x00C0 + [(Index\ Value) \bmod 8] * 8$.																													

10.14 Packet Status Register

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				
Dest Last 0	Dest Last 1	Dest Last 2	Dest Last 3	Dest Last 4	Dest Last 5	Dest Last 6	Dest Last 7	Device Dest Index					Dest Done	Dest Last	Device Result Index					Result Done	Result Last	Device Source Index					Source Done	Source Last	Device Command Index					Cmnd Done	Cmnd Last
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																																			
Addr: 0x0090-0x0093								This register reports the status of the descriptor rings. Some fields are constantly updated. Other fields, once set to 1, remain set to 1 until cleared by the host. See the individual field descriptions for details.																											
Reset Value								0000_0000_0000_0000_0000_0000_0000_0000																											
Field								Description																											
DEST LAST [0-7]								Reports which destination descriptors contain the last byte of destination data generated by a single command. Each bit represents a destination descriptor with the same index number. Once a bit is set to 1, it remains set. Writing a 1 to its bit position clears a bit.																											

DEVICE DEST INDEX	Reports the index of the Destination Descriptor to be used next by the device for destination data. This field is constantly updated. A write to this register does not affect this field. The actual byte offset relative to internal memory is $0x00C0 + [\text{Index mod } 8] * 8$. The value of DEVICE DEST INDEX increments by one after the data referenced by a destination descriptor has been completely written.
DEST DONE	Reports that data referenced by a destination descriptor has been completely written, and the DEVICE DEST INDEX has been incremented. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.
DEST LAST	Reports that data referenced by a destination descriptor has been completely written, and it contains the last byte of destination data generated by a single command. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.
DEVICE RESULT INDEX	Reports the index of the Result Descriptor to be used next by the device for a result structure. This field is constantly updated. A write to this register does not affect this field. The actual byte offset relative to internal memory is $0x0080 + [\text{Index mod } 8] * 8$. The value of DEVICE RESULT INDEX increments by one after a result structure referenced by a result descriptor has been completely written.
RESULT DONE	Reports that a result structure referenced by a result descriptor has been completely written, and the DEVICE RESULT INDEX has been incremented. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.
RESULT LAST	Reports that a result structure referenced by a result descriptor has been completely written, and it contains the last byte of the result structure generated by a single command. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.
DEVICE SOURCE INDEX	Reports the index of the Source Descriptor to be used next by the device for source data. This field is constantly updated. A write to this register does not affect this field. The actual byte offset relative to internal memory is $0x0040 + [\text{Index mod } 8] * 8$. The value of DEVICE SOURCE INDEX increments by one after the data referenced by a source descriptor has been completely read.
SOURCE DONE	Reports that data referenced by a source descriptor has been completely read, and the DEVICE SOURCE INDEX has been incremented. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.
SOURCE LAST	Reports that data referenced by a source descriptor has been completely read, and the descriptor was found to have its LAST bit set to 1. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.
DEVICE COMMAND INDEX	Reports the index of the Command Descriptor to be used next by the device for a command structure. This field is constantly updated. A write to this register does not affect this field. The actual byte offset relative to internal memory is $0x0000 + [\text{Index mod } 8] * 8$. The value of DEVICE COMMAND INDEX increments by one after the command structure referenced by a command descriptor has been completely read.

COMMAND DONE	Reports that a command structure referenced by a command descriptor has been completely read, and the <code>DEVICE COMMAND INDEX</code> has been incremented. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.
COMMAND LAST	Reports that a command structure referenced by a command descriptor has been completely read, and the descriptor was found to have its <code>LAST</code> bit set to 1. Once this bit is set to 1, it remains set. It is cleared by writing a 1 to this bit position.

10.15 Packet Interrupt Enable Register

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
reserved												Dest Done	Dest Last	reserved				Result Done	Result Last	reserved				Source Done	Source Last	reserved				Cmd Done	Cmd Last
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x0094-0x0097												This register enables the INT signal based on the status reported by the Packet Status register. All fields are R/W, returning the last value written.																			
Reset Value												xxxx_xxxx_xxxx_00xx_xx00_xxxx_00xx_xx00																			
Field												Description																			
DEST DONE												If set to 1, an interrupt is generated when the DEST DONE bit in the Packet Status register is set to 1.																			
DEST LAST												If set to 1, an interrupt is generated when the DEST LAST bit in the Packet Status register is set to 1.																			
RESULT DONE												If set to 1, an interrupt is generated when the RESULT DONE bit in the Packet Status register is set to 1.																			
RESULT LAST												If set to 1, an interrupt is generated when the RESULT LAST bit in the Packet Status register is set to 1.																			
SOURCE DONE												If set to 1, an interrupt is generated when the SOURCE DONE bit in the Packet Status register is set to 1.																			
SOURCE LAST												If set to 1, an interrupt is generated when the SOURCE LAST bit in the Packet Status register is set to 1.																			
COMMAND DONE												If set to 1, an interrupt is generated when the COMMAND DONE bit in the Packet Status register is set to 1.																			
COMMAND LAST												If set to 1, an interrupt is generated when the COMMAND LAST bit in the Packet Status register is set to 1.																			

10.16 Packet Configuration Register

313029282726252423222120191817161514131211109876543210 reserved Compress Config Encryption Config reserved																															
Note: Bit 0 is the least-significant bit. Reserved bits must be written with 0s and ignored when read.																															
Addr: 0x098-0x09B																Use this register to configure the Packet Engine. All fields are R/W, returning the last value written.															
Reset Value																xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_x00x															
Field																Description															
COMPRESS CONFIG																Set this bit to 0 to support multiple compression contexts, or set it to 1 to support only a single compression context. See section 10.9 External RAM Usage for more details.															
ENCRYPTION CONFIG																Set this bit to 0 to support 512-byte encryption contexts. This mode must be used if the RC4 encryption algorithm is required. Set this bit to 1 to support 128-byte encryption contexts. This mode does not support the RC4 encryption algorithm. See section 10.9 for more details.															

11 DC Specifications

DC Supply Voltage (V_{DD} , V_{DD2} , AV_{DD2})	-0.3V to +5.0V
DC Input Voltage	-0.3V to $V_{DD}+0.3$
Storage Temperature	-40°C to +125°C
Delay between asserting +2.5V and +3.3V power supplies (V_{DD} , V_{DD2} , AV_{DD2})	0-500ms

Figure 35. Absolute Maximum Ratings

Warning! 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.

11.1 Power Sequencing

If the +2.5V and +3.3V power supply voltages are not asserted at the same time, the possibility of reverse currents arises. To prevent damage to the device, these voltages must be enabled within the time given in the absolute maximum ratings. The power supply should be designed to assert power within the time limits given under the recommended operating conditions.

11.2 Recommended Operating Conditions

DC Supply Voltage (V_{DD})	+3.0V to +3.6V
DC Supply Voltage (V_{DD2} , AV_{DD2})	+2.2V to +2.8V
Delay between asserting +2.5V and +3.3V power supplies (V_{DD} , V_{DD2} , AV_{DD2})	0 – 100ms
Operating Temperature	0°C to +70°C

Figure 36. Recommended Operating Conditions

11.3 DC Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V _{IL}	Low level input voltage (I)				0.8	V
	Clock Input (CI)				0.8	V
V _{IH}	High level input voltage (I)		2.0			V
	Clock Input (CI)		2.4			V
I _{IL}	Low level input current (I)	V _{IN} = V _{SS} V _{DD} = 3.6V	-10			μA
I _{IH}	High level input current (I)	V _{IN} = V _{DD} V _{DD} = 3.6V			-10	μA
V _{OL}	Low level output voltage	V _{DD} = 3.0V				
	(O4)	I _{OL} = 4ma			0.4	V
	(O8)	I _{OL} = 8mA			0.4	V
V _{OH}	High level output voltage	V _{DD} = 3.0V				
	(O4)	I _{OH} = -4mA	2.4			V
	(O8)	I _{OH} = -8mA	2.4			V
I _{OZ}	High impedance output leakage current	V _O = V _{SS} or V _{DD} V _{DD} = 3.6V	-10			μA
I _{DDs}	Quiescent supply current				360	μA
C _{IN}	Input capacitance	V _{DD} = 3.3V		2.4		pF
C _{OUT}	Output capacitance	V _{DD} = 3.3V		5.6		pF
C _{I/O}	I/O capacitance	V _{DD} = 3.3V		6.6		pF
P _A	Power dissipation T _{clk} = 50 MHz	V _{DD} = 3.6V		0.41	0.60	W
		V _{DD2} = 2.8V		0.14	0.23	
		Total		0.50	0.60	
	Power dissipation T _{clk} = 66 MHz	V _{DD} = 3.6V		0.54	0.80	W
		V _{DD2} = 2.8V		0.18	0.30	
		Total		0.65	0.80	
I _{AVDD2}	PLL analog power supply current	A _{VDD2} = 2.8V			3.4	mA
I _{AVSS}	PLL analog ground current	A _{VDD2} = 2.8V			3.4	mA

Note: I=input, O=output; I/O=bi-directional

Figure 37. DC Electrical Characteristics

12 AC Specifications

Symbol	Parameter	Conditions
C_{L1}	Output load on External SRAM Interface	5pF min, 40 pF max
C_{L2}	Output load on all other pins	50 pF
V_{DD}	Supply voltage	3.3V $\pm$ 5%
V_{DD2} , AV_{DD2}	Supply voltage	2.5V $\pm$ 5%
V_{SS}	Ground potential	0V
T_A	Ambient operating temperature	0°C to +70°C

Note: See derating information below for other busload conditions.

Figure 38. AC Specification Definition

Signal	Pins	Derating ¹
External RAM data bus (high to low)	CD[15-0]	0.6ns per 10pF
External RAM data bus (low to high)	CD[15-0]	0.35ns per 10pF
External RAM address bus	CA[17-0]	0.28ns per 10pF

¹ These derating values represent typical case. For worst case derating, multiply these values by 1.75. For best case derating, multiply these values by 0.5.

Figure 39. AC Specification Derating

Number	Description	Min	Max	Units
1	Reset width	4 t_{CLK}		ns
2	First 7902 access after Reset	8 t_{CLK}		ns

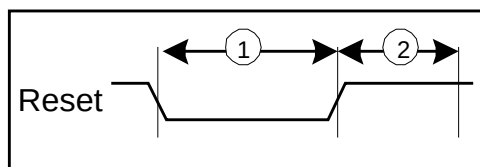


Figure 40. Reset Timing

Number	Description	PLL Condition	Min	Max	Units
1	Clock frequency ($t_{CLK} = 1 / \text{clock frequency}$)	enabled	40	66	MHz
		disabled		40	
2	Clock width high	enabled	6		ns
		disabled	9.5		
3	Clock width low	enabled	6		ns
		disabled	9.5		
4	Clock rise time from V_{IL} to V_{IH}	enabled 40-50 MHz		3	ns
		enabled 50-66 MHz		1.5	
		disabled		3	
5	Clock fall time from V_{IH} to V_{IL}	enabled 40-50 MHz		3	ns
		Enabled 50-66 MHz		1.5	
		disabled		3	
–	PLL lock time	enabled		100	μs
		disabled		N / A	
–	Clock frequency jitter	enabled		3	%
		disabled		3	

Note: Clock widths are measured to / from 1.4V. PLLE#=low (PLL Enabled), PLLE#=high (PLL Disabled)

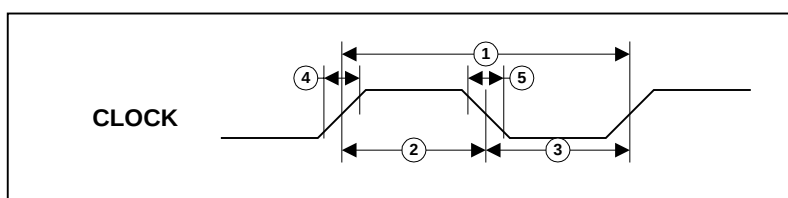


Figure 41. External Clock

Number	Description	PLL Condition	Min	Max	Units
1	A/MA, TS#, CS#, TSIZ, R/W# setup		3		ns
2	A/MA, TS#, CS#, TSIZ, R/W# hold		1		ns
3	Write data setup		3		ns
4	Write data hold		1		ns
5	$\tau_{A\#}$ high-z to low (assertion delay)	enabled	1.5	8	ns
		disabled	3	14	
6	$\tau_{A\#}$ low to high (deassertion delay)	enabled	1.5	8	ns
		disabled	3	14	
7	$\tau_{A\#}$ high to high-z	enabled	1.5	8	ns
		disabled	3	14	
8	Read data output valid delay (high-z to 0/1)	enabled	1.5	9	ns
		disabled	3	16.5	
9	Read data output invalid delay (0/1 to high-z)	enabled	1.5	9	ns
		disabled	3	16.5	

PLLE#=low (PLL Enabled), PLLE#=high (PLL Disabled)

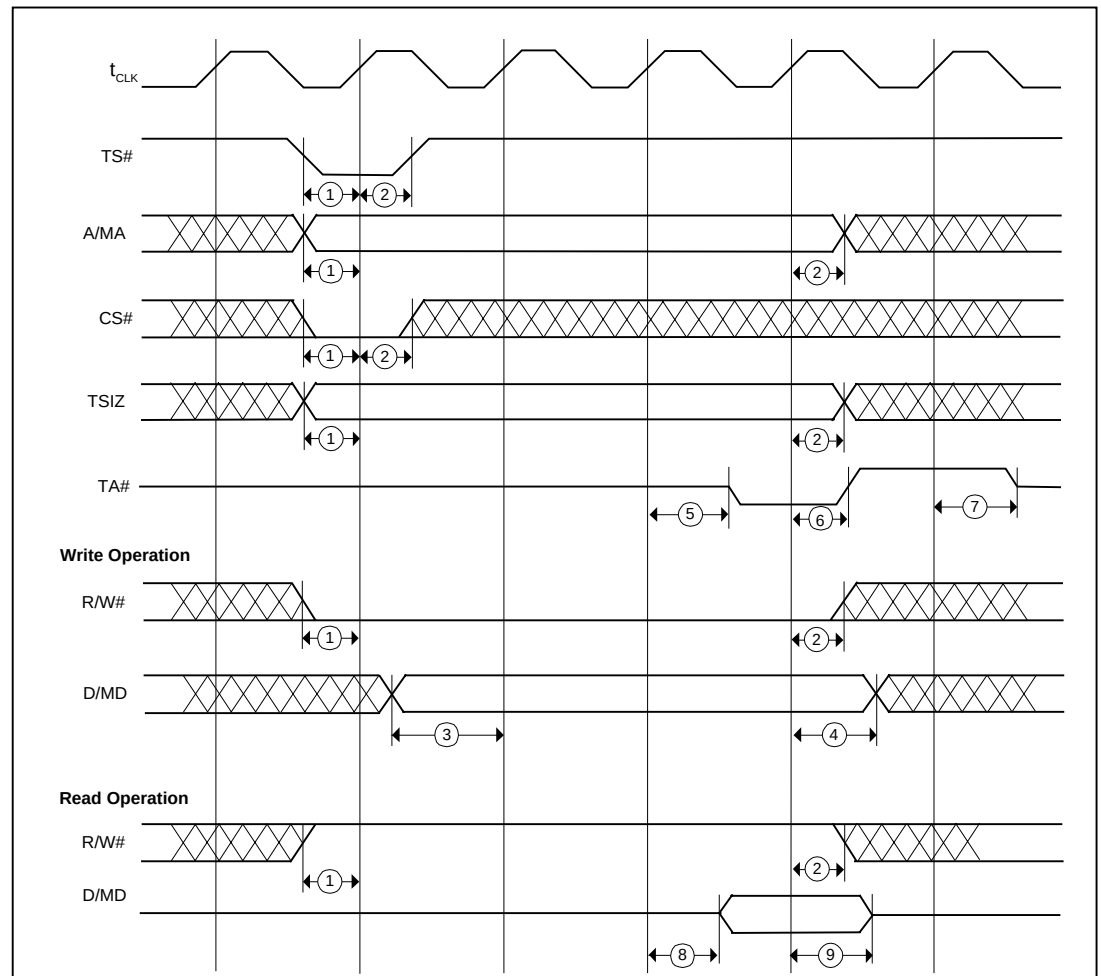


Figure 42. Read/Write CPU Timing (Single Beat)

Number	Description	Min	Max	Units
1	Data valid after CA/CLB#/CUB# valid (access time)		$2 t_{CLK} - 15$	ns
2	Data valid after COE# active		$2 t_{CLK} - 14$	ns
3	Address valid width (cycle time)	$2 t_{CLK} - 7$		
4	Data hold after COE# inactive	0	$t_{CLK} - 1$	ns
5	CLB#/CUB# Access Time		$2 t_{CLK} - 13$	ns
6	Output Enable time from CLB#/CUB#	1		ns
7	Output Disable time from CLB#/CUB#		7	ns
8	Output Enable time from COE#	1		ns
9	Output Data Hold time from CA change	3		ns

Note: $t_{CLK} = 1 / (\text{clock frequency in MHz.})$ Example: If the CLK input frequency is 50MHz, use $t_{CLK}=20$ instead of $t_{CLK}=20\text{E-}9$ when calculating the times.

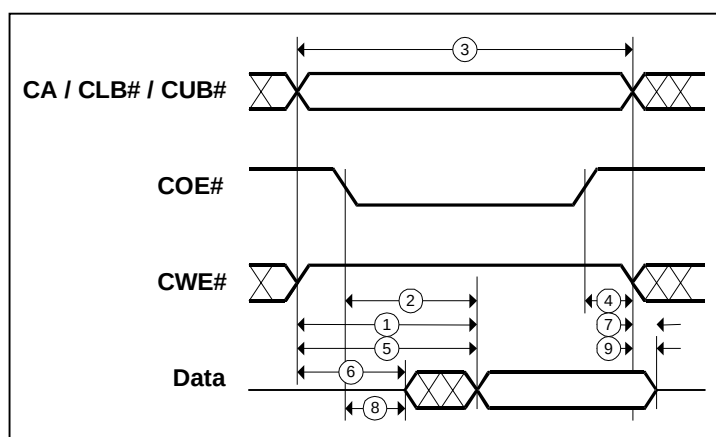


Figure 43. External SRAM Read Timing

Number	Description	Min	Max	Max
1	CA/CLB#/CUB# setup to CWE# valid	$t_{CLK} - 7$		ns
2	CWE# active width	$t_{CLK} - 4$		ns
3	CA hold after CWE# inactive	0		ns
4	Data setup before CWE# inactive	$t_{CLK} - 7$		ns
5	Data hold after CWE# inactive	0		ns
6	CA valid to CWE# inactive	$2 t_{CLK} - 9$		ns
7	Write Cycle time	$2 t_{CLK} - 7$		ns
8	CLB#/CUB# to End of Write	$2 t_{CLK} - 7$		ns

Note: $t_{CLK} = 1 / (\text{clock frequency in MHz.})$ Example: If the CLK input frequency is 50MHz, use $t_{CLK}=20$ instead of $t_{CLK}=20\text{E-}9$ when calculating the times.

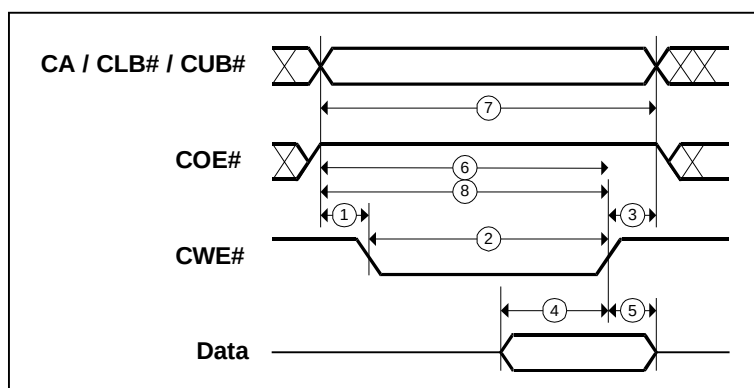


Figure 44. External SRAM Write Timing

13 Thermal Specifications

Parameter	Min	Typ.	Max	Units
Junction Temperature (Tj)	0		100	°C
Operating Temperature (Ta)	0		70	°C
Thermal resistance, (θ_{ja}) at 0 m/s airflow		24.0		°C/W
Thermal resistance, (θ_{jma}) at 1 m/s airflow		20.2		°C/W
Thermal resistance, (θ_{jma}) at 2 m/s airflow		19.0		°C/W
Thermal resistance, (θ_{jc})		7.1		°C/W

Figure 45. Thermal Specifications

14 Pin List

Important: See note below regarding the address and data bus pins.

Pin	Name	Pin	Name	Pin	Name	Pin	Name
1	TSIZ[1]	37	IRQ#	73	CD[11]	109	A[07]/MA[24]
2	TSIZ[2]	38	A[00]/MA[31]	74	CD[12]	110	A[08]/MA[23]
3	TSIZ[3]	39	A[01]/MA[30]	75	CD[13]	111	A[09]/MA[22]
4	VSS	40	A[02]/MA[29]	76	CD[14]	112	A[10]/MA[21]
5	D[19]/MD[12]	41	BMODE	77	CD[15]	113	A[11]/MA[20]
6	VDD	42	VSS	78	CWE#	114	VDD
7	D[20]/MD[11]	43	VSS	79	COE#	115	A[12]/MA[19]
8	D[21]/MD[10]	44	VSS	80	CA[16]	116	VSS
9	D[22]/MD[09]	45	CLK	81	CA[15]	117	A[13]/MA[18]
10	D[23]/MD[08]	46	NC	82	CUB#	118	VSS
11	D[24]/MD[07]	47	VDD	83	CLB#	119	VSS
12	D[25]/MD[06]	48	VDD2	84	VSS	120	D[00]/MD[31]
13	VSS	49	VDD2	85	CA[00]	121	D[01]/MD[30]
14	D[26]/MD[05]	50	AVDD2	86	CA[01]	122	D[02]/MD[29]
15	D[27]/MD[04]	51	NC	87	CA[02]	123	D[03]/MD[28]
16	D[28]/MD[03]	52	AVSS	88	CA[03]	124	D[04]/MD[27]
17	D[29]/MD[02]	53	VSS	89	CA[04]	125	VDD2
18	VDD2	54	VSS	90	VDD	126	D[05]/MD[26]
19	BURST#	55	AACK#	91	CA[05]	127	VSS
20	D[30]/MD[01]	56	TA#	92	VSS	128	D[06]/MD[25]
21	D[31]/MD[00]	57	TS#	93	CA[06]	129	D[07]/MD[24]
22	DBB#	58	VSS	94	CA[07]	130	D[08]/MD[23]
23	TEA#	59	VDD	95	CA[08]	131	D[09]/MD[22]
24	BDIP# / PSDVAL#	60	CD[00]	96	CA[09]	132	D[10]/MD[21]
25	VSS	61	CD[01]	97	CA[17]	133	D[11]/MD[20]
26	RESET#	62	CD[02]	98	CA[10]	134	D[12]/MD[19]
27	VSS	63	CD[03]	99	VDD	135	VDD
28	VSS	64	VSS	100	CA[11]	136	VSS
29	VDD	65	CD[04]	101	VSS	137	D[13]/MD[18]
30	PLLE#	66	CD[05]	102	CA[12]	138	VSS
31	CS#	67	CD[06]	103	CA[13]	139	D[14]/MD[17]
32	NDTEST#	68	CD[07]	104	CA[14]	140	D[15]/MD[16]
33	R / W#	69	CD[08]	105	A[03]/MA[28]	141	D[16]/MD[15]
34	NDPI	70	CD[09]	106	A[04]/MA[27]	142	D[17]/MD[14]
35	NDPO	71	CD[10]	107	A[05]/MA[26]	143	D[18]/MD[13]
36	NC	72	VDD	108	A[06]/MA[25]	144	TSIZ[0]

Note: For the dual-named address and data bus pins, MA[18-31] and MD[0-31] are used with processors supporting the MPC860 / 8260 bit numbering conventions and A[13-0] and D[31-0] are be used with all other processors.

Figure 46. Pin List (Numeric)

Important: See note below regarding the address and data bus pins.

Pin	Name	Pin	Name	Pin	Name	Pin	Name
38	A[00]/MA[31]	80	CA[16]	134	D[12]/MD[19]	2	TSIZ[2]
39	A[01]/MA[30]	97	CA[17]	137	D[13]/MD[18]	3	TSIZ[3]
40	A[02]/MA[29]	60	CD[00]	139	D[14]/MD[17]	6	VDD
105	A[03]/MA[28]	61	CD[01]	140	D[15]/MD[16]	29	VDD
106	A[04]/MA[27]	62	CD[02]	141	D[16]/MD[15]	47	VDD
107	A[05]/MA[26]	63	CD[03]	142	D[17]/MD[14]	59	VDD
108	A[06]/MA[25]	65	CD[04]	143	D[18]/MD[13]	72	VDD
109	A[07]/MA[24]	66	CD[05]	5	D[19]/MD[12]	90	VDD
110	A[08]/MA[23]	67	CD[06]	7	D[20]/MD[11]	99	VDD
111	A[09]/MA[22]	68	CD[07]	8	D[21]/MD[10]	114	VDD
112	A[10]/MA[21]	69	CD[08]	9	D[22]/MD[09]	135	VDD
113	A[11]/MA[20]	70	CD[09]	10	D[23]/MD[08]	18	VDD2
115	A[12]/MA[19]	71	CD[10]	11	D[24]/MD[07]	48	VDD2
117	A[13]/MA[18]	73	CD[11]	12	D[25]/MD[06]	49	VDD2
55	AACK#	74	CD[12]	14	D[26]/MD[05]	125	VDD2
50	AVDD2	75	CD[13]	15	D[27]/MD[04]	4	VSS
52	AVSS	76	CD[14]	16	D[28]/MD[03]	13	VSS
24	BDIP# / PSDVAL#	77	CD[15]	17	D[29]/MD[02]	25	VSS
41	BMODE	83	CLB#	20	D[30]/MD[01]	27	VSS
19	BURST#	45	CLK	21	D[31]/MD[00]	28	VSS
85	CA[00]	79	COE#	22	DBB#	42	VSS
86	CA[01]	31	CS#	37	IRQ#	43	VSS
87	CA[02]	82	CUB#	36	NC	44	VSS
88	CA[03]	78	CWE#	46	NC	53	VSS
89	CA[04]	120	D[00]/MD[31]	51	NC	54	VSS
91	CA[05]	121	D[01]/MD[30]	34	NDPI	58	VSS
93	CA[06]	122	D[02]/MD[29]	35	NDPO	64	VSS
94	CA[07]	123	D[03]/MD[28]	32	NDTEST#	84	VSS
95	CA[08]	124	D[04]/MD[27]	30	PLLE#	92	VSS
96	CA[09]	126	D[05]/MD[26]	33	R / W#	101	VSS
98	CA[10]	128	D[06]/MD[25]	26	RESET#	116	VSS
100	CA[11]	129	D[07]/MD[24]	56	TA#	118	VSS
102	CA[12]	130	D[08]/MD[23]	23	TEA#	119	VSS
103	CA[13]	131	D[09]/MD[22]	57	TS#	127	VSS
104	CA[14]	132	D[10]/MD[21]	144	TSIZ[0]	136	VSS
81	CA[15]	133	D[11]/MD[20]	1	TSIZ[1]	138	VSS

Note: For the dual-named address and data bus pins, MA[18-31] and MD[0-31] are used with processors supporting the MPC860 / 8260 bit numbering conventions and A[13-0] and D[31-0] are used with all other processors.

Figure 47. Pin List (Alphabetical)

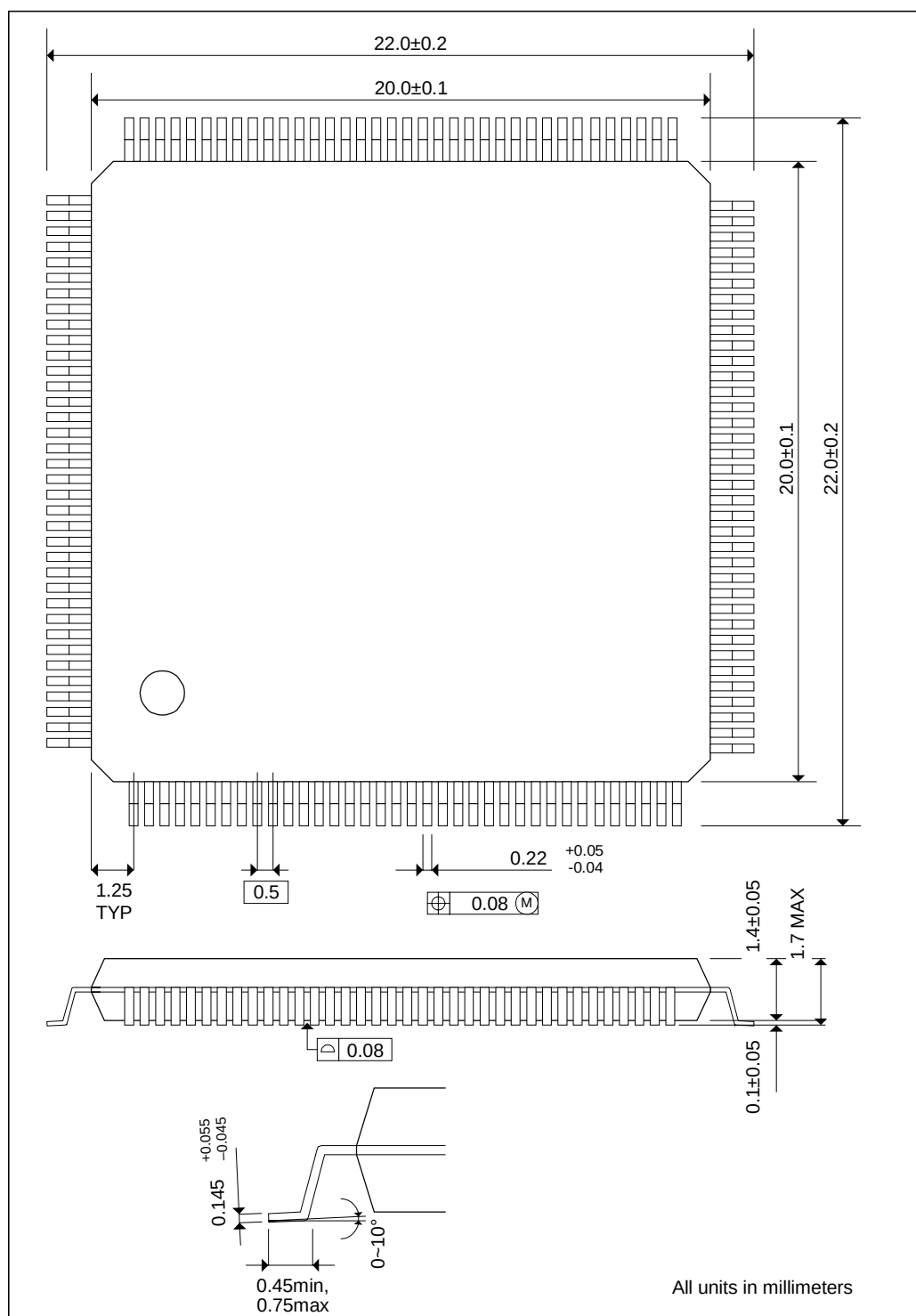


Figure 48. 144-Pin TQFP Package