DATA SHEET 7811 Network Security Processor

Network Security Processors

Hi/fn® supplies two of the Internet's most important raw materials: compression and encryption. Hi/fn is also the world's first company to put both on a single chip, creating a processor that performs compression and encryption at a faster speed than a conventional CPU alone could handle, and for much less than the cost of a Pentium® or comparable processor.

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

Hi/fn Applications Support Hotline:
408-399-3544

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.

This product is approved for export and reexport under License Exception ENC to non-government end-users worldwide except those located in Iran, Iraq, Libya, North Korea, Sudan, Syria, Yugoslavia (Serbia), and Taliban-controlled regions of Afghanistan.

Table of Contents

Figures

1 Introduction

The Hi/fn® 7811 Security Processor performs network security functions in hardware. Its pipelined architecture allows compression, encryption, and authentication to be performed in a single pass. The 7811 hardware supports the encryption, compression, and authentication algorithms, required for IPSec, PPTP, L2TP, PPP, and others.

The 7811 is used in network servers, routers, and gateways. It accelerates the security functions of virtual private networks, electronic commerce, and secure Web sessions. The 7811's PCI interface allows it to be implemented as an add-in card or on the motherboard.

A 7811-based security subsystem includes a PCI interface, SDRAM memory, and a dedicated microprocessor. Running Hi/fn's security software, such a subsystem handles network security protocols at the packet level, supporting the Internet security protocol (IPSec) in both tunnel and transport modes. The PPP, PPTP, and L2TP transmission protocols are supported in addition to TCP/IP.

1.1 Features

Security Engine

- Pipelined security engine performs encryption/decryption, compression/ decompression, and authentication
- Supports major security protocols
- Built-in DMA engines handle command and data streams
- Supports 32K simultaneous IPSec sessions
- Optional lock/unlock mode allows manufacturer-selected levels of exportability
- Backward-compatible with the Hi/fn 7751

Local CPU Interface

- Glueless CPU interface connects to a 32-bit MIPS microprocessor
- Local CPU works with 7811 to boost performance and offload host CPU
- Dedicated CPU allows sensitive operations to be isolated from the host

SDRAM Interfaces

- General purpose SDRAM interface supports up to 64 MB and increases performance and reduces PCI bus activity
- Dedicated context memory interface supports up to 64 MB for session contexts
- Memory can be made inaccessible to the host, enhancing security

PCI Interface

- Bus-mastering 32-bit, 33 MHz PCI interface

True Random Number Generator

- On-chip true-random number for public-key cryptography and IV generation

Security Barrier

- Security barrier features protect security-relevant data items such as keys
- Encryption hardware and memory can be made inaccessible from PCI
- Supports FIPS 140-1, Level 3-compliant security subsystems

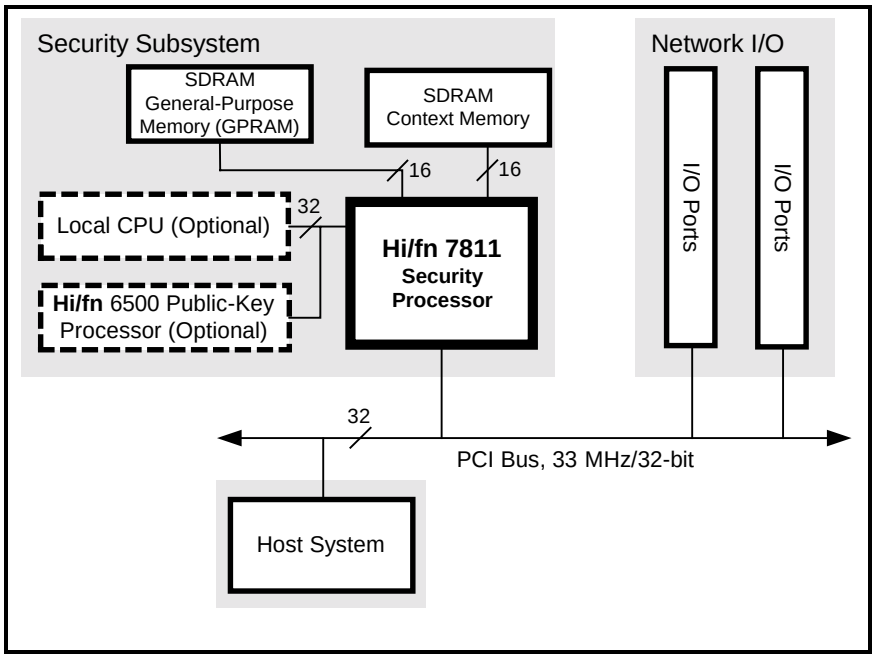


Figure 1. System concept

1.2 Description

Hi/fn's 7811 is a maximum-performance network security processor, achieving high performance through a pipelined Security Engine, two private memory buses, an attached microprocessor, and four general purpose DMA units. In addition to high performance, the 7811 is designed for high security. When the 7811 is in *protected mode,* encryption keys, session keys, and other security-relevant data items cannot be recovered from the 7811. The 7811 has many additional security features to protect its data and its operation, making it ideal for FIPS 140-1, Level 3-compliant security subsystems. These same features also enhance the security of systems that do not require the stringent FIPS 140-1 Level 3 standards.

The 7811 supports up to 64 MB of SDRAM on each of two local buses. *Context memory* holds session-oriented context information. *GPRAM (general purpose memory)* provides I/O buffering and general purpose storage. The 7811 supports 32768 independent, simultaneous IPSec sessions.

The 7811 is a PCI bus master, controlling data moving into and out of the host system with its four general purpose DMA units. When the host has packets for the 7811 to process, it creates a small DMA command structure and sets a Valid bit. The 7811 transfers the commands and data, processes the packets, and writes the transformed packets and status information back to host memory. The 7811 maintains all security information locally.

An on-chip true-random number generator (RNG) provides random data for use in public-key encryption.

Unless run in 7751 compatibility mode, an attached microprocessor controls the 7811. The processor initializes the 7811, sets up and tears down each security session, controls the 7811's four general purpose DMA engines, performs header processing, and manages the security subsystem in general. The processor adds performance to the security subsystem, reduces host overhead, and allows the

creation of a security barrier, where the security subsystem is isolated from the rest of the system and security information (such as keys) cannot be recovered by the host or an intruder who has gained control of the host. The subsystem generally includes a processor, a boot ROM, and local memory.

Hi/fn provides security software for the 7811, including a complete set of software for the 7811's MIPS processor and interface routines for the host.

With or without the MIPS CPU, the 7811 can run in 7751 compatibility mode, where it functions as an upgraded, higher-performance 7751 security processor.

Part Number	Description
7811-PB3	7811 Network Security Processor

Figure 2. Ordering information

1.3 Block Diagram

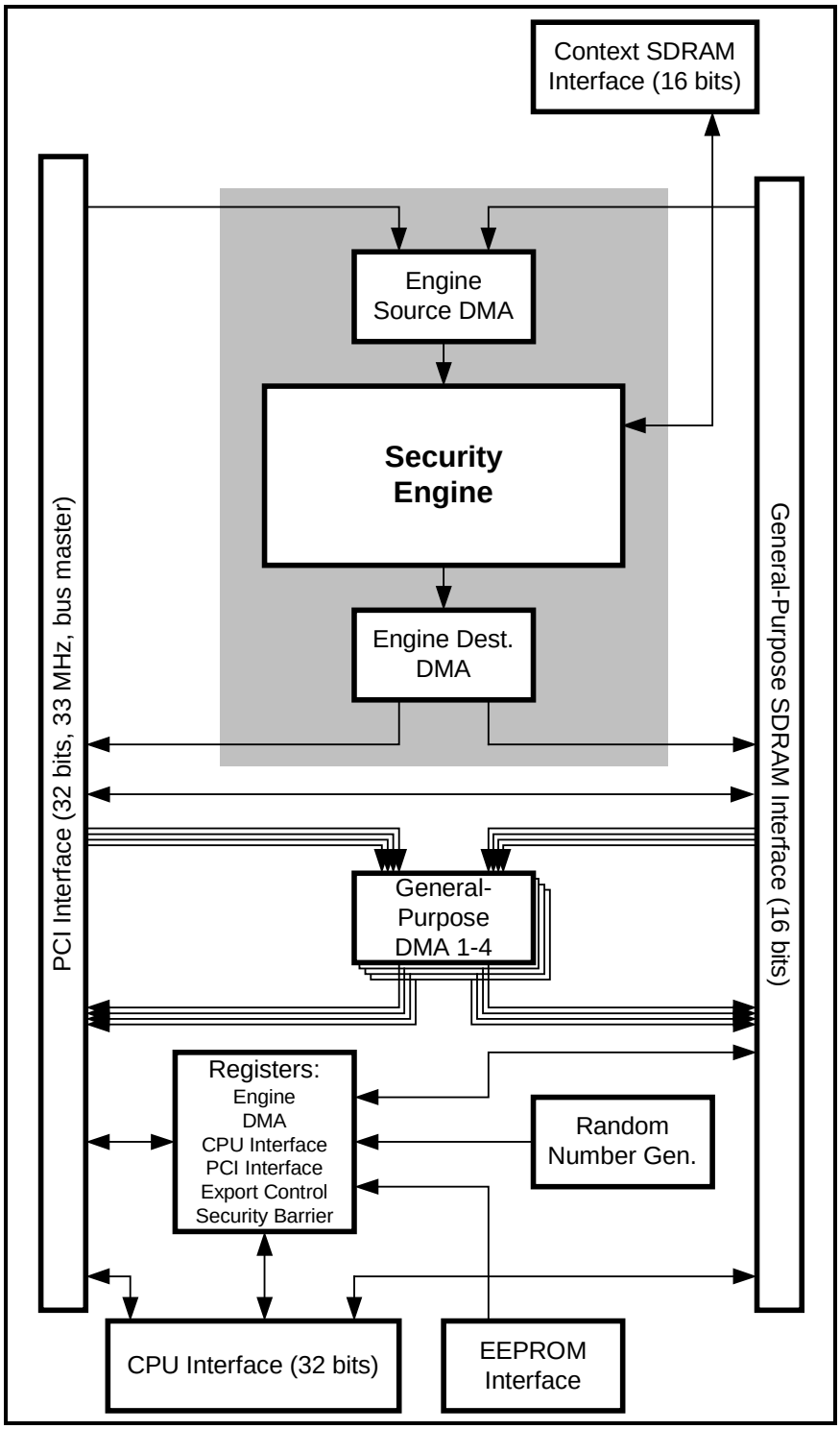


Figure 3. Block Diagram of the 7811

Block	Description	See Sections
Security Engine	Pipelined compression, encryption, authentication, and checksum/CRC/LCB engine with internal DMA engines that operate independently from the general purpose DMA engines.	2.2.1, 6
General Purpose DMA Engines (GPDMA1-4)	Four identical DMA engines that transfer commands and data to and from the 7811 subsystem. In protected mode, Only the source registers of GPDMA 1 and the destination registers of GPDMA 4 are accessible to the host; the rest are controlled by the MIPS CPU.	2.2.2
Registers	Configuration, control, and status registers.	2.2.3, 8
Random Number Generator	A hardware random-number generator used in public-key cryptography.	2.2.4
PCI Interface	Bus-mastering 33-MHz/32-bit PCI interface. All features of the 7811 are available to the host through PCI accesses, provided the 7811 is in unprotected mode. In protected mode, very little of the 7811 can be accessed through PCI. In addition to being a bus master, the 7781 is also an efficient PCI target.	2.2.5
Local CPU Interface	Glueless interface to a MIPS processor. All features of the 7811 are available to the MIPS processor through memory-mapped accesses.	2.2.6
EEPROM Interface	Controls an external 93C46 EEPROM containing configuration data, including encryption engine unlock codes that vary according exportability requirements.	2.2.7
General Purpose SDRAM (GPRAM) Interface	Controls external SDRAM containing data buffers, general purpose storage, and, optionally, code for the MIPS processor. (GPRAM is also called “packet SDRAM.”) Supports up to 64 MB of SDRAM in the same chip configurations as context SDRAM.	2.2.8
Context SDRAM Interface	Controls external SDRAM containing per-session compression, encryption, and authentication state (such as keys and compression histories). Supports up to 64 MB of SDRAM. Four-bank SDRAM devices in 8Mx8, 16Mx8, 4Mx16, 8Mx16, and 16Mx16 configurations are supported.	0

Figure 4. Functional units in the 7811

1.4 Specification Summary

Packet Protocols

- IPSec. The following are supported in both transport and tunnel modes:
  - AH, ESP, AH+ESP, AH+IPPCP, ESP+IPPCP, AH+ESP+IPPCP
- PPP transforms:
  - CCP, ECP, CCP+ECP
- PPTP
- L2TP

Encryption Algorithms

- DES, 3DES, RC4

Authentication Algorithms

- SHA, MD5

Compression Algorithms

- LZS, MPPC

Performance

- IPSec performance
  - 246 Mbps with 3DES encryption and SHA-1 authentication
  - 100 Mbps with 3DES encryption, SHA-1 authentication, and LZS compression
- Peak Performance is shown below:

Mode	Speed
LZS compression	100 Mbps
LZS decompression	200 Mbps
MPPC compression	80 Mbps
MPPC decompression	200 Mbps
3DES encryption/decryption	252 Mbps
RC4 encryption/decryption	200 Mbps
SHA authentication	301 Mbps
MD5 authentication	376 Mbps

Packaging

352-pin TBGA

Power

LVTTL CMOS device with 5V-tolerant PCI I/O. Maximum power dissipation is 3.0 watts measured at 3.6V.
The 7811 has a unique power saving feature when the MIPS interface is left unused (When used in the 7751 mode, without the GPRAM and MIPS). Pulling up the pin AD2 (signal TEST1) to VDD will ensure this low power mode. Maximum power dissipation in this mode would be 2.3W measured at 3.6V.

2 Device Architecture

2.1 Data and Control Flow

The 7811 has a security core that is based on the 7751 Security processor. Unlike the 7751, however, it is not necessary for the host to issue low-level commands to the Security Engine directly. Instead, the host sends packet streams and high-level commands to the 7811, and the 7811's local MIPS processor performs the header processing required to process the packets and issues the Security Engine commands.

By using a relatively inexpensive local processor, the intelligence of the security subsystem is greatly increased. The advantages of this are much lower host CPU and PCI bus overhead, more efficient utilization of the Security Engine, reduced host software complexity, and the ability to create a security boundary.

In general, the 7811 uses its GPDMA units to move data and command streams from host memory to GPRAM. The MIPS processor executes the commands through a combination of Security Engine commands and software (for example, while compression and encryption are performed entirely in hardware, header manipulation is generally performed in software). The Security Engine generally uses GPRAM for both input and output, as this gives the highest performance. When the security operations are finished, the MIPS processor queues the fully processed packets to be sent by a GPDMA unit to a buffer in host memory. This is normally system memory, but could be a memory-mapped device.

To the host, security subsystems using more than one 7811 look the same as a single-7811 implementation by using the same I/O and control model regardless of the number of 7811's.

2.2 Functional Units

2.2.1 Security Engine

The Security Engine is a fully pipelined unit that performs security and compression processing on packets. It encodes by performing compression, encryption, padding, and authentication field generation. It decodes by performing decryption, decompression, depadding, and authentication header testing.

The Security Engine is compatible with the Hi/fn 7751, containing functionally equivalent units and using the same registers, DMA engines, and command structures. Performance, however, has been significantly enhanced, and the memory and PCI interfaces have been merged with the higher-performance 7811 equivalents.

New functionality of the 7811 is controlled by additional registers, generally at higher addresses than those of the 7751.

Figure 5 shows the engine configurations for encoding and decoding.

Authentication and checksum/CRC/LCB processing take place in parallel with other operations.

For details of the operation of the Security Engine, see chapter 7.

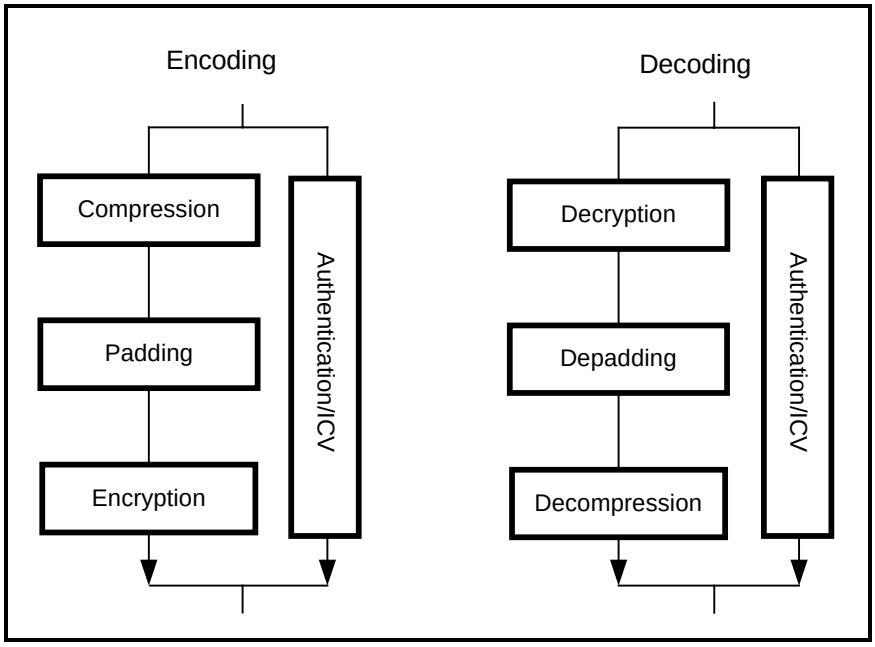


Figure 5. Pipeline configurations for encryption and decryption

2.2.2 General Purpose DMA Units (GPDMA1-GPDMA4)

The 7811 has four general purpose DMA units, GPDMA1 through GPDMA4. These units move data from one portion of memory and/or PCI address space to another, using 128-byte burst transfers for maximum efficiency.

The units are identical in structure, and in unprotected mode they are interchangeable. In protected mode, however, the host processor only has access to the Source registers of GPDMA1 and the Destination register of GPDMA4, allowing the host to specify little more than the source of data being sent to the 7811 and the destination of data coming out of the 7811.

For more information about the GPDMA units, see chapter 6.

2.2.3 Registers

The 7811 contains a variety of internal registers for initialization, status, and test purposes. These registers are partitioned into two groups. Group 0 contains Security Engine registers. Group 1 contains registers from all of the 7811's functional units. When accessed over PCI, each register group occupies 4KB of memory and has its own base address register. Group 0 uses BAR0, while Group 1 uses BAR1. When accessed by the MIPS processor, each register group has a programmable starting address.

In protected mode, most of these registers are inaccessible to PCI, though they are accessible to the MIPS processor.

Group 0 Registers

Register	Offset	Reference
Security Engine Data	0x00	Figure 57
Security Engine Control	0x04	Figure 58
Security Engine Interrupt Status	0x08	Figure 59
Security Engine Configuration	0x0C	Figure 60
Security Engine Interrupt Enable	0x10	Figure 61
Security Engine Status	0x14	Figure 62
Security Engine FIFO Status	0x18	Figure 63
Security Engine FIFO Configuration	0x1C	Figure 64

Figure 6. Group 0 registers

Group 1 Registers

Register	Offset	Description	Ref.
Security Engine Command Ring Address	0x0C	Address of the next command descriptor.	Figure 65
Security Engine Source Ring Address	0x1C	Address of the next source descriptor.	Figure 66
Security Engine Result Ring Address	0x2C	Address of the next result descriptor.	Figure 67
Security Engine Dest Ring Address	0x3C	Address of the next destination descriptor.	Figure 68
Security Engine Status and Control	0x40	Status and configuration bits for the Security Engine.	Figure 69
Security Engine Interrupt Enable	0x44	Enables interrupts on various conditions.	Figure 70
Security Engine DMA Configuration	0x48	Configures the Security Engine's DMA units.	Figure 71
PCI Address "OR" Mask	0x4C	Renders low GPRAM memory inaccessible to PCI	Figure 72
PCI Interrupt	0x50	Routes MIPS interrupts to the PCI bus	Figure 73
PCI Int. Mask	0x54	Enables the interrupts defined by the PCI Interrupt register	Figure 74
MIPS Interrupt	0x58	Routes PCI interrupts to the MIPS processor	Figure 75
MIPS Int. Mask	0x5C	Enables the interrupts defined by the MIPS Interrupt register	Figure 76
RNG Enable	0x60	Enables the random number generator	Figure 77
RNG Config	0x64	Sets RNG parameters	Figure 78
RNG Data	0x68	Random data	Figure 80
RNG Status	0x6C	RNG state information	Figure 81
MIPS SDRAM1 Address	0x70	Starting address of default mapping of GPRAM in the MIPS address space.	Figure 82

Register	Offset	Description	Ref.
MIPS SDRAM2 Address	0x74	Starting address of the secondary mapping of GPRAM in the MIPS address space. Used in multi-7811 systems.	Figure 83
MIPS Group 1 Register Address	0x78	Starting address for the Group 1 registers in the MIPS address space	Figure 84
MIPS Group 0 Register Address	0x7C	Starting address for the Group 0 registers in the MIPS address space	Figure 85
MIPS PCI1 Register Address	0x80	Starting address for PCI-mapped accesses in the MIPS address space	Figure 86
MIPS PCI2 Register Address	0x84	Starting address for a second PCI-mapped region in the MIPS address space	Figure 87
MIPS PCI1 Address Translation	0x88	Upper address bits to convert a MIPS address to PCI for the PCI1 region	Figure 88
MIPS PCI2 Address Translation	0x8C	Upper address bits to convert a MIPS address to PCI for the PCI2 region	Figure 89
MIPS Config	0x90	MIPS Config – Big/little endian selection and interrupt mapping	Figure 90
MIPS Reset	0x94	Reset bits for the MIPS processor	Figure 91
Revision ID	0x98	Silicon revision	Figure 92
EEPROM Data	0x9C	32-bit EEPROM data	Figure 93
GPDMA1 Source Addr.	0xA0	Source Command Pointer for GPDMA1.	Figure 94
GPDMA2 Source Addr.	0xA4	Source Command Pointer for GPDMA2.	Figure 95
GPDMA1 Dest Addr.	0xA8	Dest command pointer for GPDMA1.	Figure 96
GPDMA2 Dest Addr.	0xAC	Dest command pointer for GPDMA2.	Figure 97
GPDMA1_2 Arbitration	0xB0	Time-slice control for GPDMA1 and GPDMA2.	Figure 98
GPDMA1_2 Config	0xB4	Mode config for GPDMA1 and GPDMA2.	Figure 99
GPDMA1_2 Status	0xB8	Interrupt status for GPDMA1 and GPDMA2	Figure 100
GPDMA1_2 Interrupt Enable	0xBC	Interrupt enables for GPDMA1 and GPDMA2	Figure 101

Register	Offset	Description	Ref.
PCI BAR0 Shadow	0xC0	Read Only copy of BAR0 (4KB region holding Group 0 registers)	Figure 102
PCI BAR1 Shadow	0xC4	Read Only copy of BAR1. (4KB region holding Group 1 registers)	Figure 103
PCI BAR2 Shadow	0xC8	Read Only copy of BAR2. (Mapped to GPRAM, up to 64 MB in size)	Figure 104
SDRAM Config	0xCC	Read Only copy of SDRAM parameters	Figure 105
GPDMA3 Source Addr.	0xD0	Source Command Pointer for GPDMA3.	Figure 106
GPDMA4 Source Addr.	0xD4	Source Command Pointer for GPDMA4.	Figure 107
GPDMA3 Dest Addr.	0xD8	Dest command pointer for GPDMA3.	Figure 108
GPDMA4 Dest Addr.	0xDC	Dest command pointer for GPDMA4.	Figure 109
GPDMA3_4 Arbitration	0xE0	Time-slice config for GPDMA3 and GPDMA4.	Figure 110
GPDMA3_4 Config	0xE4	Mode config for GPDMA3 and GPDMA4.	Figure 111
GPDMA3_4 Status	0xE8	Interrupt status for GPDMA3 and GPDMA4.	Figure 112
GPDMA3_4 Enable	0xEC	Interrupt enables for GPDMA3 and GPDMA4.	Figure 113
Global Status	0xF0	Engine, DMA, and interrupt status.	Figure 114
Test-and-Set	0xF8	Test-and-Set semaphore register.	Figure 115

Figure 7. Group 1 register summary

2.2.4 Random Number Generator

Public-key cryptography depends on a reliable, high-speed source of random numbers. The 7811 contains a hardware random number generator (RNG) based on internal free-running oscillators whose frequencies drift relative to each other and to the 7811 internal clock. The phase relation of these signals is unpredictable, and this is used to provide a random bit stream. The random bits are mixed cryptographically with internal state derived from previous random bits, updating the internal state. The output mixing function uses this internal state to produce 32-bit random numbers at a programmable rate.

The results are pushed into the *random number FIFO,* which is a 4-element FIFO, 32 bits wide. The FIFO can be read from the RNG Data register.

It is possible to use random numbers more quickly or slowly than they can be generated (hence the FIFO). The Random Number Ready bit in the RNG Status register is set when the random number generator contains at least two random numbers. An interrupt can also be asserted when the Random Number Ready bit is set. If the host underflows the FIFO, the RNG Underflow bit will be set.

On reset, the random number generator is disabled. It must be re-enabled through the RNG Enable register. If the random number FIFO is read while the random number generator is disabled, the result is undefined.

At its maximum speed, the random number generator operates at about 2.8 Mb/s at a 90 MHz clock. For computations refer to Figure 78. RNG Config register.

2.2.5 PCI Interface

The PCI interface is a 33 MHz/32-bit PCI bus master designed for maximum-performance DMA transfers.

The PCI interface is generally used by the 7811's four DMA interfaces, with the 7811 as the bus master. GPDMA transfers use 128-byte bursts whenever possible, while the Security Engine DMA units use 64-byte bursts. The 7811 is also an efficient PCI target in situations where the host or another PCI bus master needs to access the 7811's registers or GPRAM. It supports posted writes and a single active delayed read. The 7751-compatible Security Engine is also capable of becoming a PCI bus master.

Three base-address registers are used, BAR0-BAR2. BAR0 and BAR1 are used for register access, while BAR2 provides access to GPRAM. In protected mode, access to registers and memory is restricted. See section 5.1 for the PCI memory map.

The PCI configuration space provides 64 bytes of header information to the host system. All header values have reset defaults. Many can be loaded from the external EEPROM. See section 5.4 for the PCI configuration space mapping and section 5.3 for the EEPROM mapping.

The 7811 can assert PCI interrupts under a variety of conditions, and can also route PCI interrupts to the MIPS processor.

2.2.6 Local CPU Interface

The CPU interface connects to an optional external processor that controls the 7811. This 32-bit interface is compatible with specified MIPS processors. From the point of view of the MIPS processor, the 7811 is a memory-mapped slave device.

The local MIPS CPU adds intelligence to the security subsystem, relieving the host of low-level tasks. The local CPU is used by the Hi/fn Security Platform software to create a packet-oriented subsystem. The host hands off input packets and session numbers to the security subsystem, and software running on the MIPS CPU issues the appropriate 7811 commands to process the input packet. It then assembles an output packet from the results and sends it back to the host. This greatly reduces the involvement of the host in the operation of the security subsystem, simplifying the host software and reducing PCI bus traffic.

In addition to packet-oriented processing, the MIPS CPU performs subsystem initialization, session setup, and session tear-down.

The MIPS processor can execute code out of memory on its local bus, GPRAM, or PCI memory. Execution from GPRAM or PCI memory degrades the MIPS and 7811 performance and is not recommended for use in systems requiring very high performance.

The 7811 memory spaces and functional units are mapped into the MIPS memory space through a set of address pointer registers. See section 5.2 for the MIPS memory map.

The MIPS processor can run in either big-endian or little-endian mode. The MIPS_BIGENDIAN input on the 7811 informs the 7811 of the endianness of the MIPS CPU.

The 7811 supports fast writes and MIPS-style sub-block bursting. These features are controlled by the MIPS_Config register. See Figure 90.

MIPS interrupts are supported by the 7811. The 7811 can interrupt the MIPS on a variety of user-selectable conditions.

Supported processors are the R4300 and R4310 from NEC, RV4640 and 64V474 from IDT and the RM5230 from QED.

2.2.7 EEPROM Interface

The EEPROM interface allows the storage of PCI configuration information, export-control information, and system setup information in an external 93C46 EEPROM.

2.2.8 GPRAM Interface

The GPRAM interface controls an external 16-bit SDRAM. GPRAM is used for I/O buffering of both incoming and outgoing packet data. The MIPS processor uses it for data memory and can also use it for program memory. GPRAM contains input and output packet buffers, and the Security Engine's Source, Destination, Command, and Result rings. GPRAM has greater bandwidth and lower latency than the PCI bus, allowing the 7811 to run at higher performance with less bus overhead.

Regions of GPRAM can be made inaccessible to the outside world, placing them inside the 7811's security boundary. The 7811 can support GPRAM to the extent of 64MB.

Big-Endian/Little-Endian Conversion

To support big-endian/little-endian conversion, the 7811 contains five different byte-swapping units. These are shown in Figure 8. Internally, the 7811's Security Engine operates in little-endian mode. The operating principle of 7811 endian-conversion is to store data in GPRAM in little-endian mode, regardless of source. For byte-swapping to occur, the data must flow through a path that includes one of the byte swappers, as show in Figure 8.

The byte swappers are independently controlled, as shown in the diagram. They are affected by register bits, command and descriptor bits, and an input pin, MIPS_BIGENDIAN. In the case of the GPDMA and Security Engine DMA units, the endianness of data buffers, commands, descriptors, and Write32 transfers are controlled independently.

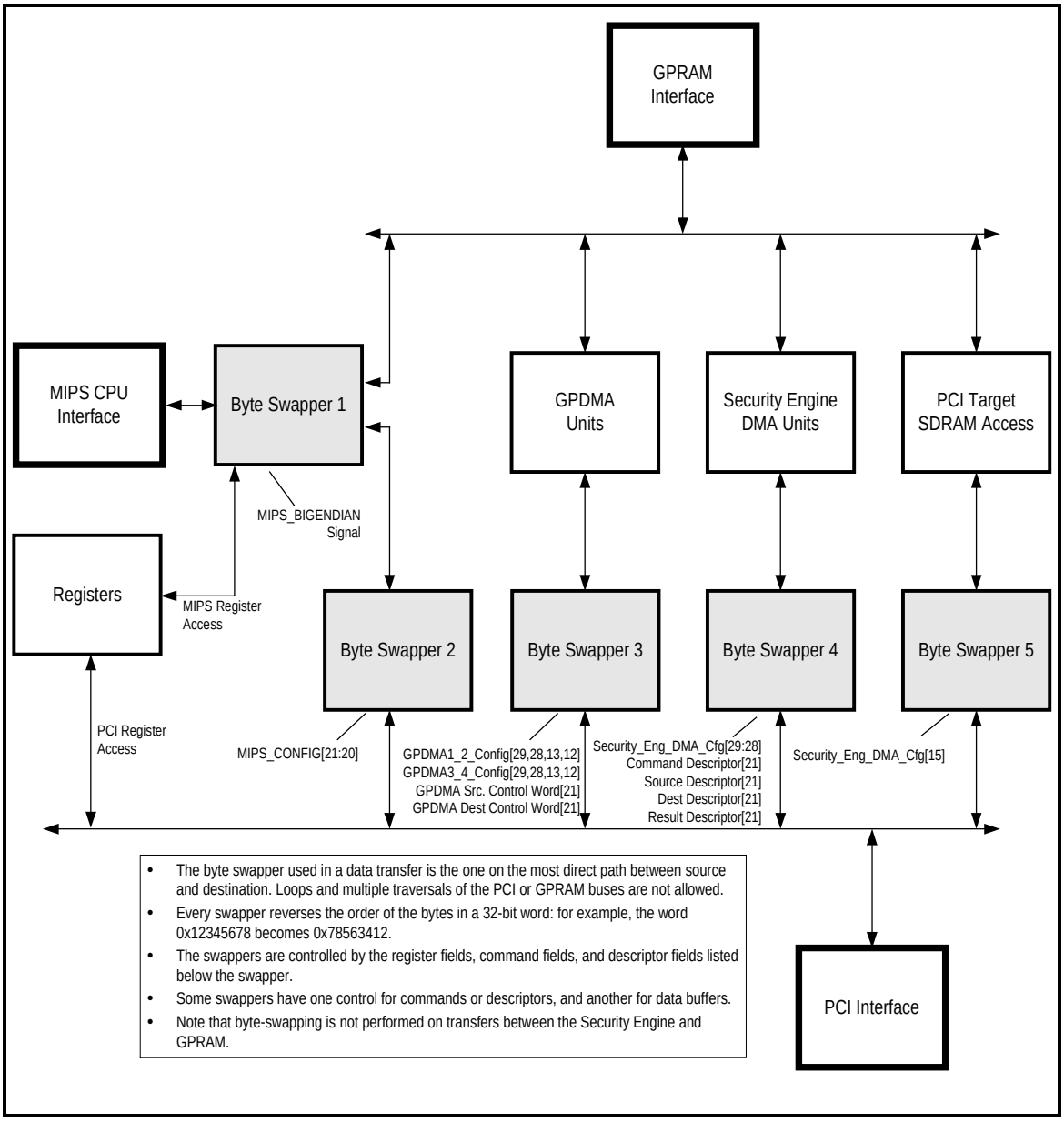


Figure 8. Big-endian conversion in the 7811

The byte-swappers do simple endian-conversion. When a byte-swapper is enabled, the order of the bytes within each 32-bit word is reversed. Bits [31:24] become bits [7:0] on output, bits [23:16] become bits [15:8], bits [15:8] become bits [23:16], and bits [7:0] become bits [31:24]. When a byte-swapper is disabled, data passes through unmodified.

Endianness Override

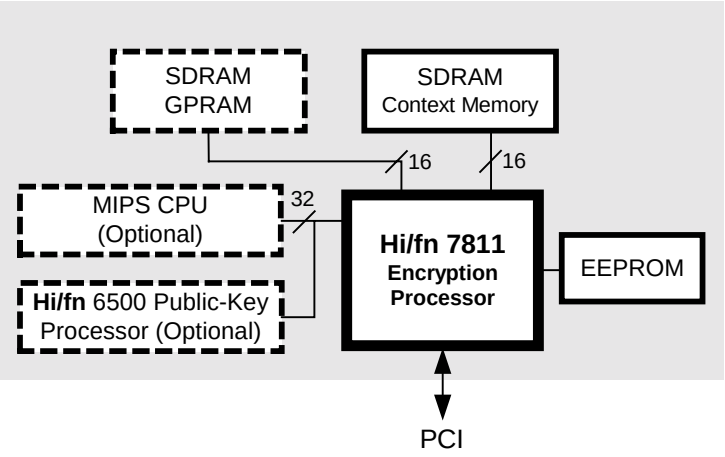
This affects the MIPS interface. It allows the MIPS processor to access data in either endianness simultaneously, which avoids software byte reordering. The following six registers are affected: SDRAM1 Address, SDRAM2 Address, MIPS Group 1 Address, MIPS Group 0 Address, MIPS PCI1 Address, and MIPS

PCI2 Address (Base-1 + 0x70 to Base-1+0x84). Each of these Group-1 registers is affected in the same way. In the production version of the 7811, if bit 0 of any of the above registers is set, then it effectively complements the state of the MIPS_BIGENDIAN pin for accesses through that register. This provides the ability to utilize a different endianness for each of the six address spaces defined by these registers.

2.2.9 Context SDRAM Interface

The context memory interface is another 16-bit SDRAM interface. Context memory contains session context information for the Security Engine. It cannot be accessed directly by the rest of the 7811. The 7811 can support SDRAM to the extent of 64MB.

3 Subsystem Configurations



Block	Description
Hi/fn 7811 Security Processor	Contains compression, encryption, and authentication engines, DMA engines, a random-number generator, bus-mastering PCI interface, CPU interface, and two SDRAM interfaces
GPRAM (Optional)	Provides data buffering and general purpose storage. However, the absence of the local CPU does not necessitate the absence of GPRAM
Context Memory	Contains the state information (compression history, encryption keys, etc.) necessary to perform compression and security processing for each active session
Local CPU (Optional)	The local CPU is used to perform header processing and manage the 7811's DMA engines. If run without a local CPU, the host CPU performs the tasks normally done by the local CPU
Hi/fn 6500 Public Key Processor (Optional)	Accelerates key exchange and public-key encryption
EEPROM	Contains system parameters and encryption engine unlock codes (for exportability).

Figure 9. Elements of a 7811 subsystem

3.1 Subsystem Variants

3.1.1 Typical Subsystem

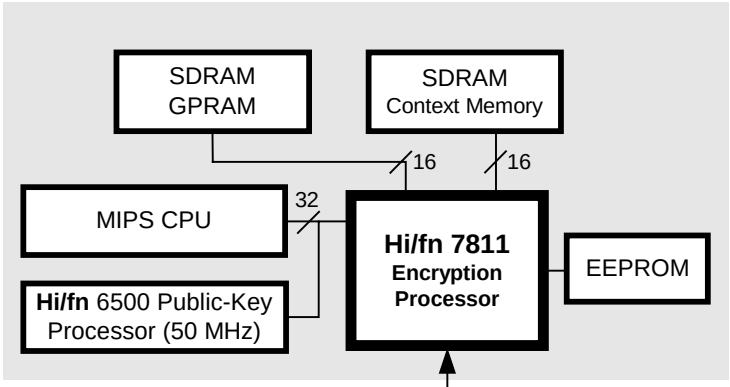


Figure 10. Typical subsystem

A typical 7811 implementation will make use of all the supported features, using the MIPS CPU for local packet processing and a 50 MHz Hi/fn 6500 for public-key encryption. Such a system can also be made FIPS 140-1, Level 3 compliant if operated in protected mode and rendered physically inaccessible to probing (typically by potting it in plastic).

3.1.2 High Performance Subsystem

For maximum performance, two 7811's can be used in the same subsystem, with each serviced by its own MIPS processor. In a two-7811 subsystem, one 7811 is used for incoming packets, and the other for outgoing packets. This is shown in Figure 11. Each 7811 has its own SDRAM and EEPROM interfaces. To the host system, each 7811 is a distinct PCI device, but the host software will see very little difference between single-7811 and multiple-7811 configurations. A 100 MHz Hi/fn 6500 is provided for maximum public-key performance. Like the typical system, the high-performance system can be made FIPS 140-1, Level 3 compliant.

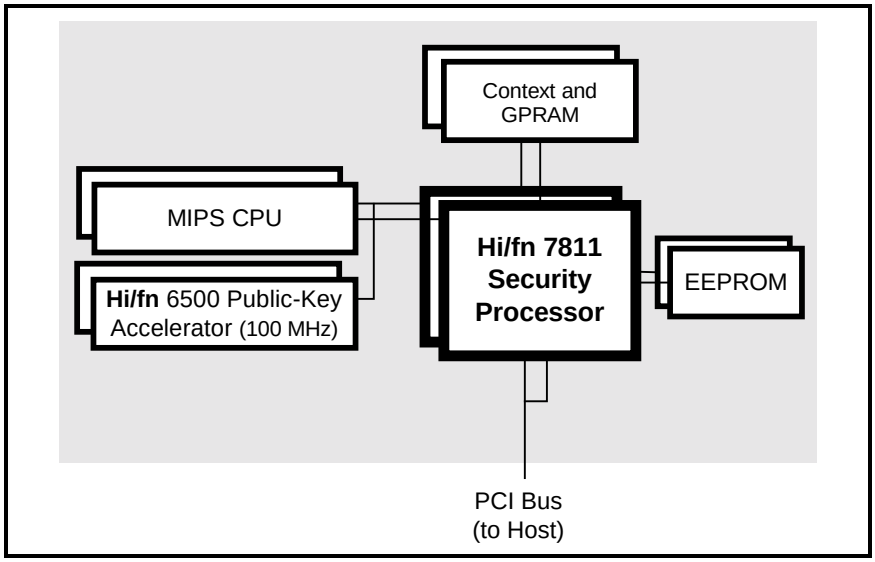


Figure 11. High Performance Subsystem

3.1.3 Minimal Subsystem (7751 Compatibility Mode)

A low cost implementation might eliminate the local CPU, GPRAM, and the Hi/fn 6500. Such a subsystem would use the host processor to perform all CPU functions. Without the MIPS CPU, the 7811 operates as an enhanced 7751. That is, it provides the same features as the Hi/fn 7751, with the addition of a random number generator, an increase in the maximum number of sessions to 32768, the use of SDRAM for context memory, and higher engine performance. This subsystem cannot be made FIPS 140-1 Level 3 compliant. This configuration is shown in Figure 12. Greater performance would be obtained if GPRAM were added.

(The Security Engine of the 7811 is compatible with the Hi/fn 7751. This means that 7751 compatibility is not a backwards-compatibility mode that is turned on and off, but an integral feature of the device.)

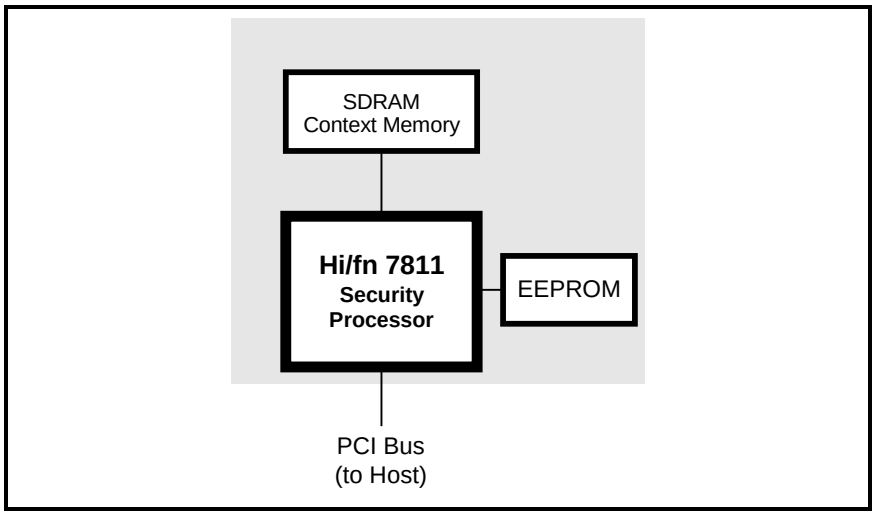


Figure 12. Minimal (no CPU) subsystem

3.1.4 Protected and Unprotected Modes

When the local MIPS CPU is used, the 7811 has two modes of operation: *unprotected mode* and *protected mode,* controlled by the PROTECT# input signal.

Unprotected mode. When running in unprotected mode, the full state of the 7811 is accessible across the PCI bus.

Protected mode. The goal of protected mode is to render security-relevant data items inaccessible. Such items include encryption keys, session keys, and access control information. In protected mode, very little of the 7811's state is accessible over PCI. Most of the 7811's registers are completely inaccessible (writes to the registers are ignored; reads return zeroes). By default, the 7811's GPRAM is inaccessible to the host. It can be partially or completely unlocked by the MIPS processor by setting bits [13:12] of the Security Engine DMA Configuration register. The host is left with almost no access to the 7811 except for the ability to tell the 7811 where to find input and output command and data buffers. In short, protected mode gives the 7811 a *security boundary,* a barrier that renders security-relevant data items inaccessible even to software that has gained complete control over the host.

The local MIPS processor retains full access to the 7811 at all times, and serves as a firewall. The software conventions used by Hi/fn are designed with protected mode in mind.

FIPS Compliance. The 7811 was designed with FIPS 140-1 Level 3 compliance in mind. FIPS 140-1 is an NIST standard for cryptographic modules. Such compliance requires security measures on a variety of levels, including physical barriers to data theft. While some of these measures depend on the system designer, the 7811 facilitates FIPS 140-1 Level 3 compliance by including all of the necessary circuit-level features.

Protected mode and FIPS 140-1 Level 3 compliance should not be confused with one another.

4 Signal Description

The 7811's signals are listed on the following pages.

If the MIPS processor is not used, the MIPS interface pins should be terminated as shown in Figure 14. If GPRAM is not used, the GPRAM interface pins should be left floating.

Signal	Type	Polarity	Description
Context SDRAM Signals			
CADDR[12:0]	Output	-	Context SDRAM multiplexed row/column address output bus.
CBA[1:0]	Output	-	Context SDRAM bank-select outputs.
CCAS#	Output	Active-Low	Context SDRAM column-address strobe output.
CCLK	Output	Rising Edge	Context SDRAM clock output.
CCS#	Output	Active-Low	Context SDRAM chip-select output
CDATA[15:0]	I/O	-	Context SDRAM data I/O bus.
CDQMH	Output	Active-High	Context SDRAM data mask output for bits 15:8.
CDQML	Output	Active-High	Context SDRAM data mask output for bits 7:0.
CRAS#	Output	Active-Low	Context SDRAM row-address strobe.
CWE#	Output	Active-Low	Context SDRAM write enable.
EEPROM Signals			
EEPROM_CS	Output	Active-High	EEPROM chip select
EEPROM_DI	Input	Active-High	EEPROM data in
EEPROM_DO	Output	Active-High	EEPROM data out
EEPROM_SK	Output	Rising Edge	EEPROM clock
MIPS Signals			
MIPS_AD[31:0]	I/O	-	MIPS SysAD bus. Multiplexed 32-bit address/data bus for the MIPS processor.
MIPS_BIGENDIAN	Input	Active-High	Data format of the MIPS processor.
MIPS_CLK	Input	Rising Edge	Clock input. This is the MIPS bus clock. A frequency-doubled clock internally derived from MIPS_CLK drives the Security Engine and the SDRAM interfaces.
MIPS_CMD[8:0]	I/O	-	MIPS SysCmd bus. Carries command/data identifier codes between the MIPS processor and the 7811.
MIPS_COLDRESET#	Output	Active-Low	MIPS ColdReset output.
MIPS_EOK#	Output	Active-Low	MIPS EOK output. Signals that the external agent (the 7811) is capable of accepting a processor request.
MIPS_EOK_EXT#	Input	Active-Low	MIPS external EOK input. EOK signal for external (non-7811) accesses.
MIPS_EREQ#	Output	Active-Low	MIPS EReq output. Asserted when the 7811 is requesting the MIPS interface bus.

Signal	Type	Polarity	Description
MIPS_EVALID#	Output	Active-Low	MIPS EValid output. Asserted when the 7811 is driving a valid address or valid data on the MIPS_SYSAD[31:0] bus
MIPS_INT[2:0]#	Output	Active-Low	MIPS interrupt bus. Three independent interrupt request pins.
MIPS_PMASTER#	Input	Active-Low	MIPS PMaster input. Asserted when the MIPS processor is the master on the MIPS bus.
MIPS_PREQ#	Input	Active-Low	MIPS PReq. Asserted when the MIPS processor is requesting the MIPS bus.
MIPS_PVALID#	Input	Active-Low	MIPS PValid input. Asserted when the MIPS processor is driving a valid address or valid data onto the MIPS_SYSAD[31:0] bus.
MIPS_RESET#	Output	Active-Low	MIPS reset output. Initiates a soft reset of the MIPS processor.
GPRAM Signals			
PADDR[12:0]	Output	-	GPRAM multiplexed row/column address output bus.
PBA[1:0]	Output	-	GPRAM bank-select outputs.
PCAS#	Output	Active-Low	GPRAM column-address strobe output.
PCLK	Output	Rising Edge	GPRAM clock output.
PCS#	Output	Active-Low	GPRAM chip-select output
PDATA[15:0]	I/O	-	GPRAM data I/O bus.
PDQMH	Output	Active-High	GPRAM data mask output for bits 15:8.
PDQML	Output	Active-High	GPRAM data mask output for bits 7:0.
PRAS#	Output	Active-Low	GPRAM row-address strobe.
PWE#	Output	Active-Low	GPRAM write enable.
PCI Signals			
PCI_AD[31:0]	I/O	-	PCI address/data bus.
PCI_CBE[3:0]#	I/O	Active-Low	PCI command/byte-enable bits.
PCI_CLK	Input	Rising Edge	PCI clock. This clock is completely asynchronous to MIPS_CLK.
PCI_DEVSEL#	I/O	Active-Low	PCI device select.
PCI_FRAME#	I/O	Active-Low	PCI cycle frame.
PCI_GNT#	Input	Active-Low	PCI bus grant.
PCI_IDSEL#	Input	Active-Low	PCI initialization device select.
PCI_INTA#	Open-Drain Output	Active-Low	PCI interrupt request.
PCI_IRDY#	I/O	Active-Low	PCI initiator ready.
PCI_LOCK#	I/O	Active-Low	PCI bus lock.
PCI_PAR	I/O	Active-High	PCI bus parity

Signal	Type	Polarity	Description
PCI_PERR#	I/O	Active-Low	PCI bus parity error.
PCI_REQ#	Output	Active-Low	PCI bus request.
PCI_RST#	Input	Active-Low	PCI reset.
PCI_STOP#	I/O	Active-Low	PCI transfer stop.
PCI_TRDY#	I/O	Active-Low	PCI target ready.
PLL Signals			
PLL_AGD	Ground	-	PLL analog ground. Should be isolated from digital ground.
PLL_DGN	Ground	-	PLL digital ground. Should be connected directly to VSS.
PLL_DVD	Power	-	PLL digital VDD. Should be connected directly to VDD.
PLL_VAA	Power	-	PLL analog VDD. Should be isolated from digital VDD.
Test Signals			
TEST_PLLIN0	Input	-	Reserved. Must be pulled down to VSS through a 1 KΩ resistor
TEST_PLLIN1	Input	-	Reserved. Must be pulled down to VSS through a 1 KΩ resistor
TEST_PLLOUT0	Output	-	PLL lock detect.
TEST_PLLOUT1	Output		Test output. Maybe left unconnected or brought to a test point
TEST_PLLOUT2	NC		No function currently
TEST[2:0]	Input		Reserved. Must be pulled down to VSS through a 1 KΩ resistor. In low power mode pin TEST[1] must be pulled up to VDD. See section 1.4
TEST_CLK_[A:D]	Input	-	Reserved. Must be pulled down to VSS through a 1 KΩ resistor.
TEST_IN[3:0]	Input	-	Reserved. Must be pulled down to VSS through a 1 KΩ resistor.
TEST_OUT[3:0]	Output	-	Reserved. Must be left unconnected.
TEST_SCAN	Input		Reserved. Must be pulled down to VSS through a 1 KΩ resistor.
Miscellaneous Signals			
LED_OUT[2:0]#	Output	Active-Low	General purpose outputs, used to drive LED's on the 7811 Evaluation Board. Driven by the LED[2:0] field of the MIPS Reset register.
MODE[2:1]	Input	-	Operating mode: 00 = reserved, 01=reserved, 10=full function, 11=function depends on software unlock codes (used for U.S. export control; see your Hi/fn representative). Use a 1 KΩ resistor to pull up or down.
NC	NC	-	No connection. Must be left unconnected in 7811 designs.
PROTECT#	Input	Active-Low	Protected/unprotected mode select. Should be tied high or low; the behavior of the 7811 is not guaranteed if PROTECT# changes state during operation.
VDD	Power	-	3.3V power supply. All VDD pins must be tied together.
MIPS#	Input	Active-Low	Pulled down for 43xx Series. Pulled up for 46xx Series
VSS	Ground	-	Ground. All VSS pins must be tied together.

Figure 13. Signal Description

SIGNAL	Connection if MIPS is not Populated
MIPS_BIGENDIAN#	VDD
MIPS_CLK#	System Clock
MIPS_COLDRESET#	NC
MIPS_EOK#	NC
MIPS_EOK_EXT#	VDD
MIPS_EREQ#	NC
MIPS_EVALID#	NC
MIPS_INT[2:0]#	NC
MIPS_PMASTER#	VDD
MIPS_PREQ#	VDD
MIPS_PVALID#	VDD
MIPS_RESET#	NC
MIPS_SYSAD[31:0]	VDD
MIPS_SYSCMD[8:0]	VDD

Figure 14. Recommended terminations if MIPS processor is not used

5 Memory Maps

The 7811 has several memory spaces:

- The PCI memory space, as seen by the host.
- The MIPS processor memory space, as seen by the MIPS processor.
- The Group 0 register space.
- The Group 1 register space.
- GPRAM.
- Context SDRAM.
- EEPROM.

These spaces interact with each other in different ways:

- PCI is accessible from the MIPS interface and the 7811. The 7811 can act as both an initiator and a target of PCI transfers. The 7811 also functions as a bridge between the MIPS interface and PCI, allowing the MIPS processor to initiate PCI transfers.
- The MIPS bus is controlled by the MIPS processor. Because the 7811 is a MIPS slave, the 7811 cannot initiate transfers on the MIPS bus. This means that the 7811 cannot act as a bridge for host-initiated transfers between PCI and the MIPS bus. This feature helps maintain the security boundary between the security subsystem and the host.
- The Group 0 and Group 1 registers are accessible from the MIPS interface at all times. The two register groups are fully accessible from PCI if the 7811 is not in protected mode. In protected mode, all the registers that might compromise sensitive information are inaccessible over PCI. Others remain accessible.
- GPRAM is mapped to both the PCI and MIPS address spaces. Its starting address in PCI is given in BAR 2, while its starting address in the MIPS address space is given by the MIPS SDRAM1 Address and MIPS SDRAM2 Address registers.
- Context SDRAM can be read and written only by the Security Engine. Context SDRAM is accessed automatically during security processing by the Security Engine. Block transfers to and from Context memory are also supported as Security Engine commands (Context Memory Read and Context Memory Write). Since the Security Engine is inaccessible to the PCI host in protected mode, this feature does not compromise security.
- The EEPROM is loaded automatically into the 7811 on reset, but is otherwise inaccessible.
- The GPDMA units can operate in either the MIPS or PCI address space. Because the 7811 is a MIPS slave, the MIPS address space is useful only when accessing GPRAM.

5.1 PCI Memory Map

The 7811 uses five regions of PCI memory. Three are allocated for PCI target cycles through the Base Address Registers. BAR0 points to the 7811's Group 0 registers, BAR1 points to the Group 1 registers, and BAR2 points to the 7811's GPRAM. The PCI addresses of these three regions are allocated by the PCI host.

The remaining two regions are used by the 7811 as the PCI bus master. These regions are used by the MIPS processor to access PCI (and, as a special case, by the GPDMA units when accessing a MIPS address that happens to be in one of these two regions). These are the MIPS PCI1 Translation and MIPS PCI2 Translation registers in the 7811. These registers give the starting addresses of two 32 MB target regions in PCI memory space. Reads and writes by the MIPS processor to the PCI1 and PCI2 regions in the MIPS memory space are translated into reads and writes in the PCI1 and PCI2 regions in the PCI memory space.

In secure modes, access to the Group 0 and Group 1 registers from PCI would be disabled. Also, part of the GPRAM would be made inaccessible. SDRAM is protected through two registers: the Security Engine DMA Configuration register and the PCI Address "OR" Mask register.

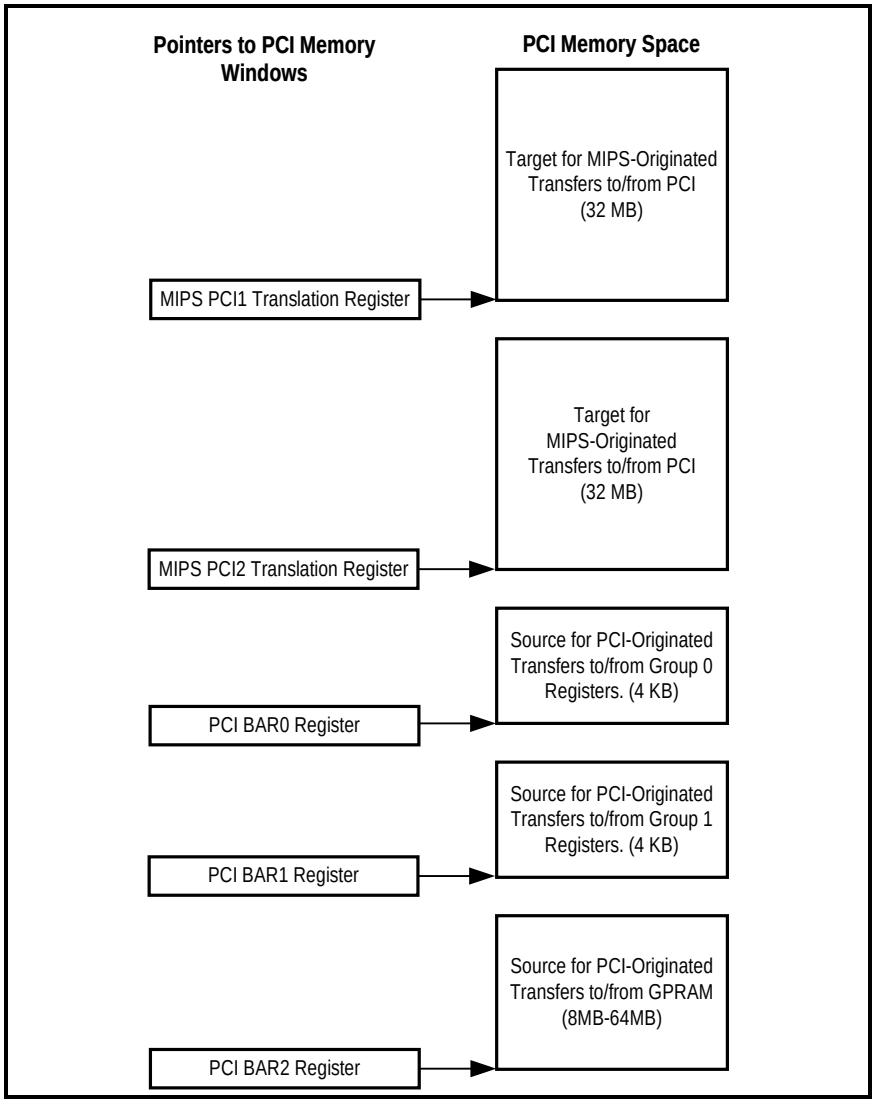


Figure 15. Usage of PCI memory space by the 7811.

5.2 MIPS Memory Map

Six 7811 registers map regions of PCI and 7811 space into the MIPS memory space, as shown in Figure 17. These regions can go anywhere in the MIPS memory space, and are generally mapped by the MIPS processor itself. Bootstrapping means that complete generality is not quite possible – the Group 1 registers must be accessible for the MIPS processor to set the starting addresses for the individual regions. The default address of the Group 1 registers is loaded from EEPROM.

Two PCI regions are mapped into the MIPS memory space, allowing the MIPS processor to issue PCI reads and writes. This is shown in Figure 16.

While the regions correspond closely to the PCI mappings, the sizes of several regions are larger (merely for convenience in hardware implementation), and the GPRAM is mapped twice.

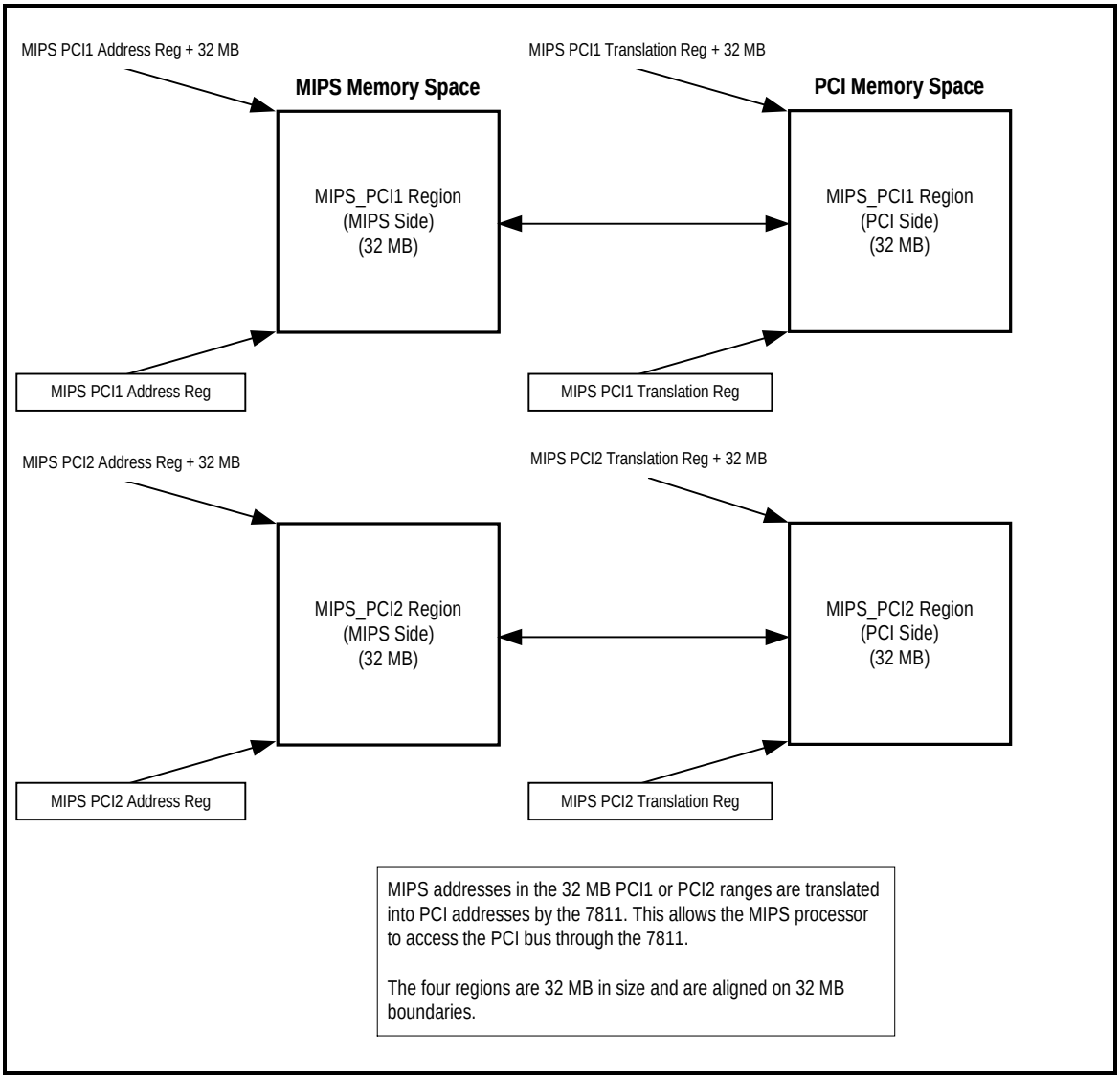


Figure 16. MIPS-to-PCI address decoding

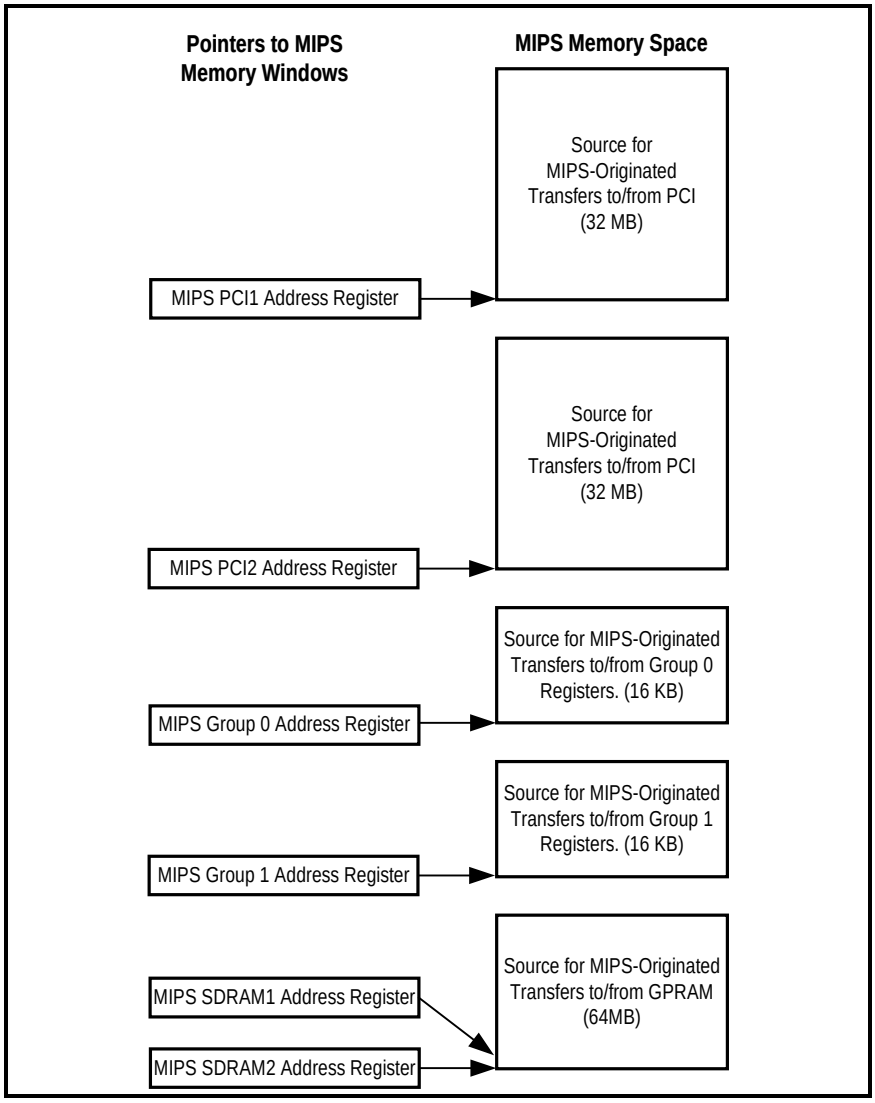


Figure 17. Usage of MIPS memory space by the 7811

5.3 EEPROM Memory Map

The 7811 uses an external 93C46 64x16-bit EEPROM to store initialization values for the PCI interface, the SDRAM interface, and the Security Engine. The EEPROM contents are defined in Figure 18. The functionality of the Security Engine may be determined by the use of "unlock codes" in address range 0x13-0x22. Hi/fn can supply information on setting the unlock codes to support differing levels of encryption functionality. This allows customers to have a single design that can be either exportable or non-exportable under U.S. regulations. If the MODE[2:1] bits are 0b10, the unlock codes are ignored, and the chip is fully functional. If they are 0b11, the functionality of the chip is determined by the unlock codes.

Address	Bit Field	Name	Default	PCI Config. Address
00	15:0	*PCI Device ID.*	0x0007	0x02-0x03
01	15:0	*PCI Vendor ID*	0x13A3	0x00-0x01
02	15:0	*PCI Class Code[23:8]*	0x0B40	0x0B-0x0A
03	15:8	*PCI Class Code[7:0]*	0x00	0x09
	7:0	*PCI Revision ID*	0x01	0x08
04	15:8	*PCI BIST*	0x00	0x0F
	7:0	*PCI Header Type*	0x00	0x0E
05	15:0	*PCI Subsystem ID*	0x0000	0x2E-0x2F
06	15:0	*PCI Subsystem Vendor ID*	0x0000	0x2C-0x2D
07	15:8	*PCI Max_Lat*	0x00	0x3F
	7:0	*PCI Min_Gnt*	0x00	0x3E
08	15:8	*PCI Interrupt Pin*	0x01	0x3D
	7:0	*Reserved:* must be set to zero	0x00	
09:0F	15:0	*Reserved:* must be set to zero	0x00	
10		*Context SDRAM Config*		
	15:11	*Reserved:* must be set to zero.		
	10:9	*Column bits:* 00 = 8, 01 = 9, 10 = 10, 11 = reserved		
	8	T_{RP}:(RAS Precharge time)0 = 2 cycles, 1 = 3 cycles		
	7	T_{RCD}:(RAS activate to command delay) 0 = 2 cycles, 1 = 3 cycles		
	6	T_{RC}:(RAS cycle time) 0 = 7 cycles, 1 = 8 cycles		
	5:3	*CAS latency*: 010 = 2 cycles, 011 = 3 cycles		
	2:0	*Refresh*: 000 = 100 MHz, 001 = 80 MHz, 010 = 66 MHz, 011 = 10 MHz, 1xx = reserved		
11		*GPRAM Config*		
	15:11	*Reserved:* must be set to zero.		
	10:9	*Column bits*: 00 = 8, 01 = 9, 10 = 10, 11 = reserved		
	8	T_{RP}: 0 = 2 cycles, 1 = 3 cycles		
	7	T_{RCD}: 0 = 2 cycles, 1 = 3 cycles		
	6	T_{RC}: 0 = 7 cycles, 1 = 8 cycles		
	5:3	*CAS latency*: 010 = 2 cycles, 011 = 3 cycles		
	2:0	*Refresh:* 000 = 100 MHz, 001 = 80 MHz, 010 = 66 MHz, 011 = 10 MHz, 1xx = reserved		
12	15:2	*Reserved:* must be set to zero.		
	1:0	*GPRAM Memory PCI Window Size (allocated to BAR2):* 00=8 MB, 01=16 MB, 10=32 MB, 11=64 MB		
13-22	15:0	*Unlock Variables*		
23	15:0	*MIPS Group 1 Address Register [31:16]*		
24	15:0	*MIPS SDRAM1 Address Register [31:16]*		
25	15:0	*EEPROM Data Register[15:0]*		
26	15:0	*EEPROM Data Register[31:16]*		

Figure 18. EEPROM memory map

5.4 PCI Configuration Space

The 7811 has a bus-mastering PCI interface that supports fast back-to-back transfers. It uses a single PCI interrupt and three base address registers.

BAR0 is a pointer to the Group 0 register space; BAR1 is a pointer to the Group 1 register space, and BAR2 is a pointer to GPRAM. BAR0 and BAR1 are

4 KB in size, while BAR2 is specified by EEPROM word 0x12, and varies between 8 and 64 MB. Registers take precedence over SDRAM, so BAR0 and BAR1 can overlap BAR2.

The PCI configuration space header is shown in Figure 19.

Field	Address	Default Value	Read/Write Access	Loaded from EEPROM?
Device ID	02-03	0x0007	Read	Yes
Vendor ID	00-01	0x13A3	Read	Yes
Status	06-07	0x0280	Read	No
Command	04-05	0x0000	R/W	No
Class Code	09-0B	0x0B4000	Read	Yes
Revision ID	08	0x01	Read	Yes
BIST	0F	0x00	Read	Yes
Header Type	0E	0x00	Read	Yes
Master Lat Timer	0D	0x00	R/W	No
Cacheline Size	0C	0x00	R/W	No
BAR0	10-13	0x000000	R/W	No
BAR1	14-17	0x000000	R/W	No
BAR2	18-1B	0x000000	R/W	No
Subsystem ID	2E-2F	0x0000	Read	Yes
Subsys Vendor ID	2C-2D	0x0000	Read	Yes
Max_Lat	3F	0x00	Read	Yes
Min_Gnt	3E	0x00	Read	Yes
Interrupt Pin	3D	0x01	Read	Yes
Interrupt Line	3C	0x00	R/W	No
Retry Timeout	41	0x80	R/W	No
TRDY Timeout	40	0x00	R/W	No

Figure 19. PCI configuration space

Bit	Description	Default	Type
15	Detect Parity Error	0	Status
14	Signaled System Error	0	Status
13	Received Master Abort Status	0	Status
12	Received Target Abort Status	0	Status
11	Signaled Target Abort Status	0	Status
10:9	Device Select Timing	01	Read Only
8	Data Parity Detected	0	Status
7	Fast Back-to-Back Capable Status Flag	1	Read Only
6	RESERVED	0	Read Only
5	66 MHz-Capable Status Flag	0	Status
4:0	RESERVED	0	Read Only

Figure 20. PCI Status Register

6 General Purpose DMA (GPDMA) Units

The 7811 has four general purpose DMA (GPDMA) units, named GPDMA1 through GPDMA4. These are efficient burst-oriented DMA units. They have a flexible command structure that uses command lists in memory. The GPDMA units are used to transfer incoming commands and data from PCI to GPRAM, and outgoing status information and data from GPRAM to PCI. They can also be used as efficient GPRAM-to-GPRAM and PCI-to-PCI transfers.

A typical use of the GPDMA units is shown in Figure 21. In this example, GPDMA1 is used to retrieve the command stream from the PCI host and store them in GPRAM. The MIPS processor interprets the command stream and uses GPDMA2 to fetch the incoming data packets, storing them in GPRAM. The MIPS processor then performs header processing, handing off the security processing to the Security Engine, which uses its own dedicated DMA engines for input and output. Once the Security Engine has finished, the MIPS processor assembles the output packet and uses GPDMA3 to transfer the output data back to the PCI host. GPDMA4 transfers status information.

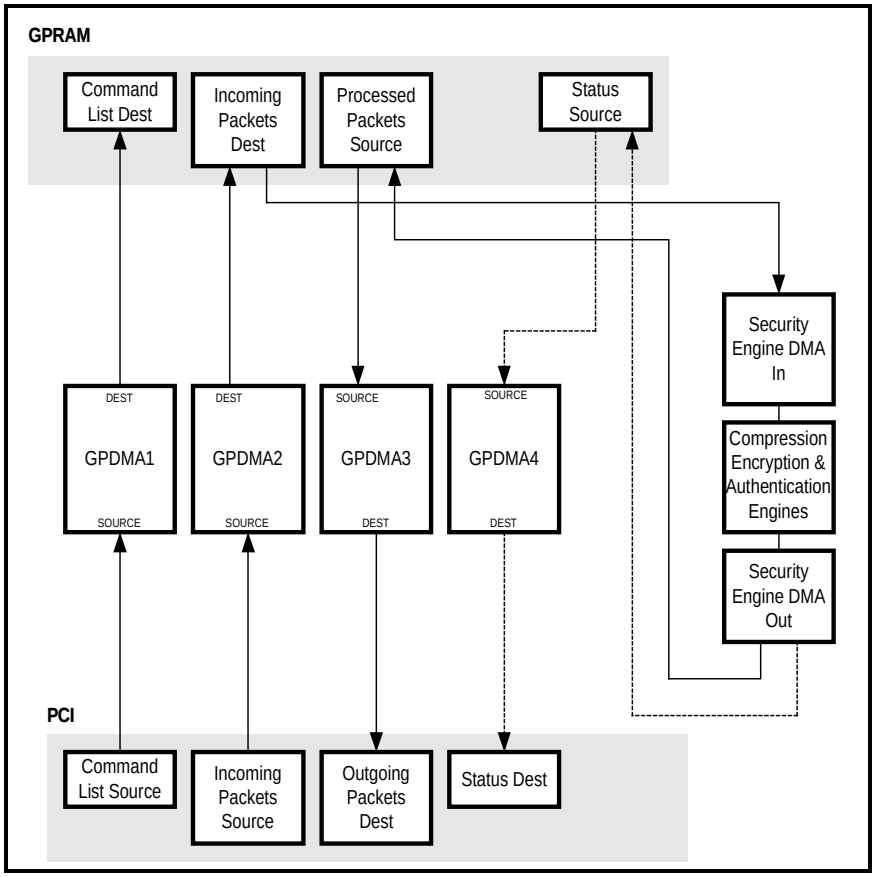


Figure 21. Application-level GPDMA example

The GPDMA units are controlled by the GPDMA registers and a list of GPDMA commands. A GPDMA command is a 16-byte data structure. Each GPDMA unit has a *Source Command Pointer register* and a *Destination Command Pointer register.* (See section 8.2 for register descriptions.) The Source Command Pointer register points to a list of GPDMA commands defining the data to be copied, while the Destination Command Pointer register points to a list of

GPDMA commands defining where the data is to go. DMA scatter/gather is supported.

The GPDMA command consists of four 32-bit dwords, aligned on 32-bit boundaries. The function of these words varies according to mode.

6.1 Data and Message Modes

The GPDMA units work in one of two modes: *Data mode* and *Message mode.*

In Data mode, the first eight bytes of the Source command are transferred to the Destination. In addition, the third 32-bit word of the Source command is a pointer to a data buffer, and the fourth 32-bit word contains the length of this buffer. The data buffer is transferred to the Destination as part of the command. Data buffers can be aligned to any byte boundary. They must be at least one byte long.

In Message mode, fifteen of the sixteen command bytes are transferred. The upper byte (bits [31:24]) of the Source Control Word is not transferred. This byte contains the Valid bit and other Destination-specific state information.

Software uses Data mode for transferring packet streams, using the eight bytes of parameter information to identify the session number and other necessary information to identify to software what is to be done with the data. Message mode is used for control streams between the host and the MIPS processor.

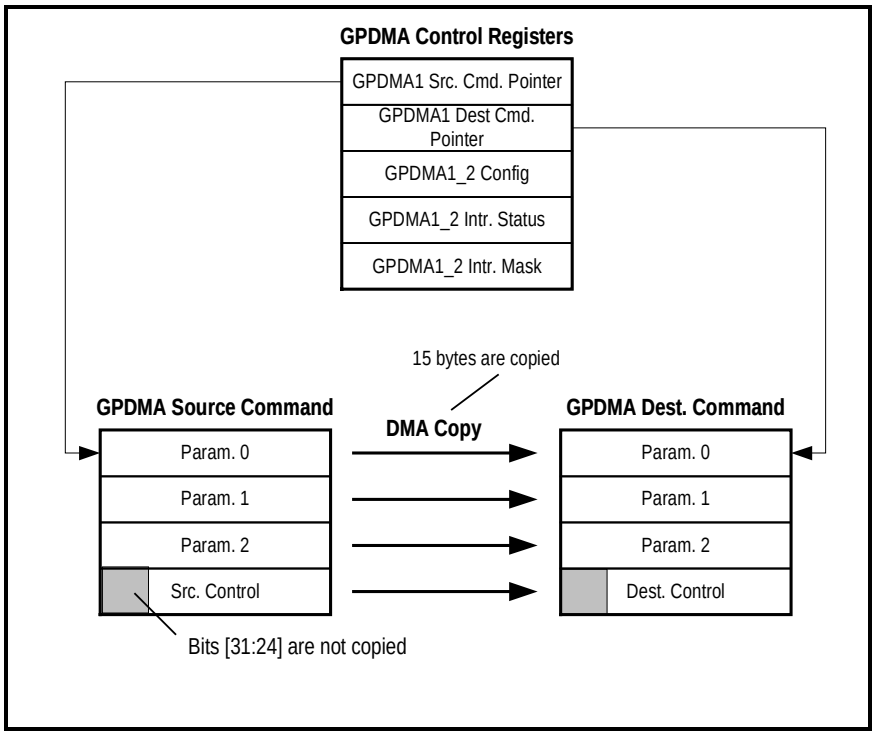


Figure 22. GPDMA operation in Message mode

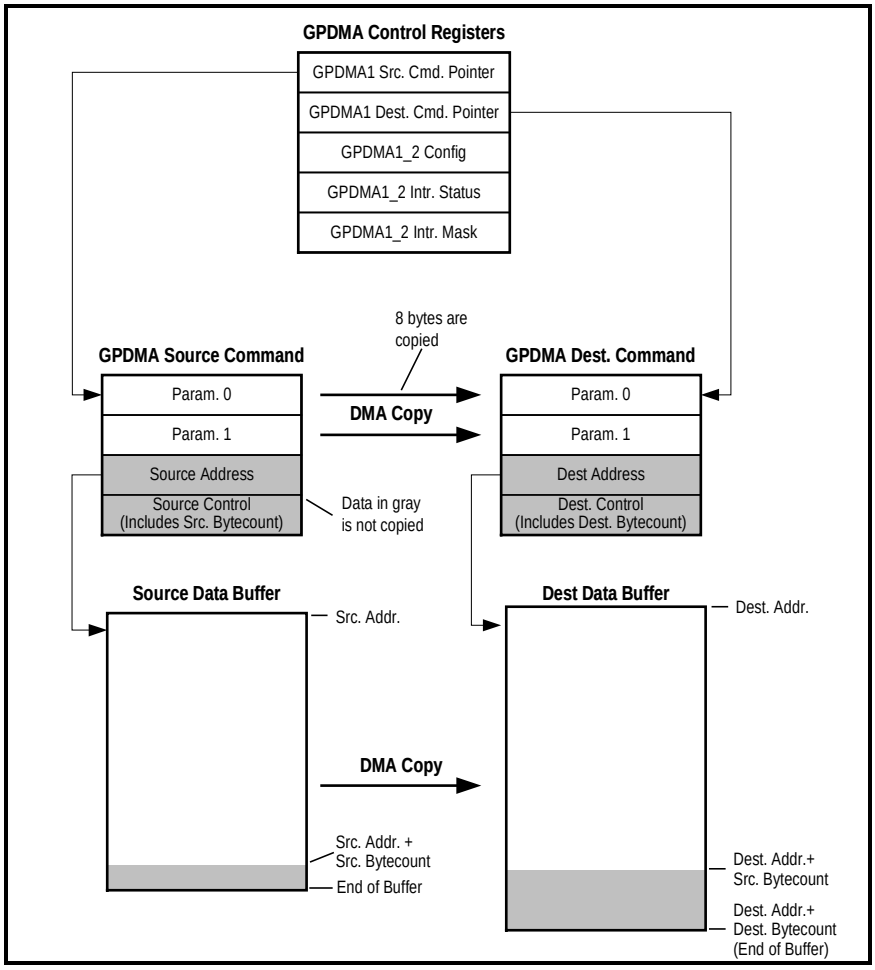


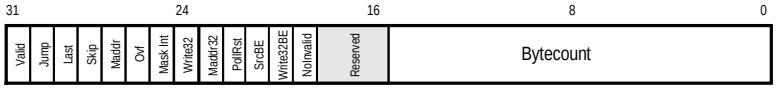
Figure 23. GPDMA operation in Data mode

Word 0	Source Parameter 0
Word 1	Source Parameter 1
Word 2	Source Parameter 2 or Source Data Address
Word 3	Source Control

Field	Description
Source Param 0	*Parameter 0.* Transferred to the Dest command structure. If the Write32 bit is set, the 32 bits of Src Param 1 are written to the dword-aligned 32-bit address given in Src Param 0. That is, mem[Src Param 0] := Src Param 1.
Source Param 1	*Parameter 1*. Transferred to the Dest command structure. If the Write32 bit is set, this field provides the data for a 32-bit write to the address in Src Param 0. That is, mem[Src Param 0] := Src Param 1.
Source Param 2	*If the Jump bit is one,* this field is a 32-bit pointer to the next GPDMA command *If the Jump bit is zero:* If the GPDMA unit is in Message mode, this is Source Parameter 2. In Data mode, this field is a 32-bit pointer to the data to be transferred.
Source Control	*GPDMA unit source control parameters*. This 32-bit word is divided into control fields, shown in Figure 25

Figure 24. GPDMA Source command structure

Src Control



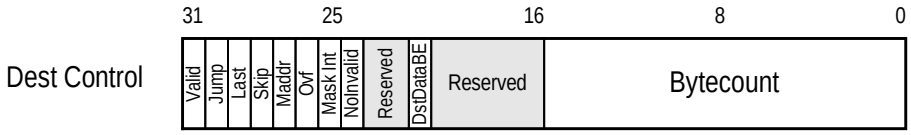
Field	Bits	Description
Valid	31	If this bit is set, the GPDMA unit assumes that the command structure is valid. Data transfers will begin after both the Source and Destination Valid bits are set. After the transfer has finished, both valid bits are cleared by the GPDMA unit.
Jump	30	If this bit is set, Src Param 2 is used as a 32-bit pointer to the next GPDMA command. If this bit is zero, the next GPDMA command is assumed to being immediately following the current command.
Last	29	In Data mode, this bit is set on the last fragment of the current data stream. In Message mode, this bit is copied from the Source command.
Skip	28	If set, the next GPDMA structure in the list is skipped. Not valid if the Jump bit is set.
Maddr	27	*MIPS address.* If set, the address in Src Data Address is considered to be a MIPS addresses. Otherwise, it is considered to be a PCI addresses. (The 7811 maintains two address maps, one for the MIPS processor and one for PCI.)
Ovf	26	*Overflow.* Set by the 7811 if the Dest buffer overflowed on the transfer.
Mask Int	25	*Mask Done Interrupt.* If set, the Done interrupt will be suppressed. If zero, the Done interrupt behaves as determined by the GPDMA Interrupt Enable register.
Write32	24	*Write 32-bit word after transfer.* This feature allows a word to be written to a specified address after the transfer is complete. This feature must be enabled in the appropriate Config register (DMA1_2 Config or DMA3_4 Config before use, or the Write32 bit will be ignored. If this condition is met, the 32-bit word in Src Param 1 will be written to the 32-bit address given in Src Param 0. That is, mem[Src Param 0] := Src Param 1. This write occurs after the DMA transfer is complete. This allows user-defined semaphores to be passed to arbitrary dword addresses. Not valid if the Jump bit is set.
Maddr32	23	If set, the address in Src Param 0 is considered to be a MIPS address. Otherwise, it is considered to be a PCI address.
PollRst	22	*Poll Timer Reset.* In Write32 mode, reset the Security Engine's invalid poll timer at the completion of the command, causing the Security Engine to poll for new commands immediately.
SrcBE	21	*Source Data Big-Endian.* If set to one, the data in the source buffer is big-endian. Setting this bit to 1 however does not affect data stored in the GPRAM. If zero, it is little-endian. Only valid in Data mode; reserved in Message mode.
Write32BE	20	*Write32 Data Big-Endian.* If set to one, the data in Source Parameter 1 is big-endian. Setting this bit to 1 however does not affect data stored in the GPRAM. If zero, it is little-endian. Valid only in Data mode; reserved in Message mode.
NoInvalid	19	*No Invalidation Cycle.* If set to one, the GPDMA unit will not invalidate the Source Command when the command has completed (the Valid bit will not be set to zero; in fact, no write will be made to the Source Command at all). If zero, it will invalidate the command as usual.
Reserved	18:16	Passed to Dest GPDMA command in Message mode. No operation in Data mode.
Bytecount	15:0	If the GPDMA unit is in Data mode, this gives the number of bytes to transfer, starting at the address in Src Param 2. Otherwise, this field is transferred to the Dest command structure but is not interpreted. Bytecount must be nonzero for the command to be executed. That is, in Data mode the GPDMA unit treats Bytecount = 0 as being equivalent to Valid = 0.

Figure 25. Bit fields in the Source Control word

Word 0	Dest. Parameter 0
Word 1	Dest. Parameter 1
Word 2	Dest. Parameter 2 or Dest. Data Address
Word 3	Dest. Control

Field	Description
Dest Param 0	Dest *Parameter 0.* The GPDMA unit copies Source Parameter 0 to Dest Parameter 0.
Dest Param 1	*Dest Parameter 1*. The GPDMA unit copies Source Parameter 1 to Dest Parameter 1.
Dest Param 2	*Dest Parameter 2/Data Address.* If the GPDMA unit is in Message mode, this is a copy of Source Parameter 2. In Data mode, this field is a 32-bit pointer to the Dest data buffer.
Dest Control	GPDMA unit destination control parameters. This 32-bit word is divided into fields, shown in Figure 27.

Figure 26. GPDMA Dest Command structure



Field	Bits	Description
Valid	31	If this bit is set, the GPDMA unit assumes that the command structure is valid. Data transfers will begin after both the Source and Destination Valid bits are set. After the transfer has finished, both valid bits are cleared by the GPDMA unit.
Jump	30	If this bit is set, Src Param 2 is used as a 32-bit pointer to the next GPDMA command. If this bit is zero, the next GPDMA command is assumed to being immediately following the current command.
Last	29	If this bit is set, it indicates that this is the last fragment of the current data stream.
Skip	28	If set, the next GPDMA structure in the list is skipped. Not valid if the Jump bit is set.
Maddr	27	*MIPS address.* If set, the address in Dest Data Address is considered to be MIPS addresses. Otherwise, it are considered to be a PCI addresses. (The 7811 maintains two address maps, one for the MIPS processor and one for PCI.)
Ovf	26	Overflow. If set, the Dest buffer overflowed while this command was being executed.
Mask Int	25	*Mask Done Interrupt.* If set, the Done interrupt will be suppressed. If zero, the Done interrupt behaves as determined by the GPDMA Interrupt Enable register.
NoInvalid	24	*No Invalidation Cycle*. If this bit is set to one, disable the invalidation cycle at the end of the command. The Valid bit will not be cleared; in fact, no write to the command will take place. If this bit is set to zero, invalidation takes place normally.
Reserved	23:22	Passed from the GPDMA Source command structure in Message mode. Reserved in Data mode.
DestDataBE	21	*Dest Data Big-Endian.* If this bit is set, the Dest Data Buffer is in big-endian format. Setting this bit to 1 however does not affect data stored in the GPRAM. If zero, the data is little-endian.
Reserved	20:16	Passed from the GPDMA Source command structure in Message mode. Reserved in Data mode.
Bytecount	15:0	In Data mode, this field must be initialized to the maximum size of the data buffer, in bytes. When the transfer is complete, the GPDMA unit overwrites this field with the size of the data transferred, in bytes. Bytecount must be nonzero for the command to be executed. That is, in Data mode the GPDMA unit treats Bytecount = 0 as being equivalent to Valid = 0 In Message mode, this field is a copy of the corresponding Source Control bits.

Figure 27. Bit fields in the Dest Control word

6.2 Handshaking

The Valid bit in the fourth dword provides handshaking for the transfer. The GPDMA unit waits for the Valid bits to be set on both the Source and Destination command structures. Once both bits are set, the GPDMA unit copies the Source command structure to the Destination, overwriting part of the Destination command structure. If the GPDMA unit is in Data mode, the GPDMA unit copies the data pointed to by the address pointer in the source command structure. It then writes it to the buffer pointed to by the address pointer in the destination command structure.

6.3 Scatter/Gather

The GPDMA units support scatter/gather DMA:

- Fragmented source data can be gathered and written to a single Dest buffer.
- Fragmented source data can be gathered and scattered to multiple Dest buffers.
- Non-fragmented source data can be written to a single Dest buffer.
- Non-fragmented source data can be scattered to multiple Dest buffers.

Scatter/gather behavior is controlled by the Last command bit, which is part of every command's Control Word, and by the Software Last configuration bits in the GPDMA1_2 and GPDMA3_4 Config registers (see Figure 99 and Figure 111). Each GPDMA unit has its own Software Last configuration bit.

6.3.1 Last Command Bit

The Last bit indicates that the current command is the last command of the data transfer. On Source commands, the Last bit always indicates the final fragment of a data transfer. For example, a transfer that is split into two fragments would be transferred with two Source commands. The Last bit would be zero on the first command, and one on the second command. When transferring non-fragmented data, the Last bit is set on every Source command.

On the Dest side, usage of the Last bit can be somewhat more complicated, depending on the hardware and software requirements of the host system and the 7811 software running on the MIPS processor. If fragmentation can be avoided on the Dest side (by allocating non-fragmented buffers that can contain the largest packet), then the Dest side is simple. When a Source command has the Last bit set, the Dest side will finish its current command and advance to its next command. When a Source data stream is fragmented, as indicated by a series of commands with the Last bit clear, the Dest side treats this as a single command, putting the data into a single buffer and advancing to the next command after finishing the last Source command.

It's important to realize that there is no one-to-one command correspondence. A stream of ten Source commands might use only one Dest command. With fragmentation on the Dest side, a single Source command might require several Dest commands. The configuration of one side is not visible to the other; that is, one does not have to know how the Source side is set up in order to set up the Dest side, or vice versa. Configuration revolves around satisfying the local fragmentation requirements of the host side for both incoming and outgoing data, and also for the MIPS processor.

6.3.2 Software Last Register Bit

If the SW_Last (Software Last) bit in the GPDMA Config register is zero (hardware-controlled Last), the Dest side operates as follows: If the current Dest buffer overflows, the GPDMA unit clears the Valid and Last bits in the current Dest command. It then advances to the next Dest command and begins writing to this new command's data buffer. This continues until a Source command is encountered with its Last bit set. When the final data is written, the final Dest buffer has its Valid bit reset and its Last bit set. The GPDMA unit ignores the initial state of the Dest-side Last bits. In other words, the GPDMA unit uses as many Dest command and Dest data buffers as necessary to store the Source data stream up until a Source Last bit is seen.

If the SW_Last bit is set to one (software-controlled Last), the Dest side uses a more rigid command allocation. This is useful in Windows NT, for example, where the OS might respond to a request to allocate a 1500-byte buffer by returning a fragmented buffer with 1024 bytes in the first fragment and 476 in the second. As far as Windows NT is concerned, this is one buffer, and the 7811 has to treat it as such when writing data.

The mechanism is to use one Dest command per buffer fragment, with the Last bit set on the final fragment, and the bytecount fields set to the sizes dictated by the OS. The GPDMA unit will write the first 1024 bytes to the first data buffer, then write the last 476 bytes to the second buffer.

Because the commands are tied closely to individual buffers, the GPDMA unit cannot advance to a third command if the second buffer overflows (for instance, if the Source data had 1520 bytes when only 1500 were allocated in the two Dest commands). Instead, the GPDMA unit discards any leftover data and sets the Overflow bit in both the current Source and Dest commands (and optionally asserts an interrupt). Processing continues with the next command.

The contents of the Dest buffer are undefined, though it is guaranteed that the GPDMA unit will not write beyond the end of the buffer (it may not write anything at all).

6.4 Data Mode Example

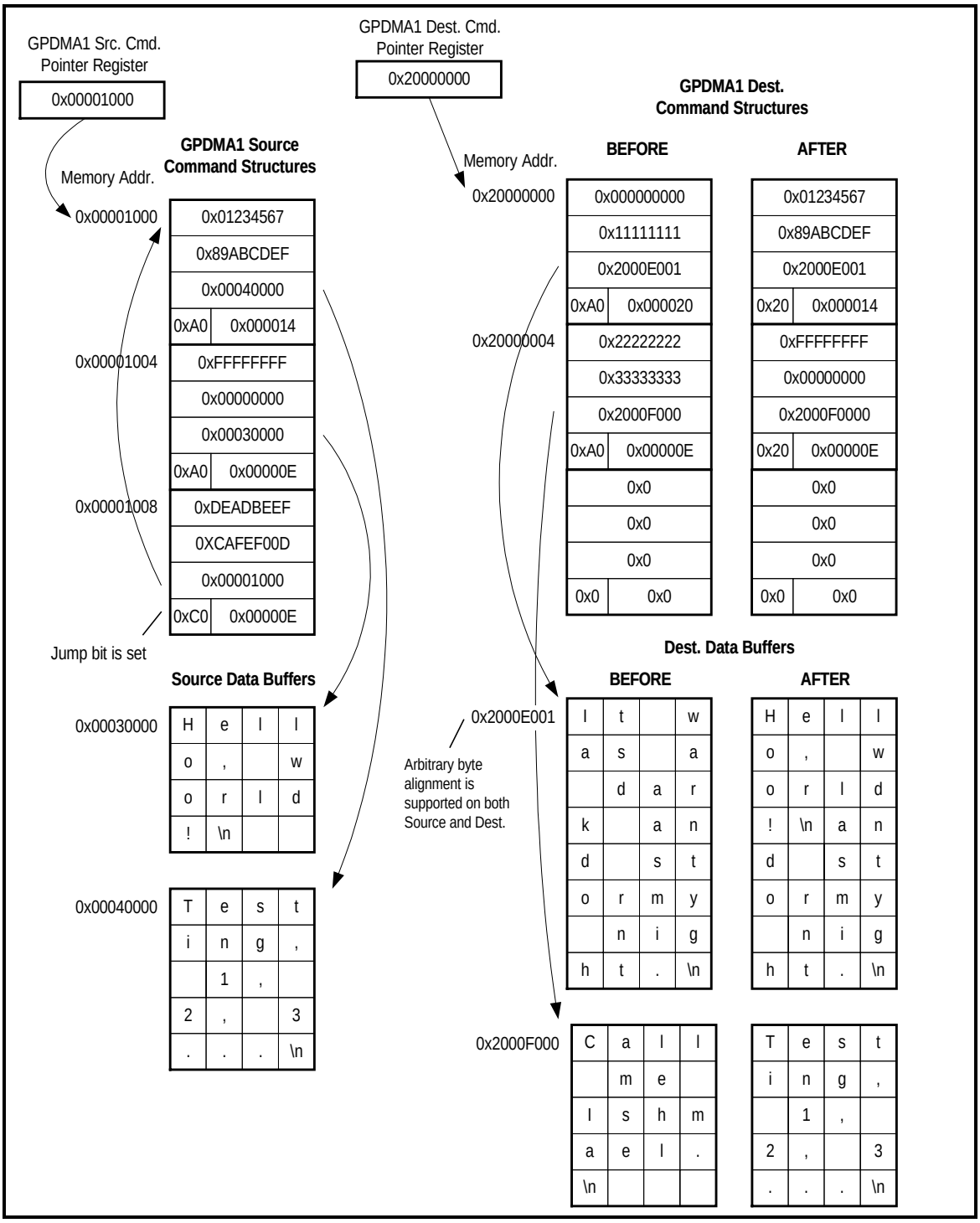


Figure 28. GPDMA example

Figure 28 gives an example of a GPDMA unit at work. For simplicity, this example involves no fragmentation. The GPDMA1 Source Command Pointer register points to a list of three GPDMA source commands. The first two

commands specify data buffers for the unit to transfer. The last command has its Jump bit set, indicating that it contains an address pointing to a non-contiguous command. In this case, it points to the starting address of the first buffer. This type of buffer arrangement is called a *command ring*.

The GPDMA Dest command pointer register points to a list of destination commands. In this case, there are also three commands, the first two of which define data buffers. These commands are not organized into a ring. The third command is a dummy command with its Valid bit set to zero. The GPDMA will process the first two Dest commands and then wait for the third command's Valid bit to be set. Software on the Dest side will presumably write a new address to the Dest Command Pointer once it has processed the first two buffers.

While both sides have the same number of data buffers, this is not true in general. In many cases, one side of the transfer will be managed by the host, while the other will be managed by the MIPS CPU. Neither will be aware of how many data buffers have been allocated by the other, or how many commands are in the command ring on the other side.

For the purposes of this example, we will assume that the host is creating the Source commands and data, which are in host memory, while the MIPS processor is managing the Dest commands and data in GPRAM.

The sequence of operations is as follows:

1. On the host (Source) side, the host software initializes its command ring. The most important operations are the creation of the third command, which contains the jump back to the first command, and writing zeroes to the Valid bits of the first two commands. The host writes the address of the first command to the GPDMA1 Source Command Pointer register.
2. On the 7811 (Destination) side, the MIPS processor initializes the Dest command list and writes the address of the first command to the GPDMA1 Dest command pointer register. Since the data buffers pointed to by the commands are not being used, the commands' Valid bits can be set.
3. Once the command structures and address registers are initialized, it is safe to start the GPDMA unit (in this example, we use GPDMA1). The MIPS processor sets the GPDMA1 Active bit in the GPDMA1_2 Config register. (In unprotected mode, the host can set the bit, but it is only accessible to the MIPS processor in protected mode.)
4. At this point the GPDMA1 unit is completely set up, but the Source commands are dummy commands with their Valid bits clear.
5. When the host has data it wants to send to the 7811, it updates the first command. In the example, there are 0x14 bytes of data at address 0x40000. The host writes the Source Data Address word and the byte count to the Bytecount field.
6. The Source Param 0 and Source Param 1 words are used to pass information to the software that will use the data. For example, Source Param 0 might contain the session number of the data, and Source Param 1 might contain processing parameters. These values are not interpreted by the 7811.
7. The host sets the Valid bit as part of its write to the last word of the command. This activates the command.

8. The GPDMA unit polls the Valid bits at a programmable rate. When it sees that both Valid bits are set, the transfer begins. It transfers the first two words of the command from Source to Destination. It reads the Source Data Address and Source Bytecount parameters and uses them to copy the data buffer from Source to Destination. The Dest Data Address points to the target buffer, and the Dest Bytecount parameter gives the maximum size of the buffer.
9. When the copy is complete, the GPDMA unit zeroes the Valid bits on both Source and Destination and updates the Dest Bytecount to reflect the number of bytes transferred. The Dest Overflow bit is set if a buffer overflow took place.
10. Once the transfer is complete, the GPDMA unit executes the next command. Commands are assumed to be arranged linearly in memory unless a Jump command is encountered. In that case the next command is at the Jump address, or if the Skip bit is set on the current command, which causes the next command to be ignored.

In Message mode, operation is very similar, except that the Source/Dest Data Address field becomes Source Param 2. Fifteen of the sixteen command bytes are transferred, but no data buffers are transferred. Control flow is the same as in data mode.

6.5 GPDMA Arbitration and Polling

Each GPDMA unit has a time-slice value that indicates the maximum amount of time that it can remain in control of the bus. This is set in the GPDMA1_2 and GPDMA3_4 Arbitration registers. Possible values range from 0 cycles to 1 M engine cycles, in increments of 16 cycles. When this timeout occurs, the GPDMA unit relinquishes the bus after the completion of its current command.

A GPDMA unit that has not been enabled does not have a time-slice.

Valid bits are tested by polling. Polling takes place during a GPDMA unit's time-slice. The frequency of polling is determined by two values: the Poll Timer and the Invalid Poll Timer. Both time-slice values occur in the GPDMA1_2 and GPDMA3_4 Config registers. Values for the Poll Timer range from 0 to 496 engine cycles. The Poll Timer is used the first time a Source or Dest command is polled. If the Valid bit is set, subsequent polling is set by either the Poll Timer or the Invalid Poll Timer, whichever gives the greatest interval. The Invalid Poll Timer's range is 256-2048 engine cycles. This two-timer mechanism allows a new command to be examined right away, while reducing the polling frequency while waiting for the current command to become valid.

When a GPDMA unit becomes idle, either due to detecting a zero Valid bit or because it is waiting for a polling interval to expire, it relinquishes its time slice.

The Poll Timer is shared between two GPDMA units (GPDMA 1 and 2, GPDMA 3 and 4). The Invalid Poll Timer is specified separately for each GPDMA unit.

7 The Security Engine

The Security Engine is a pipelined compression/encryption unit with its own Source and Destination DMA units. Where there are a few configuration registers, most of the operation of the Security Engine is controlled on a per-command basis through the command stream that is read by the Source DMA unit.

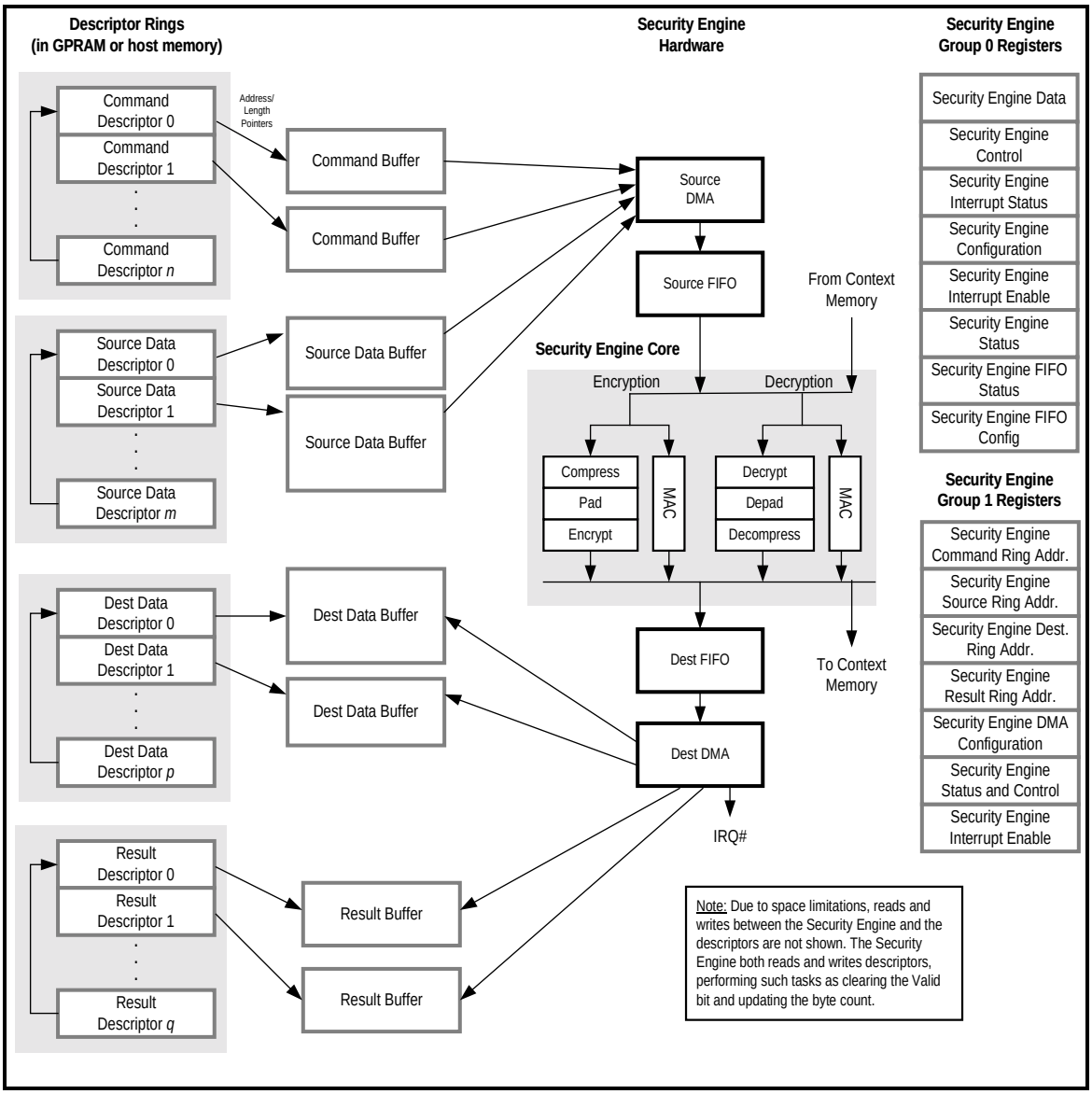


Figure 29. Security engine block diagram

Command and data flow is controlled by four data structures called *descriptors*. A descriptor is analogous to a GPDMA command. Like GPDMA commands, descriptors can be arranged into lists, often with the Jump bit set in the final descriptor in the list. This creates a *descriptor ring.* (The terminology for the Security Engine refers to descriptor lists as “rings,” regardless of whether the last descriptor contains a jump to the first.)

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.

7.1 Operation

In protected mode, the Security Engine is controlled solely by the MIPS processor. Even if hostile software gains control of the host, the host will not be able to control the Security Engine. The PCI host cannot send data to the Security Engine directly. Instead, the host sends a command stream to the MIPS processor using one of the GPDMA units. The MIPS processor services these commands by copying the data stream to GPRAM using a GPDMA unit, then executing the appropriate Security Engine commands. Finally, the MIPS processor sends the results back to the host through another GPDMA unit.

The advantages of this system are:

- A security barrier is made possible.
- Host software is both simpler and more general; the host software doesn't need to deal with the details of the Security Engine hardware.
- Host CPU loading is minimized.
- The number of PCI transfers is minimized, as header processing and multi-pass processing is done from GPRAM, not over PCI.
- Performance is increased.

The Security Engine is controlled by only a few registers: an address register for each of the four descriptor rings, a configuration register, a status/control register, and an interrupt register. Most of the Security Engine's operation is controlled by the descriptors.

DMA operation is similar to that of the GPDMA units. The CPU (either the host processor or the MIPS processor) creates descriptors and data buffers. The Valid bit of the Security Engine descriptors is analogous to those in GPDMA commands, indicating whether the descriptor is ready to use. The Last bit also works in a way similar to that of the GPDMA unit.

While similar to GPDMA commands, Security Engine descriptors have significant differences. Message mode and Write32 mode are not supported. The GPDMA units can be used for any block-transfer task, while the Security Engine descriptors move data only into and out of the Security Engine.

Command setup and execution works as follows:

1. The CPU writes source commands to command buffers and source data to data buffers. Fragmentation is allowed.
2. The CPU initializes command and source data descriptors that point to the command and source data buffers. Fragmentation is handled through the use of multiple command and source descriptors (one per fragment). The Last bit in each descriptor indicates the fragmentation status. If the Last bit is zero, there is an additional fragment following the current descriptor. If the Last bit is one, this is the final fragment. The Valid bit is set on each descriptor as the last step of initialization.
3. The CPU creates "empty" dest data and results descriptors. An empty descriptor has its pointers initialized to point to available buffers and its

bytecount field set to the length of each buffer. Fragmentation is controlled by the Destination Last bit is initialized according to the same rules used in the GPDMA engines. The descriptors have their Valid bits set as the last step in each descriptor's initialization.

4. The four address registers (Security Engine Source Address, Command Address, Result Address, and Destination Address registers) are initialized to point to the first descriptor in each descriptor ring.
5. The Security Engine is enabled by writing 0b10 to each of the four Ring Control fields in the Security Engine Status and Control register. See Figure 69.

Once operation has begun, the Security Engine will use and invalidate descriptors automatically. Because the Security Engine clears the Valid bits on descriptors when it finishes with them, there is no danger of it re-executing commands when it jumps to the first descriptor in the descriptor ring. The CPU re-uses descriptors by reinitializing them and setting the Valid bit as the last step of initialization.

Interrupts can be enabled on a variety of conditions, using the Security Engine Interrupt Enable Register. These interrupts can be attached to either PCI or the MIPS processor, according to the setting of the MIPS Config register.

On the input side, the Security 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.

Note: For historical reasons, Security Engine command and data structures are represented as a series of 16-bit words. The Security Engine's DMA units transfer them as 32-bit dwords, however. The mapping between 16-bit and 32-bit representations of these data structures is as follows:

Figure 30. Mapping between 16-bit and 32-bit representations of the Security Engine's commands and descriptors

7.1.1 Data Alignment

All input to the Security Engine can be aligned arbitrarily on any byte boundary. All output of the Security Engine must be aligned to a 32-bit boundary. This means that the software handling the Security Engine's output must be careful to align result descriptors, result buffers, dest data descriptors, and dest data buffers to 32-bit boundaries.

7.1.2 Address Mapping

The descriptors have a MIPSaddr bit to indicate that the descriptor's address pointer refers to a MIPS address. This allows MIPS-based software to use native MIPS addresses when programming Security Engine commands. Similarly, software running on the PCI host can use PCI addresses.

Because the 7811 is a PCI master, the Security Engine can access memory across the PCI bus in addition to GPRAM. Because the 7811 is a MIPS slave, the Security Engine cannot access memory devices over the MIPS bus.

7.2 Registers

Register	Offset	Reference
Security Engine Data	0x00	Figure 57
Security Engine Control	0x04	Figure 58
Security Engine Interrupt Status	0x08	Figure 59
Security Engine Configuration	0x0C	Figure 60
Security Engine Interrupt Enable	0x10	Figure 61
Security Engine Status	0x14	Figure 62
Security Engine FIFO Status	0x18	Figure 63
Security Engine FIFO Configuration	0x1C	Figure 64

Figure 31. Security Engine registers in the Group 0 register space

Register	Offset	Description	Reference
Security Engine Command Ring Address	0x0C	Address of the next command descriptor.	Figure 65
Security Engine Source Ring Address	0x1C	Address of the next source descriptor.	Figure 66
Security Engine Result Ring Address	0x2C	Address of the next result descriptor.	Figure 67
Security Engine Dest Ring Address	0x3C	Address of the next destination descriptor.	Figure 68
Security Engine Status and Control	0x40	Status and configuration bits for the Security Engine.	Figure 69
Security Engine Interrupt Enable	0x44	Enables interrupts on various conditions.	Figure 70
DMA Configuration	0x48	Configures the Security Engine's DMA units.	Figure 71
Note: For a complete listing of the Group 1 registers, see Figure 7.			

Figure 32. Security Engine registers in the Group 1 register space

7.3 Descriptors

7.3.1 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
Valid	Jump	Last	rsrvd			Mskdne	NoInvld	Add32	MIPSadr	BE	reserved					Command Length															
Command Pointer/Jump Pointer																															

Word	Bit Field	Description
0	31	*Valid.* The software creating the descriptor sets the Valid bit to one to indicate that the command is complete and ready to be processed. The 7811 sets the Valid bit back to zero when it is done with the descriptor. The Security Engine also considers the descriptor to be invalid if the Command Length field is zero and the Jump bit is not set. The Security Engine polls the descriptor periodically to determine if it has become valid.
	30	*Jump.* If set to one, the next word in the descriptor contains the address of the next descriptor. This address is loaded into the Security Engine Command Ring Address
	29	*Last.* If set to one, this descriptor points to the last (or only) fragment of a command.
	28:26	*Reserved.* Must be set to zero.
	25	*Mask Done.* If set to one, inhibit the setting of the Command Done bit in the Security Engine Status/Control register.
	24	*NoInvalid.* The Valid bit will not be set to zero when the command has completed.
	23	*Add32.* Increment the descriptor pointer by 32 bytes (instead of eight bytes).
	22	*MIPSaddr.* The Command/Jump Pointer is a MIPS address.
	21	*BE.* The command buffer is in big-endian format. **
	20:16	*Reserved.* Must be set to zero.
	15:0	*Command length.* Length (in bytes) of the command structure pointed to by the command pointer. The command descriptor is not valid until this field is non-zero, unless the Jump bit is set.
1	31:0	*Command pointer/Jump pointer.* A 32-bit pointer to the command.

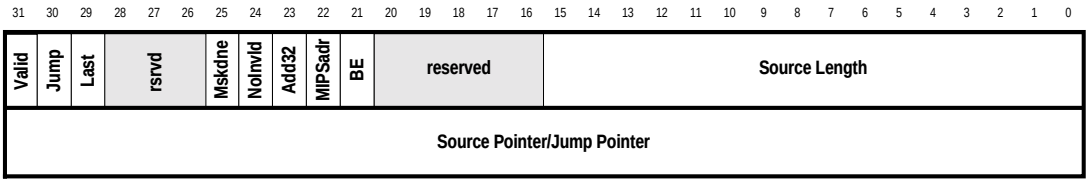
Description

The Command descriptor contains bit fields that control descriptor ring operation, a pointer to a command, and the length of the command. If the Jump bit is set, the pointer is a jump address (pointing to the next descriptor), not a command address.

** Setting this bit to 1 has no effect on data stored in the GPRAM. It only affects data in the PCI memory space.

Figure 33. Command descriptor

7.3.2 Source Descriptor



Word	Bit Field	Description
0	31	*Valid.* The software creating the descriptor sets the Valid bit to one to indicate that the descriptor is complete and ready to be processed. The 7811 sets the Valid bit back to zero when it is done with the descriptor. The Security Engine also considers the descriptor to be invalid if the Source Length field is zero and the Jump bit is not set. The Security Engine polls the descriptor periodically to determine if it has become valid.
	30	*Jump.* If set to one, the next word in the descriptor contains the address of the next descriptor. This address is loaded into the Security Engine Source Ring Address
	29	*Last.* If set to one, this descriptor points to the last (or only) fragment of a source buffer.
	28:26	*Reserved.* Must be set to zero.
	25	*Mask Done.* If set to one, inhibit the setting of the Source Done bit in the Security Engine Status/Control register.
	24	*NoInvalid.* The Valid bit will not be set to zero when the command has completed.
	23	*Add32.* Increment the descriptor pointer by 32 bytes (instead of eight bytes).
	22	*MIPSaddr.* The Source/Jump Pointer is a MIPS address.
	21	*BE.* The source buffer is in big-endian format. **
	20:16	*Reserved.* Must be set to zero.
	15:0	*Source length.* Length (in bytes) of the source structure pointed to by the source pointer. The descriptor is not valid until this field is non-zero, unless the Jump bit is set.
1	31:0	*Source pointer/Jump pointer.* A 32-bit pointer to the source buffer.

Description

The Source descriptor is identical in format to the Command descriptor. It contains bit fields that control descriptor ring operation, a pointer to a source data buffer, and the length of the data. If the Jump bit is set, the pointer is a jump address, not a data address.

** Setting this bit to 1 has no effect on data stored in the GPRAM. It only affects data in the PCI memory space.

Figure 34. Source descriptor

7.3.3 Destination Descriptor

31	30	29	28	27	26	25	24	23	22	21	20 … 16	15 … 0	1 0
Valid	Jump	Last	rsrvd	Over	rsrvd	Mskdne	NoInvld	Add32	MIPSadr	BE	reserved	Dest Data Length	00
Dest Data Pointer/Jump Pointer													00

Word	Bit Field	Description
0	31	*Valid.* The software creating the descriptor sets the Valid bit to one to indicate that the descriptor is complete and ready to be processed. The 7811 sets the Valid bit back to zero when it is done with the descriptor. The Security Engine also considers the descriptor to be invalid if the Dest Data Length field is zero and the Jump bit is not set. The Security Engine polls the descriptor periodically to determine if it has become valid.
	30	*Jump.* If set to one, the next word in the descriptor contains the address of the next descriptor. This address is loaded into the Security Engine Dest Ring Address
	29	*Last.* If set to one, this descriptor points to the last (or only) fragment of a Dest buffer.
	28	*Reserved.* Must be set to zero.
	27	*Overflow.* If set, one of the buffers associated with the current command overflowed.
	26	*Reserved.* Must be set to zero.
	25	*Mask Done.* If set to one, inhibit the setting of the Dest Done bit in the Security Engine Status/Control register.
	24	*NoInvalid.* The Valid bit will not be set to zero when the command has completed.
	23	*Add32.* Increment the descriptor pointer by 32 bytes (instead of eight bytes).
	22	*MIPSaddr.* The Dest/Jump Pointer is a MIPS address.
	21	*BE.* The Dest buffer is in big-endian format. **
	20:16	*Reserved.* Must be set to zero.
	15:0	*Dest length.* Length (in bytes) of the Dest data pointed to by the Dest Data Pointer. Unless the Jump bit is set, this field must be non-zero. The host initializes this field to the buffer size in bytes. This size must be a multiple of four bytes. The Security Engine updates it with the actual length, in bytes, of the data placed in the buffer.
1	31:0	*Source pointer/Jump pointer.* A 32-bit pointer to the dest buffer.

Description

The Dest descriptor contains bit fields that control descriptor ring operation, a pointer to a dest data buffer, and the length of the data. If the Jump bit is set, the pointer is a jump address, not a data address. The descriptor and the dest data buffer must both be aligned to 32-bit boundaries, and the length of the dest data buffer must be initialized by the hosts to a multiple of four bytes.

** Setting this bit to 1 has no effect on data stored in the GPRAM. It only affects data in the PCI memory space.

Figure 35. Dest descriptor

7.3.4 Result Descriptor

31	30	29	28	27	26	25	24	23	22	21	20 … 16	15 … 1	0
Valid	Jump	Last	rsrvd	Over	Dst Ovf	Mskdne	NoInvld	Add32	MIPSadr	BE	reserved	Result Length	00
Result Pointer/Jump Pointer													00

Word	Bit Field	Description
0	31	*Valid.* The software creating the descriptor sets the Valid bit to one to indicate that the descriptor is complete and ready to be processed. The 7811 sets the Valid bit back to zero when it is done with the descriptor. The Security Engine also considers the descriptor to be invalid if the Result Length field is zero and the Jump bit is not set. The Security Engine polls the descriptor periodically to determine if it has become valid.
	30	*Jump.* If set to one, the next word in the descriptor contains the address of the next descriptor. This address is loaded into the Security Engine Dest Ring Address
	29	*Last.* If set to one, this descriptor points to the last (or only) fragment of a Dest buffer.
	28	*Reserved.* Must be set to zero.
	27	*Overflow.* If set, one of the buffers associated with the current command overflowed.
	26	*Dest Overflow.* If set to one, one of the Dest descriptors overflowed. (Copied from bit [27] of the Dest descriptor.)
	25	*Mask Done.* If set to one, inhibit the setting of the Result Done bit in the Security Engine Status/Control register.
	24	*NoInvalid.* The Valid bit will not be set to zero when the command has completed.
	23	*Add32.* Increment the descriptor pointer by 32 bytes (instead of eight bytes).
	22	*MIPSaddr.* The Result/Jump Pointer is a MIPS address.
	21	*BE.* The Result buffer is in big-endian format. **
	20:16	*Reserved.* Must be set to zero.
	15:0	*Result length.* Length (in bytes) of the Result data pointed to by the Result Pointer, in bytes. Unless the Jump bit is set, this field must be non-zero. The host initializes this field to dest buffer size in bytes. This size must be a multiple of four bytes. The Security Engine updates it with the actual length of the data placed in the buffer.
1	31:0	*Result pointer/Jump pointer.* A 32-bit pointer to the result.

Description

The Result descriptor contains bit fields that control descriptor ring operation, a pointer to a Result data buffer, and the length of the data. If the Jump bit is set, the pointer is a jump address, not a data address. The descriptor and the result buffer must both be aligned to 32-bit boundaries, and the length of the result buffer must be initialized by the host to a multiple of four bytes.

** Setting this bit to 1 has no effect on data stored in the GPRAM. It only affects data in the PCI memory space.

Figure 36. Result descriptor

7.4 Encode/Decode Command Structures

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.

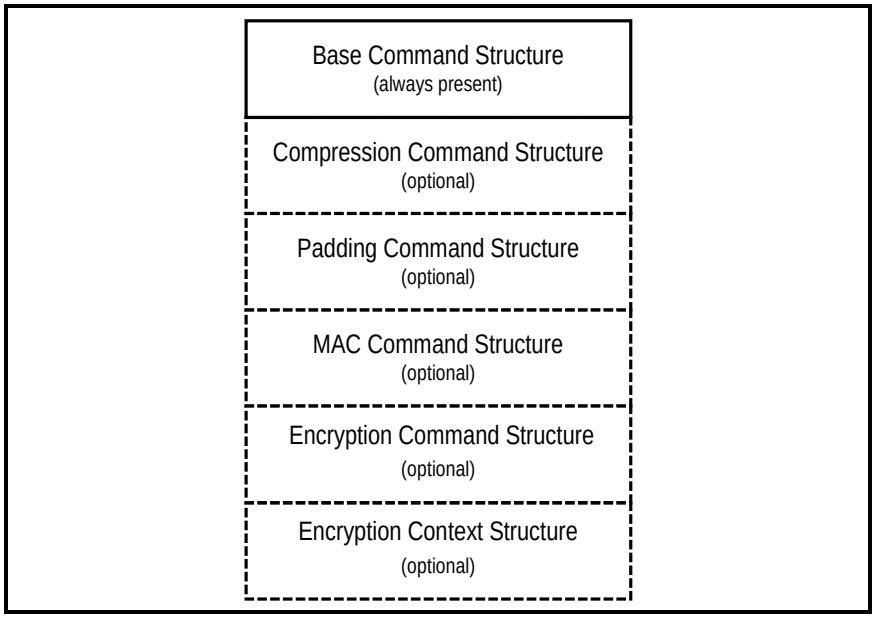


Figure 37. Command structures

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

7.4.1 Base Command Structure

Order in memory	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Command			Reserved	Encrypt	MAC	Pad	Comp	Reserved		Dest Align		Session Number[14:12]			Ignore Dest Cnt
1	Total Source Cnt (D17-D16)		Total Dest Cnt (D17-D16)		Session Number[11:0]											
2	Total Source Count (D15-D0)															
3	Total Dest Count (D15-D0)															

Refer to Figure 30 for mapping between 16 and 32 bit representations.

16-bit Word	Bit Field	Description
0	15:13	*Command opcode*. Gives the basic command to execute. 0=Encode, 1=Decode, 2=Read RAM, 3=Write RAM, 4-7= Reserved
	12	*Reserved.* Must be set to zero.
	11	*Encrypt.* If set to one, enable the encryption/decryption unit. If zero, it is disabled.
	10	*MAC.* If set to one, the MAC unit is enabled. If zero, it is disabled.
	9	*Pad.* If set to one, the padding unit is enabled. If zero, it is disabled.
	8	*Compress.* If one, the compression/decompression unit is enabled. If zero, it is disabled.
	7:6	*Reserved.* Must be set to zero.
	5:4	*Dest Align.* Specifies the number (0-3) of garbage bytes to emit before the start of the Dest data stream. These bytes are counted in the Dest Count.
	3:1	*Session Number[14:12]*. Upper bits of the session number. The lower bits are in word 1. Setting bit 3 to 1 indicates the use of IPPCP. Hence max sessions is now 32K.
	0	*Ignore Dest Count.* If set to one, do not terminate the command when the Total Dest Count decrements to zero. Set to zero for normal operation.
1	15:14	*Total Source Count[17:16]*. Upper two bits of the Total Source Count.
	13:12	*Total Dest Count[17:16]*. Upper two bits of the Total Dest Count.
	11:0	*Session Number.* Gives the least-significant 12 bits of the session number. Used for indexing compression and encryption context.
2	15:0	*Total Source Count[15:0]*. The length in bytes of the source data, plus any bytes skipped by the Source Align field. This count is decremented for each source byte processed, and when the count reaches zero, the command is terminated (if the Ignore Source Count bit is zero).
3	15:0	*Total Dest Count[15:0]*. The length in bytes of the Dest data, plus any bytes skipped by the Dest Align field. This count is decremented for each Dest byte produced, and when the count reaches zero, the command is terminated (if the Ignore Dest Count bit is zero).

Description

The Encode and Decode commands start with the Base Command structure, followed by the command structures for all enabled stages: Encryption, MAC, Pad, and Compress. If a stage is disabled, its command structure must not be present. A command must be at least sixteen bytes long (counting the eight bytes of the Base Command Structure), and should be padded with zeroes as necessary to guarantee this. For example, a command that consists of only the Base Command and Pad Command structure would be 12 bytes long, and would require four bytes of zeroes after the Pad Command structure.

Figure 38. Base command structure

7.4.2 Compress Command Structure

Order in memory	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Comp Source Cnt (D17-D16)		0b000001000									Clear Hist	Update History	Strip 0/ Restart	reserved	MPPC
1	Comp Header Count (D15-D0)															
2	Comp Source Count (D15-D0)															
3	reserved															

Refer to Figure 30 for mapping between 16 and 32 bit representations

16-bit Word	Bit Field	Description
0	15:14	*Comp Source Count[17:16]*. High-order bits of the source count. See the description of word 2, below.
	13:5	*Reserved.* Must be set to 0b000001000.
	4	*Clear History.* If set to one, clear the session's compression history before compression or decompression begins. If zero, use the history stored as part of the session context. 1) The *Clear History* bit has to be set on all stateless compression and decompression commands. If not set, there is the possibility of a corrupt or malicious packet exposing data from a prior history. 2) This bit must be set on the 1st packet in the stateful compression and decompression sessions.
	3	*Update History.* Only valid during decompression. If one, 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.
	2	*Strip 0/Restart.* See the text below this table.
	1	*Reserved.* Must be set to zero.
	0	*MPPC.* Selects the compression algorithm. If one, MPPC is used. If zero, LZS is used.
1	15:0	*Comp Header Count[15:0]*. Selects the number of bytes of header data to pass through the unit unmodified.
2	15:0	*Comp Source Count[15:0]*. Low-order 16 bits of the source count. After passing through the bytes specified in the Compressed Header Count, the compression/decompression unit processes the number of bytes given in the Comp Source Count. After compressing or decompressing the specified number of source bytes, the unit returns to pass-through mode. The Total Source Count (in the Base Command structure) will also end processing. When either counter has expired, the compression/decompression unit will flush out its internal data and (if compressing in LZS mode) append an End Marker to the decompressed data stream.
3	15:0	*Reserved.* Must be set to zero.

Description

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) are discarded.

Figure 39. Compress command structure

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.

In MPPC mode (the MPPC bit is set to one), this bit is known as the RESTART bit, and is used to implement the “restart” function of the MPPC protocol.

Enabling the Strip 0 feature generally reduces the size of the compressed data stream by one byte. If this bit is set to one 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 will be zero approximately 88% of the time. If the last byte is eliminated, it will not be counted by the Dest Counter. If padding is enabled, then Strip 0 should not be used.

On a Decompress operation, a 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 inserted byte will not be counted by the Source Counter.

If the Strip 0 mode is enabled during a Decompress operation, the 7751 must know the exact number of source bytes so that it can insert the zero at the correct location in the data stream. The pad length must contain the exact number of padding bytes.

Many data communication standards define this 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, 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 zero.

7.4.3 Pad Command Structure

Order in memory	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Pad Source Count (D17-D16)		Pad Count Mode	reserved	Pad Length				reserved						Pad Algorthm	
1	Pad Source Count (D15-D0)															

Refer to Figure 30 for mapping between 16 and 32 bit representations

16-bit Word	Bit Field	Description
0	15:14	*Pad Source Count[17:16]*. High-order bits of the source count. See the description of word 1, below.
	13	*Pad Count Mode.* If set to one, 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 zero, the Pad Source Counter starts decrementing after the last header byte, as set by the Pad Header Count field, has been passed through. The Pad Header Count is initialized by the Encrypt Header Count field in the Encryption command structure.
	12	*Reserved.* Must be set to zero.
	11:8	*Pad Length*. Adjusts the number of padding bytes by the specified number of bytes. When encoding, padding is increased. When decoding, padding is decreased.
	7:2	*Reserved.* Must be set to zero.
	1:0	*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.
1	15:0	*Pad Source Count[15:0]*. 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 Count. Once Pad Source Count bytes have been processed, or the last byte as defined by the Total Source Count has been processed, the padding processing unit stops, and data is once more passed through unmodified.

Description

During encoding, the padding unit rounds out data fields to a modulo eight length, which is required by many protocols.

Figure 40. Pad command structure

7.4.4 MAC Command Structure

Order in memory	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	MAC Source Cnt (D17-D16)		MAC Count Mode	reserved	New Key	reserved	MAC Position		reserved	Insert MAC	MAC Result	Truncate MAC	MAC Mode		MAC Algorithm	
1	MAC Header Count (D15-D0)															
2	MAC Source Count (D15-D0)															
3	reserved															

Refer to Figure 30 for mapping between 16 and 32 bit representations

Word	Field	Description
0	15:14	*MAC Source Count[17:16]*. High-order bits of the source count. The lower-order bits are in word 2.
	13	*MAC Count Mode.* If set to one, 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 zero, the MAC Source Counter begins after the last header byte has been passed through, as determined by the MAC Header Count field.
	12	*Reserved.* Must be set to zero.
	11	*New Key.* If set to one, a new key is to be supplied in the Encryption Context structure.
	10	*Reserved.* Must be set to zero.
	9:8	*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.
	7	*Reserved.* Must be set to zero.
	6	*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 status bit.
	5	*MAC Result.* If this bit is set to one, the calculated MAC will be written into the MAC Result structure. If zero, the calculated MAC will not be written into the Result structure.
	4	*Truncate MAC.* If this bit is set to one, the MAC will be truncated to twelve bytes. The most-significant bytes will be stripped. If zero, the MAC will be full-length, which is 20 bytes for SHA and 16 bytes for MD5.
	3:2	*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.
	1:0	*MAC Algorithm.* Determines the MAC hashing algorithm, as follows: 00: SHA; 01: MD5; 1x: Reserved.
1	15:0	*MAC Header Count.* The number of Source bytes to skip before MAC processing begins. This field is ignored if the MAC Count Mode bit is set to one.
2	15:0	*MAC Source Count[15:0]*. Least-significant 16 bits of the MAC Source Count. This gives the number of source bytes to process before resuming pass-through operation.
3	15:0	*Reserved.* Must be set to zero.

Description

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.

Figure 41. Mac command structure

7.4.5 Encryption Command Structure

Order in memory	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Encrypt Source Cnt (D17-D16)		Encrypt Count Mode	New IV	New Key	reserved				Clear Encrypt Context	reserved	Encrypt Mode		reserved	Encrypt Algorithm	
1	Encrypt Header Count (D15-D0)															
2	Encrypt Source Count (D15-D0)															
3	reserved															

Refer to Figure 30 for mapping between 16 and 32 bit representations

16-bit Word	Bit Field	Description
0	15:14	*Encrypt Source Count[17:16].* High-order bits of the source count.
	13	*Encrypt Count Mode.* If set to one, 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 zero, the Encrypt Source Counter begins after the last header byte has been passed through, as determined by the Encrypt Header Count field.
	12	*New IV.* If set to one, a new DES/3DES encryption initialization vector (IV) is to be 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 bit must be zero.
	11	*New Key.* If set to one, a new key is to be supplied in the Encryption Context structure.
	10:7	*Reserved.* Must be set to zero.
	6	*Clear Encryption Context.* If set to one, the encryption context will not be saved to context 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 zero, context is updated normally.
	5	*Reserved.* Must be set to zero.
	4:3	*Encryption 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.
	2	*Reserved.* Must be set to zero.
	1:0	*Encryption Algorithm.* Determines the encryption algorithm, as follows: 00=DES, 01=3DES, 10=RC4, 11=reserved.
1	15:0	*Encryption Header Count.* The number of Source bytes to skip before encryption processing begins. This field is ignored if the Encrypt Count Mode bit is set to one.
2	15:0	*Encrypt Source Count[15:0].* Least-significant 16 bits of the Encrypt Source Count. This gives the number of source bytes to process before resuming pass-through operation.
3	15:0	*Reserved.* Must be set to zero.

Figure 42. Encryption command structure

7.4.6 Encryption Context Structure

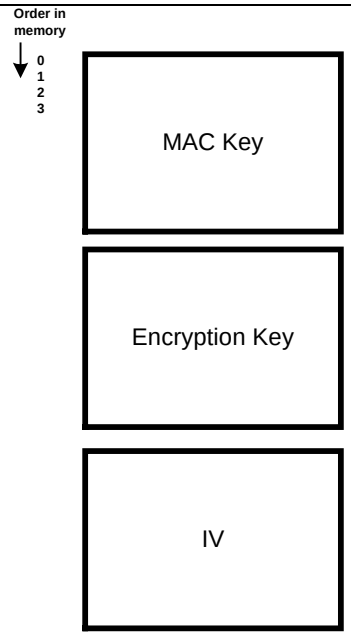
If the encryption command calls for a new key, MAC key, or IV, this must be provided to the Security Engine. The data takes the form of an *Encryption Context Structure,* with the format described in Figure 29.

Due to a peculiarity of the Security Engine, the Encryption Context can be placed in either a command buffer or a data buffer. It is either appended to the command or pre-pended to the data

In the command case, the Encryption Context structure can be considered to be another command structure, one that follows the usual command structures. It can be physically appended to the command in the same command buffer or

(more typically), it will be placed in its own command buffer as the last segment of a multi-descriptor command. See Figure 44.

The data case is similar, but the Encryption Context precedes the Source Data stream instead of following it.



Field	Description
MAC Key	The MAC Key is a 64-byte data structure consisting of the key followed by padding to make 64 bytes. The padding bytes are zeroes. If the New Key field of the MAC Command structure is zero, this field is omitted.
Encryption Key	The Encryption Key is a data structure whose length varies according to the encryption algorithm. DES uses an eight-byte key. Triple-DES uses a 24-byte key. RC4 uses a 260-byte key, of which the four final bytes are always zero. If the actual key is less than 256 bytes, it is repeated to fill 256 bytes. If the New Key field of the Encryption Command structure is zero, this field is omitted.
IV	IV is eight bytes long for DES and Triple-DES. RC4 does not use an IV. If the New IV field of the Encryption Command structure is zero, this field is omitted.

Description

The Encryption Context Structure allows keys and IVs to be given to the Security Engine. Each of these fields is controlled by a bit in the Command Structure. Unless the Command calls for the presence of one of these fields, it must be omitted.

Figure 43. Encryption Context structure

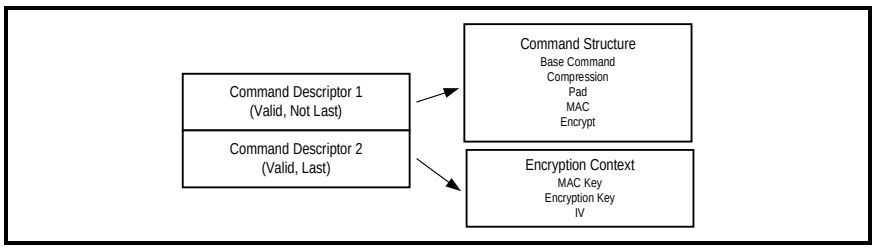


Figure 44. Typical use of descriptors for a command that requires encryption context

7.5 Read RAM/Write RAM Command Structures

The Security Engine can read and write the Context RAM using the Read RAM and Write RAM commands, with opcodes of 2 and 3, respectively.

The Read RAM and Write RAM commands are not needed in ordinary operation; they are used for debugging and RAM diagnostics.

7.5.1 Read RAM Command Structure

Command Buffer Contents

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Command (02 for Read RAM)				Reserved											
1	Reserved		Total Dest Cnt (D17-D16)		Reserved	Start Address (A24-A14)										

Source Data Buffer Contents

0	Reserved	Start Address (A13-A0)
1	Total Dest. Count (D15-D0)	

Refer to Figure 30 for mapping between 16 and 32 bit representations

16-bit Word	Bit Field	Description
Command Buffer		
0	15:13	*Command opcode*. The opcode for Read RAM is 2 (0b010).
	12:0	*Disable* Dest *FIFO.* If set to one, all Dest data is discarded. If zero, normal operation.
1	15:14	*Reserved.* Must be set to zero.
	13:12	*Total Dest Count[17:16]*. High-order bits of the Total Dest Count.
	11	*Reserved.* Must be set to zero.
	10:0	*Start Address[24:14]*. High-order bits of the address within the Context RAM from which data is to be read.
Source Data Buffer		
0	15:14	*Reserved.* Must be set to zero.
	13:0	*Start Address[13:0]*. Lower-order bits of the Context RAM starting address. Start Address specifies the address in 16-bit words. Since the Context RAM is accessed 16 bits at a time, it is necessary to divide the address by two and program these lower 14 bits.
1	15:0	*Total Dest. Count[15:0]*. Lower-order bits of the Total Dest Count, which is the length in bytes of the Dest data to be read, indicating that bit 0 must be zero.

Description

The Read RAM command reads a block of data from the Context RAM. Unlike other commands, the Read RAM command has its command structure split, with the first 32 bits in the Command Buffer and the last 32 bits in the Source Data Buffer. This is due to the fact that in the Read RAM command there is no data that is sent to the engines through the source descriptor. The Command Buffer must be padded to 16 bytes with zeroes.

For bits [13:0] of command 0 of the Source Data Buffer, if it was required to read address 0x800000, the value that needs to be programmed into the respective Start Address fields of the Read RAM command structure needs to be: 0x800000 >> 1 = 0x400000. Failure to do this will result in accessing every other entry and thus only half the CRAM is actually available.

It must be noted that the Dest Count is in bytes and the address is a 16 bit address. If we intended to read 8 bytes from location 0x100 we would need to set the Dest Count to 8 (bytes) and the address would be 0x100 shifted right once i.e. 0x80.

Figure 45. Read RAM command structure

7.5.2 Write RAM Command Structure

Order in memory	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Command (03 for Write RAM)			reserved												
1	Total Source Cnt (D17-D16)		reserved			Start Address (A24-A14)										
2	Total Source Count (D15-D0)															
3	reserved		Start Address (A13-A0)													

Refer to Figure 30 for mapping between 16 and 32 bit representations

16-bit Word	Bit Field	Description
0	15:13	*Command opcode*. The opcode for Write RAM is 3 (0b011).
	12:0	*Reserved.* Must be set to zero.
1	15:14	*Total Source Count[17:16]*. High-order bits of the Total Source Count.
	13:11	*Reserved.* Must be set to zero.
	10:0	*Start Address[24:14]*. High-order bits of the address within the Context RAM to which data will be written.
2	15:0	*Total Source Count[15:0]*. The lower-order 16 bits of the Total Source Count (the number of bytes to be written), indicating that bit 0 must be zero.
	15:14	*Reserved.* Must be set to zero.
3	13:0	*Start Address[13:0]*. Lower-order bits of the Start Address, which is the address to which data will be written. Start Address specifies the address in 16-bit words.

Description

The Write RAM command takes a block of data from the Source Data buffer and writes it to Context RAM. The command must be padded to 16 bytes with zeroes.
It must be noted that the Source Count is in bytes and the address is a 16 bit address. If we intended to write 8 bytes in location 0x100 we would need to set the source count to 8 (bytes) and the address would be 0x100 shifted right once i.e 0x80.

Figure 46. Write RAM command structure

7.6 Source Structures

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

7.7 Dest Structures

The Destination Data structure is a byte stream containing the output of the Security Engine. It will be aligned to a 32-bit boundary and will be padded to a multiple of 32 bits wide. The number of valid bytes is given in the Total Dest Count field of the Base Result structure.

7.8 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.

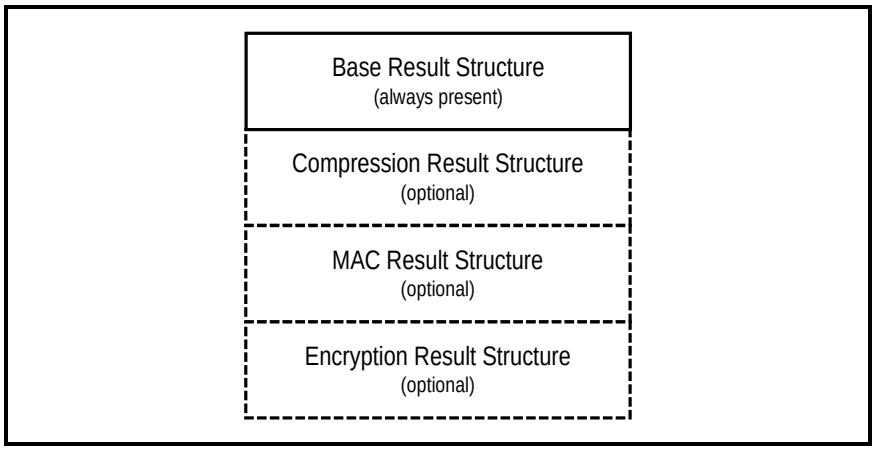


Figure 47. Result structures

7.8.1 Base Result Structure

Order in memory	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Reserved						Dest Overrun	Reserved								
1	Total Source Cnt (D17-D16)		Total Dest Cnt (D17-D16)		Reserved											
2	Total Source Count (D15-D0)															
3	Total Dest Count (D15-D0)															

Refer to Figure 30 for mapping between 16 and 32 bit representations

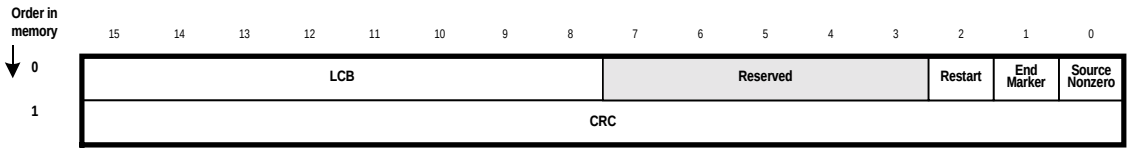
16-bit Word	Bit Field	Description
0	15:10	*Reserved.*
	9	*Dest Overrun.* This bit is set when the command produced more data than specified in the TOTAL DEST COUNT field of the Base Command Structure, and the IGNORE DEST COUNT bit was set to zero.
	8:0	*Reserved*
1	15:14	*Total Source Count[17:16]*. Upper two bits of the Total Source Count.
	13:12	*Total Dest Count[17:16]*. Upper two bits of the Total Dest Count.
	11:0	*Reserved*
2	15:0	*Total Source Count[15:0]*. Will decrement for every data phase byte processed from the source FIFO. It will not decrement for any extra bytes resulting from non-word alignment in the last transfer.
3	15:0	*Total Dest Count[15:0]*. Will decrement for every data phase byte that enters the destination FIFO. It will not decrement for any extra bytes resulting from non-word alignment in the last transfer.

Description

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.

Figure 48. Base Result Structure

7.8.2 Compression Result Structure



Refer to Figure 30 for mapping between 16 and 32 bit representations

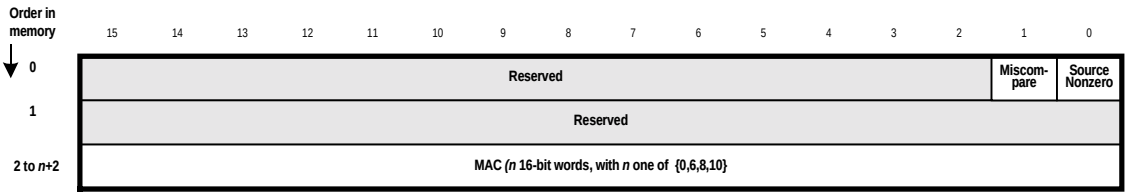
16-bit Word	Bit Field	Description
0	15:8	*Source LCB*. The LCB (longitudinal check byte) of the uncompressed data. For compression, this is the Source data. For decompression, this is the Dest data. This field is valid only if the compression/decompression unit was the only enabled unit in the command. The LCB is initialized to 0xFF, and each uncompressed data byte is exclusive-ORed to it. ($LCB_n = LCB_{n-1}$ XOR $data_n$.)
	7:3	*Reserved.*
	2	*Restart.* In MPPC mode, this bit indicates (if set to one) that the data was moved to the front of the compression context. If set to zero, the data was not moved. This bit is undefined for LZS compression
	1	*End Marker.* Defined only for LZS decompression. If set to one, an LZS End Marker was encountered in the Source data stream. If zero, no End Marker was encountered. LZS decompression ends when an End Marker is encountered.
	0	*Source Nonzero.* When set to one, this bit indicates that the command was terminated before the Comp Source Counter reached zero.
1	15:0	*CRC.* The 16-bit CRC of the command. The CRC is performed on the uncompressed data, using the equation $x^{16}+x^{12}+x^5+1$. At the start of each command, the CRC is initialized to 0xFFFF.

Description

The Compression Result Structure returns information about compression/decompression operations.

Figure 49. Compression Result Structure

7.8.3 MAC Result Structure



Refer to Figure 30 for mapping between 16 and 32 bit representations

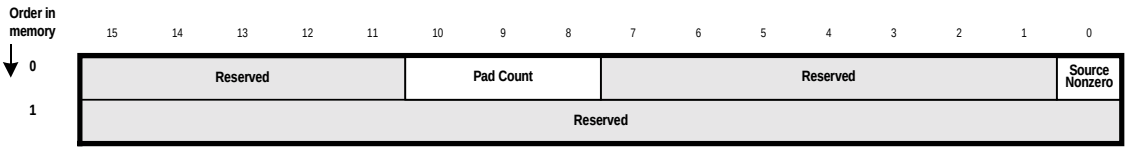
16-bit Word	Bit Field	Description
0	15:2	*Reserved.*
	1	*Miscompare.* Valid only during decode. If set to one, the MAC unit detected a miscompare (authentication failure). If zero, the computed MAC and the embedded MAC were the same.
	0	*Source Nonzero.* When set to one, this bit indicates that the command was terminated before the MAC Source Counter reached zero. This information is also reflected in the Security Engine Status register.
1	15:0	*Reserved.*
2 to *n*+2	15:0	*MAC.* The result of the MAC unit. The length of the MAC is determined by the MAC algorithm, the MAC Result bit, and the MAC Truncate bit (all of which are fields in the MAC Command Structure). If the MAC Result bit is zero, no MAC is returned. Otherwise, the MAC will be 12, 16, or 20 bytes long.

Description

The MAC Result Structure returns information about MAC operations.

Figure 50. MAC Result Structure

7.8.4 Encryption Result Structure



Refer to Figure 30 for mapping between 16 and 32 bit representations

16-bit Word	Bit Field	Description
0	15:11	*Reserved.*
	10:8	*Pad Count.* The number of bytes of padding added by the padding engine.
	7:1	*Reserved*
	0	*Source Nonzero.* When set to one, this bit indicates that the command was terminated before the Encryption Source Counter reached zero.
1	15:0	*Reserved.*

Description

The Encryption Result Structure returns information about encryption operations.

Figure 51. Encryption Result Structure

7.9 Context RAM Usage

An individual session's context memory requirements depend on the commands used by the session and the algorithms chosen. Encryption and authentication require context memory, as does compression with persistent compression history.

The 7811 has several context memory maps. These are set according to Security Engine Configuration register bits, as shown in Figure 52. All but one of these modes are compatible with the 7751. These modes allocate the same amount of RAM for every session. A new mode, the *multi-size* memory mode, allows both large and small context buffers to be allocated at the same time. The selection of a large or small buffer is based on the session number.

Compression Config.	Encryption Config.	Session Numbers	Description
0	0	0-0x7FFF	Single-History mode. Each session history is 512 bytes. One 32 KB compression history is shared by all sessions
0	1	0-0x7FFF	As above, but session history is 128 bytes.
1	x	0-0x3FFF	Multi-Size Mode, large sessions. Sessions are each 16 KB, starting at address 0 and building up.
1	0	0x4000-0x7FFF	Multi-Size Mode, small sessions. Sessions are 512 bytes, starting at the highest context memory address and building down.
1	1	0x4000-0x7FFF	Multi-Size Mode, small sessions. As above, but sessions are 128 bytes.

Figure 52. Context memory modes

7.9.1 Single-Size Modes

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

Stateless : The single-history option can be used when all sessions are guaranteed to use only stateless compression algorithms. All sessions share a single 32 KB compression history. The incremental per-session context is either 128 or 512 bytes, depending on the setting of the Encryption Configuration bit in the Security Engine Configuration register. The 32 KB context area starts at address 0 in Context RAM, and individual sessions start at 32 KB and work upwards in increments of 128 or 512 bytes. Single-history mode supports 32K sessions.

Stateful : The multiple-history option combines encryption and MAC context with compression history in a single 16 KB 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 Context RAM. As all sessions are 16 KB in size, the address of session *n* is 16384**n*. The 7811 multiple-history mode can address a total of 16 K sessions. The maximum memory addressing capability of the 7811 being 64MB and each session requiring 16K memory, the maximum number of stateful sessions is only 64MB/16K = 4K sessions.

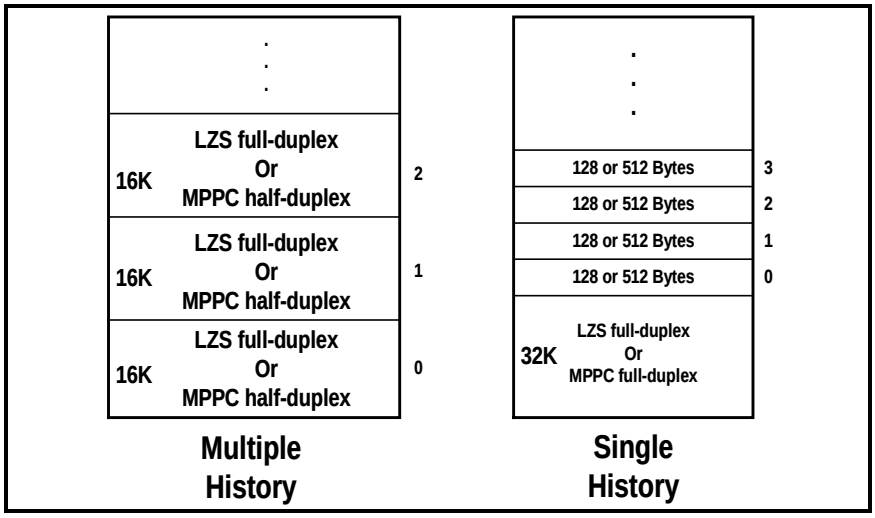


Figure 53. Context RAM memory usage in single-size modes

7.9.2 Multi-size Mode

Multi-size mode allows both stateless and stateful context sessions to be allocated at the same time. If bit 14 of the session number (Base Command Structure) is a '1', a small context buffer is allocated in the upper memory address space. If bit 14 of the session number is a zero, a large buffer is allocated from a low memory address, using the same rules as multiple-history mode. The allocation algorithm is given in Figure 55 and illustrated in Figure 54.

Although the examples assume static memory allocation (that is, memory will be divided between stateless and stateful sessions at system initialization), dynamic allocation can also be used. In this case the MIPS software will have to check to make sure that a new stateless session is not overlapping the memory space of an existing stateful session, or vice versa.

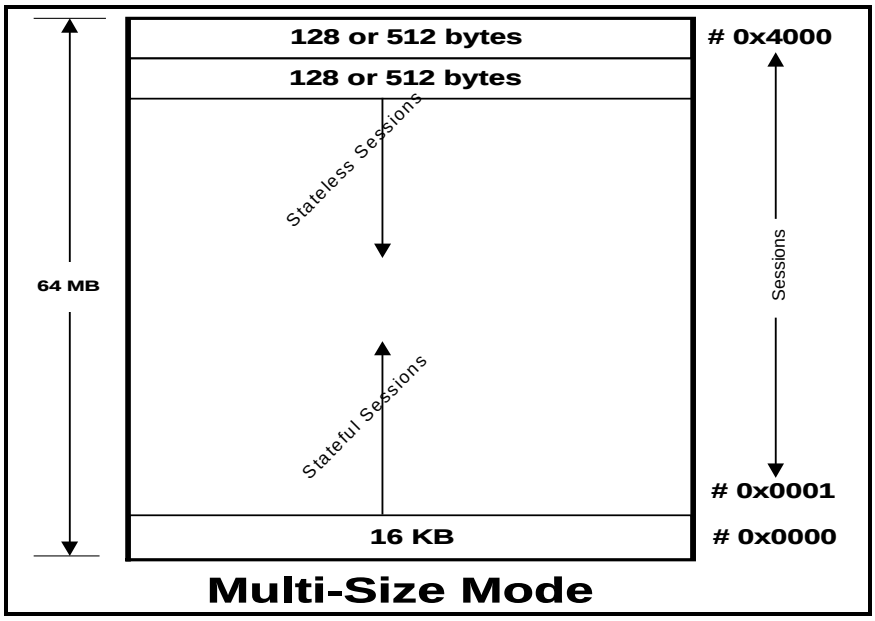


Figure 54. Multi-size mode example

```
/* The Encryption_Configuration bit in the Security Engine
Configuration register sets the size of a “small-context” session:
128 or 512 bytes. */

if Security_Engine_Configuration.Encryption_Configuration = 1
   SmallSessionSize := 0x80
else
   SmallSessionSize := 0x200
endif

/* The session number determines whether the session is a small
session or a large (16 KB of context memory) session.  Large
sessions start from address zero and move up; small sessions start
at address 0x0000000 and build down. */

if SessionNumber[14] = 0
   SessionBaseAddress := 0x4000 * SessionNumber[13:0]
else
   SessionBaseAddress := 0x0000000 – (SessionNumber[13:0] + 1 ) *
SmallSessionSize
endif
```

Figure 55. Multi-size mode memory allocation algorithm

Sessions in multi-size mode are allocated as if the system has 64 MB of context RAM. If the context RAM is smaller than 64 MB, the small-context sessions will be aliased into the upper region of context memory.

The example in Figure 56 shows the relationship between context RAM size and the total number of sessions. In this example, LZS compression is assumed. MPPC compression will reduce the number of stateful sessions by a factor of two.

In each case, the available memory is split between stateful sessions and stateless sessions. This provides at least enough room for 16K stateless sessions. Thus, the number of stateless sessions is (in practice) independent of context RAM size.

The number of stateful sessions depends entirely on the amount of installed RAM. One session is dedicated to “stateless” compression history, and the rest are available for use. With memory sizes in the 16MB-64MB range, for the specific case this varies from 894-3966 stateful sessions. With MPPC compression, this number would be halved.

Context Memory (CRAM) size in MBytes	Stateless Context Size in bytes	Number of Stateless Sessions	Stateful Context Size	Number of Stateful Sessions	Total Sessions
16	128 bytes	16K	16 KB	895	17279
	512 bytes	16K	16 KB	511	16895
32	128 bytes	16K	16 KB	1919	18303
	512 bytes	16K	16 KB	1535	17919
64	128 bytes	16K	16 KB	3967	20351
	512 bytes	16K	16 KB	3583	19967

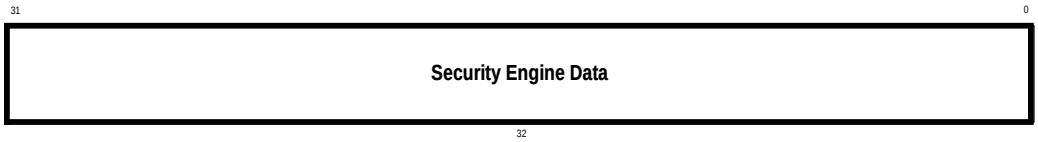
Example: *Memory size = 16 MB*
If the total memory size available is 16 MB, session #0 will occupy the lowest portion of memory and take 16KB. This is used as a scratch pad for the stateless sessions. The stateless sessions grow downwards from the highest memory location and the stateful sessions grow upwards from the lowest memory address. The assumption made in the example shown above is that the maximum number of stateless sessions possible have been allocated (16K). This would leave a total of (16MB-(128*16K)) = 14MB. Of this 16KB is used as scratch pad for the stateless sessions leaving (14MB-16KB) for stateful sessions. Since the memory required for each stateful session is 16KB there would be a total of 895 sessions. In this specific case a total of 17279 sessions are possible.

Figure 56. Number of Sessions in Multi-size mode

8 Registers

8.1 Group 0 Registers

Security Engine Data	**(Read/Write)**	**Base0+0x00**



Bit Field	Description
31:0	*Data.* See below.

Register Description

This register is used to write data directly into the Security Engine's Source FIFO or read directly from its Dest FIFO. The Data register must not be used by 7811 software because it bypasses the DMA channels, which can cause the device to hang. The Data register is provided for low-level debugging.

Overflowing the Source FIFO or underflowing the Dest FIFO will cause the Data Error bit to be set in the Security Engine Status register. If enabled, an interrupt will also be asserted.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is undefined.

Figure 57. Security Engine Data register

Security Engine Control	**(Read/Write)**	**Base0+0x04**

Bit Field	Description
31:2	*Reserved.* Must be set to zero.
1	*Reserved.* Must be set to one.
0	*Security Engine Reset.* Setting this bit to one causes a software reset of the Security Engine, which is equivalent to a hardware reset except that the Security Engine Configuration register is unaffected. The Source and Dest FIFOs are reset and the Security Engine pipeline is reset (operations in progress are abandoned). Because a soft reset causes operations to be abandoned, rather than completed, some information in Context RAM may be corrupted by a soft reset. The Security Engine Reset bit will return to zero after the reset sequence is complete. Until it returns to zero, the Security Engine registers should be left alone, except reading the Security Engine Control register to poll this bit.

Register Description

The Security Engine Control register controls some of the Security Engine's basic functionality.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is undefined.

Figure 58. Security Engine Control register

Security Engine Interrupt Status	**(Read/Write)**	**Base0+0x08**

Bit Field	Description
31:16	*Reserved.* Must be set to zero.
15	*Invalid Command.* If one, an illegal command was issued to the Security Engine. This bit could be set if the chips' MODE[2:1] pins are 00 or 01. It could also be set if the MODE[2:1] pins are 11 and the chip was not unlocked.
14	*Data Error.* If one, the Security Engine Data Register was read or written when it was not ready.
13	*Source FIFO Ready.* If one, the Source FIFO is ready for more data.
12	*Dest FIFO Ready.* If one, the Dest FIFO is ready for more data.
11:10	*Reserved.* Must be set to zero.
9	*Dest Overrun.* If one, the command produced more output bytes than specified in the Dest Count command parameter.
8	*Reserved.* Must be set to zero.
7	*Source Command.* If one, the Source FIFO is in the command phase.
6	*Source Context.* If one, the Source FIFO is in the context phase.
5	*Source Data.* If one, the Source FIFO is in the data phase.
4	*Dest Data.* If one, the Dest FIFO is in the data phase.
3	*Reserved.* Must be set to zero.
2	*Dest Result.* If one, the Dest FIFO is in the result phase.
1:0	*Reserved.* Must be set to zero.

Register Description

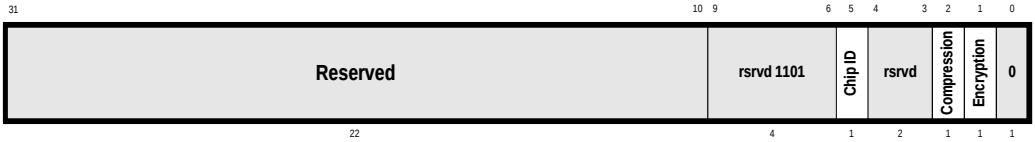
The Security Engine Interrupt Status register reports on the status of the Security Engine. The bits correspond to interrupt conditions. When one of these conditions takes place, the corresponding bit is set to one. The bits in this register are persistent; to clear a bit, the user must write a one to the corresponding bit position. This register is used to support interrupt service routines. Software polling can also use the Security Engine Status register, which does not have persistent bits, but shows the state of the Security Engine in real time (except after fatal errors, in which case the register contents are frozen).

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

At Reset, the value of this register is 0x00000000.

Figure 59. Security Engine Interrupt Status register

Security Engine Configuration (Read/Write) Base0+0x0C



Bit Field	Description
31:10	*Reserved.* Must be set to zero.
9:6	*Reserved.* Must be set to 0b1101.
5	*Chip ID.* Setting this bit to one causes the next read of the Security Engine Status register to return the chip ID rather than the current status. After the Status register is read, it returns to its normal function and this bit is reset to zero. This is a 7751 compatibility feature and would not normally be used in 7811-specific code.
4:3	*Reserved.* Must be set to zero.
2	*Compression Configuration.* Enables single-history or multi-history modes of context RAM allocation. 0 = single history, 1 = multi-history. Must be set to one for multi-size mode. See section 7.9.
1	*Encryption Configuration.* Selects the size of the encryption/MAC context. 0 = 512 bytes of context, 1 = 128 bytes. A 128-byte context will not accommodate RC4 encryption.
0	*Reserved.* Must be set to zero.

Register Description

The Security Engine Control register controls some of the Security Engine's basic functionality.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 60. Security Engine Configuration register

Security Engine Interrupt Enable (Read/Write) Base0+0x10

Bit Field	Description
31:16	*Reserved.* Must be set to zero.
15	*Invalid Command.* If one, an illegal command was issued to the Security Engine.
14	*Data Error.* If one, the Security Engine Data Register was read or written when it was not ready.
13	*Source FIFO Ready.* If one, the Source FIFO is ready for more data.
12	*Dest FIFO Ready.* If one, the Dest FIFO is ready for more data.
11:10	*Reserved.* Must be set to zero.
9	*Dest Overrun.* If one, the command produced more output bytes than specified in the Dest Count command parameter.
8	*Reserved.* Must be set to zero.
7	*Source Command.* If one, the Source FIFO is in the command phase.
6	*Source Context.* If one, the Source FIFO is in the context phase.
5	*Source Data.* If one, the Source FIFO is in the data phase.
4	*Dest Data.* If one, the Dest FIFO is in the data phase.
3	*Reserved.* Must be set to zero.
2	*Dest Result.* If one, the Dest FIFO is in the result phase.
1:0	*Reserved.* Must be set to zero.

Register Description

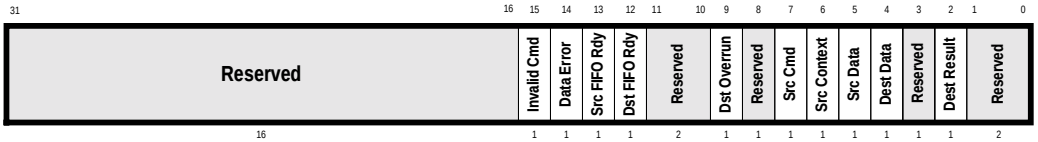
The Security Engine Interrupt Enable register selects the conditions under which interrupts will be asserted. Setting a bit to one enables the associated interrupt; zero disables it. The bit mappings are the same as in the Security Engine Interrupt Status and Security Engine Status registers. See the Security Engine Interrupt Status register for details of the bit mappings.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 61. Security Engine Interrupt Enable register

Security Engine Status	**(Read Only)**	**Base0+0x14**



Bit Field	Description
31:16	*Reserved.*
15	*Invalid Command.* If one, an illegal command was issued to the Security Engine.
14	*Data Error.* If one, the Security Engine Data Register was read or written when it was not ready.
13	*Source FIFO Ready.* If one, the Source FIFO is ready for more data.
12	*Dest FIFO Ready.* If one, the Dest FIFO is ready for more data.
11:10	*Reserved.*
9	*Dest Overrun.* If one, the command produced more output bytes than specified in the Dest Count command parameter.
8	*Reserved.*
7	*Source Command.* If one, the Source FIFO is in the command phase.
6	*Source Context.* If one, the Source FIFO is in the context phase.
5	*Source Data.* If one, the Source FIFO is in the data phase.
4	*Dest Data.* If one, the Dest FIFO is in the data phase.
3	*Reserved.*
2	*Dest Result.* If one, the Dest FIFO is in the result phase.
1:0	*Reserved.*

Register Description

The *Security Engine Status* register reflects the state of the Security Engine in real time. Some of these status bits are asserted for very brief periods and should be polled from the Interrupt Status register. Bits in the *Interrupt Status* register are persistent until cleared.

If the *Chip ID* bit is set to 1 in the *Security Engine Configuration* register, the *Security Engine Status* register must read 0x1120h

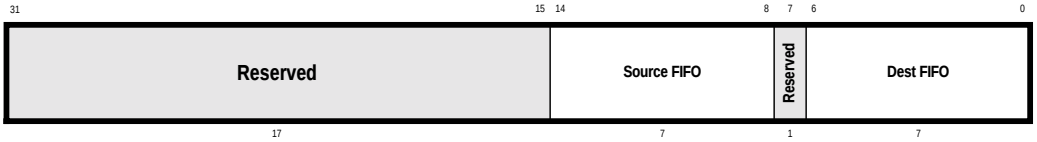
This register is Read Only.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 62. Security Engine Status register

Security Engine FIFO Status	(Read Only)	Base0+0x18



Bit Field	Description
31:15	*Reserved.*
14:8	*Source FIFO Free Space.* The number of bytes of free space in the Source FIFO, expressed as a seven-bit unsigned binary number.
7	*Reserved.*
6:0	*Dest FIFO Data.* The number of bytes of data in the Dest FIFO, expressed as a seven-bit unsigned binary number.

Register Description

The Security Engine FIFO Status register reports the amount of free space in the Security Engine's Source and Dest FIFOs. These FIFOs are each 64 bytes deep.

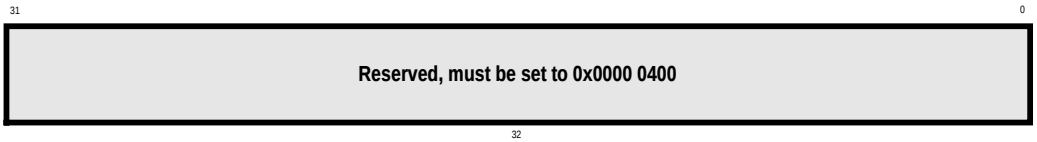
This register is provided for backward compatibility. The Security Engine's DMA units deal with the FIFOs transparently. The FIFOs rarely, if ever, require direct manipulation.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is 0x00004040.

Figure 63. Security Engine FIFO Status register

Security Engine FIFO Configuration	(Read/Write)	Base0+0x1C



Bit Field	Description
31:0	*Reserved.* Must be set to 0x0000 0400.

Register Description

The Security Engine FIFO Configuration register is another register provided for backward compatibility. It must be initialized to 0x00010001.

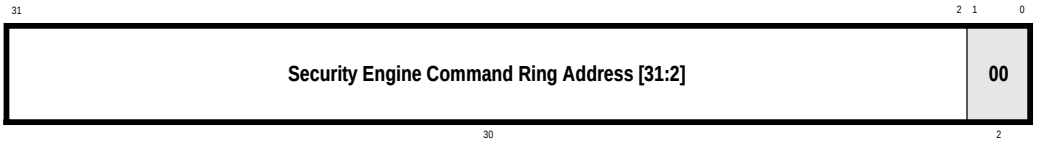
Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 64. Security Engine FIFO Configuration register

8.2 Group 1 Registers

Security Engine Command Ring Address (Read/Write)	**Base1+0x0C**



Bit Field	Description
31:0	Address of the Security Engine DMA command ring. This is a dword-aligned byte address; bits [1:0] must be set to 0b00.

Register Description

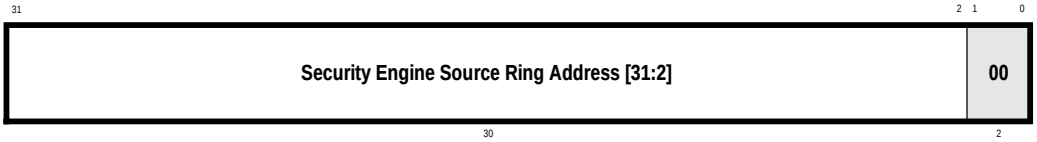
Points to the next Security Engine Command Descriptor.

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 65. Security Engine Command Ring Address register

Security Engine Source Ring Address (Read/Write)	**Base1+0x1C**



Bit Field	Description
31:0	Address of the Security Engine DMA source ring. This is a dword-aligned byte address; bits [1:0] must be set to 0b00.

Register Description

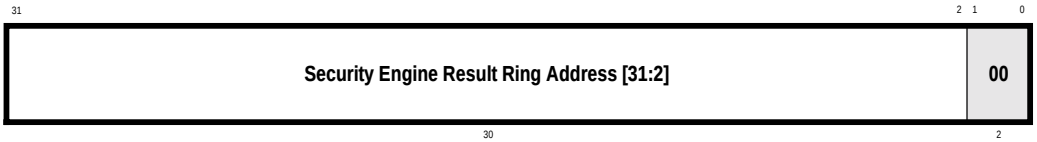
Points to the next Security Engine Source Descriptor.

Protected mode. In protected mode, PCI reads return 0x00. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 66. Security Engine Source Ring Address register

Security Engine Result Ring Address (Read/Write)	**Base1+0x2C**



Bit Field	Description
31:0	Address of the Security Engine DMA result ring. This is a dword-aligned byte address; bits [1:0] must be set to 0b00.

Register Description

Points to the next Security Engine Result Descriptor.

Protected mode. In protected mode, PCI reads return 0x00. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 67. Security Engine Result Ring Address register

Security Engine Dest Ring Address (Read/Write)	**Base1+0x3C**

Bit Field	Description
31:0	Address of the Security Engine DMA destination ring. This is a dword-aligned byte address; bits [1:0] must be set to 0b00.

Register Description

Points to the next Security Engine Dest Descriptor.

Protected mode. In protected mode, PCI reads return 0x00. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 68. Security Engine Dest Ring Address register

Security Engine Status and Control (Read/Write) Base1+0x40

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. Ring Ctl	Dst PCI Abort	Dst Done	Dst Last	Dst Waiting	Dst Over	Reserved	Res Ring Ctl	Res PCI Abort	Res Done	Res Last	Res Waiting	Res Over	Reserved	Src Ring Ctl	Src PCI Abort	Src Done	Src Last	Src Waiting	Illegal Write	Illegal Read	Cmd Ring Ctl	Cmd PCI Abrt	Cmd Done	Cmd Last	Cmd Waiting	Reserved	Engine IRQ
2	1	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1

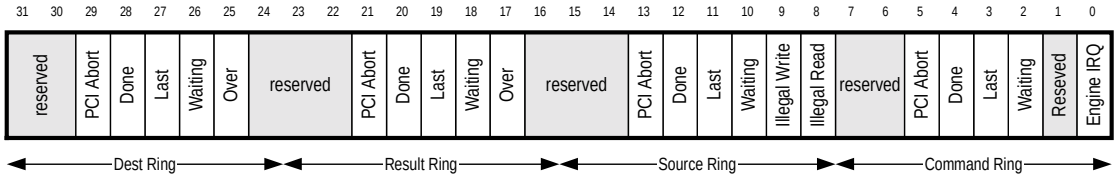
Register Description

31:30	*Dest Ring Control.* Controls the polling of Security Engine Dest descriptors: 00 = NOP (no effect; used when updating other fields in the register) 01 = Disable. After the current descriptor is processed, the Dest ring will be disabled. 10 = Enable. Enables the polling of the Dest descriptors. 11 = Reserved.
29	*Dest PCI Abort.* A PCI initiator or target abort occurred while processing a Dest descriptor. This is a fatal error and will halt descriptor processing on all four descriptor engines.
28	*Dest Done.* If one, the Dest descriptor processing is complete.
27	*Dest Last.* If one, the Dest descriptor processing is complete, and the descriptor had its Last bit set.
26	*Dest Waiting.* If one, the current Dest descriptor does not have its Valid bit set to one.
25	*Dest Over.*
24	*Reserved.* Must be set to zero.
23:22	*Result Ring Control.* Controls the polling of the Security Engine Result descriptors. The encoding is the same as with the Dest Ring Control field.
21	*Result PCI Abort.* A PCI initiator or target abort occurred while processing a Result descriptor. This is a fatal error and will halt descriptor processing on all four descriptor engines.
20	*Result Done.* If one, the Result descriptor processing is complete.
19	*Result Last.* If one, the Result descriptor processing is complete, and the descriptor had its Last bit set.
18	*Result Waiting.* If one, the current Result descriptor does not have its Valid bit set to one.
17	*Result Over.*
16	*Reserved.* Must be set to zero.
15:14	*Source Ring Control.* Controls the polling of the Security Engine Source descriptors. The encoding is the same as with the Dest Ring Control field.
13	*Source PCI Abort.* A PCI initiator or target abort occurred while processing a Source descriptor. This is a fatal error and will halt descriptor processing on all four descriptor engines.
12	*Source Done.* If one, the Source descriptor processing is complete.
11	*Source Last.* If one, the Result descriptor processing is complete, and the descriptor had its Last bit set.
10	*Source Waiting.* If one, the current Source descriptor does not have its Valid bit set to one.
9	*Illegal Write.* An illegal write was attempted to a protected region.
8	*Illegal Read.* An illegal read was attempted to a protected region.
7:6	*Command Ring Control.* Controls the polling of the Security Engine Command descriptors. The encoding is the same as with the Dest Ring Control field.
5	*Command PCI Abort.* A PCI initiator or target abort occurred while processing a Command descriptor. This is a fatal error and will halt descriptor processing on all four descriptor engines.

Security Engine Status and Control (Read/Write) Base1+0x40	
4	*Command Done.* If one, the Command descriptor processing is complete.
3	*Command Last.* If one, the Command descriptor processing is complete, and the descriptor had its Last bit set.
2	*Command Waiting.* If one, the current Command descriptor does not have its Valid bit set to one.
1	*Reserved.* Must be set to zero.
0	*Engine IRQ.*

Figure 69. Security Engine Status and Control register

Security Engine Interrupt Enable (Read/Write) Base1+0x44



Register Description

Enables interrupts on many of the conditions associated with the Security Engine Status/Control register (Base1 + 0x40).

The interrupts associated with this register can be mapped to PCI or MIPS interrupts. This mapping is set in the MIPS Config register.

Protected mode. In protected mode, PCI reads return 0x00. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 70. Security Engine Interrupt Enable register

Security Engine DMA Configuration (Read/Write) Base1+0x48

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		InboundBE	OutboundBE	DAddrMIPS	RAddrMIPS	SAddrMIPS	CAddrMIPS	Polling Frequency								TRamBE	Reserved	ProtMem		Reserved	Invalid Poll Scaler			Reserved			Software Last	Reserved	1	DMAReset#	MSTReset#

Bit Field	Description
31:30	*Reserved.* Must be set to zero.
29	*Inbound descriptor is big-endian.* **
28	*Outbound descriptor is big-endian.* **
27	*DAddrMIPS*. The Dest Data Ring address is a MIPS address if one, PCI if zero.
26	*RAddrMIPS*. The Result Ring address is a MIPS address if one, PCI if zero.
25	*SAddrMIPS.* The Status Ring address is a MIPS address if one, PCI if zero.
24	*CAddrMIPS.* The Command Ring address is a MIPS address if one, PCI if zero.
23:16	*Polling Frequency.* Determines how frequently descriptors will be polled, in units of 16 engine clock cycles. That is, a value of 0x01 gives a value of 16 cycles, 0x02 gives 32 cycles, and so on. A value of 0x00 disables polling.
15	*TRamBE.* If one, GPRAM appears to be big-endian to the host when the 7811 is a PCI target. If zero, GPRAM appears to be little-endian. This bit is meaningful only when the 7811 is the PCI target (slave). It has no effect when the 7811 is the PCI bus master. It does not affect register accesses.
14	*Reserved.* Must be set to zero.
13:12	*ProtMem.* Sets the size of protected memory. 00 = all GPRAM is protected (inaccessible from PCI). 01 = 4 MB is protected. 10 = 8 MB is protected. 11 = 16 MB is protected. Protected memory starts at address zero of GPRAM and extends upwards. This mechanism works for PCI target transfers only. For GPDMA transfers, memory protection is implemented through the PCI Address "OR" Mask (register 0x4C).. Protection is only effective when the 7811 is in protected mode.
11	*Reserved.* Must be set to zero.
10:8	*Invalid Poll Scaler.* This three-bit field determines how frequently a descriptor will be polled to see if the Valid bit has been set. Polling will occur according to this field or the Polling Frequency field, whichever gives the longer interval. This is used to reduce the frequency of polling while waiting for a valid descriptor, while polling more frequently when valid descriptors are available. This field is encoded as follows (in terms of engine clock cycles): 0 = 256 cycles, 1 = 512 cycles, 2=768 cycles, 3 = 1024 cycles 4 = 1280 cycles, 5 = 1536 cycles, 6=1792 cycles, 7=2048 cycles.
7:5	*Reserved.* Must be set to zero.
4	*Software Last.* This has the same function as Software Last in the GPDMA units.
3	*Reserved*. Must be set to zero.
2	*Reserved.* Must be set to one.
1	*DMAReset#.* Setting this bit to zero resets the descriptor units.
0	*MSTReset#.* Master reset for the Security Engine. Setting this bit to zero resets the entire Security Engine, including the security pipeline and the descriptor units.

Register Description

Security Engine DMA Configuration (Read/Write)	**Base1+0x48**
This register contains 7751-compatible bit fields and additional 7811-specific bit fields. ** Setting any of these bits to 1 has no effect on data stored in the GPRAM. They only affect data in the PCI memory space. *Protected mode.* In protected mode, PCI reads return zero and PCI writes are ignored. *On reset*, the value of this register is 0x00000000.	

Figure 71. Security Engine DMA Config register

PCI Address "OR" Mask	**(Read/Write)**	**Base1+0x4C**
31 — 0 PCI Address "OR" Mask 32		

Bit Field	Description
31:0	*PCI Address "OR" Mask.* This register implements memory protection for GPDMA transfers between PCI and GPRAM. See below.

Register Description

In GPDMA transfers between PCI and GPRAM, the value in this register is ORed with the GPRAM address. This allows address bits to be forced to one. For example, a system with 8 MB of total GPRAM and 4 MB of protected memory might set the mask to 0x00400000. All accesses to the lower 4 MB would silently be converted to accesses to the upper 4 MB.

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 72. PCI Address "OR" mask register

PCI Interrupt	(Read/Write)	Base1+0x50

31 — 0

PCI Interrupt [31:0]

32

Bit Field	Description
31:0	*PCI Interrupt.* The 7811 assets a PCI interrupt whenever a write to this register sets one or more bits that correspond to bits set in the PCI Interrupt Mask register.

Register Description

The PCI Interrupt register is used by the MIPS processor to generate PCI interrupts under software control. The value written to this register is ORed with the value in the PCI Interrupt Mask register. If the result is non-zero, the 7811 asserts a PCI interrupt.

This mechanism allows interrupts to be generated under software control, a parameter passed through the register, and the interrupts to be masked as needed by the software.

Protected mode. In both protected and unprotected modes, this register is accessible for reads and writes to both PCI and the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 73. PCI Interrupt register

PCI Interrupt Mask	(Read/Write)	Base1+0x54

31 — 0

PCI Interrupt Mask

32

Bit Field	Description
31:0	*PCI Interrupt Mask.* Enables the function of the PCI interrupt register.

Register Description

When a write occurs to the PCI Interrupt register, the PCI Interrupt register is ORed with the PCI Interrupt Mask register. If the result is non-zero, the 7811 asserts a PCI interrupt.

Protected mode. In both protected and unprotected modes, this register is accessible for reads and writes to both PCI and the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 74. PCI Interrupt Enable register

MIPS Interrupt	(Read/Write)	Base1+0x58

31 — 0

MIPS Interrupt [31:0]

32

Bit Field	Description
31:0	*MIPS Interrupt.* The 7811 assrets a MIPS interrupt whenever a write to this register sets one or more bits that correspond to bits set in the MIPS Interrupt Mask register.

Register Description

The MIPS Interrupt register is used by the host processor to generate MIPS interrupts under software control. The value written to this register is ORed with the value in the MIPS Interrupt Mask register. If the result is non-zero, the 7811 asserts a MIPS interrupt. This will be mapped to one of the three MIPS interrupt pins according to the settings of the MIPS Config register (0x90).

This mechanism allows interrupts to be generated under software control, a parameter passed through the register, and the interrupts to be masked as needed by the software.

Protected mode. In both protected and unprotected modes, this register is accessible for reads and writes to both PCI and the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 75. MIPS Interrupt register

MIPS Interrupt Mask	(Read/Write)	Base1+0x5C

31 — 0

MIPS Interrupt Mask

32

Bit Field	Description
31:0	*MIPS Interrupt Mask.* Enables the function of the PCI interrupt register.

Register Description

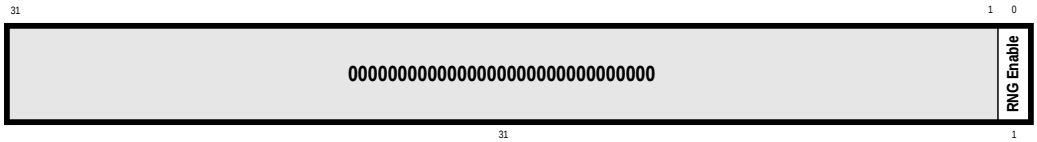
When a write occurs to the MIPS Interrupt register, MIPS PCI Interrupt register is ORed with the MIPS Interrupt Mask register. If the result is non-zero, the 7811 asserts a MIPS interrupt.

Protected mode. In unprotected mode, this register can be read and written by both PCI and the MIPS processor. In protected mode, this register is accessible only to the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 76. MIPS Interrupt Mask register

RNG Enable	(Read/Write)	Base1+0x60



Bit Field	Description
31:1	*Reserved.* Must be set to zero.
0	*RNG Enable.* Enables the function of the random number generator. 1 = enable RNG, 0 = disable RNG.

Register Description

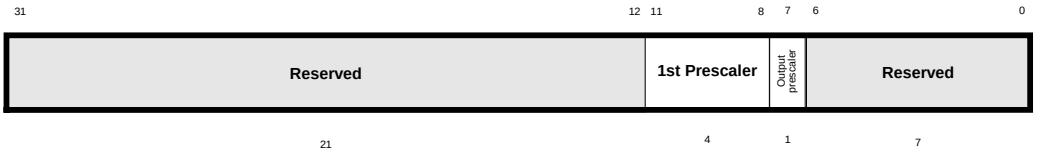
This register enables or disables the functionality of the random number generator.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 77. RNG Enable register

RNG Config	(Read/Write)	Base1+0x64



Bit Field	Description
31:12	Reserved. Must be written as zeroes.
11:8	First prescaler value. See Figure 79
7	Output prescaler value: 0 = 1024 (default) 1 = 512
6:0	Reserved. Must be written as zeroes

Register Description

The two prescalers determine the speed at which the random number generator produces output. A new 32-bit random number is generated every (first prescaler*output prescaler) engine cycles. The first prescaler value can be set to values between 2 and 32K. The output prescaler can be set to 512 or 1024. Their product can vary from 1 K cycles to 32 M cycles.

At its maximum speed, the random number generator operates at about 2.8 Mb/s at a 90 MHz clock:

(1024 cycles per random number / 32 bits per random number) = 32 cycles per bit,
90,000,000 cycles per second / 32 cycles per bit = 2,812,500 bits per second.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

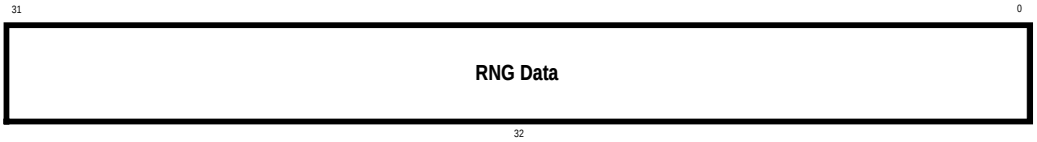
On reset, (hard or soft) both prescalers are set to zero, giving a speed of one random number per 32 M cycles, or about 90 b/s at a 90 MHz clock

Figure 78. RNG Config register

Value	Meaning
000x	32K
0010	16K
0011	8K
0100	4K
0101	2K
0110	1K
0111	512
1000	256
1001	128
1010	64
1011	32
1100	16
1101	8
1110	4
1111	2

Figure 79. Decoding of the first prescaler field

RNG Data	**(Read Only)**	**Base1+0x68**



Bit Field	Description
31:0	*Random data.* Pulls the top entry off the Random Number FIFO.

Register Description

This Read Only register provides random data from the Random Number FIFO. Once a random number is read, it is discarded, thus preventing the same random number from being used twice.

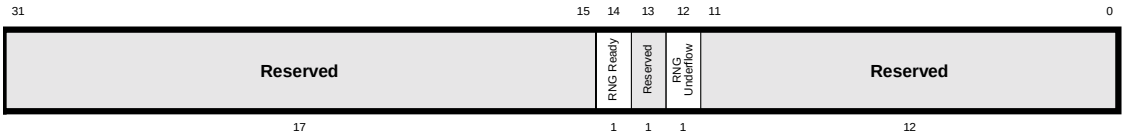
Random numbers are produced at a fixed rate. It is possible to underflow the Random Number FIFO, at which point the RNG Underflow bit in the RNG Status register will be set. To prevent this, the Random Number Ready bit in the RNG Status register should be polled before the RNG Data register is read. If the Random Number Ready bit is set, at least two 32-bit random numbers are available in the FIFO. The value of the RNG Data is undefined during a FIFO underflow.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is undefined.

Figure 80. RNG Data register

RNG Status	(Read/Write)	Base1+0x6C



Bit Field	Description
31:15	*Reserved.* Must be set to zero.
14	*RNG Ready.* If one, the random number generator has at least two valid 32-bit random numbers in its FIFO.
13	*Reserved*. Must be set to zero.
12	*RNG Underflow.* If one, an attempt was made to read more data than existed in the RNG FIFO. Once set, this bit is persistent until the RNG is reset.
11:0	*Reserved.* Must be set to zero.

Register Description

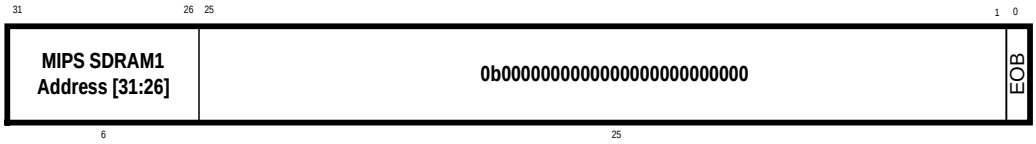
This register contains the state of the random number generator.

Protected mode. In protected mode, this register is only accessible to the MIPS processor.

On reset, the value of this register is 0x00000000.

Figure 81. RNG Status register

MIPS SDRAM1 Address	(Read/Write)	Base1+0x70



Bit Field	Description
31:26	Upper 6 bits of the 32-bit starting address of GPRAM, as seen by the MIPS processor.
25:1	Must be set to zero (that is, SDRAM must be mapped to a 64 MB boundary).
0	EOB (Endianness Override Bit) : Setting this bit to 1, complements the state of the MIPS_BIGENDIAN pin (AD1), providing a different endianness for the register space defined by this register.

Register Description

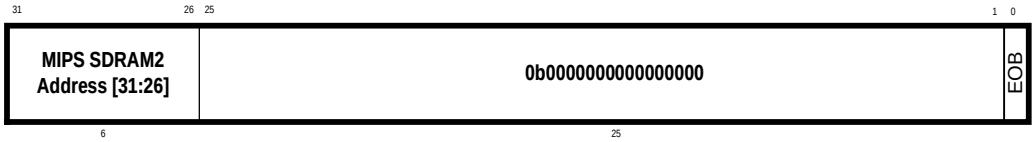
This register sets one of the two addresses at which the MIPS processor can access GPRAM. The MIPS SDRAM1 Address register is initialized from EEPROM, while the MIPS SDRAM2 Address register is not. In multi-7811 configurations, the MIPS processor must map each 7811's GPRAM to a different address range. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. In protected mode, PCI reads return 0x00. PCI writes are ignored.

On reset, bits [15:0] are set to zero, and bits [31:16] are read from EEPROM address 0x24.

Figure 82. MIPS SDRAM1 Address register

MIPS SDRAM2 Address	(Read/Write)	Base1+0x74



Bit Field	Description
31:26	Upper 6 bits of the 32-bit starting address of GPRAM, as seen by the MIPS processor.
25:1	Must be set to zero (that is, SDRAM must be mapped to a 64 MB boundary).
0	EOB (Endianness Override Bit) : Setting this bit to 1, complements the state of the MIPS_BIGENDIAN pin (AD1), providing a different endianness for the register space defined by this register.

Register Description

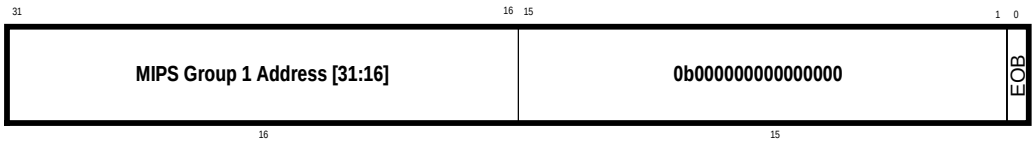
This register sets one of the two addresses at which the MIPS processor can access GPRAM. The MIPS SDRAM1 Address register is initialized from EEPROM, while the MIPS SDRAM2 Address register is not. In multi-7811 configurations, the MIPS processor must map each 7811's GPRAM to a different address range. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. In protected mode, PCI reads return 0x00. PCI writes are ignored.

On reset, the value of this register is 0x10000000.

Figure 83. MIPS SDRAM2 Address register

MIPS Group 1 Address	(Read/Write)	Base1+0x78



Bit Field	Description
31:16	*MIPS Group 1 Register Address [31:16].* Upper 16 bits of the 32-bit starting address of the Group 1 register space, as seen by the MIPS processor.
15:1	Must be set to 0x0000 (that is, the register space must begin on a 64 KB boundary).
0	EOB (Endianness Override Bit) : Setting this bit to 1, complements the state of the MIPS_BIGENDIAN pin (AD1), providing a different endianness for the register space defined by this register.

Register Description

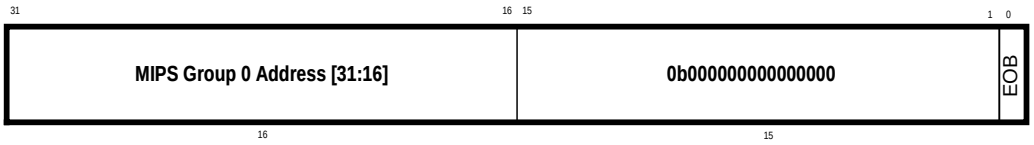
This register sets the address at which the 7811 accesses Group 1 registers. In multi-7811 configurations, the MIPS processor must map each 7811's registers to a different address range. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, bits [15:0] are set to zero, and bits [31:16] are read from EEPROM address 0x23.

Figure 84. MIPS Group 1 Address register

MIPS Group 0 Address	(Read/Write)	Base1+0x7C



Bit Field	Description
31:16	*MIPS Group 0 Register Address [31:16]*. Upper 16 bits of the 32-bit starting address of the Group 0 register space, as seen by the MIPS processor.
15:1	Must be set to 0x0000 (that is, the register space must begin on a 64 KB boundary).
0	EOB (Endianness Override Bit) : Setting this bit to 1, complements the state of the MIPS_BIGENDIAN pin (AD1), providing a different endianness for the register space defined by this register.

Register Description

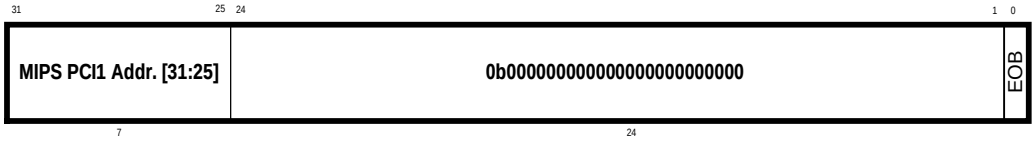
This register sets the address at which the 7811 accesses Group 0 registers. In multi-7811 configurations, the MIPS processor must map each 7811's registers to a different address range. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x30000000.

Figure 85. MIPS Group 0 Address register

MIPS PCI1 Address	(Read/Write)	Base1+0x80



Bit Field	Description
31:25	*MIPS PCI1Address [31:25]*. Upper 7 bits of the 32-bit starting address of the PCI memory space, as seen by the MIPS processor.
24:1	Must be set to zero (that is, the mapping must start on a 32 MB boundary).
0	EOB (Endianness Override Bit) : Setting this bit to 1, complements the state of the MIPS_BIGENDIAN pin (AD1), providing a different endianness for the register space defined by this register.

Register Description

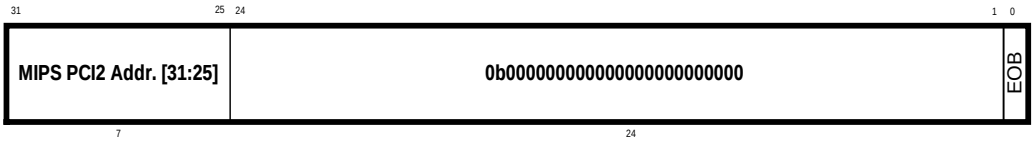
This register sets the address at which the 7811 accesses PCI Writes to the PCI1 region are translated to PCI addresses with the MIPS PCI1 Translation register. The MIPS PCI2 register provides identical functionality, allowing two non-contiguous regions of PCI memory to be accessed independently. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x40000000.

Figure 86. MIPS PCI1 Address register

MIPS PCI2 Address	(Read/Write)	Base1+0x84



Bit Field	Description
31:25	*MIPS PCI2 Address [31:25]*. Upper 7 bits of the 32-bit starting address of a PCI memory space, as seen by the MIPS processor.
24:1	Must be set to zero (that is, the mapping must start on a 32 MB boundary).
0	EOB (Endianness Override Bit) : Setting this bit to 1, complements the state of the MIPS_BIGENDIAN pin (AD1), providing a different endianness for the register space defined by this register.

Register Description

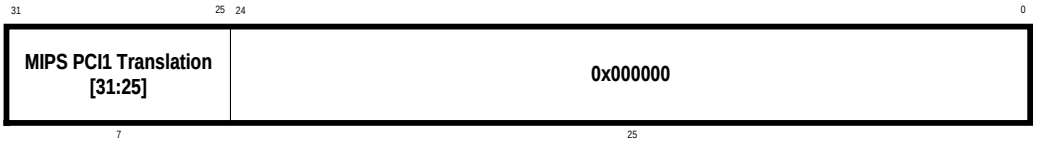
This register sets the address at which the 7811 accesses PCI Writes to the PCI2 region are translated to PCI addresses with the MIPS PCI2 Translation register. The MIPS PCI1 register provides identical functionality, allowing two non-contiguous regions of PCI memory to be accessed independently. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x50000000.

Figure 87. MIPS PCI2 Address register

MIPS PCI1 Translation	(Read/Write)	Base1+0x88



Bit Field	Description
31:25	*MIPS PCI1 Translation [31:25]*. These seven bits replace the upper 7 bits of a MIPS address when accessing the PCI1 region.
24:0	Must be set to zero.

Register Description

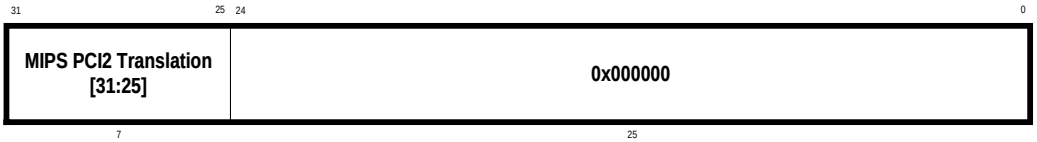
This register provides address translation for the PCI1 region. When the MIPS processor accesses an address in the PCI1 region, the upper seven bits of this register replace the upper seven bits of the original address. This translated address is used for the PCI transfer. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. base (1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. This register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is 0x00000000.

Figure 88. MIPS PCI1 Translation register

MIPS PCI2 Translation	(Read/Write)	Base1+0x8C



Bit Field	Description
31:25	*MIPS PCI2 Translation [31:25]*. These seven bits replace the upper 7 bits of a MIPS address when accessing the PCI2 region.
24:0	Must be set to zero.

Register Description

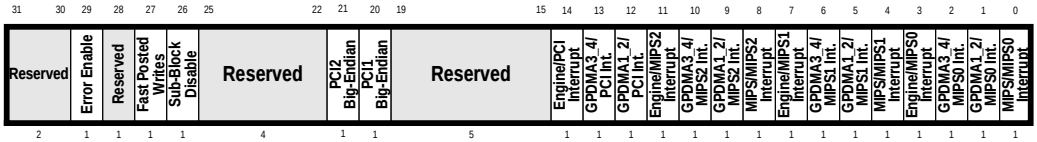
This register provides address translation for the PCI2 region. When the MIPS processor accesses an address in the PCI2 region, the upper seven bits of this register replace the upper seven bits of the original address. This translated address is used for the PCI transfer. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side.

Protected mode. This register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is 0x00000000.

Figure 89. MIPS PCI2 Translation register

MIPS Config **(Read/Write)** **Base1+0x90**

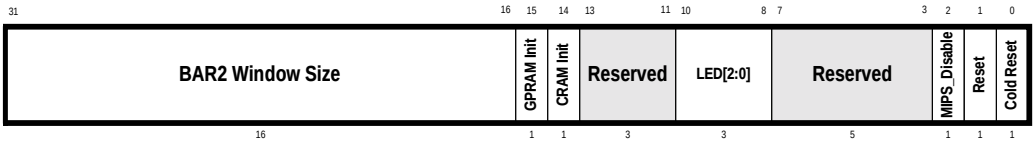


Bits	Description
31:30	*Reserved*. Must be set to zero.
29	*Error Enable.* If set to one, PCI errors that happen on cycles initiated by the MIPS processor will be reported on the MIPS command bus. If zero, PCI errors will not be propagated to the MIPS command bus, but the status bit in GPDMA1_2_Status will still be set.
28	*Reserved*. Must be set to zero.
27	*Fast Posted Writes.* If the MIPS CPU is producing zero-wait-state bursts, this bit will enable fast posted write processing if set to one, and disable it if set to zero. This will increase performance with aligned 16- and 32-byte writes.
26	*Sub-block disable.* Instead of using MIPS-style sub-block bursting, use linear address bursting. 1 = disable sub-block bursting, 0 = enable sub-block bursting. MIPS processors require sub-block bursting.
25:22	*Reserved*. Must be set to zero.
21	*PCI2 Big-endian*. Determines whether big-endian byte swapping is performed on transfers to or from the PCI2 region. 0 = Do not perform big-endian conversion. 1 = Perform big-endian conversion.
20	*PCI1 Big-endian*. Determines whether big-endian byte swapping is performed on transfers to or from the PCI1 region. Same encoding as above.
15:19	Reserved. Must be set to zero.
14	*Engine/PCI Interrupt.* If one, map the Security Engine interrupt to the PCI interrupt.
13	*GPDMA3_4/PCI Interrupt.* If one, map the GPDMA3_4 interrupt to the PCI interrupt.
12	*GPDMA1_2/PCI Interrupt.* If one, map the GPDMA1_2 interrupt to the PCI interrupt.
11	*Engine/MIPS2 Interrupt.* If one, map the Security Engine interrupt to the MIPS INT2 interrupt.
10	*GPDMA3_4/MIPS2 Interrupt.* If one, map the GPDMA3_4 interrupt to the MIPS INT2 interrupt.
9	*GPDMA1_2/MIPS2 Interrupt.* If one, map the GPDMA1_2 interrupt to the MIPS INT2 interrupt.
8	*MIPS/MIPS2 Interrupt.* If one, map the MIPS interrupt (caused by a write to the MIPS Interrupt register) to the MIPS INT2 interrupt.
7	*Engine/MIPS1 Interrupt.* If one, map the Security Engine interrupt to the MIPS INT1 interrupt.
6	*GPDMA3_4/MIPS1 Interrupt.* If one, map the GPDMA3_4 interrupt to the MIPS INT1 interrupt.
5	*GPDMA1_2/MIPS1 Interrupt.* If one, map the GPDMA1_2 interrupt to the MIPS INT1 interrupt.
4	*MIPS/MIPS1 Interrupt.* If one, map the MIPS interrupts (caused by a write to the MIPS Interrupt register) to the MIPS INT1 interrupt.
3	*Engine/MIPS0 Interrupt.* If one, map the Security Engine interrupt to the MIPS INT0 interrupt.
2	*GPDMA3_4/MIPS0 Interrupt.* If one, map the GPDMA3_4 interrupt to the MIPS INT0 interrupt.
1	*GPDMA1_2/MIPS0 Interrupt.* If one, map the GPDMA1_2 interrupt to the MIPS INT0 interrupt.
0	*MIPS/MIPS0 Interrupt.* If one, map the MIPS interrupts (caused by a write to the MIPS Interrupt register) to the MIPS INT0 interrupt.

MIPS Config	**(Read/Write)**	**Base1+0x90**
This register selects address formats and interrupt mappings for the 7811. This register is available for writing only after a MIPS reset (by setting bit 0 and 1 of the MIPS reset register i.e. (base1 + 0x94) to 1's). Prior to this reset, the register cannot be modified from the PCI side. *Protected mode.* In protected mode, PCI reads return zero and PCI writes are ignored. *On reset*, the value of this register is 0x00004000.		

Figure 90. MIPS Config register

MIPS Reset	(Read/Write)	Base1+0x94



Bit Field	Description
31:16	*BAR2 Window Size.* Reports the size of installed packet and context SDRAMs, as given in the EEPROM word 0x12.
15	*GPRAM Init.* If one, GPRAM has been initialized and can be accessed by software. If zero, initialization has not completed and GPRAM should not be accessed.
14	*Context RAM Init.* If one, context memory has been initialized. If zero, initialization has not completed and the Security Engine should not be enabled.
13:11	*Reserved.* Must be set to zero.
10:8	*LED[2:0]*. These bits are inverted and driven onto external pins. Can be used to drive status LEDs.
13:3	*Reserved.* Must be set to zero.
2	*MIPS Disable.* If one, disables the MIPS interface after the current bus transaction, if any. Shutting down the MIPS interface before resetting the MIPS processor will prevent possible bus-hang conditions. At least one microsecond should elapse between setting MIPS Disable and resetting the MIPS processor.
1	*MIPS Reset.* Setting this bit to zero asserts the MIPS_RESET# pin, causing a "warm" reset of the MIPS processor. Setting this bit to one de-asserts the MIPS_RESET# pin after a delay of 16K clock cycles. If the bit is set to zero during the 16K cycle delay, the delay is aborted and the pin is asserted immediately. This signal should not be set to zero until after the MIPS Disable bit has been set to one.
0	*MIPS Coldreset*. Setting this bit to zero asserts the MIPS_COLDRESET# pin, causing a "cold" reset of the MIPS processor. Setting this bit to one de-asserts the MIPS_COLDRESET# pin after a delay of 64K cycles (as specified in the MIPS specification). In addition, a zero-to-one transition on this bit will cause the MIPS_RESET# pin to be asserted, both during the coldreset delay and for 16K cycles afterwards. If the bit is set to zero during the 64K cycle delay, the delay is aborted and the pin is asserted immediately. This signal should not be set to zero until after the MIPS Disable bit has been set to one.

Register Description

This register allows the MIPS processor to be reset by the host processor.

Protected mode. This register is fully accessible in both protected and unprotected modes.

On reset, the upper bits mirror EEPROM location 0x12; the lower 16 bits are 0x0000.

Figure 91. MIPS Reset register

Revision Number	**(Read Only)**	**Base1+0x98**

31 — 0

Revision Number

32

Bit Field	Description
31:0	*Revision number.* Contains a revision code assigned by Hi/fn.

Register Description

This register contains the revision number of the 7811, represented as an unsigned 32-bit integer, initially 0x00000001 (for samples). For production parts this would be 0x00000010. This register is the one the host software needs to read, to determine which patches if any to load.

Protected mode. This Read Only register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is unchanged.

Figure 92. Revision Number register

EEPROM Data	**(Read Only)**	**Base1+0x9C**

31 — 0

EEPROM Data

32

Bit Field	Description
31:0	*EEPROM Data.* Contains 32 bits of user-defined data from the EEPROM.

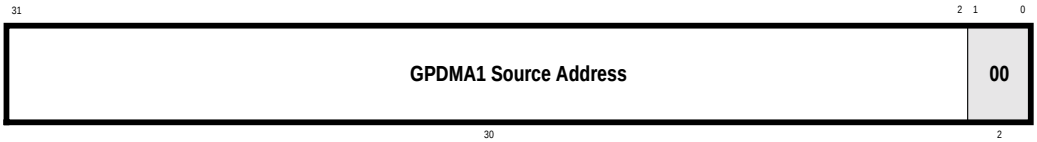
Register Description

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, bits [31:16] are loaded from EEPROM address 0x26. Bits [15:0] are loaded from EEPROM address 0x25.

Figure 93. EEPROM data register

GPDMA1 Source Address	(Read/Write)	Base1+0xA0



Bit Field	Description
31:0	*GPDMA1 Source Address*

Register Description

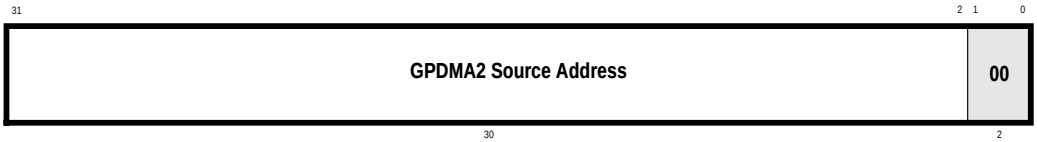
This register should not be modified by software unless the GPDMA1 unit is disabled. Of the four GPDMA Source Address registers, this is the only one that is accessible to the host in Protected mode. The others are controlled by the MIPS processor alone.

Protected mode. This register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is 0x00000000.

Figure 94. GPDMA1 Source Address register

GPDMA2 Source Address	(Read/Write)	Base1+0xA4



Bit Field	Description
31:0	*GPDMA Source Address.*

Register Description

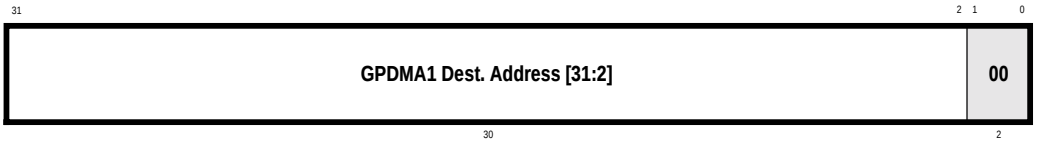
This register contains a pointer to a list of GPDMA2 source commands. See the description of the GPDMA Source Address register for details. Unlike the GPDMA1 Source Address register, this register is not accessible to the PCI bus in protected mode

Protected mode. In protected mode, the PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 95. GPDMA2 Source Address register

GPDMA1 Dest Address	(Read/Write)	Base1+0xA8



Bit Field	Description
31:0	*GPDMA1 Dest Address.*

Register Description

This register contains a pointer to a list of GPDMA1 destination commands. It is initialized by software and is updated by the GPDMA1 unit thereafter. For example, once the current command has finished executing, the GPDMA1 unit will add 16 to the register so it will point to the next command. The Jump and Skip16 bits of the Dest command can also affect this register.

This is a dword-aligned byte address. That is, bits 1:0 must be 0b00.

This register should not be modified by software unless the GPDMA1 unit is disabled.

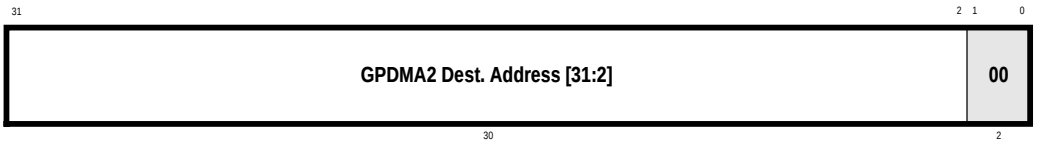
Of the four GPDMA Dest Address registers, only the GPDMA4 Dest Address register is accessible to the host in Protected mode. The Dest registers for GPDMA units 1-3 are controlled by the MIPS processor alone.

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 96. GPDMA1 Dest Address register

GPDMA2 Dest Address	(Read/Write)	Base1+0xAC



Bit Field	Description
31:0	*GPDMA2* Dest *Address.*

Register Description

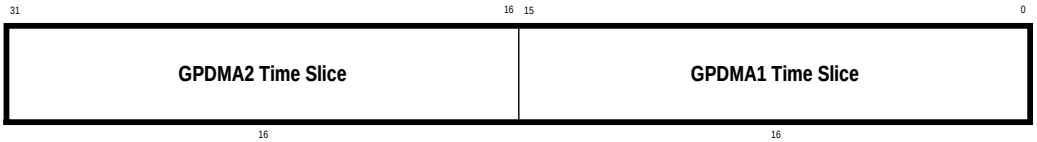
This register contains a pointer to a list of GPDMA2 destination commands. See the description of the GPDMA1 register.

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 97. GPDMA2 Dest Address register

GPDMA1_2 Arbitration	(Read/Write)	Base1+0xB0



Bit Field	Description
31:16	*GPDMA2 Time Slice.* The amount of time that GPDMA2 can run before relinquishing control to GPDMA1.
15:0	*GPDMA1 Time Slice.* The amount of time that GPDMA1 can run before relinquishing control to GPDMA2.

Register Description

This register sets the time-slice arbitration between GPDMA1 and GPDMA2. The time-slice unit is 16 system clock cycles. When a time slice expires, control will pass to the other GPDMA unit after current command has completed.

If one of the GPDMA units is disabled or idle, such as when it is waiting for the Source Valid bit to be set in the command, it relinquishes its time slice without waiting for a timeout.

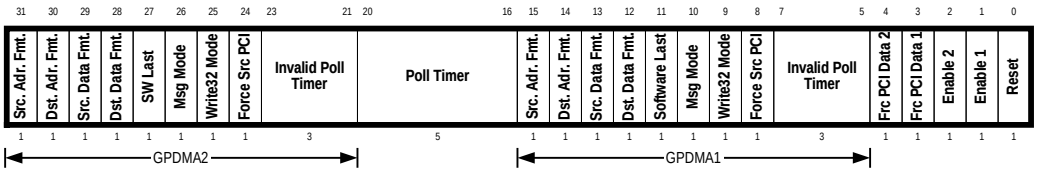
Under no circumstances must the time slice value for either GPDMA be less than the value of the "Poll Timer" of the GPDMA1_2 Config register. Setting either GPDMA1 time slice or GPDMA2 time slice to a value less than the "Poll Timer" value will disable that GPDMA from ever getting an opportunity to transfer data.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 98. GPDMA1_2 Arbitration register

GPDMA1_2 Config **(Read/Write)** **Base1+0xB4**

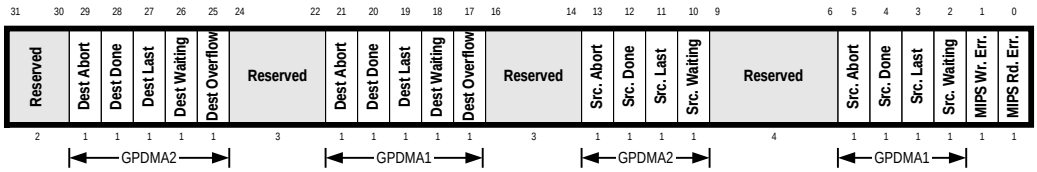


Bits	Description
31	*GPDMA2 Source Address Format.* Selects address type for the GPDMA2 Source Address register. 1 = MIPS address, 0 = PCI address.
30	*GPDMA2* Dest *Address Format.* Selects address type for the GPDMA2 Dest Address register. 1 = MIPS address, 0 = PCI address.
29	*GPDMA2 Source Command Format.* Selects endianness for GPDMA2 Source data. 1 = big-endian, 0 = little-endian. Applies to commands only. **
28	*GPDMA2 Dest Command Format.* Selects endianness for GPDMA2 Dest data. 1 = big-endian, 0 = little-endian. Applies to commands only. **
27	*GPDMA2 Software Last.* 1 = Software Last mode, 0 = Hardware Last mode. Applies only to Dest commands, and only in Data mode. There are two ways of dealing with buffer fragmentation on the Dest side. In Software Last mode, the Last bit in Dest commands is set by software to indicate the final fragment of an output buffer. The GPDMA unit never alters the Last bit on Dest commands in this mode. Software Last mode is used when the host's operating system (for example, Windows NT) allocates fragmented memory buffers to be used as a single virtual buffer. A separate Dest Command must be created to point to each buffer fragment and give its size. In Hardware Last mode, the GPDMA unit ignores the initial state of the Last bit in Dest commands. If the data overflows the data buffer of the current command, the GPDMA unit clears the Last bit in the current Dest command, advances to the next Dest command, and begins writing to its buffer. When the last byte of data is written, the GPDMA unit sets the Last bit of the final Dest command. Hardware Last mode is typically used where the OS returns non-fragmented buffers.
26	*GPDMA2 Message Mode.* Determines the mode of GPDMA2. 0 = Data mode, 1 = Message mode.
25	*GPDMA2 Write32 Mode*. Write32 mode specifies that, on commands with the Source Last bit set, the 32-bit data in Source Param 0 is written to the 32-bit address in Source Param 1. This is a flexible semaphore and control mechanism. 1 = Write32 mode enabled, 0 = Write32 mode disabled.
24	*Force GPDMA2 Source to PCI.* Force all Source command addresses on GPDMA2 to be PCI addresses, regardless of where they fall in the memory map. 1 = force, 0 = don't force.
23:21	*GPDMA2 Invalid Poll Timer.* This three-bit field determines how frequently a command will be polled to see if the Valid bit has been set. Polling will occur according to this field or the Poll Timer field, whichever gives the longer interval. This is used to reduce the frequency of polling while waiting for a valid descriptor, while polling more frequently when valid descriptors are available. This field is encoded as follows (in terms of engine clock cycles): 0 = 0 cycles, 1 = 256 cycles, 2 = 512 cycles, 3 = 768 cycles 4 = 1024 cycles, 5 = 1280 cycles, 6 = 1536 cycles, 7 = 1792 cycles.
20:16	*GPDMA1_2 Poll Timer.* Sets the polling frequency for GPDMA1 and GPDMA2. This is the rate at which the Valid bits in the Source and Dest commands are tested, in units of 16 engine clock cycles. **The value of this timer must always be less than either of GPDMA1 time slice or GPDMA2 time slice for correct operation.**
15	*GPDMA1 Source Address Format.* Selects address type for the GPDMA1 Source Address register. 1 = MIPS address, 0 = PCI address.

GPDMA1_2 Config	**(Read/Write)**	**Base1+0xB4**
14	*GPDMA1* Dest *Address Format.* Selects address type for the GPDMA1 Dest Address register. 1 = MIPS address, 0 = PCI address.	
13	*GPDMA1 Src. Command Format.* Selects endianness for GPDMA1 Source data. 1 = big-endian, 0 = little-endian. Applies to commands only. **	
12	*GPDMA1* Dest *Command Format.* Selects endianness for GPDMA1 Dest data. 1 = big-endian, 0 = little-endian. Applies to commands only. **	
11	*GPDMA1 Software Last.* 1 = Software Last mode, 0 = Hardware Last mode.	
10	*GPDMA1Message Mode.* Determines the mode of GPDMA1. 0 = Data mode, 1 = Message mode.	
9	*GPDMA1 Write32 Mode*. Write32 mode specifies that, on commands with the Source Last bit set, the 32-bit data in Source Param 0 is written to the 32-bit address in Source Param 1. This is a flexible semaphore and control mechanism. 1 = Write32 mode enabled, 0 = Write32 mode disabled.	
8	*Force GPDMA1 Source to PCI.* Force all Source command addresses on GPDMA1 to be PCI addresses, regardless of where they fall in the memory map. 1 = force, 0 = don't force.	
7:5	*GPDMA1 Invalid Poll Timer.* This three-bit field determines how frequently a command will be polled to see if the Valid bit has been set. Polling will occur according to this field or the Poll Timer field, whichever gives the longer interval. This is used to reduce the frequency of polling while waiting for a valid descriptor, while polling more frequently when valid descriptors are available. This field is encoded as follows (in terms of engine clock cycles): 0 = 0 cycles, 1 = 256 cycles, 2 = 512 cycles, 3 = 768 cycles 4 = 1024 cycles, 5 = 1280 cycles, 6 = 1536 cycles, 7 = 1792 cycles.	
4	*Force GPDMA2 Data to PCI.* Force data buffer accesses to the PCI bus, regardless of where they fall in the memory map. 1 = force, 0 = don't force.	
3	*Force GPDMA1 Data to PCI.* Force data buffer accesses to the PCI bus, regardless of where they fall in the memory map. 1 = force, 0 = don't force.	
2	*GPDMA2 Enable.* 1 = GPDMA2 enabled, 0 = GPDMA2 disabled.	
1	*GPDMA1 Enable.* 1 = GPDMA1 enabled, 0 = GPDMA1 disabled.	
0	*GPDMA1_2 Reset.* Resets GPDMA1 and GPDMA2. 0 = reset, 1 = don't reset. **This should not be performed until both engines have been disabled for one millisecond**.	
This register sets the operating parameters for GPDMA1 and GPDMA2. ** Setting any of these bits to 1 has no effect on data stored in the GPRAM. They only affect data in the PCI memory space. *Protected mode.* In protected mode, PCI reads return zero. PCI writes are ignored. *On reset*, the value of this register is 0x00000000.		

Figure 99. GPDMA1_2 Config register

GPDMA1_2 Status **(Read/Write)** **Base1+0xB8**



Bit Field	Description
31:30	*Reserved*. Must be set to zero.
29	*GPDMA2* Dest *Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred
28	*GPDMA2* Dest *Done.* 1 = A Dest command has completed. 0 = No Dest command has completed.
27	*GPDMA2 Dest Last.* 1 = A Dest Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.
26	*GPDMA2 Dest Waiting.* 1 = The GPDMA unit is waiting for a Dest Valid bit or Bytecount to become nonzero on Dest 0 = the GPDMA unit is not waiting for Dest Valid or Bytecount.
25	*GPDMA2 Dest Overflow*. 1 = The Dest data buffer overflowed and at least one byte of data was lost. 0 = No overflow occurred. Overflow is a meaningful condition only in Software Last mode. Overflow takes place when there is more data than will fit in the Dest command's data buffer, but the Dest Last bit is set, indicating that the GPDMA unit is not to use the next command's buffer for the remaining data.
24:22	*Reserved.* Must be set to zero.
21	*GPDMA1* Dest *Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred
20	*GPDMA1* Dest *Done.* 1 = A Dest command has completed. 0 = No Dest command has completed.
19	*GPDMA1* Dest *Last.* 1 = A Dest Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.
18	*GPDMA1 Dest Waiting.* 1 = The GPDMA unit is waiting for a Source Valid bit or Bytecount to become nonzero on Dest 0 = the GPDMA unit is not waiting for Dest Valid or Bytecount.
17	*GPDMA1 Dest Overflow*. 1 = The Dest data buffer overflowed and at least one byte of data was lost. 0 = No overflow occurred. Overflow is a meaningful condition only in Software Last mode. Overflow takes place when there is more data than will fit in the Dest command's data buffer, but the Dest Last bit is set, indicating that the GPDMA unit is not to use the next command's buffer for the remaining data.
16:14	*Reserved.* Must be set to zero.
13	*GPDMA2 Source Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred
12	*GPDMA2 Source Done.* 1 = A Source command has completed. 0 = No Dest command has completed.
11	*GPDMA2 Source Last.* 1 = A Source Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.
10	*GPDMA2 Source Waiting.* 1 = The GPDMA unit is waiting for a Source Valid bit or Bytecount to become nonzero on Dest 0 = the GPDMA unit is not waiting for Source Valid or Bytecount.
9:6	*Reserved.* Must be set to zero.
5	*GPDMA1 Source Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred
4	*GPDMA1 Source Done.* 1 = A Source command has completed. 0 = No Dest command has

GPDMA1_2 Status	(Read/Write)	Base1+0xB8
	completed.	
3	*GPDMA1 Source Last.* 1 = A Source Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.	
2	*GPDMA1 Source Waiting.* 1 = The GPDMA unit is waiting for a Source Valid bit or Bytecount to become nonzero on Dest 0 = the GPDMA unit is not waiting for Source Valid or Bytecount.	
1	*MIPS Write Error.* A fatal error occurred during a MIPS write to PCI.	
0	*MIPS Read Error.* A fatal error occurred during a MIPS read from PCI.	

This register reports the status of GPDMA1 and GPDMA2. All bits in this register are persistent. Once set, they remain set until cleared. To clear a bit, write a one to this register in the desired bit position. For example, to clear the GPDMA1 Source Done bit, write a 1 to bit 4 of this register. Writing a zero has no effect on the corresponding register bit.

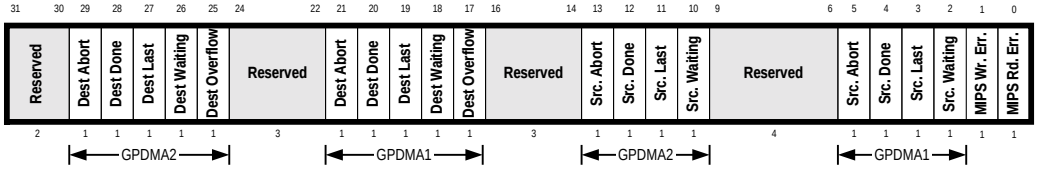
This register is intended to support interrupt service routines. If software polling is necessary, the Global Status register should be polled instead.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 100. GPDMA1_2 Status register

GPDMA1_2 Interrupt Enable (Read/Write) Base1+0xBC



Bit Field	Description
31:30	*Reserved*. Must be set to zero.
29	GPDMA2 Dest Abort.
28	GPDMA2 Dest Done.
27	GPDMA2 Dest Last.
26	GPDMA2 Dest Waiting.
25	GPDMA2 Dest Overflow.
24:22	*Reserved.* Must be set to zero.
21	GPDMA1 Dest Abort.
20	GPDMA1 Dest Done.
19	GPDMA1 Dest Last.
18	GPDMA1 Dest Waiting.
17	GPDMA1 Dest Overflow.
16:14	*Reserved.* Must be set to zero.
13	GPDMA2 Source Abort.
12	GPDMA2 Source Done.
11	GPDMA2 Source Last.
10	GPDMA2 Source Waiting.
9:6	*Reserved.* Must be set to zero.
5	GPDMA1 Source Abort.
4	GPDMA1 Source Done.
3	GPDMA1 Source Last.
2	GPDMA1 Source Waiting.
1	MIPS Write Error.
0	MIPS Read Error.

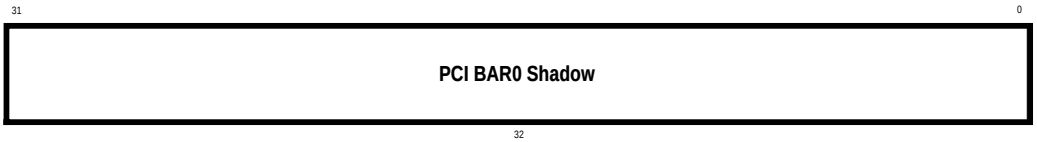
This register enables interrupts on the selected conditions. Its structure is the same as the GPDMA1_2 Status register. A one bit enables the selected interrupt, and a zero bit disables it. The interrupt is sent to PCI, to the MIPS processor, or both. This is determined by the MIPS Config register.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 101. GPDMA1_2 Interrupt Enable register

PCI BAR0 Shadow	(Read Only)	Base1+0xC0



Bit Field	Description
31:0	*BAR0 Shadow.* Contains a Read Only copy of the PCI Base Address 0 pointer for this device.

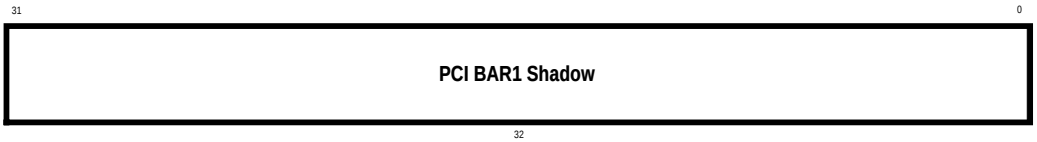
Register Description

Protected mode. This Read Only register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is undefined.

Figure 102. PCI BAR0 Shadow register

PCI BAR1 Shadow	(Read Only)	Base1+0xC4



Bit Field	Description
31:0	*BAR1 Shadow.* Contains a Read Only copy of the PCI Base Address 1 pointer for this device.

Register Description

Protected mode. This Read Only register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is undefined.

Figure 103. PCI BAR1 Shadow register

PCI BAR2 Shadow	(Read Only)	Base1+0xC8

31 — 0

PCI BAR2 Shadow

32

Bit Field	Description
31:0	*BAR2 Shadow.* Contains a Read Only copy of the PCI Base Address 2 pointer for this device.

Register Description

Protected mode. This Read Only register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is undefined.

Figure 104. PCI BAR2 Shadow register

SDRAM Config	(Read Only)	Base1+0xCC

31 — 0

SDRAM Config

32

Bit Field	Description
31:0	*SDRAM Config*. Contains a Read Only copy of the EEPROM SDRAM timing and configuration parameters.

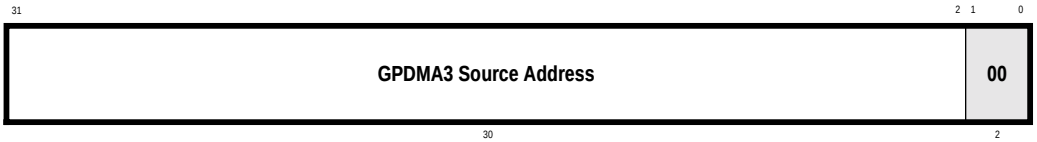
Register Description

Protected mode. In protected mode, this register is accessible only to the MIPS processor.

On reset, bits [30:16] are loaded from EEPROM address 0x11 (bit [31] is forced to one). Bits [14:0] are loaded from EEPROM address 0x10 (bit [15] is forced to one).

Figure 105. SDRAM Config register

GPDMA3 Source Address	(Read/Write)	Base1+0xD0



Bit Field	Description
31:0	*GPDMA3 Source Address.*

Register Description

This register contains a pointer to a list of GPDMA3 source commands. See the description of the GPDMA1 Source Address register for details.

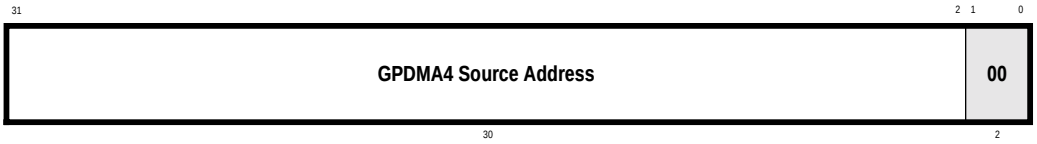
Unlike the GPDMA1 Source Address register, this register is not accessible to PCI in protected mode.

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 106. GPDMA3 Source Address register

GPDMA4 Source Address	(Read/Write)	Base1+0xD4



Bit Field	Description
31:0	*GPDMA4 Source Address.*

Register Description

This register contains a pointer to a list of GPDMA4 source commands. See the description of the GPDMA1 Source Address register for details.

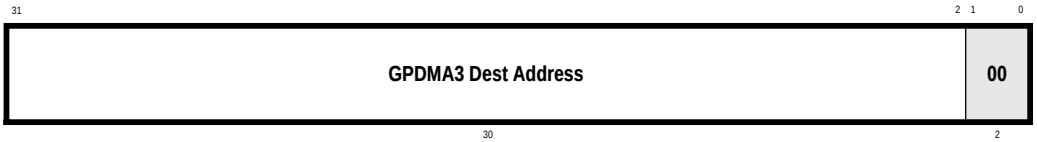
Unlike the GPDMA1 Source Address register, this register is not accessible to PCI in protected mode.

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 107. GPDMA4 Source Address register

GPDMA3 Dest Address	(Read/Write)	Base1+0xD8



Bit Field	Description
31:0	*GPDMA3 Dest Address.*

Register Description

This register contains a pointer to a list of GPDMA3 destination commands. It is initialized by software, and is updated by the GPDMA3 unit thereafter. For example, once the current command has finished executing, the GPDMA3 unit will add 16 to the register so it will point to the next command. The Jump and Skip16 bits of the Dest command can also affect this register.

This register contains a byte address. Any byte address is valid; there are no alignment restrictions on GPDMA commands.

This register should not be modified by software unless the GPDMA3 unit is disabled.

Of the four GPDMA Dest Address registers, only the GPDMA4 Dest Address register is accessible to the host in Protected mode. The Dest registers for GPDMA units 1-3 are controlled by the MIPS processor alone.

Protected mode. In protected mode, PCI reads return zero and PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 108. GPDMA3 Dest Address register

GPDMA4 Dest Address	(Read/Write)	Base1+0xDC

Bit Field	Description
31:0	*GPDMA4* Dest *Address.*

Register Description

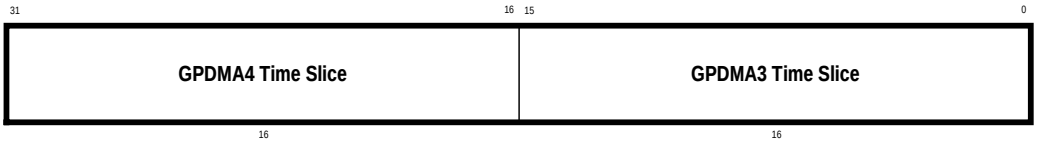
This register contains a pointer to a list of GPDMA4 destination commands. See the description of the GPDMA3 Dest Address register. Unlike the GPDMA3 Dest Address register, however, this register is accessible to PCI in protected mode.

Protected mode. This register is fully accessible to PCI in both protected and unprotected modes.

On reset, the value of this register is 0x00000000.

Figure 109. GPDMA4 Dest Address register

GPDMA3_4 Arbitration	(Read/Write)	Base1+0xE0



Bit Field	Description
31:16	*GPDMA4 Time Slice.* The amount of time that GPDMA4 can run before relinquishing control to GPDMA3.
15:0	*GPDMA3 Time Slice.* The amount of time that GPDMA3 can run before relinquishing control to GPDMA4.

Register Description

This register sets the time-slice arbitration between GPDMA3 and GPDMA4. The time-slice unit is 16 system clock cycles. When a time slice expires, control will pass to the other GPDMA unit when the current command has completed.

If one of the GPDMA units is disabled or idle, such as when it is waiting for the Source Valid bit to be set in the command, it relinquishes its time slice without waiting for a timeout.

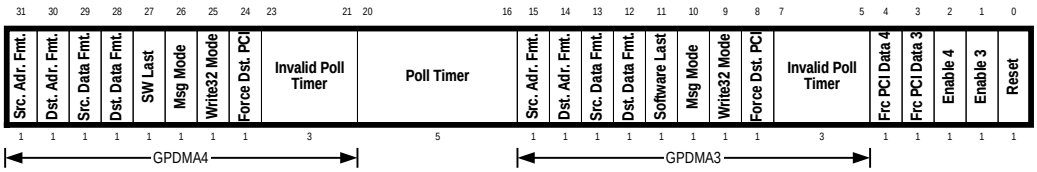
Under no circumstances must the time slice value for either GPDMA be less than the value of the "Poll Timer" of the GPDMA3_4 Config register. Setting either GPDMA3 time slice or GPDMA4 time slice to a value less than the "Poll Timer" value will disable that GPDMA from ever getting an opportunity to transfer data.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 110. GPDMA3_4 Arbitration register

GPDMA3_4 Config	(Read/Write)	Base1+0xE4



Bits	Description
31	*GPDMA4 Source Address Format.* Selects address type for the GPDMA4 Source Address register. 1 = MIPS address, 0 = PCI address.
30	*GPDMA4* Dest *Address Format.* Selects address type for the GPDMA4 Dest Address register. 1 = MIPS address, 0 = PCI address.
29	*GPDMA4 Src. Command Format.* Selects endianness for GPDMA4 Source data. 1 = big-endian, 0 = little-endian. Applies to commands only. **
28	*GPDMA4 Dest Command Format.* Selects endianness for GPDMA4 Dest data. 1 = big-endian, 0 = little-endian. Applies to commands only. **
27	*GPDMA4 Software Last.* 1 = Software Last mode, 0 = Hardware Last mode. Applies only to Dest commands, and only in Data mode. There are two ways of dealing with buffer fragmentation on the Dest side. In Software Last mode, the Last bit in Dest commands is set by software to indicate the final fragment of an output buffer. The GPDMA unit never alters the Last bit on Dest commands in this mode. Software Last mode is used when the host's operating system (for example, Windows NT) allocates fragmented memory buffers to be used as a single virtual buffer. A separate Dest Command must be created to point to each buffer fragment and give its size. In Hardware Last mode, the GPDMA unit ignores the initial state of the Last bit in Dest commands. If the data overflows the data buffer of the current command, the GPDMA unit clears the Last bit in the current Dest command, advances to the next Dest command, and begins writing to its buffer. When the last byte of data is written, the GPDMA unit sets the Last bit of the final Dest command. Hardware Last mode is typically used where the OS returns non-fragmented buffers.
26	*GPDMA4 Message Mode.* Determines the mode of GPDMA4. 0 = Data mode, 1 = Message mode.
25	*GPDMA4 Write32 Mode*. Write32 mode specifies that, on commands with the Source Last bit set, the 32-bit data in Source Param 0 is written to the 32-bit address in Source Param 1. This is a flexible semaphore and control mechanism. 1 = Write32 mode enabled, 0 = Write32 mode disabled.
24	*Force GPDMA4 Dest to PCI.* Force all GPDMA Dest accesses to the PCI bus, regardless of where they fall in the memory map. 1 = force, 0 = don't force.
23:21	*GPDMA4 Invalid Poll Timer.* This three-bit field determines how frequently a command will be polled to see if the Valid bit has been set. Polling will occur according to this field or the Poll Timer field, whichever gives the longer interval. This is used to reduce the frequency of polling while waiting for a valid descriptor, while polling more frequently when valid descriptors are available. This field is encoded as follows (in terms of engine clock cycles): 0 = 0 cycles, 1 = 256 cycles, 2 = 512 cycles, 3 = 768 cycles 4 = 1024 cycles, 5 = 1280 cycles, 6 = 1536 cycles, 7 = 1792 cycles.
20:16	*GPDMA3_4 Poll Timer.* Sets the polling frequency for GPDMA3 and GPDMA4. This is the rate at which the Valid bits in the Source and Dest commands are tested, in units of 16 engine clock cycles. . **The value of this timer must always be less than either of GPDMA3 time slice or GPDMA4 time slice for correct operation.**
15	*GPDMA3 Source Address Format.* Selects address type for the GPDMA3 Source Address register. 1 = MIPS address, 0 = PCI address.

GPDMA3_4 Config	(Read/Write)	Base1+0xE4
14	*GPDMA3* Dest *Address Format.* Selects address type for the GPDMA3 Dest Address register. 1 = MIPS address, 0 = PCI address. This should not be performed until both engines have been disabled for one millisecond.	
13	*GPDMA3 Src. Command Format.* Selects endianness for GPDMA3 Source data. 1 = big-endian, 0 = little-endian. Applies to commands only. **	
12	*GPDMA3* Dest *Command Format.* Selects endianness for GPDMA3 Dest data. 1 = big-endian, 0 = little-endian. Applies to commands only. **	
11	*GPDMA3 Software Last.* 1 = Software Last mode, 0 = Hardware Last mode.	
10	*GPDMA3 Message Mode.* Determines the mode of GPDMA3. 0 = Data mode, 1 = Message mode.	
9	*GPDMA3 Write32 Mode*. Write32 mode specifies that, on commands with the Source Last bit set, the 32-bit data in Source Param 0 is written to the 32-bit address in Source Param 1. This is a flexible semaphore and control mechanism. 1 = Write32 mode enabled, 0 = Write32 mode disabled.	
8	*Force GPDMA3 Dest to PCI.* Force all GPDMA3 Dest accesses to the PCI bus, regardless of where they fall in the memory map. 1 = force, 0 = don't force.	
7:5	*GPDMA3 Invalid Poll Timer.* This three-bit field determines how frequently a command will be polled to see if the Valid bit has been set. Polling will occur according to this field or the Poll Timer field, whichever gives the longer interval. This is used to reduce the frequency of polling while waiting for a valid descriptor, while polling more frequently when valid descriptors are available. This field is encoded as follows (in terms of engine clock cycles): 0 = 0 cycles, 1 = 256 cycles, 2 = 512 cycles, 3 = 768 cycles 4 = 1024 cycles, 5 = 1280 cycles, 6 = 1536 cycles, 7 = 1792 cycles.	
4	*Force GPDMA4 Data to PCI.* Force all data buffers on GPDMA4 to the PCI bus, regardless of where they fall in the memory map. 1 = force, 0 = don't force.	
3	*Force GPDMA3 Data to PCI.* Force all data buffers on GPDMA3 to be the PCI bus, regardless of where they fall in the memory map. 1 = force, 0 = don't force.	
2	*GPDMA4 Active.* 1 = GPDMA4 enabled, 0 = GPDMA4 disabled.	
1	*GPDMA3 Active.* 1 = GPDMA3 enabled, 0 = GPDMA3 disabled.	
0	*GPDMA3_4 Reset.* Resets GPDMA3 and GPDMA4. 0 = reset, 1 = don't reset. **This should not be performed until both engines have been disabled for one millisecond**.	

This register sets the operating parameters for GPDMA3 and GPDMA4.

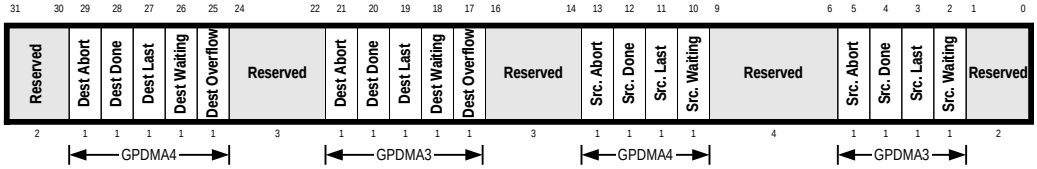
** Setting any of these bits to 1 has no effect on data stored in the GPRAM. They only affect data in the PCI memory space.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 111. GPDMA3_4 Config register

GPDMA3_4 Status **(Read/Write)** **Base1+0xE8**



Bit Field	Description
31:30	*Reserved*. Must be set to zero.
29	*GPDMA4* Dest *Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred
28	*GPDMA4* Dest *Done.* 1 = A Dest command has completed. 0 = No Dest command has completed.
27	*GPDMA4 Dest Last.* 1 = A Dest Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.
26	*GPDMA4 Dest Waiting.* 1 = The GPDMA unit is waiting for a Dest Valid bit or Bytecount to become nonzero on Dest. 0 = the GPDMA unit is not waiting for Dest Valid or Bytecount.
25	*GPDMA4 Dest Overflow*. 1 = The Dest data buffer overflowed and at least one byte of data was lost. 0 = No overflow occurred. Overflow is a meaningful condition only in Software Last mode. Overflow takes place when there is more data than will fit in the Dest command's data buffer, but the Dest Last bit is set, indicating that the GPDMA unit is not to use the next command's buffer for the remaining data.
24:22	*Reserved.* Must be set to zero.
21	*GPDMA3* Dest *Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred
20	*GPDMA3* Dest *Done.* 1 = A Dest command has completed. 0 = No Dest command has completed.
19	*GPDMA3* Dest *Last.* 1 = A Dest Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.
18	*GPDMA3 Dest Waiting.* 1 = The GPDMA unit is waiting for a Source Valid bit or Bytecount to become nonzero on Dest. 0 = the GPDMA unit is not waiting for Dest Valid or Bytecount.
17	*GPDMA3* Dest *Overflow*. 1 = The Dest data buffer overflowed and at least one byte of data was lost. 0 = No overflow occurred. Overflow is a meaningful condition only in Software Last mode. Overflow takes place when there is more data than will fit in the Dest command's data buffer, but the Dest Last bit is set, indicating that the GPDMA unit is not to use the next command's buffer for the remaining data.
16:14	*Reserved.* Must be set to zero.
13	*GPDMA4 Source Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred
12	*GPDMA4 Source Done.* 1 = A Source command has completed. 0 = No Dest command has completed.
11	*GPDMA4 Source Last.* 1 = A Source Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.
10	*GPDMA4 Source Waiting.* 1 = The GPDMA unit is waiting for a Source Valid bit or Bytecount to become nonzero on Dest. 0 = the GPDMA unit is not waiting for Source Valid or Bytecount.
9:6	*Reserved.* Must be set to zero.
5	*GPDMA3 Source Abort.* 1 = A PCI Initiator Abort or Target Abort occurred. 0 = No abort has occurred

GPDMA3_4 Status	(Read/Write)	Base1+0xE8
4	*GPDMA3 Source Done.* 1 = A Source command has completed. 0 = No Dest command has completed.	
3	*GPDMA3 Source Last.* 1 = A Source Last bit of one has been encountered. 0 = No Dest Last bit of one has been encountered.	
2	*GPDMA3 Source Waiting.* 1 = The GPDMA unit is waiting for a Source Valid bit or Bytecount to become nonzero on Dest. 0 = the GPDMA unit is not waiting for Source Valid or Bytecount.	
1:0	*Reserved.* Must be set to zero.	

This register reports the status of GPDMA3 and GPDMA4. All bits in this register are persistent. Once set, they remain set until cleared. To clear a bit, write a one to this register in the desired bit position. For example, to clear the GPDMA3 Source Done bit, write a 1 to bit 4 of this register. Writing a zero has no effect on the corresponding register bit.

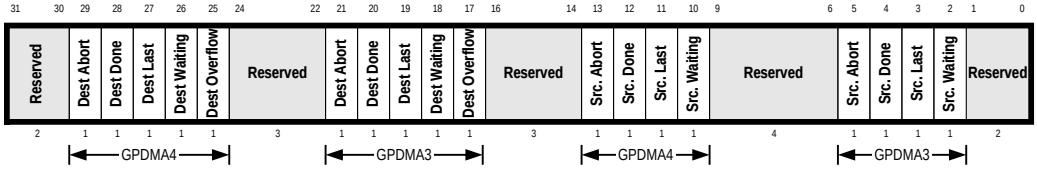
This register is intended to support interrupt service routines. If software polling is necessary, the Global Status register should be polled instead.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 112. GPDMA3_4 Status register

GPDMA3_4 Interrupt Enable (Read/Write) Base1+0xEC



Bit Field	Description
31:30	*Reserved*. Must be set to zero.
29	GPDMA4 Dest Abort.
28	GPDMA4 Dest Done.
27	GPDMA4 Dest Last.
26	GPDMA4 Dest Waiting.
25	GPDMA4 Dest Overflow.
24:22	*Reserved.* Must be set to zero.
21	GPDMA3 Dest Abort.
20	GPDMA3 Dest Done.
19	GPDMA3 Dest Last.
18	GPDMA3 Dest Waiting.
17	GPDMA3 Dest Overflow.
16:14	*Reserved.* Must be set to zero.
13	GPDMA4 Source Abort.
12	GPDMA4 Source Done.
11	GPDMA4 Source Last.
10	GPDMA4 Source Waiting.
9:6	*Reserved.* Must be set to zero.
5	GPDMA3 Source Abort.
4	GPDMA3 Source Done.
3	GPDMA3 Source Last.
2	GPDMA3 Source Waiting.
1:0	*Reserved.* Must be set to zero.

This register enables interrupts on the selected conditions. Its structure is the same as the GPDMA3_4 Status register. A one bit enables the selected interrupt, and a zero bit disables it. The interrupt is sent to PCI, to the MIPS processor, or both. This is determined by the MIPS Config register.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 113. GPDMA3_4 Interrupt Enable register

Global Status	**(Read/Write)**	**Base1+0xF0**

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
SE Dst. Done	SE Dst. Wait	SE Res. Done	SE Res. Wait	SE Src. Done	SE Src. Wait	SE Cmd Done	SE Cmd Wait	Dest Done	Dest Wait	Src. Done	Src. Wait	Dest Done	Dest Wait	Src. Done	Src. Wait	Dest Done	Dest Wait	Src. Done	Src. Wait	Dest Done	Dest Wait	Src. Done	Src. Wait	Error	MIPS R/W Err.	SP Interrupt	PCI Interrupt	MIPS Interrupt	SE Interrupt	3_4 Interrupt	1_2 Interrupt
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
								GPDMA4				GPDMA3				GPDMA2				GPDMA1											

Bit Field	Description
31	*Security Engine* Dest *Done*. The Security Engine's Destination operation has completed and its Valid bit has been cleared.
30	*Security Engine* Dest *Waiting*. The Security Engine is waiting for the Valid bit to be set in the Dest descriptor.
29	*Security Engine Result Done*. The Security Engine's Result operation has completed and its Valid bit has been cleared.
28	*Security Engine Result. Waiting*. The Security Engine is waiting for the Valid bit to be set in the Result descriptor.
27	*Security Engine Source Done*. The Security Engine's Source operation has completed and its Valid bit has been cleared.
26	*Security Engine* Dest *Waiting*. The Security Engine is waiting for the Valid bit to be set in the Source descriptor.
25	*Security Engine Command Done*. The Security Engine's Command operation has completed and its Valid bit has been cleared.
24	*Security Engine Command Waiting*. The Security Engine is waiting for the Valid bit to be set in the Command descriptor.
23	*GPDMA4* Dest *Done*. The GPDMA4 Dest command has completed and its Valid bit has been cleared.
22	*GPDMA4* Dest *Wait*. GPDMA4 is waiting for the Valid bit to be set in the Dest command.
21	*GPDMA4 Source Done*. The GPDMA4 Source command has completed and its Valid bit has been cleared.
20	*GPDMA4 Source Wait*. GPDMA4 is waiting for the Valid bit to be set in the Source command.
19	*GPDMA3* Dest *Done*. The GPDMA4 Dest command has completed and its Valid bit has been cleared.
18	*GPDMA3 Dest Wait*. GPDMA4 is waiting for the Valid bit to be set in the Dest command.
17	*GPDMA3 Source Done*. The GPDMA4 Source command has completed and its Valid bit has been cleared.
16	*GPDMA3 Source Wait*. GPDMA4 is waiting for the Valid bit to be set in the Source command.
15	*GPDMA2 Dest Done*. The GPDMA4 Dest command has completed and its Valid bit has been cleared.
14	*GPDMA2 Dest Wait*. GPDMA4 is waiting for the Valid bit to be set in the Dest command.
13	*GPDMA2 Source Done*. The GPDMA4 Source command has completed and its Valid bit has been cleared.
12	*GPDMA2 Source Wait*. GPDMA4 is waiting for the Valid bit to be set in the Source command.
11	*GPDMA1* Dest *Done*. The GPDMA4 Dest command has completed and its Valid bit has been cleared.
10	*GPDMA1 Dest Wait*. GPDMA4 is waiting for the Valid bit to be set in the Dest command.
9	*GPDMA1 Source Done*. The GPDMA4 Source command has completed and its Valid bit has been cleared.
8	*GPDMA1 Source Wait*. GPDMA4 is waiting for the Valid bit to be set in the Source command.
7	*Error*. Set on any abort or overflow

Global Status	**(Read/Write)**	**Base1+0xF0**
6	*MIPS R/W Error.* A fatal error occurred during a MIPS transfer to or from PCI.	
5	*Security Pipeline Interrupt.* Set when an enabled interrupt occurs.	
4	*PCI Interrupt.* Set when an enabled interrupt occurs.	
3	*MIPS Interrupt.* Set when an enabled interrupt occurs.	
2	*Security Engine Interrupt.* Set when an enabled 7751 interrupt occurs.	
1	*GPDMA3_4 Interrupt.* Set when an enabled interrupt occurs.	
0	*GPDMA1_2 Interrupt.* Set when an enabled interrupt occurs.	

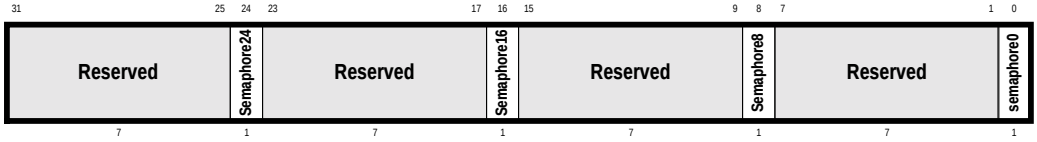
This register reports the status of various parts of the 7811. This register, not the Interrupt registers, should be used for status polling.

Protected mode. In protected mode, PCI reads return zero. PCI writes are ignored.

On reset, the value of this register is 0x00000000.

Figure 114. Global Status register

Test-and-Set	**(Read/Write)**	**Base1+0xF8**



Bit Field	**Description**
31:25	*Reserved.* Must be set to zero.
24	*Semaphore24*. Writing a one to this bit sets it to zero; writing a zero to this bit has no effect.
23:17	*Reserved.* Must be set to zero.
16	*Semaphore16*. Writing a one to this bit sets it to zero; writing a zero to this bit has no effect.
15:9	*Reserved.* Must be set to zero.
8	*Semaphore8*. Writing a one to this bit sets it to zero; writing a zero to this bit has no effect.
7:1	*Reserved.* Must be set to zero.
0	*Semaphore0*. Writing a one to this bit sets it to zero; writing a zero to this bit has no effect.

Register Description

This register implements a simple destructive-read semaphore protocol, which would typically be used between the host and MIPS processors.

Reads must be 32 bits wide. Writes can be byte, word, or dword writes. Reading this register returns its current state and then sets all four active bits (Semaphore24, Semaphore16, Semaphore8, and Semaphore0) to one. Writes a one to the active bits clears them; writing zeroes to them has no effect.

Protected mode. This register is fully accessible in both protected and unprotected modes.

On reset, the value of this register is 0x00000000.

Figure 115. Test-and-Set register

9 Clock Generation and Reset

9.1 Clock Generation

The MIPS_CLK input is the master clock for the 7811. It serves as the MIPS bus clock and as the input to an internal PLL. This PLL doubles the MIPS_CLK frequency, and the 2x clock drives the Security Engine, the Context SDRAM interface, and the GPRAM interface. Thus, a 7811 running with a 45 MHz MIPS_CLK has a 90 MHz Security Engine and 90 MHz SDRAM interfaces.

The MIPS_CLK must thus be provided whether a MIPS processor is used or not. Because MIPS_CLK is used as an input to a PLL, it should never be disabled.

PCI_CLK is the master clock for the PCI interface, which is fully asynchronous to the rest of the 7811. PCI_CLK should follow the PCI specification for 33 MHz bus operation.

9.2 Reset

PCI_RST# is the master reset signal for the 7811. PCI_RST# halts all the internal engines and resets most of the registers, leaving them with the values specified in the individual register descriptions PCI_RST# should meet the PCI specification for 33 MHz bus operation.

10 Testability

10.1 NAND Tree

The NAND tree in the 7811 starts with signal TEST_IN0, and ends at signal TEST_PLLOUT2. To enable this test mode, TEST0, TEST1, and TEST_IN2 must be set high, and TEST2 must be set low. The order of inputs in the NAND tree is as shown in figure Figure 116.

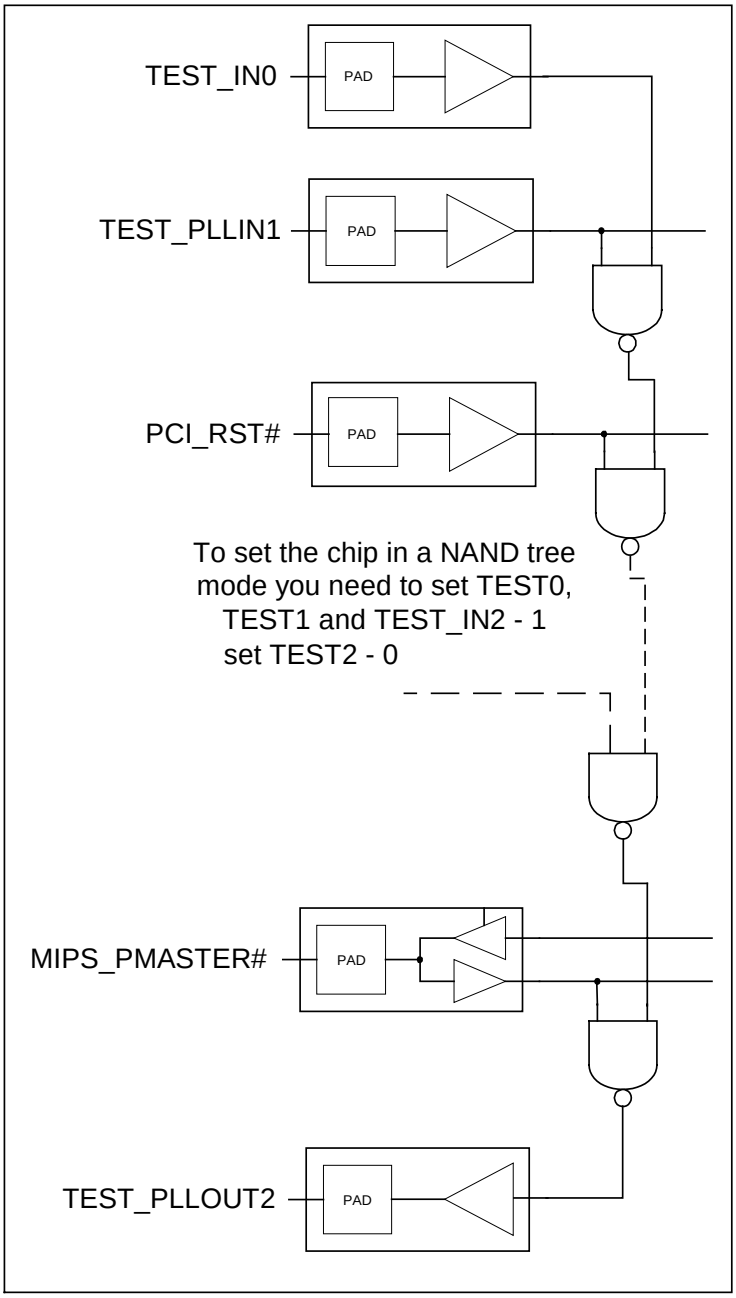


Figure 116. Pin ordering for NAND Tree

11 DC Specifications

11.1 Absolute Maximum Ratings

DC Supply Voltage (V_{DD})	-0.3 V to +4.5 V
DC Input Voltage (V_{IN})	-0.3 V to V_{DD}+0.3 V
DC Output Voltage (V_{OUT})	-0.3 V to V_{DD}+0.3 V
Storage Temperature (T_{stg})	-40°C to +125°C

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

Figure 117. Recommended operating conditions

11.2 Recommended Operating Conditions

DC Supply Voltage (V_{DD})	+3.0V to +3.6V
Operating Temperature (T_A)	0°C to +70°C
Junction Temperature (T_J)	0°C to +85°C

Figure 118. Recommended operating conditions

11.3 DC Specifications, 3.3V Interfaces

The following specifications are for the 3.3V interfaces. The PCI interface DC specifications are identical to that in the PCI Specification, version 2.1.

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{IL}	Low level input voltage (I, SI)				0.8	V
	Clock Input (CI)				0.8	V
V_{IH}	High level input voltage (I,SI)		2.0			V
	Clock Input (CI)		2.4			V
V_H	Schmitt hysteresis			0.8		V
I_{IL}	Low level input current (I,SI)	VIN = VSS	-10			µA
	With Pull-up (PI)	VDD = 3.6V	-40			µA
I_{IH}	High level input current	VIN = VDD VDD = 3.3V			10	µA
V_{OL}	Low level output voltage	VDD = 3.0V				
	(02)	IOL = 2mA			0.4	V
	(04)	IOL = 4mA			0.4	V
	(06)	IOL = 6mA			0.4	V
	(08)	IOL = 8mA			0.4	V
V_{OH}	High level output voltage	VDD = 3.0V				
	(02)	IOH = -2mA	2.4			V
	(04)	IOH = -4mA	2.4			V
	(06)	IOH = -6mA	2.4			V
	(08)	IOH = -8mA	2.4			V
I_{OZ}	High impedance leakage current	VO = VSS VDD = 3.6V	-10			µA
I_{DD}	Quiescent supply current				10	µA
C_{IN}	Input capacitance	VDD = 3.3V		2.4		pF
C_{OUT}	Output capacitance	VDD = 3.3V		5.6		pF
$C_{I/O}$	Input/Output capacitance	VDD = 3.3V		6.6		pF
P_A	Power dissipation	VDD = 3.3V			3.5	W

Figure 119. DC electrical characteristics

Symbol	Parameter	Conditions
CL2	Load on bus	50 pF
VDD	Supply voltage	3.3V ± 0.3V
VSS	Ground potential	0V
TA	Ambient operating temperature	0°C to +70°C

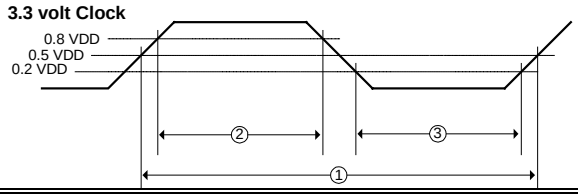
Figure 120. Test conditions

12 AC Specifications

Most of the 7811's signals are part of well-defined industry-standard interfaces, and comply to the timing requirements of these interfaces.

12.1 Master Clock (MIPS_CLK) Timing

The master clock for the 7811 is the MIPS_CLK pin. Its timing is shown in Figure 121. An internal PLL doubles the MIPS_CLK signal. This frequency-doubled clock is used for the SDRAM interface and Security Engine. The lock time for this PLL is also given in Figure 121.



Number	Description	Min	Max	Units
1	t_{cyc}, MIPS_CLK Cycle Time	22	40	ns
2	t_{high}, MIPS_CLK High Time	3.5		ns
3	t_{low}, MIPS_CLK Low Time	3.5		ns
-	MIPS_CLK Rise/Fall Time		4	ns
-	PLL Lock Time		100	µs

Figure 121. MIPS_CLK parameters

12.2 PCI Timing

The timing of the 7811's 32-bit, 33 MHz PCI interface is identical to that of the *PCI Local Bus Specification, Revision 2.1.*

12.3 MIPS Interface Timing

The timing of the MIPS interface is compatible with the MIPS R4300 processor. The MIPS interface speed is determined by MIPS_CLK.

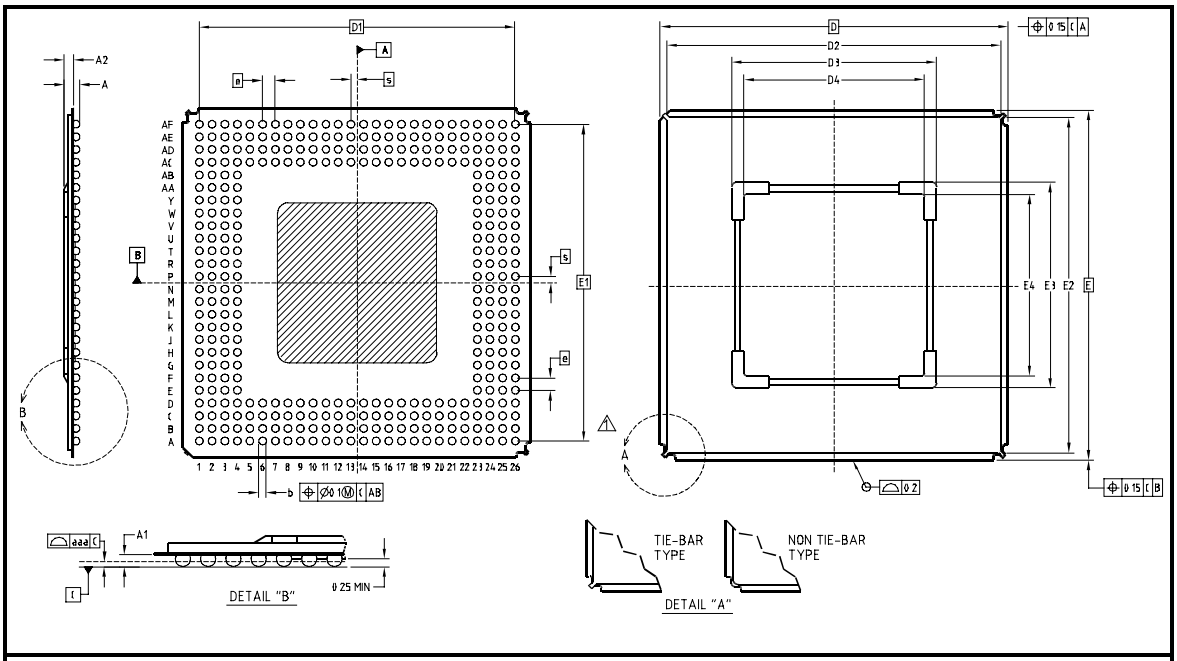
12.4 SDRAM Timing

When MIPS_CLK is run at 45 MHz or less, the timing of the SDRAM interface is compatible with 125 MHz JEDEC-compatible SDRAMs, as given in JEDEC Standard No. 21-C.

The 7811 can also support certain 100 Mhz JEDEC-compatible devices, one of them being the Toshiba TC59S6408BFT-10.

13 Physical Specifications

13.1 Package Dimensions



352-pin TBGA Package			
Symbol	**Min (mm)**	**Nom (mm)**	**Max (mm)**
A	-	-	1.7
A1	0.5	0.6	0.7
A2	-	-	1.0
b	0.60	0.75	0.90
D	35.0 BSC		
D1	31.75 BSC		
D2	33.4	33.6	33.8
D3			
D4			
E	35.0 BSC		
E1	31.75 BSC		
E2	33.4	33.6	33.8
E3			
E4			
e	1.27 BSC		
s	0.635 BSC		
aaa	0.15		

Figure 122. Package dimensions

THIS PAGE INTENTIONALLY BLANK

13.2 Pin Configuration

	A	B	C	D	E	F	G	H	J	K	L	M	N
26	MIPS_AD31	VDD	VSS	MIPS_AD27	MIPS_AD24	VSS	VDD	MIPS_AD17	PROTECT#	MIPS_CLK	NC	PLL_DGN	VSS
25	MIPS_AD30	MIPS_AD29	VSS	VDD	MIPS_AD25	MIPS_AD22	MIPS_AD19	MIPS_AD18	MIPS_EVALID#	VSS	PLL_DVD	PLL_AGD	VSS
24	VSS	VDD	VDD	MIPS_AD28	MIPS_AD26	MIPS_AD21	VDD	MIPS_AD16	MIPS_EOK#	VDD	VDD	TEST_PLL_OUT1	VDD
23	VSS	VSS	VSS	VSS	MIPS_AD23	MIPS_AD20	VSS	MIPS#	VSS	VSS	VDD	PLL_VAA	VSS
22	VSS	VSS	VDD	VSS									
21	CDATA0	CDATA15	VSS	VDD									
20	CDATA14	CDATA1	VDD	VSS									
19	CDATA3	CDATA13	CDATA2	CDATA12									
18	VSS	VDD	CDATA11	CDATA4									
17	CDATA5	CDATA10	VDD	VSS									
16	CDATA7	CDATA8	CDATA9	CDATA6									
15	CWE#	CDQML	CDQMH	CCAS#									
14	VDD	VSS	CCLK	VSS									
13	CRAS#	CCS#	VDD	VSS									
12	CBA0	CADDR11	CADDR12	NC									
11	CADDR9	CADDR10	CBA1	VDD									
10	VSS	CADDR8	VDD	VSS									
9	VDD	CADDR7	CADDR0	TEST_SCAN									
8	CADDR6	VDD	CADDR1	TEST_PLLIN1									
7	VSS	CADDR2	VDD	VSS									
6	NC	CADDR3	CADDR5	TEST_PLLIN0									
5	VSS	CADDR4	VDD	TEST_PLL_OUT0									
4	VSS	PCI_AD31	PCI_AD30	VSS	PCI_AD25	VDD	VSS	MODE1	PCI_AD17	VSS	PCI_TRDY#	VSS	PCI_PAR
3	PCI_CLK	PCI_RST#	VDD	PCI_AD27	PCI_IDSEL	LED_OUT1	VDD	PCI_AD20	PCI_AD16	VDD	PCI_DEVSEL#	PCI_PERR#	PCI_AD14
2	PCI_INTA#	PCI_GNT#	PCI_AD28	VSS	PCI_CBE3#	VSS	PCI_AD22	PCI_AD19	PCI_CBE2#	PCI_FRAME#	PCI_LOCK#	PCI_SERR#	PCI_AD15
1	VSS	PCI_REQ#	PCI_AD29	PCI_AD26	PCI_AD24	PCI_AD23	PCI_AD21	PCI_AD18	VSS	PCI_IRDY#	PCI_STOP#	VDD	PCI_CBE1#
	A	B	C	D	E	F	G	H	J	K	L	M	N

Figure 123. Pin configuration, columns A-N

P	R	T	U	V	W	Y	AA	AB	AC	AD	AE	AF	
NC	MIPS_PMASTER#	MIPS_CMD8	NC	MIPS_CMD5	MIPS_CMD2	MIPS_COLDRESET#	MIPS_INT1#	MIPS_AD14	MIPS_AD11	MIPS_AD8	MIPS_AD5	MIPS_AD2	26
VDD	MIPS_EREQ#	MIPS_PVALID#	VSS	VDD	MIPS_CMD3	MIPS_RESET#	MIPS_INT0#	MIPS_AD13	VDD	MIPS_AD9	MIPS_AD6	MIPS_AD3	25
VSS	MIPS_EOK_EXT#	MIPS_CMD7	VDD	VSS	MIPS_CMD1	VDD	MIPS_INT2#	MIPS_AD12	MIPS_AD10	VDD	MIPS_AD4	MIPS_AD1	24
VDD	MIPS_PREQ#	MIPS_CMD6	VSS	MIPS_CMD4	MIPS_CMD0	VSS	MIPS_AD15	VSS	VSS	MIPS_AD7	TEST_IN3	MIPS_AD0	23
									VSS	EEPROM_DO	EEPROM_DI	VSS	22
									EEPROM_SK	EEPROM_CS	NC	VSS	21
									VSS	VDD	VDD	TEST_CLK_A	20
									LED_OUT2	VSS	TEST_CLK_B	VSS	19
									TEST_CLK_C	TEST_CLK_D	VSS	VSS	18
									VSS	VDD	VSS	VSS	17
									VSS	VDD	PDATA0	PDATA15	16
									PDATA13	PDATA1	PDATA14	PDATA2	15
									VSS	VDD	PDATA3	PDATA12	14
									PDATA10	PDATA4	PDATA11	PDATA5	13
									PDATA6	VSS	PDATA7	PDATA9	12
									PDQMH	PWE#	PDATA8	PDQML	11
									VSS	VDD	PCLK	PCAS#	10
									VDD	VSS	PCS#	PRAS#	9
									PBA1	PADDR11	PADDR12	PBA0	8
									VSS	VDD	PADDR10	PADDR9	7
									VSS	VDD	PADDR0	PADDR8	6
									VDD	VSS	PADDR1	PADDR7	5
VSS	VDD	PCI_AD9	VSS	VDD	PCI_AD2	VSS	VDD	VDD	VSS	PADDR6	VSS	VSS	4
VDD	PCI_AD12	VSS	VDD	PCI_AD5	PCI_AD1	VDD	LED_OUT0	TEST0	TEST_IN2	VDD	PADDR5	PADDR2	3
MODE2	PCI_AD11	PCI_AD8	PCI_AD7	PCI_AD4	VSS	TEST_OUT0	TEST_OUT2	TEST_PLL_OUT2	TEST_IN1	TEST1	VSS	PADDR3	2
PCI_AD13	PCI_AD10	PCI_CBE0#	PCI_AD6	PCI_AD3	PCI_AD0	TEST_OUT1	TEST_OUT3	VSS	TEST_IN0	MIPS_BIG ENDIAN	TEST2	PADDR4	1
P	R	T	U	V	W	Y	AA	AB	AC	AD	AE	AF	

Figure 124. Pin configuration, columns P-AF

Signal	PIN	Signal	PIN	Signal	Pin	Signal	Pin	Signal	Pin
CADDR0	C9	MIPS_AD4	AE24	PCI_AD26	D1	TEST_PLLIN1	D8	VSS	AC10
CADDR1	C8	MIPS_AD5	AE26	PCI_AD27	D3	TEST_PLLOUT0	D5	VSS	AC14
CADDR10	B11	MIPS_AD6	AE25	PCI_AD28	C2	TEST_PLLOUT1	M24	VSS	AC16
CADDR11	B12	MIPS_AD7	AD23	PCI_AD29	C1	TEST_PLLOUT2	AB2	VSS	AC17
CADDR12	C12	MIPS_AD8	AD26	PCI_AD3	V1	TEST_SCAN	D9	VSS	AC20
CADDR2	B7	MIPS_AD9	AD25	PCI_AD30	C4	TEST0	AB3	VSS	AC22
CADDR3	B6	MIPS_BIGENDIAN	AD1	PCI_AD31	B4	TEST1	AD2	VSS	AC23
CADDR4	B5	MIPS_CLK	K26	PCI_AD4	V2	TEST2	AE1	VSS	AC4
CADDR5	C6	MIPS_CMD0	W23	PCI_AD5	V3	VDD	A14	VSS	AC6
CADDR6	A8	MIPS_CMD1	W24	PCI_AD6	U1	VDD	A9	VSS	AC7
CADDR7	B9	MIPS_CMD2	W26	PCI_AD7	U2	VDD	AB4	VSS	AD12
CADDR8	B10	MIPS_CMD3	W25	PCI_AD8	T2	VDD	AC25	VSS	AD19
CADDR9	A11	MIPS_CMD4	V23	PCI_AD9	T4	VDD	AC5	VSS	AD5
CBA0	A12	MIPS_CMD5	V26	PCI_CBE0#	T1	VDD	AC9	VSS	AD9
CBA1	C11	MIPS_CMD6	T23	PCI_CBE1#	N1	VDD	AD10	VSS	AE17
CCAS#	D15	MIPS_CMD7	T24	PCI_CBE2#	J2	VDD	AD14	VSS	AE18
CCLK	C14	MIPS_CMD8	T26	PCI_CBE3#	E2	VDD	AD16	VSS	AE2
CCS#	B13	MIPS_COLDRESET#	Y26	PCI_CLK	A3	VDD	AD17	VSS	AE4
CDATA0	A21	MIPS_EOK#	J24	PCI_DEVSEL#	L3	VDD	AD20	VSS	AF17
CDATA1	B20	MIPS_EOK_EXT#	R24	PCI_FRAME#	K2	VDD	AD24	VSS	AF18
CDATA10	B17	MIPS_EREQ#	R25	PCI_GNT#	B2	VDD	AD3	VSS	AF19
CDATA11	C18	MIPS_EVALID#	J25	PCI_IDSEL	E3	VDD	AD6	VSS	AF21
CDATA12	D19	MIPS_INT0#	AA25	PCI_INTA#	A2	VDD	AD7	VSS	AF22
CDATA13	B19	MIPS_INT1#	AA26	PCI_IRDY#	K1	VDD	AE20	VSS	AF4
CDATA14	A20	MIPS_INT2#	AA24	PCI_LOCK#	L2	VDD	B18	VSS	B14
CDATA15	B21	MIPS_PMASTER#	R26	PCI_PAR	N4	VDD	B24	VSS	B22
CDATA2	C19	MIPS_PREQ#	R23	PCI_PERR#	M3	VDD	B26	VSS	B23
CDATA3	A19	MIPS_PVALID#	T25	PCI_REQ#	B1	VDD	B8	VSS	C21
CDATA4	D18	MIPS_RESET#	Y25	PCI_RST#	B3	VDD	C10	VSS	C23
CDATA5	A17	MODE1	H4	PCI_SERR#	M2	VDD	C13	VSS	C25
CDATA6	D16	MODE2	P2	PCI_STOP#	L1	VDD	C17	VSS	C26
CDATA7	A16	NC	A6	PCI_TRDY#	L4	VDD	C20	VSS	D10
CDATA8	B16	NC	AE21	PCLK	AE10	VDD	C22	VSS	D13
CDATA9	C16	NC	D12	PCS#	AE9	VDD	C24	VSS	D14
CDQMH	C15	NC	L26	PDATA0	AE16	VDD	C3	VSS	D17
CDQML	B15	NC	P26	PDATA1	AD15	VDD	C5	VSS	D2
CRAS#	A13	NC	U26	PDATA10	AC13	VDD	C7	VSS	D20
CWE#	A15	PADDR0	AE6	PDATA11	AE13	VDD	D11	VSS	D22
EEPROM_CS	AD21	PADDR1	AE5	PDATA12	AF14	VDD	D21	VSS	D23
EEPROM_DI	AE22	PADDR10	AE7	PDATA13	AC15	VDD	D25	VSS	D4
EEPROM_DO	AD22	PADDR11	AD8	PDATA14	AE15	VDD	F4	VSS	D7
EEPROM_SK	AC21	PADDR12	AE8	PDATA15	AF16	VDD	G24	VSS	F2
LED_OUT0	AA3	PADDR2	AF3	PDATA2	AF15	VDD	G26	VSS	F26
LED_OUT1	F3	PADDR3	AF2	PDATA3	AE14	VDD	G3	VSS	G23
LED_OUT2	AC19	PADDR4	AF1	PDATA4	AD13	VDD	K24	VSS	G4
MIPS#	H23	PADDR5	AE3	PDATA5	AF13	VDD	K3	VSS	J1
MIPS_AD0	AF23	PADDR6	AD4	PDATA6	AC12	VDD	L23	VSS	J23
MIPS_AD1	AF24	PADDR7	AF5	PDATA7	AE12	VDD	L24	VSS	K23
MIPS_AD10	AC24	PADDR8	AF6	PDATA8	AE11	VDD	M1	VSS	K25

Signal	PIN	Signal	PIN	Signal	Pin	Signal	Pin	Signal	Pin
MIPS_AD11	AC26	PADDR9	AF7	PDATA9	AF12	VDD	N24	VSS	K4
MIPS_AD12	AB24	PBA0	AF8	PDQMH	AC11	VDD	P23	VSS	M4
MIPS_AD13	AB25	PBA1	AC8	PDQML	AF11	VDD	P25	VSS	N23
MIPS_AD14	AB26	PCAS#	AF10	PLL_AGD	M25	VDD	P3	VSS	N25
MIPS_AD15	AA23	PCI_AD0	W1	PLL_DGN	M26	VDD	R4	VSS	N26
MIPS_AD16	H24	PCI_AD1	W3	PLL_DVD	L25	VDD	U24	VSS	P24
MIPS_AD17	H26	PCI_AD10	R1	PLL_VAA	M23	VDD	U3	VSS	P4
MIPS_AD18	H25	PCI_AD11	R2	PRAS#	AF9	VDD	V25	VSS	T3
MIPS_AD19	G25	PCI_AD12	R3	PROTECT#	J26	VDD	V4	VSS	U23
MIPS_AD2	AF26	PCI_AD13	P1	PWE#	AD11	VDD	Y24	VSS	U25
MIPS_AD20	F23	PCI_AD14	N3	TEST_CLK_A	AF20	VDD	Y3	VSS	U4
MIPS_AD21	F24	PCI_AD15	N2	TEST_CLK_B	AE19	VDD	AA4	VSS	V24
MIPS_AD22	F25	PCI_AD16	J3	TEST_CLK_C	AC18	VSS	A1	VSS	W2
MIPS_AD23	E23	PCI_AD17	J4	TEST_CLK_D	AD18	VSS	A10	VSS	Y23
MIPS_AD24	E26	PCI_AD18	H1	TEST_IN0	AC1	VSS	A18	VSS	Y4
MIPS_AD25	E25	PCI_AD19	H2	TEST_IN1	AC2	VSS	A22		
MIPS_AD26	E24	PCI_AD2	W4	TEST_IN2	AC3	VSS	A23		
MIPS_AD27	D26	PCI_AD20	H3	TEST_IN3	AE23	VSS	A24		
MIPS_AD28	D24	PCI_AD21	G1	TEST_OUT0	Y2	VSS	A4		
MIPS_AD29	B25	PCI_AD22	G2	TEST_OUT1	Y1	VSS	A5		
MIPS_AD3	AF25	PCI_AD23	F1	TEST_OUT2	AA2	VSS	A7		
MIPS_AD30	A25	PCI_AD24	E1	TEST_OUT3	AA1	VSS	AB1		
MIPS_AD31	A26	PCI_AD25	E4	TEST_PLLIN0	D6	VSS	AB23		

Figure 125. Pin configuration, alphabetical