Sun-3 Architecture Manual

Version 2.0 Preliminary

Company Confidential

Sun Microsystems Inc.

16 May 1985

Table of Contents

1. Introduction

This document describes the Sun-3 architecture. It is intended as a reference for Sun-3 software, hardware, and systems implementors.

Features. The Sun-3 architecture is an extension of the Sun-2 architecture. Its main features are support of the 68020 CPU, the 68881 FPP, 8 KB pages, eight 256 MB contexts, and a 32-bit VMEbus. The Sun-3 architecture does not necessarily apply to any future CPUs.

Nomenclature The nomenclature used in the Sun-3 architure includes terminology common to the 68020 processor terminology. Examples include word definition, byte addressing, address space defition, etc. Deviations from this nomenclature are noted. Minus active signal names are suffixed with a minus (-).

Scope. This Sun-3 architecture manual describes all devices which may be included in a Sun-3 CPU board plus all boards which are interconnected with the CPU board over a Sun defined memory bus. The manual does not describe devices accessible over the system bus (VMEbus).

Implementation. The main part of this document is independent of particular implementations of the architecture. Implementation specific data, such as timing information, need to be specified for each implementation.

Correctness. An important goal of this document is correctness. Please report any errors, omissions, or oversights immediately so they can be corrected in future revisions.

1.1. Definitions

In the subsequent description of the Sun-3 architecture the following abbreviations are used:

CPU: Central Processing Unit

DVMA: Direct Virtual Memory Access

MMU: Memory Management Unit

PMEG: Page Map Entry Group

POR: Power-On-Reset

ECC: Error Correcting Code

1.2. Architecture Overview

The Sun-3 architecture is divided into three spaces: the CPU space, Control space, and Device space.

The CPU space comprises all references of the 68020 CPU in function code 7.

The Control space is the core of the Sun-3 architecture. It includes the Sun-3 memory management unit (the "MMU") as well as all other Sun-3 architecture extensions to the CPU, such as the bus error register, the system enable register, the diagnostic register, and the ID-PROM. The ID-PROM contains a unique serial number and indicates the implementation type of the architecture.

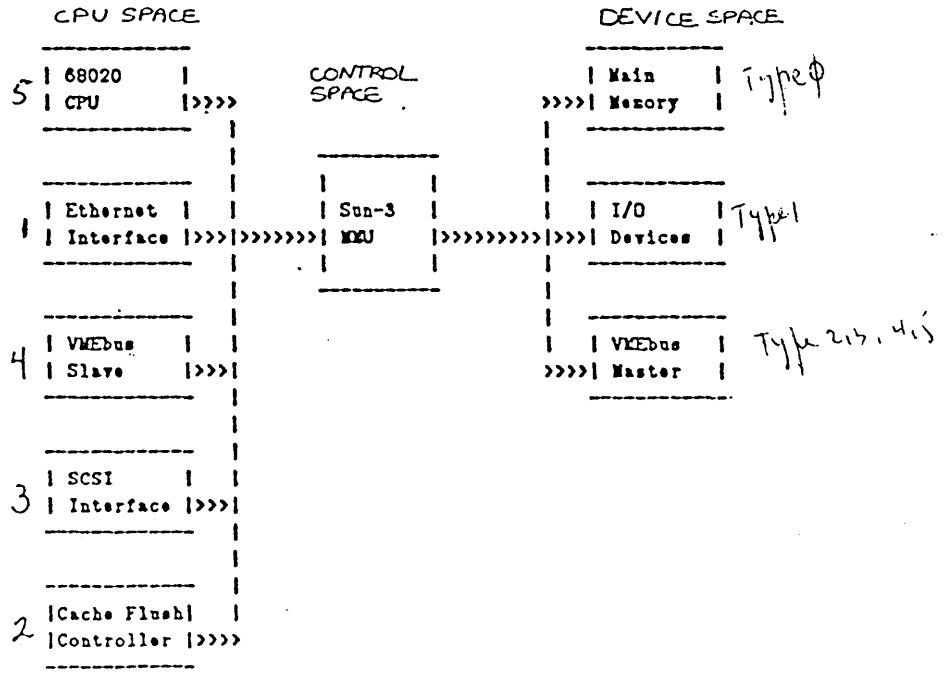
The Device space of the Sun-3 architecture defines what devices exist in the architecture and how they are accessed. These devices include main memory, the system bus (VMEbus), and I/O devices.

The CPU may directly access one device, the Floating Point Accelerator, in Device Space without MMU address translation and protection checking.

All other CPU accesses to Device Space pass through the MMU and thus are translated and protected in an identical fashion. In addition, direct memory accesses by DVMA masters such as the Ethernet, SCSI, and the VMEbus slave interface also pass through the memory management and thus operate in a fully protected environment.

The figure below illustrates how the CPU, MMU, and devices are interconnected in the Sun-3 architecture. The CPU sends out a virtual address that is translated by the MMU into a physical address. The VMEbus Master Interface, Main Memory, Video Memory, and I/O Devices are addressed with physical addresses on the right side of the MMU.

DVMA devices and the CPU must arbitrate for and share the virtual address bus on the left side of the MMU. Among the possible DVMA devices are the Ethernet interface, SCSI interface, VMEbus Slave interface, and a Cache Flush Controller, depending on the implementation. Of these devices, the Ethernet has the highest priority, a Cache Flush Controller second highest, the SCSI third highest, the VMEbus slave interface fourth highest, and the CPU lowest priority.



1.3. Implementation Configurations

The Sun-3 architecture allows each implementation of the Sun-3 architecture to have its own configuration of devices. In addition, configurations may provide certain options in terms of main memory size and I/O devices. With this configuration flexibility come a number of optional bits in registers. These Sun-3 implementation configurations are treated uniformly as follows:

- *Machine Configuration.* The machine type in the IDPROM indicates which devices the machine has and which are valid options. Addresses for all devices defined in Sun-3 are fixed by the Sun-3 architecture.

- *Optional Control Space devices.* Control Space devices are decoded from fields of address bits. Most devices are uniquely identified by decoding A<31..28>, but some devices may require the use of additional subfields for unique device identification. Address bits outside defined fields are ignored on Control Space accesses. Accesses to unimplemented devices or undefined codes are discussed in the Bus Error Register section.

- *Optional Main Memory.* The minimum and maximum main memory for each Sun-3 machine type are implementation dependent. The smallest minimum main memory size for any Sun-3 configuration is 2 megabytes. The size of the main memory physical address space is also implementation dependent. Any access to a main memory physical address beyond the range of the maximum physical address space is undefined. Accesses to memory that is addressable, but not physically present, are discussed in the Bus Error Register section.

- *Optional I/O Devices.* I/O devices are defined as Type 1 devices in Device Space. Most I/O devices are uniquely identified by decoding physical address bits A<20..17>, but some devices may require the use of additional subfields for unique device identification. Address bits outside defined fields are ignored on accesses to Type 1 devices in Device Space. Accesses to I/O devices which are optional to an implementation but not installed or to codes undefined in an implementation are discussed in the Bus Error Register section.

- *High Order Virtual Address Bits.* The Sun-3 architecture has 28 bit virtual addressing in Device Space. The treatment of virtual address bits A<31..28> for CPU bus cycles in Device Space depends on whether the implementation supports the Floating Point Accelerator (FPA) option. For those implementations that support the FPA, the only valid codes for A<31..28> are 0x0, 0xE, and 0xF. Code 0xE accesses the FPA, and codes 0x0 plus 0xF access other Device Space devices. Treatment of other codes are discussed in the Bus Error section. For those implementations that do not support the FPA, CPU virtual address bits A<31..28> are ignored in Device Space.

- *Optional Bits in Registers.* The architecture defines certain register bits, such as bits of the System Enable register, which control devices that may be optional or may not exist for a given implementation. These register bits must exist for all implementations but will have no affect in implementations for which the device is non-existent.

- *Unused bits in Page Map.* Unimplemented bits in the Page Map Physical Address and Reserved fields read back as 0's. Unused Page Map control bits may be read and written but have no affect.

- *Unused bits in the Cache Tags.* For those implementations that support a cache, the Unused cache tag bits D<7..0> and unused Virtual Address tags are undefined. The Reserved tag bits D<29..28> and D<11> may be read and written but have no affect.

2. Address Spaces

The Sun-3 architecture uses three address spaces: CPU Space, Control Space, and Device Space. These address spaces are decoded with different processor function codes.

The following table describes how different CPU function codes are mapped to the CPU, Control, and Device space.

FC	Address Space
0	Reserved
1	Device Space (User Data)
2	Device Space (User Program)
3	Control Space
4	Reserved
5	Device Space (Supervisor Data)
6	Device Space (Supervisor Program)
7	CPU Space

2.1. CPU Space

CPU space consists of all cycles that use function code 7. These include coprocessor cycles, interrupt acknowledge, breakpoint and ring-protection cycles.

2.2. Control Space

Control space consists of all cycles that use function code 3. These include accesses to the memory management unit (the "MMU"), the bus error register, the system enable register, the user enable register, the diagnostic register, the ID-PROM, several cache related functions (if implemented), and an (optional) bypass path to the UART.

2.3. Device Space

Device space includes all devices that are accessed by the CPU with data or program space instructions. These devices include main memory, the VMEbus Master interface, I/O devices, and so on. All devices in Device Space, with one exception, are accessed via the MMU. This allows these devices to be protected, shared, and managed in a uniform manner in a multiprocess environment. The one exception is an optional Floating Point Accelerator, which is accessed by direct decoding of CPU virtual address bits A<31..28>. The chapter on the FPA describes how this device is protected and shared.

3. CPU Space

CPU space consists of all cycles that use function code 7. This includes coprocessor cycles, interrupt, breakpoint and ring-protection cycles.

3.1. CPU Space Cycles

In the Sun-3 architecture, address bits A<17..16> are decoded to determine the type of CPU Space cycle as follows:

CYCLE TYPE	A17	A16	CYCLE CONCLUSION
BREAKPOINT CYCLE	0	0	BERR
RINGPROTECTION	0	1	BERR
COPROCESSOR CYCLE	1	0	DSACK/BERR
INTERRUPT ACK. CYCLE	1	1	AVEC/DSACK/BERR

Breakpoint and ringprotection cycles are terminated with Bus Error.

Interrupt acknowledge cycles are discussed in the chapter on CPU Interrupts.

3.2. Coprocessor Cycles

The execution of "F-line" opcodes results in coprocessor bus cycles. There are eight possible coprocessors, identified by address bits A15 through A13. Sun-3 systems support one coprocessor, the MC68881 Floating Point coprocessor (FPP). This coprocessor is identified by A<15..13> = '001' and is is enabled by bit D<6> of the System Enable register.

"F-line" opcode instructions will terminate with an unimplemented instruction exception if external hardware asserts the bus error signal to terminate the coprocessor cycle. Three conditions cause this exception: first, access to any coprocessor except the FPP (i.e., any coprocessor code except A<15..13> = '001'); second, access to the FPP while disabled (i.e., with EN.FPP = 0); and third, access to an enabled (EN.FPP = 1) but non-responding FPP. None of these conditions cause changes to the Bus Error register.

4. Control Space

Control Space includes the Sun-3 memory management unit and all Sun-3 architectural extensions to the CPU. These extensions include the bus error register, the system enable register, the user enable register, the diagnostic register, the ID-PROM, cache functions (if implemented), and an (optional) bypass path to the UART.

4.1. Access to Control Space Devices

Space devices). Control Space devices are decoded from fields of address bits. Most devices are uniquely identified by decoding A<31..28>, but some devices may require the use of additional subfields for unique device identification. Address bits outside defined fields are ignored on Control Space accesses. Accesses to unimplemented devices or undefined codes are discussed in the Bus Error Register section.

Control Space devices are identified by decoding fields of address bits. Most devices are uniquely identified by decoding A<31..28>, but some devices may require the use of additional subfields for unique device identification. The Flush Cache Control Space operation requires decoding data bits D<1..0> to uniquely identify the type of flush operation.

Each Control Space device has an associated address space for accessing elements within the device. (For some devices this may be a null space.) These address spaces are shown in the summary table below. In this table, CX indicates the Context ID register.

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
ID PROM	0x0		BYTE	READ	A<4..0>
PAGE MAP	0x1		LONG	R/W	CX<2..0>, A<27..13>
SEGMENT MAP	0x2		BYTE	R/W	CX<2..0>, A<27..17>
CONTEXT REG.	0x3		BYTE	R/W	
SYSTEM ENABLE	0x4		BYTE	R/W	
USER DVMA ENABLE	0x5		BYTE	R/W	
BUS ERROR REG.	0x6		BYTE	READ	
DIAGNOSTIC REG.	0x7		BYTE	WRITE	
CACHE TAGS	0x8		LONG	R/W	A<16..4>*
CACHE DATA	0x9		LONG	R/W	A<16..2>*
FLUSH CACHE SET (CONTEXT)	0xA	'01'	N/A	WRITE	CX<2..0>, A<16..4>**
FLUSH CACHE SET (PAGE)	0xA	'10'	N/A	WRITE	CX<2..0>, A<27..13>, A<12..4>**
FLUSH CACHE SET (SEGMENT)	0xA	'11'	N/A	WRITE	CX<2..0>, A<27..17>, A<16..4>**
BLOCK COPY (READ)	0xB		N/A	READ	CX<2..0>, A<27..4>
BLOCK COPY (WRITE)	0xB		N/A	WRITE	CX<2..0>, A<27..4>
Unused	0xC..0xE				
UART BYPASS	0xF		BYTE	R/W	A<2..1>

* Depends on size of cache: see Control Space Operations for the Cache
** Depends on sizes of cache and Flush Set: see cache Control Space Ops

The UART is normally accessed through the MMU as a Type 1 device in Device Space. Sun-3 implementations may provide an optional bypass path in Control Space to access the UART directly.

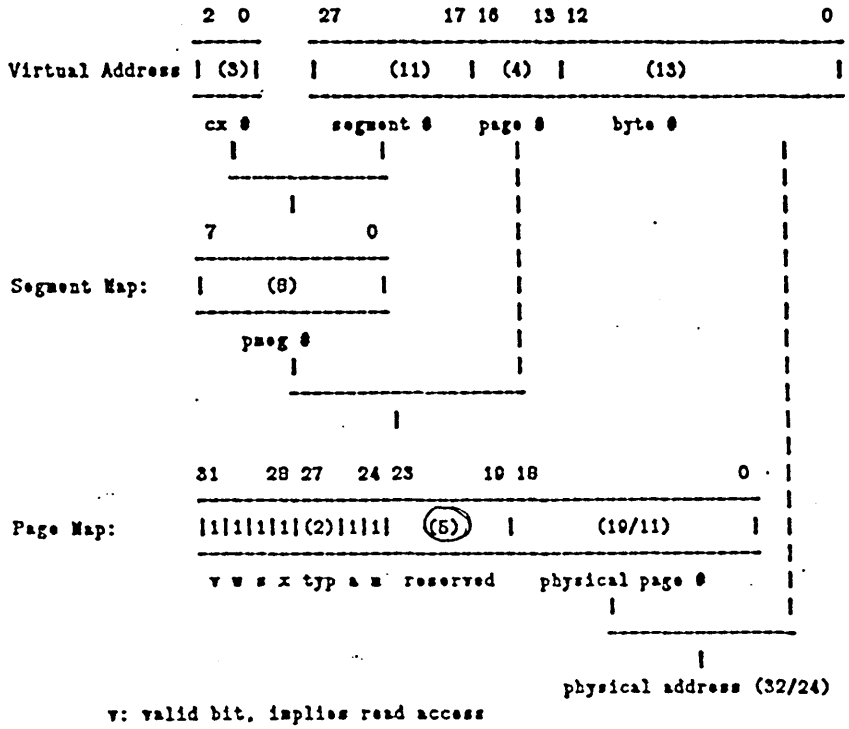
Accesses to Control Space devices which are optional to an implementation but not installed or to codes undefined in an implementation are discussed in the Bus Error Register section.

4.2. Sun-3 Memory Management Unit Summary

4.2.1. MMU: Summary

page size:	8 KBytes
segment size:	128 KBytes
process size:	256 MBytes
# of contexts:	8
# of segments/context:	2048
# of pages/segment:	16
# of pmegs:	256
# of pages total:	4096
# of segments total:	16384

4.2.2. MMU: Address Translation



v: valid bit, implies read access
w: write access bit
s: system access bit
x: don't cache bit
a: accessed bit
m: modified bit

typ=00: main memory
typ=01: Sun-3 I/O space
typ=10: VMEbus 16-bit data
typ=11: VMEbus 32-bit data

4.3. MMU Overview

The memory management consists of a context register, a segment map, and a page map. Virtual addresses from the processor and DVMA devices are translated into intermediate addresses by the segment map and then into physical addresses by the page map. The MMU uses a page size of 8K bytes and a segment size of 128K bytes. Eight contexts with an address space of 256M bytes each are provided.

4.4. Contexts

The Sun-3 MMU is divided into 8 distinct address spaces or "contexts". The current context is selected by means of a 3-bit *context* register. The same context applies to both user and supervisor state.

Initialization: None

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
CONTEXT REG.		0x3	BYTE	R/W	

CONTEXT REGISTER

BIT	NAME	MEANING
D<2..0>	CX<2..0>	3 bit Context I.D.
D<3>	Reserved	Reserved: Reads back as 0
D<7..4>	Undefined	Contents undefined when read

4.5. Segment Map

The segment map has 16384 entries. It is indexed by the 3 bits of the current context register and the 11 most significant bits of the virtual address, bits 27 through 17. Thus, the segment map is divided into 8 contexts of 2048 entries each. Segment map entries are 8 bits wide, pointing to a page map entry group (*pmeg*).

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
SEGMENT MAP		0x2	BYTE	R/W	CX<2..0>, A<27..17>

4.6. Page Map

The page map contains 4096 page entries each mapping an 8K byte page. Page map entries are composed of a valid bit, protection field, don't cache bit, type field, accessed and modified bits, and a page number.

The page map is divided into 256 sections of 16 entries each. Each section is pointed to by a segment map entry and is called a page map entry group, or *pmeg*.

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
PAGE MAP		0x1	LONG	R/W	CX<2..0>, A<27..13>

4.6.1. Valid Bit

The valid bit means that the page entry is valid. It also allows read and execute access to the page.

4.6.2. Write Bit

The write bit allows write access to the page.

4.6.3. Supervisor Bit

If the supervisor bit is set, the read and write access protection apply only to supervisor accesses, and user accesses are prohibited. If the supervisor bit is clear, then read and write access protection apply both to supervisor accesses and user accesses.

4.6.4. Don't Cache Bit

If this bit is set then the page referenced will not be cached in a Sun-3 external cache. The Don't Cache bit has no affect on the use of the MC68020 internal instruction cache. It is meaningful only for implementations of the Sun-3 architecture that include an external cache. In these implementations, the bit only has meaning for Type 0 pages in Device Space. In machines without a cache the bit can be read and written but has no effect.

4.6.5. PageType

The 2-bit page type field provides for four physical address spaces, each starting at a physical address of 0. The four types are:

```
PMAP<27..26>: TYPE
        0 - Main Memory
        1 - I/O Devices
        2 - VMEbus 16-bit data
        3 - VMEbus 32-bit data
```

4.6.6. Statistics Bits: Accessed and Modified

The accessed bit is set whenever a page is accessed, on read or write cycles, through the MMU. The modified bit is set whenever a page is modified (written into) through an MMU page access. The statistics bits are automatically updated for all cycles requiring an MMU access except if the page is invalid or protected.

```
PMAP<25>: ACCESSED
        0 - Not Accessed
        1 - Accessed
PMAP<24>: MODIFIED
        0 - Not Modified
        1 - Modified
```

In Sun-3 implementations with a cache, the accessed and modified bits are only updated on memory accesses that "miss" the cache. The sections "The MMU Access Bit" and "Modified Bits for the Cache and MMU" in the Cache chapter further discuss statistic bit updates in systems with a cache.

4.6.7. Reserved Field

The reserved field in the page map has no function. It can be written into, but it always reads back as 0.

PMAP<24..10>: RESERVED

4.6.8. Physical Page Number

The page number field in conjunction with the byte address generates the physical address. The physical page number field is either 11 bits or 19 bits wide. In conjunction with the 13-bit physical byte number, the 11-bit physical page number field generates a 24-bit physical address field, whereas the 19-bit physical page number field generates a 32-bit physical address field. In case of the 11-bit physical page number field, the unused map bits <18..12> have no function; they can be written into, but they always reads back as 0.

The size of the physical address space for each device is limited to be no larger than the size of the physical address field in the MMU, but may be less. For Type 0 space (main memory), the size of the physical address space is implementation dependent. For Type 1 space (I/O space), the size of the physical address space is device dependent. For Type 2 and 3 (VMEbus) space, the physical address space may be 32, 24, or 16 bits.

PMAP<18..0>: PAGE NUMBER

4.7. ID PROM

The purpose of the ID PROM is to provide information about the machine. This includes basic information on the machine type, a unique serial number for software licensing and distribution, a unique Ethernet address, the date of manufacturing, and a checksum. In addition, the ID PROM stores configuration data for the machine.

The ID PROM is a 32 byte bipolar PROM that is not modifiable.

REGISTER/MAP	A<31..28> D<1..0>	SIZE	TYPE	ADDRESS FIELD
ID PROM	0x0	BYTE	READ	A<4..0>

I/D PROM

BYTE	NAME	MEANING
0x00	ID PROM 0	ID Prom byte 0
0x01	ID PROM 1	ID Prom byte 1
0x02	ID PROM 2	ID Prom byte 2
...		
0x1F	ID PROM 31	ID Prom byte 31

The content of the ID PROM is as follows:

Entry	Field	
(1)	Format	1 Byte
(2)	Machine Type	1 Byte
(3)	Ethernet Address	6 Bytes
(4)	Date	4 Bytes
(5)	Serial Number	3 Bytes
(6)	Checksum	1 Byte
(7)	Reserved	16 Bytes

In detail:

(1) *Format.* The format of the ID PROM.

(2) *Machine Type.* A number specifying an implementation of the architecture.

(3) *Ethernet Address.* This is the unique 48-bit Ethernet address assigned by Sun to this machine.

(4) *Date.* The date the ID PROM was generated. It is in the form of a 32-bit long word which contains the number of seconds since January 1, 1970.

(5) *Serial Number.* This is a 3-byte serial number.

(6) *Checksum.* The checksum is defined such that the longitudinal XOR of the first 16 bytes of the PROM including the checksum yields 0.

(7) *Reserved.* This field will be specified in a future revision of this document.

4.8. System Enable Register

The System Enable Register enables system facilities and allows booting. This register can be read and written under software control and is cleared on Power On (hardware reset) and Watchdog reset, but not upon CPU reset. Bits are assigned as follows:

Initialization: See chapter on CPU Reset

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
SYSTEM ENABLE	0x4		BYTE	R/W	

The fields of the system enable register are as follows:

SYSTEM ENABLE REGISTER

BIT	NAME	MEANING
D<0>	EN.DIAG	Read back diagnostic switch
D<1>	EN.FPA	Enable Floating Point Accelerator
D<2>	EN.COPY	Enable copy mode to video memory, if present
D<3>	EN.VIDEO	Enable video display
D<4>	EN.CACHE	Enable external cache if present
D<5>	EN.SDVMA	Enable system DVMA if present
D<6>	EN.FPP	Enable floating point processor if present
D<7>	EN.BOOT-	Enable Boot State (0 => boot, 1 => normal)

When cleared on reset, all bits are initialized to 0. In this state, boot state is active whereas all other enabled devices are disabled.

EN.DIAG. This bit reads back the external diagnostic switch. A "0" bit read means that the switch is in its normal state (not-diagnostic), whereas a "1" means that the switch is activated (diagnostic).

EN.FPA. Enable the Floating Point Accelerator, if present. If this bit is deasserted, then accesses to the FPA cause a bus error, setting the FPAENERR error bit in the Bus Error register. If the bit is asserted, accesses are directed to the FPA. In the later case, if no FPA is present, then the access will result in a TIMEOUT bus error recorded in the Bus Error register.

EN.COPY. This bit enables the copy update mode to the video memory, if present.

EN.VIDEO. This bit enables the video signal to the video monitor, if present,

EN.CACHE. This bit enables the external cache, if present.

EN.SDVMA. This bit enables the system DVMA from the system bus (VMEbus), if present.

EN.FPP. This bit enables the floating point coprocessor (FPP), if present. See the section "Coprocessor Cycles" in the CPU Space chapter for a description of instruction exceptions resulting from accesses to the FPP coprocessor while (EN.FPP = 0).

EN.BOOT-. Boot state (minus active) forces all supervisor program fetches to the EPROM device independent of the setting of the memory management. All other types of references are unaffected and will be mapped as during normal operation of the processor.

4.9. User DVMA Enable Register

On implementations of the architecture that allow user DVMA, this register controls which contexts have DVMA access. For each context, a separate enable bit is provided.

Initialization: See chapter on CPU Reset

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
USER DVMA ENABLE	0x5		BYTE	R/W	

The fields of the user DVMA enable register are as follows:

USER DVMA ENABLE REGISTER

BIT	NAME	MEANING
D<0>	EN.CX0	Enable User DVMA to Context 0
D<1>	EN.CX1	Enable User DVMA to Context 1
D<2>	EN.CX2	Enable User DVMA to Context 2
D<3>	EN.CX3	Enable User DVMA to Context 3
D<4>	EN.CX4	Enable User DVMA to Context 4
D<5>	EN.CX5	Enable User DVMA to Context 5
D<6>	EN.CX6	Enable User DVMA to Context 6
D<7>	EN.CX7	Enable User DVMA to Context 7

When cleared after reset (see the chapter CPU Reset), all bits are initialized to 0. In this state, all user DVMA is disabled.

4.10. Bus Error Register

When an error occurs during a CPU bus cycle to Control Space or Device Space whose cause is identified synchronously with the bus cycle, then that bus cycle terminates with a Bus Error signal, and the cause of the error is captured in the Bus Error register. The Bus Error register may also be written when coprocessor cycles in CPU Space time out, causing an unimplemented instruction exception (see below). However, no other bus cycles alter the Bus Error register. This includes, in particular, DVMA cycles and other CPU Space cycles.

The Bus Error register always latches the cause of the most recent bus error for the CPU bus cycles indicated above. Thus, in the case of stacked bus errors, the information relating to the earlier bus errors is lost.

The Bus Error register is a read-only register.

Initialization: None

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
BUS ERROR REG.	0x6		BYTE	READ	

The fields of the Bus Error register are defined as follows:

BUS ERROR REGISTER

BIT	NAME	MEANING
D<0>	Unused	
D<1>	Unused	
D<2>	FPAENERR	FPA Enable error
D<3>	FPABERR	FPA bus error response
D<4>	VMEBERR	VMEbus Bus Error
D<5>	TIMEOUT	Timeout Error
D<6>	PROTERR	Protection Error
D<7>	INVALID	Invalid Page

In more detail, the bus error conditions are as follows:

- *FPAENERR* is set, in implementations that support the Floating Point Accelerator, whenever the CPU attempts to access the FPA while its enable bit is reset (EN.FPA = 0).

- *FPABERR* is set whenever the Floating Point Accelerator signals an error during a CPU bus access. The FPA has no interrupt capability; all errors must be reported as bus errors.

- *VMEBERR* indicates that a CPU cycle to the VMEbus was acknowledged with a VMEbus bus error.

- *TIMEOUT* results from accessing non-existing devices, both on-board and off-board, during those CPU bus cycles which may record bus errors. For Sun-3 systems with caches, timeouts may also result during Write Back cycles; these are reported through the Memory Error Control register, not the Bus Error register. All bus error timeout conditions are summarized below.

- *PROTERR* indicates a protection violation resulting from an attempted access to a Device Space page during a CPU bus cycle. Protection errors resulting from cache accesses are discussed in the section "Definition of Cache Protection" in the chapter on the Sun-3 Cache. MMU protection errors are detected during MMU address translation. Two errors are possible.

First, an attempt to write into a valid page whose Write access bit is reset causes a PROTERR. Second, an attempt to access with User permission a valid page marked Supervisor access also causes a PROTERR.

- *INVALID* means that the valid bit in the page map was not set during a CPU bus cycle to a Device Space page.

Timeout errors, resulting from accesses to non-existing devices, may occur on CPU bus cycles to Control Space or Device Space.

Control Space devices are decoded from fields of address bits; most devices are decoded from A<31..28>. Accesses to invalid codes within defined fields or to unimplemented Control Space devices will set the TIMEOUT bit in the Bus Error register.

For accesses to Device Space, the treatment of CPU virtual address bits A<31..28> depends on whether the implementation supports the Floating Point Accelerator (FPA). In implementations that support the FPA, any Device Space accesses from the CPU in which virtual address bits A<31..28> = (0x1 through 0xD) result in a TIMEOUT bus error. If the FPA is enabled (with EN.FPA = 1), accesses to a non-existing FPA also cause a TIMEOUT bus error. In implementations that do not support the FPA, A<31..28> are ignored.

Other accesses to Device Space may cause timeout bus errors, as follows. Each implementation defines the size of its own Type 0 physical address space. This address space includes main memory and (optional) frame buffers. Any Device Space access to either main memory or a frame buffer address space which is addressable, but not physically present, results in a TIMEOUT bus error. This may include, in particular, the Direct Access address space for the Copy-Mode frame buffer and the address space for the Video-RAM frame buffer.

Sun-3 I/O devices are Type 1 devices identified by fields of address bits; most devices are uniquely identified by A<20..17>. Accesses to devices which are optional but not installed, or to invalid device codes within defined fields, result in a TIMEOUT bus error.

Accesses to Type 2 or 3 Device Space (the VMEbus Master interface) may result in a TIMEOUT bus error if the addressed VMEbus device fails to respond within a defined time interval, which is specified for each implementation.

4.11. Diagnostic Register

The diagnostic register drives an 8-bit LED display for displaying error messages. A "0" bit written will cause the corresponding LED to light up, a "1" bit to be dark. Upon power-on-reset, the diagnostic register is initialized to 0 causing all LEDs to light up.

Initialization: See chapter on CPU Reset

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
DIAGNOSTIC REG.	0x7		BYTE	WRITE	

5. Device Space

Device space includes all the devices of the system that are accessed through memory management. This includes main memory, video memory, the system bus, and input/output devices.

In the following, each device is described in terms of its address, initialization, interrupts, exceptions, reference, and register mapping.

Not all devices are present in all implementations of the architecture. The address assignment of all Sun-3 devices, however, is fixed. Which devices are present are described in the implementation document for each machine type. However, the following devices are required for all implementations:

- Main Memory
- Memory Error Registers
- Interrupt Register
- EPROM
- EEPROM
- Clock

5.1. Sun-3 Physical Address Map

The Sun-3 physical address map is fixed for all implementations of Sun-3 architecture. In the map below, the Type code for each device and the base address (in 32 bit hex notation) for the device's address space are shown.

The base address shown (below) for each device may be divided into address subfields: one (or more) to identify the device and one to define the device's address space.

Most Type 1 devices are uniquely identified by decoding physical address bits A<20..17>, but some devices may require the use of additional fields for unique device identification. Address bits outside defined fields are ignored on accesses to Type 1 devices in Device Space. Accesses to I/O devices which are optional to an implementation but not installed or to invalid device codes within defined fields are discussed in the Bus Error Register section.

Type 2 and Type 3 spaces are VMEbus spaces. The size of the VMEbus address space is identified by decoding high order physical address fields. (A<31..24> = 0xFF) identifies the 24 bit field, and (A<31..16> = 0xFFFF) identifies the 16 bit field.

The number of physical address bits which define a device's address space depends on the MMU Type field and the implementation. The size of Type 0 physical address space for main memory is implementation dependent. The size of the address space required for each Type 1 device varies according to the device. For Type 2 and 3 (VMEbus) devices, the 32 bit VMEbus address space excludes the top 16 MB, and the 24 bit VMEbus address space excludes the top 64 KB.

SUN-3 PHYSICAL ADDRESS ASSIGNMENTS

TYPE	PHYSICAL BASE ADDRESS	DEVICE	PHYSICAL ADDRESS SPACE
	Note: Size of Type 0 address space is implementation dependent		
0	0x00000000	Main Memory	Impl. Depend.
0	0x00100000	Main-Memory Frame Buffer	A<16..0>
0	0x00100000	Copy-Mode Frame Buffer - Copy Mode access	A<16..0>
0	0xFF000000	Copy-Mode Frame Buffer - Direct access	A<16..0>
0	0xFF000000	Video-RAM Frame Buffer	A<20..0>*
		* Maximum; Minimum is A<17..0> (Impl. depend.)	
	Note: Type 1 devices identified from Address fields: see each device		
1	0x00000000	Keyboard/Mouse	A<2..1>
1	0x00002000	Serial Port	A<2..1>
1	0x00004000	EEPROM	A<10..0>
1	0x00006000	Time of Day Clock	A<4..0>
1	0x00008000	Memory Error Registers	A<2..0>
1	0x0000A000	Interrupt Register	(1 byte)
1	0x0000C000	Intel Ethernet Interface	(1 byte)
1	0x0000E000	Color Map	A<9..0>
1	0x00010000	EPROM	A<15..0>
1	0x00012000	AMD Ethernet Interface	A<1>
1	0x00014000	SCSI Interface	A<4..0>
1	0x00016000	(reserved)	
1	0x00018000	(reserved)	
1	0x0001A000	(reserved)	
1	0x0001C000	Data Encryption Proc.	A<1>
1	0x0001E000	ECC Memory Registers	A<7..0>
	Note: Type 2 space is VMEbus 16 bit data space Address fields A<31..24> and A<31..16> identify address spaces		
2	0x00000000	VMEbus 32-bit address space	A<31..0>
2	0xFF000000	VMEbus 24-bit address space	A<23..0>
2	0xFFFF0000	VMEbus 16-bit address space	A<15..0>
	Note: Type 3 space is VMEbus 32 bit data space Address fields A<31..24> and A<31..16> identify address spaces		
3	0x00000000	VMEbus 32-bit address space	A<31:.0>
3	0xFF000000	VMEbus 24-bit address space	A<23..0>
3	0xFFFF0000	VMEbus 16-bit address space	A<15..0>
	Note: FPA is identified by decoding Virtual Address bits A<31..28>		
N/A	0xE0000000 (Virtual)	Floating Point Accelerator	A<12..0>

5.2. Main Memory

Main memory is the primary system memory. It is contiguous in physical addresses. The minimum size is implementation dependent but is at least 2 Megabytes. The number of MMU address bits decoded to define the maximum physical address space is also implementation dependent. See the Bus Error register section on the handling of accesses to addressable but not-existing memory.

Initialization: Software

TYPE	DEVICE ADDR	DEVICE	PHYS. SPACE
0	Contiguous, starting at 0	Main Memory	Implementation Depend.

ADDR	REGISTER	DATA	TYPE
0	MEMORY 0	LONG	READ-WRITE
....			

In most implementations, main memory is built from dynamic RAM chips. The dynamic RAMs are refreshed in hardware.

Main memory must be protected by either parity or ECC check bits. In both cases, the protection bits (parity or ECC check bits) must be initialized by writing memory before use. Errors detected in using memory are enabled and reported through the Memory Error registers, below.

In implementations with ECC memory, the amount of memory data accessed in response to a CPU or DVMA bus cycle may be larger than required by the CPU or DVMA device. Memory errors detected in unused locations during the access may be reported to the Memory Error registers.

5.3. Frame Buffer: Data Organization

Sun-3 implementations may have no video memory, monochrome video memory, or color video memory. Monochrome video memory may be implemented using a main-memory frame buffer, a copy-mode frame buffer, or a video-RAM frame buffer. Color video memory must use the video-RAM frame buffer.

Additional, external display devices and frame buffers can be added to those implementations of the architecture that include a system bus. These external frame buffers are not within the scope of this document.

There are two data organizations for Sun-3 video memory: one for monochrome and one for color. The data organization determines how video memory data are displayed on the screen.

The data organization for monochrome video memory is as follows. Data bit 15 of Word 0 of the frame buffer is the first visible pixel in the upper left corner of the display. Consecutive words are displayed along the horizontal scanline left to right. After <display-width> number of pixels have been displayed, the next word is displayed at the beginning of the next horizontal line, up to <display-height> number of lines. <display-width> and <display-height> are implementation constants. The display data polarity is such that "1" bits are black on the screen and "0" bits are white.

```
N = <display-width> / 16
M = <display-height>

 15          0 15          0 15          0  15          0
+-------------+-------------+-------------+-------------+
| WORD 0      | WORD 1      | ...         | WORD N-1    |
+-------------+-------------+-------------+-------------+
| WORD N      | WORD N+1    | ...         | WORD 2*N-1  |
+-------------+-------------+-------------+-------------+
| ...         | ...         | ...         | ...         |
+-------------+-------------+-------------+-------------+
| WORD (M-1)*N | ...        | ...         | WORD M*N-1  |
+-------------+-------------+-------------+-------------+
```

The data organization for color video memory is as follows. Each byte of video data corresponds to one display pixel. A color map, which is a Type 1 device described elsewhere, translates the byte into the display pixel. Each bit within the byte corresponds to a memory plane; eight planes are supported. By convention, bit 0 refers to plane 0, bit 1 to plane 1, etc.

The high order byte in color video memory maps to the first visible pixel in the upper left corner of the display. Consecutive bytes are displayed as consecutive pixels along the horizontal scanline left to right. After <display-width> number of bytes are mapped to pixels, the next byte maps to a pixel displayed at the beginning of the next horizontal line, up to <display-height> number of lines. <display-width> and <display-height> are implementation constants.

```
N = <display-width>
M = <display-height>

 7           0 7           0 7           0  7           0
+-------------+-------------+-------------+-------------+
| BYTE 0=>PIX 0 | BYTE 1    | ...         | BYTE N-1    |
+-------------+-------------+-------------+-------------+
| BYTE N=>PIX N | BYTE N+1  | ...         | BYTE 2*N-1  |
+-------------+-------------+-------------+-------------+
| ...         | ...         | ...         | ...         |
+-------------+-------------+-------------+-------------+
| BYTE (M-1)*N | ...        | ...         | BYTE M*N-1  |
+-------------+-------------+-------------+-------------+
```

5.3.1. No Frame Buffer

If there is no frame buffer, then the Video Enable Bit and the Copy Enable Bit of the System Enable Register are not used.

5.3.2. Main-Memory Frame Buffer

In this alternative the frame buffer is resident in main memory and the video display is refreshed out of main memory.

The visible display area starts at main memory address 1 megabyte and extends to the size of the display. The maximum size of the visible display area is 128 kilobytes.

Initialization: Software
Interrupt: Level 4 Autovector

TYPE	DEVICE ADDR	DEVICE	PHYS. SPACE
0	0x00100000	Main-Memory Frame Bfr	A<16..0>

ADDR	REGISTER	DATA	TYPE
0x00100000	FB 0x00000000	LONG	READ-WRITE
....			
0x0011FFFC	FB 0x0001FFFC	LONG	READ-WRITE

The only relevant bit in the System Enable register is the Video Enable bit, which turns the display on and off. The Copy Enable bit has no effect.

5.3.3. Copy-Mode Frame Buffe

In this configuration, the frame buffer is loc ted in a dedicated 128K byte video memory. This video memory is dual ported: one port performs vide refresh, and the second port provides processor access.

Initialization: Software
Interrupt: Level 4 Autovector

TYPE	DEVICE ADDR	DEVICE	PHYS. SPACE
0	0x00100000	Copy-Mode Frame Bfr - Copy Mode access	A<16..0>
0	0xFF000000	Copy-Mode Frame Bfr - Direct access	A<16..0>

ADDR	REGISTER	DATA	TYPE
0x00100000 or 0xFF000000	FB 0x000C0000	LONG	READ-WRITE
....			
0x0011FFFC or 0xFF01FFFC	FB 0x0001FFFC	LONG	READ-WRITE

Relevant bits in the System Enable register are the Video Enable Bit and the Copy Enable Bit. The Video Enable bit turns the display on and off. The Copy Enable bit enables the copy mode (see below).

The video memory can be updated in two ways. First, it can be read and written directly, like memory. As such, it is visible as a 128 KByte block of memory locations starting at address 0xFF000000. Second, the video memory can be written in copy mode as a side effect of writing into a special region of main memory. This region starts at address 0x00100000.

Main memory shadows video memory in the range of physical addresses starting at 1 megabytes and extending for 128 kilobytes. This area of main memory is called the *copy region*. If the copy enable bit in the system enable register is set, then data written into this copy region is also written into the video memory at the same location within the 128K region. A read from the copy region returns data from the main memory and does not affect the video memory.

Note that the physical main memory and the Direct Access address space for the Copy-Mode frame buffer need not form a contiguous address space. If the Copy-Mode frame buffer is unimplemented, an attempt to access the Direct Access address space may result in a bus error. See the Bus Error register section for details.

5.3.4. Video-RAM Frame Buffer

In this configuration, the frame buffer is located in a dedicated area of memory which is dual ported: one port performs video refresh, and the second port provides processor access. The size of the Video-RAM frame buffer varies by implementation, in 256 KB increments, from 256 KB minimum to 2 MB maximum.

Initialization: Software
Interrupt: Level 4 Autovector

TYPE	DEVICE ADDR	DEVICE	PHYS. SPACE
0	0xFF000000	Video-RAM Frame Bfr	A<20..0>*

* Maximum; Minimum is A<17..0> (Impl. depend.)

ADDR	REGISTER	DATA	TYPE
0xFF000000	FB 0x00000000	LONG	READ-WRITE
....			
0xFF1FFFFC*	FB 0x001FFFFC*	LONG	READ-WRITE

* Address for 2 MB maximum

The only relevant bit in the System Enable register is the Video Enable bit, which turns the display on and off. The Copy Enable bit has no effect.

The video memory is updated by reading and writing it directly, like memory. Note that the physical main memory and the address space for the Video-RAM frame buffer need not form a contiguous address space. If the Video-RAM frame buffer is unimplemented, an attempt to access its address space may result in a bus error. See the Bus Error register section for details.

5.4. Color Map

The Color Map maps bytes in color video memory into display pixels. It contains three 256 byte sections: one each, for red, green, and blue.

Initialization: Software
Interrupt: Level 4 Autovector

TYPE	DEVICE ADDR	DEVICE	PHYS. SPACE
1	0x0000E000	Color Map	A<9..0>

ADDR	REGISTER	DATA	TYPE
0x000	RED MAP 0x00	BYTE	READ_WRITE
....			
0x0FF	RED MAP 0xFF	BYTE	READ_WRITE
0x100	GREEN MAP 0x00	BYTE	READ_WRITE
....			
0x1FF	GREEN MAP 0xFF	BYTE	READ_WRITE
0x200	BLUE MAP 0x00	BYTE	READ_WRITE
....			
0x2FF	BLUE MAP 0xFF	BYTE	READ_WRITE

The Color Map is generally written during vertical retrace. Following the assertion of a Level 4 interrupt for the color map, the map update must complete within 600 microseconds (the vertical retrace time) to avoid being visible in the display. Longer updates will complete but the display appearance may be affected.

5.5. Memory Error Registers

All Sun-3 implementations have either parity memory or error correcting (ECC) memory. ECC memory corrects (and optionally reports) single bit errors, and it detects and reports double bit errors. Errors detected in both versions of memory are reported with the Memory Error registers described in this section. In addition, other errors resulting from Sun-3 cache operations are also be reported in the Memory Error registers.

The Memory Error registers consist of a control and an address register. If an error occurs, the control register stores information relevant to the error, and the address register freezes the virtual address of the bus cycle in error.

Errors are reported via the non-maskable level 7 interrupt. In case of multiple (stacked) memory errors, the information relating to the first error is latched in the memory error registers. The interupt is held pending and the error information in both memory error registers is latched (frozen) until it is cleared (unfrozen) by a write to bits <31..24> of the Memory Error Address register. This write does not affect the enable bits D6:D4 of the Memory Error Control register. Disabling the PARITY CHECK bit in the Memory Error Control register for parity memory also clears the interrupt and error information. Finally, Power On and Watchdog resets clear the entire Memory Error Control register.

Initialization: See chapter on CPU Reset
Interrupt: Level 7 Autovector

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00080000	Memory Error Registers	A<2..0>

ADDR	REGISTER	DATA	TYPE
0	MEMORY ERROR CONTROL	BYTE	READ-WRITE
4	MEMORY ERROR ADDRESS	LONG	READ-ONLY

The memory error address register freezes the virtual address, the context number, and a DVMA/CPU identifier of the memory bus cycle in error. For implementations with ECC memory, the Memory Error Address register does not record the failing address on Correctable Errors (bit D<0> of the Memory Error Control register). The physical addresses for CE's are captured in the Correctable Error register on each memory board. (See the section "The Correctable Error Register" in the chapter on ECC Memory.)

MEMORY ERROR ADDRESS REGISTER

BIT	NAME	TYPE	MEANING
D<27..00>	VA<27..00>	read-only	Virtual Address (28 bit)
D<30..28>	CX<2..0>	read-only	Context Number (3 bit)
D<31>	DVMA-BIT	read-only	DVMA cycle error

The control register has two formats, one for parity memory and one for ECC memory. The two formats are structured similarly: D<3..0> record the type of error, D<6..4> are enable bits, and D<7> is the interrupt bit. The control register for ECC memory records both ECC errors and errors resulting from Sun-3 cache operations (if implemented).

5.5.1. Error Control Reg for Parity Memory

For systems equipped with parity main memory, the memory error control register provides the necessary control and information to deal with parity errors.

It stores the information on the byte(s) causing the parity error, sets a pending parity error interrupt, and provides functions to test parity error checking.

MEMORY ERROR CONTROL REGISTER for Parity Memory

BIT	NAME	TYPE	MEANING
D<0>	PARITY ERROR 00	read-only	Parity Error, bits D07:D00
D<1>	PARITY ERROR 08	read-only	Parity Error, bits D15:D08
D<2>	PARITY ERROR 16	read-only	Parity Error, bits D23:D16
D<3>	PARITY ERROR 24	read-only	Parity Error, bits D31:D24
D<4>	PARITY CHECK	read-write	Enable parity checking
D<5>	PARITY TEST	read-write	Test by inverting parity
D<6>	PARITY INT ENBL	read-write	Parity interrupt enable
D<7>	PARITY INTRPT	read-only	Parity interrupt (level 7)

- The four parity error bits are set when a parity error is detected in the corresponding byte.
- Parity check is set to enable parity checking on memory read cycles.
- Parity test is set to write parity with the inverse polarity to test the operation of the parity error circuitry. With parity test off, correct parity is generated on all memory write cycles.
- Parity interrupt enable enables level 7 interrupts if a parity error is detected.
- Parity interrupt is true if a parity interrupt is pending.

5.5.2. Error Control Reg for ECC Memory

For systems equipped with ECC main memory, the Memory Error Control register provides the necessary control and information to deal with both cache related and ECC errors.

MEMORY ERROR CONTROL REGISTER for ECC Memory

BIT	NAME	TYPE	MEANING
D<0>	CE	read-only	Correctable Error
D<1>	UE	read-only	Uncorrectable Error
D<2>	WBACKERR	read-only	Write Back error
D<3>	WB TIMEOUT	read-only	Write Back cycle time out
D<4>	ENABLE CE	read-write	Enable CE recording
D<5>	BUSHOLD	read-write	Hold memory bus mastership
D<6>	ENABLE INT	read-write	Enable error interrupts
D<7>	ERROR INT	read-only	Error interrupt

- The CE bit records a correctable error. It functions somewhat differently from other error bits D<3..1>. If enabled by the ENABLE CE bit D<4>, the CE bit is set if the ERROR STATUS bit D<0> of the Correctable Error Reg on ANY memory board is active (indicating a CE on that board). It resets only if ALL Error Status bits are reset. In particular, resetting the Memory Error Register by writing to the Memory Address Register does NOT reset the CE bit. (See also the section "The Correctable Error Register" in the chapter on ECC

Memory.)

- The UE bit records an Uncorrectable error.
- WBACKERR is set whenever there is a translation exception during a Write Back bus cycle for implementations with a cache. See also the section "Cache Error Conditions" in the Sun-3 Cache chapter.
- The WB TIMEOUT error bit is set whenever a bus time out is detected on an Write Back bus cycle for implementations with a cache. Write Back cycles are asynchronous with respect to CPU and DVMA bus cycles. See also the section "Cache Error Conditions" in the Sun-3 Cache chapter.
- The address captured in the Memory Error Address register for WBACK and TIMEOUT errors is that of the cache block which caused the error.
- ENABLE CE enables the recording of a Correctable error as bit D<0>.
- The BUSHOLD bit causes the CPU board to retain memory bus mastership for those implementations that support a multi-master memory bus.
- ENABLE INT enables an interrupt to the processor (bit D<7>) for any of the error conditions D<3..0>.
- The ERROR INT bit signals a level 7 interrupt to the processor.

The Memory Error Address register acts in conjunction with the Memory Error Control register for ECC memory to freeze the failing virtual address for all error conditions recorded in Error register bits D<3..1> (not CE's). The physical addresses for CE's are captured in the Correctable Error Reg on each memory board.

On WBACKERR and WB TIMEOUT errors, the failing virtual address is that of the former cache entry being written in the Write Back bus cycle. The DVMA-BIT, D<31> of the address register, is set to 0.

5.6. Clock

The timer is an Intersil 7170 time-of-day clock with battery backup. The timer crystal has a frequency of 32.768 kHz. It is expected that the clock interrupt output is driven in the 100 Hz periodic mode. This clock interrupt output signal causes an interrupt request on level 5 or 7 via the interrupt register, if the respective levels are enabled.

```
Interrupt:      Level 5 or 7 autovector
Initialization: None
Reference:      Intersil 7170 Data Sheet
```

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00060000	Time of Day Clk A<4..0>	

ADDR	REGISTER	DATA	TYPE
0	CLOCK REG 0x0	BYTE	READ-WRITE
.....			
0x11	CLOCK REG 0x11	BYTE	READ-WRITE

5.7. Interrupt Register

The interrupt register provides for the generation of software interrupts and controls the video and clock hardware interrupts on the board. It has the following fields:

Initialization: See chapter on CPU Reset
Interrupt: Level 1,2,3,4,5,7, autovectored

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x000A0000	— Interrupt Reg	(1 byte)

ADDR	REGISTER	DATA	TYPE
0	INTERRUPT REGISTER	BYTE	READ-WRITE

INTERRUPT REGISTER

BIT	NAME	TYPE	MEANING
D<0>	EN.INT	read-write	Enable all Interrupts
D<1>	EN.INT1	read-write	Software Interrupt Level 1
D<2>	EN.INT2	read-write	Software Interrupt Level 2
D<3>	EN.INT3	read-write	Software Interrupt Level 3
D<4>	EN.INT4	read-write	Enable Video Intrpt Levl 4
D<5>	EN.INT5	read-write	Enable Clock Intrpt Levl 5
D<6>	EN.INT6	read-write	(reserved)
D<7>	EN.INT7	read-write	Enable Clock Intrpt Levl 7

EN.INT. This bit enables all interrupts. If this bit is off, no interrupts will occur.

EN.INT<3..1>. These bits cause software interrupts on the corresponding level. The interrupt request caused by an EN.INT<3..1> bit stays active until software clears the corresponding bit.

EN.INT4 enables video interrupt requests on level 4. When enabled, a level 4 interrupt request is set at the rising edge of vertical retrace. The level 4 interrupt request is cleared by momentarily turning off the EN.INT4 bit. This bit has no effect in implementations which have no memory frame buffers.

EN.INT5 enables clock interrupt requests on level 5. When enabled, a level 5 interrupt request is set on the rising edge of the clock interrupt output. The level 5 interrupt request is cleared by momentarily turning off the EN.INT5 bit.

EN.INT6 is a reserved bit. It can be read and written but has no effect.

EN.INT7 enables clock interrupt requests on level 7. When enabled, a level 7 interrupt request is set on the rising edge of the clock interrupt output. The level 7 interrupt request is cleared by momentarily turning off the EN.INT7 bit.

5.8. EPROM

The EPROM device consists of one 27128, 27256, or 27512 type EPROM providing 16K, 32K, or 64K bytes of PROM storage, respectively.

```
Initialization: None
Interrupt:      None
Reference:      None
```

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00100000	EPROM	A<15..0>

ADDR	REGISTER	DATA	TYPE
0	EPROM BYTE 0	BYTE	READ-ONLY
1	EPROM BYTE 1	BYTE	READ-ONLY
....			

Unlike other devices, the EPROM is addressed directly with virtual address bits from the CPU. Thus, even though each 8K page of EPROM data must be enabled with its own entry in the page map, the translated address bits A<15..13> from the MMU are ignored. The EPROM is addressed instead with address bits A<15..0> (for 64KB) from the CPU.

The EPROM device is also accessed in boot state. In boot state, all supervisor program fetches are forced to fetch from the EPROM device, independent of the setting of the memory management.

5.9. EEPROM

The EEPROM device consists of one 2816 type EEPROM providing 2K Bytes of electrically erasable storage.

```
Initialization: None
Interrupt:      None
Reference:      None
```

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00040000	EEPROM	A<10..0>

ADDR	REGISTER	DATA	TYPE
0	EEPROM BYTE 0	BYTE	READ-WRITE
1	EEPROM BYTE 1	BYTE	READ-WRITE
....			

To modify the EEPROM, each byte must be written separately. After writing each byte Software must guarantee at least a 10 millisecond pause before the EEPROM can be read or written again.

Bytes can be read successively with no delays.

5.10. Serial Port

Serial ports are implemented with the Zilog 8530 SCC (Serial Communication Controller). The SCC features two high-speed, fully symmetrical and highly programmable serial channels with built-in baud-rate generators. Channel A is connected to the UART A, channel B to UART B. The clock input to the SCC is a 4.9152 MHz clock, independent of the CPU clock.

Software must guarantee the device recovery time of 1.6 microseconds.

The SCC is mapped as follows: ··

```
Interrupt:  --  Level 6 Vectored (preferred) or Autovectored
Initialization: See chapter on CPU Reset
Reference:      Zilog 8530 SCC data sheet
Recovery Time:  1.6 microseconds
```

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00020000	Serial Port	A<2..1>

ADDR	REGISTER	DATA	TYPE
0	CH B CONTROL	BYTE	READ-WRITE
2	CH B DATA	BYTE	READ-WRITE
4	CH A CONTROL	BYTE	READ-WRITE
6	CH A DATA	BYTE	READ-WRITE

5.11. Keyboard/Mouse UART

These serial ports are implemented with the Zilog 8530 SCC (serial communication controller). The SCC features two high-speed, fully symmetrical and highly programmable serial channels with built-in baud-rate generators. Channel A is connected to the Keyboard, channel B to the mouse. The clock input to the SCCs is a 4.9152 MHz clock, independent of the CPU clock. Control lines are not used.

Software must guarantee the device recovery time of 1.6 microseconds.

The SCC is mapped as follows:

```
Interrupt:      Level 6 Vectored (preferred) or Autovectored
Initialization: See chapter on CPU Reset
Reference:      Zilog 8530 SCC data sheet
Recovery Time:  1.6 microseconds
```

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00000000	Keyboard/Mouse UART	A<2..1>

ADDR	REGISTER	DATA	TYPE
0	CH B CONTROL	BYTE	READ-WRITE
2	CH B DATA	BYTE	READ-WRITE
4	CH A CONTROL	BYTE	READ-WRITE
6	CH A DATA	BYTE	READ-WRITE

5.12. Encryption Processor

The Encryption processor is an AMD 8068 data ciphering processor providing high-speed NBS DES encryption. To access an internal register in the 8068, the address register must be written first. Once the address register is setup, the selected register can be accessed repeatedly.

```
Initialization: None
Interrupts:     None
Reference:      AMD 8068 data sheet.
Recovery Time:  1.6 microseconds
```

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x001C0000	Data Encryption Processor	A<1>

ADDR	REGISTER	DATA	TYPE
0	DATA REGISTER	BYTE	READ-WRITE
2	ADDRESS REGISTER	BYTE	WRITE-ONLY

5.13. AMD Ethernet Interface

The AMD Ethernet Interface uses the 7990 chip for DVMA transfers. In its maximum DVMA configuration, the 7990 accesses the top 16 Megabytes of the current virtual address space with a supervisor data function code. (See also the chapter on DVMA Devices.) The 7990 must be configured in BCON=0 mode in its CSR register.

The 7990 must access physical memory in TYPE 0 space only; bus cycles to non-existing memory or to non-Type 0 space will not complete, and the 7990 will post a timeout error. The 7990 can also post a timeout error because of a protection error. Neither error is reported to the CPU. Memory errors (either parity errors or uncorrectable errors) on read operations are not reported to the 7990 but are reported only to the CPU, as interrupts through the Memory Error registers.

Initialization: See chapter on CPU Reset
Interrupts: Level 3, Autovector
Reference: AMD 7990 data sheet.

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00120000	AMD Ethernet Interface	A<23..0>

ADDR	REGISTER	DATA	TYPE
0	DATA PORT	WORD	READ-WRITE
2	CONTROL PORT	WORD	READ-WRITE

5.14. Intel Ethernet Interface

The Intel Ethernet Interface uses the Intel 82586 chip for DVMA transfers. In its maximum DVMA configuration, the 82586 accesses the top 16 Megabytes of the current virtual address space with a supervisor data function code. (See also the chapter on DVMA Devices.)

The 82586 must access physical memory in TYPE 0 space only; bus cycles to non-existing memory or to non-Type 0 space will not complete, but will set the ERR bit D<1> in the Ethernet Control register. Other conditions which set the ERR bit include a protection error or a memory error (either parity error or uncorrectable error) on read operations.

Once the ERR bit is set, Sun-3 implementations of the Intel Ethernet Interface must inhibit further Ethernet transfers until the RESET* signal in the Ethernet Control register is asserted (RESET* = 0).

The 82586 is connected to the system in a permanent byte-reversed mode; that is, within each word, 82586 bits 0 through 7 are mapped to MC68020 bits 8 through 15, and visa versa. This causes Ethernet data to be stored in memory in CPU byte order, whereas 82586 control blocks in memory are byte swapped.

Overall operation of the Ethernet Interface is controlled by the Ethernet Control register, which has the following definition.

```
Initialization: See chapter on CPU Reset
Interrupts:     Level 3, Autovector
Reference:      Intel 82586 data sheet.
```

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x000C0000	Intel Ethernet Interface	A<23..0>

ADDR	REGISTER	DATA	TYPE
0	CONTROL REGISTER	BYTE	READ-WRITE

The fields of the Ethernet control register are assigned as follows:

ETHERNET CONTROL REGISTER

BIT	NAME	TYPE	MEANING
D<0>	INT	read-only	Interrupt Pending
D<1>	ERR	read-only	Error Pending
D<2>	0	read-only	0
D<3>	0	read-only	0
D<4>	INTEN	read-write	Interrupt Enable
D<5>	CA	read-write	Channel Attention
D<6>	LOOPB*	read-write	Loopback
D<7>	RESET*	read-write	Reset

INT signals Interrupt from the 82586 or an error pending condition (ERR=1).

ERR indicates that a Bus Error occured during an 82586 channel operation, inhibiting further channel activity. To reset the *ERR* condition, the *RESET** bit must be set active (RESET* = 0).

RESET bit in the Ethernet control register must be activated.

INTEN enables 82586 interrupts to the CPU.

CA signals channel attention to the 82586.

*LOOPB** controls whether the front-end encoder/decoder is configured in loopback mode (LOOPB* = 0) or connected to the transceiver cable (LOOPB* = 1).

*RESET** initializes the 82586 when active (RESET* = 0) and allows normal operation when inactive (RESET* = 1). It also clears the *ERR* condition when active.

5.15. SCSI Interface

The SCSI interface provides DVMA access to Type 0 space and supports programmed I/O accesses from the CPU. In its maximum DVMA configuration, SCSI accesses the top 16 Megabytes of the current virtual address space with a supervisor data function code. (See also the chapter on DVMA Devices.)

The SCSI Control logic uses two addressable VLSI devices: the NCR 5380 SBC (SCSI Bus Controller) and the AMD 9516 UDC (Universal DMA Controller). The 5380 SBC is a byte oriented chip which implements disconnect/reconnect and has high-current drivers allowing it to be connected directly to the SCSI bus. To minimize the memory bandwidth requirement, byte-to-word packing is done with external logic, and the packed words are transferred to memory under control of the 9516 UDC. An 8 KB FIFO queue between the SCSI Bus Controller and the DMA Controller buffers packed SCSI data to improve performance.

For additional information on these chips, consult the appropriate data sheets.

5.15.1. SCSI Addressable Devices

The following addressable devices are implemented in the SCSI Control logic:

Interrupt: Level 2 Autovector
Initialization: See chapter on CPU Reset

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x00140000	SCSI Interface	A<23..0>

ADDR	REGISTER	DATA	TYPE
A<4..0>='00nnn'	SCSI BUS CONTROLLER	BYTE	READ-WRITE
A<4..1>='100n'	UNIVERSAL DMA CNTLR	WORD	READ-WRITE
A<4..1>='1010'	FIRSTBYTE REG	WORD	READ-ONLY
A<4..1>='1100'	SCSI STATUS REG	WORD	READ-WRITE

Note: 'n' indicates bits decoded by the chip

The SCSI Status register provides control and status information. Bits 0 to 3 are writeable and are cleared by system reset. Bits 9 to 15 are read-only and provide status information. The Firstbyte register, used with the Status register, records the leftover byte in an odd length transfer.

SCSI STATUS REGISTER

BIT	NAME	TYPE	MEANING
D<0>	SCSI_RES-	read-write	SCSI Reset (Minus active)
D<1>	FIFO_RES-	read-write	FIFO Reset (Minus active)
D<2>	SBC_INT_EN	read-write	SBC Interrupt Enable
D<3>	SEND	read-write	Send data to disk
D<4..8>	Unused	read-only	Read back as 0's
D<9>	SBC_INT	read-only	SBC Interrupt pending
D<10>	FIFO_EMPTY	read-only	FIFO empty
D<11>	FIFO_FULL	read-only	FIFO full
D<12>	SECONDBYTE	read-only	Second byte
D<13>	SCSI_BERR	read_only	SCSI bus error
D<14>	SCSI_BCNFL	read-only	SCSI bus conflict
D<15>	XFR_ACTIVE	read-only	Transfer active

FIRSTBYTE REGISTER

BIT	NAME	TYPE	MEANING
D<0..7>	Unused	read-only	Unused data byte
D<8..15>	LAST BTYE	read-only	Last byte, unaligned xfr

- SCSI_RES- (minus active) resets both the 5380 SBC and the 9516 UDC during system reset and should be asserted in case of SCSI_BERR or SCSI_BCNFL errors.

- FIFO_RES- (minus active) initializes the FIFO controls and should be asserted when the SEND direction is established or after SEND has been established.

- SBC_INT_EN gates the interrupt output from the 5380 SBC onto auto-vector interrupt level 2.

- SEND indicates the direction of data transfer is to the disk.

- FIFO_EMPTY and FIFO_FULL indicate the state of the fifo buffer.

- SECONDBYTE indicates that an odd number of bytes were transferred on the SCSI bus. Since DVMA transfer is by words, the leftover byte needs to be read from the Firstbyte Register bits 8 to 15.

- SCSI_BERR indicates that an error occured during a DVMA cycle to memory. Possible errors include translation errors, protection errors, accesses to addressable but unimplemented memory (Type 0 space) or non-Type 0 space, and memory data errors (parity or uncorrectable errors).

- XFR_ACTIVE indicates that the 5380 SBC is actively transferring data on the SCSI bus. The cpu should not attempt to access the SBC during this time. If it should do so, SCSI_BCNFL will be set.

5.15.2. SCSI Programming Guide

The 5380 SBC can be programmed for either DVMA or programmed IO transfer. If programmed IO is used, the 9516 UDC should be programmed off and FIFO_RES- asserted (negative). To use DVMA, the 5380 SBC must be programmed for "non-block DMA" mode. The 5380 SBC can be programmed to interrupt at level 2 on any of the following conditions:

- selection/reselection
- EOP (not used)
- SCSI bus reset
- SCSI bus memory error
- Bus phase mismatch (e.g. at the end of SCSI data transfer)
- loss of SCSI bus BSY signal

For DVMA transfers,the 9516 UDC must be programmed to the "flyby transaction, single operation" mode. All DVMA transfers are done in word mode. The UDC generates a 24 bit address in the master mode and contains 2 channels, of which only channel 1 is used. The UDC can be programmed to interrupt at level 2 on any of the following conditions:

- pattern match (not used)
- EOP
- chain aborted

The EOP interrupt can be generated either internally on a terminal count or externally if SCSI_BERR or SCSI_BCNFL occurs. Chain-aborted interrupt referrs to the UDC's auto-chaining operation, which allows the UDC to use DMA to transfer from memory all the words needed to initialize its internal registers. The processor need only supply the UDC with the starting address of the auto-chain memory block.

Several registers internal to the UDC are described as slow-readable, meaning they take 2 microseconds to read. These are the pattern register, the mask register, interrupt vector register, and channel mode registers. The slow-readable registers are not accessible in the Sun-3 implementation and attempts to read them will return unknown data. Another peculiarity of the UDC is that commands to it must be separated by 4 UDC clock cycles; the UDC clock rate is implementation dependent. This requirement will be a software responsibility.

The 8K FIFO is used to buffer DVMA disk transfers. It takes about 16 ms for the disk to transfer 8K bytes, and about 4 ms for DVMA to fill or empty the FIFO, depending on the implementation clock rate, the DVMA controller, and other memory activity. The disk and DVMA operations can be concurrent.

5.15.3. SCSI FIFO Testability

To verify that all 8K bytes of the FIFO buffer are reading and writing properly, the disk side of the FIFO must be disabled so that no data is transferred to the disk. (This can be done by programming the 5380 SBC chip to not transfer data.) To fill the FIFO, the 9516 DMA controller chip is programmed to do an 8K byte transfer, as in normal disk operations. Polling the FIFO_FULL bit in the SCSI STATUS register will verify that the FIFO is filled.

To empty the FIFO, the 9516 DMA controller chip is programmed to do an 8K byte transfer from disk to memory, but without enabling the 5380 SBC chip and without resetting the FIFO. This will cause all 8K bytes to read back into memory.

5.16. VMEbus Interface

The VMEbus interface is dual-ported. The VMEbus Master Interface provides access from the CPU to the VMEbus, whereas the VMEbus Slave Interface provides access from the VMEbus to the CPU for DVMA transfers. (See the chapter on DVMA Devices, below.)

The chart below lists VMEbus options supported by Sun-3 systems.

- Address Bus Option: A32 MASTER; A32 SLAVE
- Data Bus Option: D32 MASTER, D32 SLAVE
- Timeout Period: 100 microseconds minimum excluding bus acquisition
- Arbiter Option: ONE (single level), can be disabled
- Requestor Option: ROR (release on request)
- Interrupt Handler Options: IH(1-7)

Support for VMEbus read-modify-write cycles requires clarification. Read-modify-write cycles for the VMEbus Slave interface are executed as atomic cycles with respect to other VMEbus requests. Atomicity between VMEbus Slave accesses to main memory and CPU accesses to main memory, however, is not guaranteed.

Read-modify-write cycles from the CPU to the VMEbus Master interface are guaranteed to be atomic with respect to other VMEbus requests. They may, however, not appear on the bus in the read-modify-write format defined in the VMEbus specification.

Normally a Sun-3 CPU board acts as VMEbus arbiter. Jumpers, however, are provided to disable the arbiter and permit operation as a non-arbiting VMEbus board.

Neither the slave nor the master interface supports sequential access modes. Finally, the Sun-3 CPU board does not support an interrupter function to the VMEbus.

5.17. VMEbus Master Interface

The VMEbus Master Interface uses two page map Type codes: one for 16-bit data, and one for 32-bit data. For each Type code, three VMEbus address spaces are supported: 4 Gbytes minus the top 16 MBytes for 32-bit addressing, the top 16 MBytes minus the top 64 KBytes for 24-bit addressing, and the top 64 KBytes for 16-bit addressing.

Sun-3 implementations of the VMEbus Master interface place no alignment restrictions on CPU data; all MC68020 alignments will be supported for VMEbus cycles. Data transfers from the CPU into D8 VMEbus ports must be byte transfers only; attempts to perform word or longword CPU bus cycles into D8 ports will yield erroneous results.

The timeout period for transfers is implementation dependent but a minimum of 100 microseconds.

```
Initialization: See the chapter on CPU Reset
Interrupts:     Level 1 through 7, Vectored
Exceptions:     Timeout period minimum 100 microseconds
Reference:      Motorola VMEbus Specification
```

TYPE	DEVICE ADDR	DEVICE	PHYS. SPACE
	Note: Type 2 space is VMEbus 16 bit data space 32 bit decode of MMU address identifies VMEbus address space		
2	0x00000000	VMEbus A32 space	A<31..0>
2	0xFF000000	VMEbus A24 space	A<23..0>
2	0xFFFF0000	VMEbus A16 space	A<15..0>
	Note: Type 3 space is VMEbus 32 bit data space 32 bit decode of MMU address identifies VMEbus address space		
3	0x00000000	VMEbus A32 space	A<31..0>
3	0xFF000000	VMEbus A24 space	A<23..0>
3	0xFFFF0000	VMEbus A16 space	A<15..0>

6. DVMA Devices

There are four DVMA devices defined for Sun-3: the AMD Ethernet interface, the Intel Ethernet interface, the SCSI interface, and the VME Slave interface. In addition, Sun-3 implementations with caches may implement the Cache Flush operation in Control Space as a DVMA request.

All DVMA devices may access main memory data (Type 0 space) only. Access to addressable but non-existing physical memory or to devices other than memory will not complete. How these and other error conditions are reported depends on the DVMA device and is described in the Device Space chapter (or the Cache chapter, for the Cache Flush operation).

DVMA requests are assigned the following priority: first, Ethernet; second, the Cache Flush; third, the SCSI; and fourth, the VMEbus Slave. All DVMA requests have higher priority than the CPU.

All DVMA devices must map addresses from a DVMA device address space into a Sun-3 virtual address space. This space is identified by a 3 bit Context ID, a 28 bit virtual address, and protection access codes. System DVMA accesses are supported from Ethernet and SCSI, plus both System and User DVMA accesses from the VMEbus. No protection checking is done during the Cache Flush operation.

The following sections discuss how DVMA address spaces map into the CPU virtual address space for each device. The Device Space chapter includes information on addressing controls for the AMD and Intel Ethernet devices and the SCSI interface. The section on Control Space operations in the Sun-3 Cache chapter discusses the Cache Flush operation.

6.1. Ethernet and SCSI DVMA

The two Ethernet devices (AMD and Intel) and the SCSI interface map address spaces the same. Each device's 16MB address space is mapped into the highest 16 MB of the CPU virtual address space for the current context, as identified by the Context ID register.

All three devices support only System DVMA and are assigned Supervisor Data access for protection checking. The handling of protection errors, as well as page faults and memory data errors, depends on the device; see the Device Space chapter. All memory data errors are reported to the CPU as interrupts, if enabled.

Ethernet and SCSI DVMA are always enabled and are not affected by the EN.SDVMA bit in the System Enable register or the User DVMA Enable register.

The Intel Ethernet section should also be consulted about the ordering of control block data in memory.

6.2. The VMEbus Slave Interface

The VMEbus Slave interface supports both System and User DVMA into the CPU virtual address space. The following features are included in both modes of transfers.

- Byte, Word, and Longword transfers are supported.
- Both System DVMA accesses and User accesses are defined entirely by the VMEbus 24 or 32 bit VMEbus address and the address modifiers identifying the address mode, AM<5..4>. The VMEbus Address Modifier bits AM<2..0> defining the access protection of the VMEbus request are ignored. The 16 bit address space is also ignored.
- Access to non-existing memory or other (non-Type 0) devices results in a VMEbus Bus Error return.

- A VMEbus Bus Error is also signalled if the DVMA cycle encounters a page fault, protection error, or (on read cycles only) a memory error. Memory errors, which include parity errors or uncorrectable double bit errors, are also reported to the CPU as interrupts, if enabled. Protection checking is discussed below.

- Implementations of DVMA can offer high-bandwidth burst modes of transfer that allow fast DVMA devices to increase throughput by eliminating repeated bus arbitration. Implementations must specify whether this feature is supported and what the device requirements must be to activate this transfer mode.

6.2.1. System DVMA

System DVMA responds to the lowest megabyte of the VMEbus address range in both the 24-bit and 32-bit VMEbus address spaces. This 1MB space is mapped into the highest megabyte in the virtual address space of the current context, as indicated by the Context ID register. System DVMA is enabled via the EN.SDVMA bit in the System Enable register. System DVMA cycles use Supervisor access for protection checking; a VMEbus Bus Error is signalled if the page being accessed is not valid or if the access has a protection violation. This error is not visible to the CPU.

VME-Address A24, A32	Virtual Address
[0x00000000..0x00FFFFFF]	[0xFFF00000..0xFFFFFFFF]

6.2.2. User DVMA

User DVMA responds to the most significant 2 GBytes of the VMEbus 32-bit address space (i.e., A<31>=1). The 16-bit and 24-bit address spaces are ignored. The VMEbus User DVMA address space contains two fields. VMEbus A<30..28> map into a 3 bit context ID, and VMEbus A<27..1> with DS0 and DS1 map directly into the 28 bit virtual address space for this context. User DVMA is enabled via the User DVMA enable register, which has one bit per context. If a context is not enabled for user DVMA, then the CPU does not respond to the corresponding addresses on the VME cycle at all; this allows sharing of the upper 2 gigabytes of the VME address space with other VME devices.

User DVMA cycles have User access for protection checking; a VMEbus Bus Error is signalled if the page being accessed is not valid or the access has a protection violation. This bus error is not visible to the CPU. VMEbus masters that expect bus error support from the CPU must then post an interrupt to the CPU and must make the appropriate information about the bus error available to the CPU.

VME-Address	Virtual Address
[0x80000000..0x8FFFFFFF]	CX=0 [0x00000000..0x0FFFFFFF]
[0x90000000..0x9FFFFFFF]	CX=1 [0x00000000..0x0FFFFFFF]
[0xA0000000..0xAFFFFFFF]	CX=2 [0x00000000..0x0FFFFFFF]
[0xB0000000..0xBFFFFFFF]	CX=3 [0x00000000..0x0FFFFFFF]
[0xC0000000..0xCFFFFFFF]	CX=4 [0x00000000..0x0FFFFFFF]
[0xD0000000..0xDFFFFFFF]	CX=5 [0x00000000..0x0FFFFFFF]
[0xE0000000..0xEFFFFFFF]	CX=6 [0x00000000..0x0FFFFFFF]
[0xF0000000..0xFFFFFFFF]	CX=7 [0x00000000..0x0FFFFFFF]

7. CPU Reset

There are five possible reset sources for Sun-3 configurations: Power-On reset, VMEbus SYSRESET-, Watchdog reset, a User Reset switch, and CPU reset. Two of these, the VMEbus SYSRESET- and the User Reset switch, are optional by configuration.

The following chart indicates the devices or system bus signals that are reset by each source. Following the chart, each reset source is described.

DEVICE/ VMEbus SIGNAL	POWER ON	VMEbus RESET	WATCH-DOG	USER SWITCH	CPU RESET
MC68020 CPU	Y	B	Y	Y	Y
MC68881 FPP	Y	B	Y	Y	Y
VMEbus Signals					
VMEbus SYSRESET-	P		P	P	P
VMEbus SYSFAIL-	Y		Y	Y	Y
Control Space					
System Enable reg	Y	B	Y	Y	
User DVMA Enable reg	Y	B	Y	Y	
Diagnostic reg	Y	B	Y	Y	
Device Space					
Floating Point Acc.	Y	B	Y	Y	
Memory Error Cntl reg	Y	B	Y	Y	
Interrupt reg	Y	B	Y	Y	Y
Serial Port	Y	B	Y	Y	
Keyboard/Mouse	Y	B	Y	Y	
AMD Ethernet Intrface	Y	B	Y	Y	Y
Intel Ethernet I/F	Y	B	Y	Y	Y
SCSI Interface	Y	B	Y	Y	
ECC Memory Enable reg	Y	B	Y	Y	
ECC Chip Diag. reg	Y	B	Y	Y	

Y = Always reset the indicated device during the indicated reset condition.
B = Bus jumper option: Reset indicated CPU devices if VMEbus SYSRESET- is driven active (=0) from some VMEbus board.
P = CPU-to-bus jumper option: Drive VMEbus SYSRESET- active (=0) during indicated reset condition for CPU board.

Power-On Reset. Power-On Reset (POR) is active for 100 milliseconds minimum after the power supply voltage reaches 4.5V. POR resets the CPU and clears the System Enable register forcing boot state, and it resets the diagnostic register, lighting all the LEDs. Other devices reset during Power-On reset are indicated above.

VMEbus SYSRESET-. The VMEbus SYSRESET- must be activated by a jumper in configurations with a system bus, normally only in the event that the CPU board is not the VMEbus arbiter. If so jumpered, the CPU board does not itself drive SYSRESET- or SYSFAIL- if the bus SYSRESET- is active.

Watchdog Reset. The Sun-3 architecture provides a watchdog circuit which generates a signal equivalent to power-on reset (POR) whenever the CPU halts with a double bus fault. A Watchdog Reset results in the same devices being reset as the Power-On reset.

User Reset Switch. An optional User Reset Switch resets the same devices as the Power-On Reset.

CPU Reset. CPU Reset results from the CPU executing a Reset instruction.

8. CPU Interrupts

8.1. Interrupt Sources

There are two sources of interrupts for Sun-3 systems: VMEbus interrupts and interrupts from devices defined in the Sun-3 architecture.

The VMEbus interrupt handler for Sun-3 systems supports all levels (1:7) of VMEbus interrupts. VMEbus interrupts have lower priority than onboard device interrupts at the same level.

A list of the interrupt assignments for devices defined in the Sun-3 architecture is in the table below:

Level	Device(s)
7	Memory Error or Clock
6	SCCs
5	Clock
4	Video (Sun-3 Frame Buffer or Color Map)
3	Ethernet or System Enable Register 3
2	SCSI or System Enable Register 2
1	System Enable Register 1

8.2. Interrupt Acknowledge Cycles

Interrupt acknowledge cycles are CPU Space cycles identified by A<17..16> = '11'. The interrupt vector number resulting from the execution of an interrupt acknowledge cycle may be a vector fetched from the interrupting device, an autovector corresponding to the level of the interrupt acknowledge cycle, or a spurious interrupt vector.

All Type 1 devices defined in the Sun-3 architecture use autovectored interrupts, except for the SCC UART's. Either vectored or autovectored interrupts may be implemented for UART's, with vectored being the preferred implementation. Devices on the VMEbus use vectored interrupts.

A spurious interrupt vector results whenever an interrupt acknowledge cycle terminates by the assertion of the bus error signal. Certain implementations of Sun-3 report spurious interrupts. In particular, the failure of a device to return a vectored interrupt may be reported as a spurious interrupt. The contents of the Bus Error register are not affected by the reporting of spurious interrupts.

9. The Sun-3 Cache Architecture

9.1. The Sun-3 Cache: Introduction and Overview

9.1.1. The Sun-3 Cache: Introduction

The Sun-3 cache architecture describes a set of caches with a common structure. In particular, all Sun-3 caches are direct mapped caches (i.e., one way set associative) with a fixed block size. They vary only in the number of cache blocks.

The number of cache blocks, together with other options indicated in the architecture, are Sun-3 cache implementation parameters. These parameters must be specified as part of the product specification.

9.1.2. The Sun-3 Cache: Overview

The Sun-3 cache is organized as a direct mapped virtual addressed cache containing 16 byte blocks (or lines). Its size is variable, from 1K blocks (16K bytes) up to 8K blocks (128K bytes) for the largest allowable Sun-3 cache.

Sun-3 caches support 8 virtual contexts with 28 bit virtual address spaces.

For each 16 byte cache block there is a corresponding cache tag field. The tag field contains address information to uniquely identify the block data plus protection and control information.

In the smallest Sun-3 cache, a cache block and its corresponding tag field are addressed by A<13..4>; in the largest Sun-3 cache, the block and tags are addressed by A<16..4>. To uniquely identify the virtual address for a cache block, its tags contain the 3 bit Context ID (CID) field plus a varying number of virtual address bits, depending on the cache size.

The tag field for the smallest Sun-3 cache contains address bits A<27..14>. Together with the cache address, these tags uniquely identify the block address, A<27..4>. Similary, the largest cache tags include A<27..17>. (See below for a description of all tag bits.)

The Sun-3 cache contains only data from main memory. That is, each cache block's virtual address must translate into a physical address from a Type 0 page in Device Space. Data from any Type 0 page which is not marked "Don't Cache" in the MMU may be cached; this includes, in particular, data accessible by User or System DVMA.

9.2. The Sun-3 Cache: System Programming Requirements

Sun-3 caches are virtual address caches; implicit with the use of a virtual address cache is a set of system programming restrictions which MUST be adhered to in order to guarantee system data consistency. In addition, other restrictions on the mapping of Supervisor pages are required for optimal cache performance.

9.2.1. Data Consistency: Overview

Virtual addressing allows aliasing: the possibility of multiple virtual addresses mapping to the same physical address. If a Sun-3 cache were used without system page mapping restrictions, any two arbitrary virtual addresses could occupy any two arbitrary cache locations and still map to the same physical

address. When cache blocks are modified, Sun-3 cache hardware provides NO data consistency checking between different cache blocks. Data can become inconsistent when changes at one cache location are not seen at another cache location. Ultimately, the data at the common physical address in main memory are going to include only part of the Write modifications from the several cache locations.

The Sun-3 cache architecture solves this data consistency problem by providing two distinct mechanisms. Both mechanisms require the interaction of software with special cache hardware to ensure consistent data. Briefly, the first mechanism requires that all alias addresses which map to the same data must match in their low order 17 bits (modulo 128KB) IF these data are to be cached. The second mechanism restricts data from being cached through the use of a "Don't Cache" bit which is defined for each page in the MMU Page Map.

Both of these mechanisms are explained in more detail below.

9.2.2. Data Consistency through Modulo 128K Addressing

The first, and prefered, method of guaranteeing data consistency is through restricting the use of alias addressing for cache data. To guarantee that all alias virtual addresses map to a common cache location, it is REQUIRED that any two alias virtual addresses must match in their low order 17 bits (i.e., modulo 128K). This applies to alias addresses within the same context as well as aliases between contexts.

Note that to guarantee data consistency, ANY write to a page requires that ALL references to that page (read or write) adhere to this restriction. No requirement is placed on alias addressing to Read Only pages.

Also note that other means are available to guarantee limited data consistency, for example, in alias addressing between User contexts. If the operating system flushes the cache through "Flush Cache Set (Context Match)" commands in Control Space at the time of User context switches, then data consistency between User contexts is assured. The cache flush, however, is relatively slow and does nothing for alias addressing within the context. Further, cache blocks with Supervisor protection are unaffected by this flush command.

The hardware controls to guarantee data consistency within a single cache block are described in the section "Cache Misses and Data Consistency" below.

9.2.3. Data Consistency through Don't Cache Pages

The second mechanism to ensure data consistency is through the use of a "Don't Cache Page" bit in the MMU Page Map. If this control bit is set for a page, then all data accesses to this page are made directly to and from main memory. In bypassing the cache, the virtual cache data consistency problem is avoided.

Since alias addressing is possible, if a page is marked "Don't Cache" in one MMU Page Map entry, then it must be marked "Don't Cache" in all alias Page Map entries. Data consistency is not guaranteed otherwise.

Generally, direct memory data accesses are much slower than cache accesses. Frequent references to "Don't Cache" pages will consequently harm system performance.

9.2.4. Mapping of Supervisor Pages

There are two additional requirements for the mapping of Supervisor pages on systems implementing a Sun-3 cache. These requirements are not related to cache data consistency, but rather are fundamental to the definition of a cache "hit"; see the section "Definition of a Cache Hit" below.

First, the Sun-3 cache architecture requires that all Supervisor pages must have identical address

mappings across all contexts.

Second, for protection consistency, the Sun-3 cache architecture requires that if a page is marked as having Supervisor access within one context, then that page must be marked as having Supervisor access for all contexts.

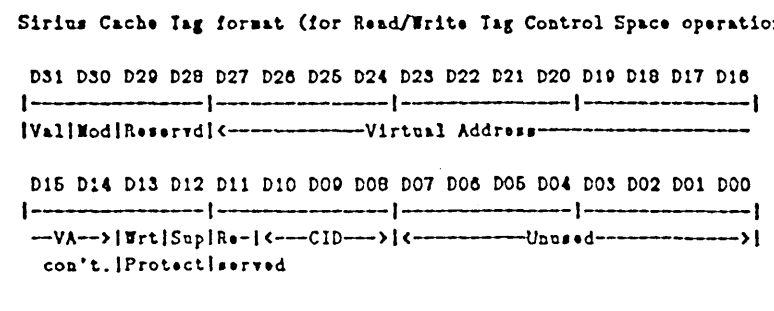
These requirements ensure that, with the cache hit and protection definitions stated below, no differences in data protection should be discernable whether a Sun-3 cache is enabled or disabled.

9.3. The Sun-3 Cache: Its Structure and Operation

9.3.1. Cache Tags

Sun-3 cache tags are listed below. The use of these tags in cache control is explained in the next section.

- Valid (1 bit): Indicates that the tags and data at the addressed cache block are both valid. Neither tags nor data have meaning if the Valid bit is reset.
- Modified (1 bit): Indicates that the cache block has been modified by one (or more) Write cycles.
- Virtual Address field (from 11 to 14 bits, depending on cache size): Virtual address bits which, together with the cache address, uniquely identify a cache block in a 28 bit virtual address space. For a 16K byte Sun-3 cache, the Virtual Address field is A<27..14>; for a 128K byte Sun-3 cache, the VA field is A<27..17>.
- Protection (2 bits): Write Allowed and Supervisor Access protection bits, identical to those in the MMU. The use of the Protection bits is explained below under Definition of Cache Protection.
- Context ID (CID) field (3 bits): Identifies the Context for the cache block.
- Reserved (3 bits): May be read or written, but have no effect.
- Unused VA bits (Variable; depends on cache size): Unused VA bits are undefined.
- Unused (8 bits): Unused bits (D<7..0>) are undefined.



9.3.2. Accessing the Cache

Cache operations include read cycles, write cycles, and Control Space operations. To access the cache for any of these operations, generally the following information is required: a Device Space/Control Space indicator (Function Code bits; implementation dependent); a 3 bit context I.D. (from the CID register or VME address bits A<30..28>); the 28 bit virtual address (A<27..0>); the Supervisor/User access bit; a Read/Write bit; and the data transfer size (Size bits; encoding is implementation dependent). On write cycles, up to one longword of data is also required; the data alignment is implementation dependent.

All of this information is required for cache read and write cycles. On Control Space operations, some may not be required; see the Control Space section, below. Timing requirements at the cache interface for this information are implementation dependent.

9.3.3. Definition of a Cache Hit

In this section, the cache "hit" is defined for read and write cycles and for the Block Copy (Read) and Block Copy (Write) Control Space operations. The "hit" criteria for other Control Space operations are explained in the Control Space section, below.

There are two requirements for a cache hit. First, the access address A<27..4> must match the cache virtual address tags plus the cache block address.

Second, either the access context ID must match the cache Context ID tags, or the cache Supervisor protection tag must be set. If the cache Supervisor Access tag is set, then the cache block may be accessed regardless of the access context ID.

This "hit" definition does not necessarily imply a valid cache access, since it takes no account of protection checking (below).

9.3.4. Definition of Cache Protection

Cache protection checking is defined for read and write cycles and for the Block Copy (Read) and Block Copy (Write) Control Space operations. Protection checking in inhibited on all other cache specific Control Space operations and during "Write Back" cycles (see Cache Access and Block Replacement, below).

There are three requirements. First, no cache protection violation can result unless there is a cache hit. Second, if the access has User protection, a protection violation results on a hit if either the cache block has Supervisor protection, or if the access attempts to write into a cache block whose Write Allowed tag is reset. Third, if the access has Supervisor protection, a protection violation results on a hit only if the access attempts to write into a cache block whose Write Allowed tag is reset.

A protection violation terminates the bus cycle with a bus error, while setting the Protection Error bit in the Bus Error register (on CPU bus cycles).

9.3.5. Enabling the Cache

The "Enable External Cache" bit, D<4> of the System Enable register, determines whether the cache is enabled for read and write cycles. In Boot state, the cache is disabled. If disabled, all cache accesses "miss" the cache, no cache blocks are written back to memory, and memory data are directly read from or written to main memory. Block Copy (Read) and Block Copy (Write) Control Space operations operate from main memory, ignoring the cache. All other Control Space operations for the cache, however, remain unaffected by the Enable bit.

9.3.6. Cache Initialization

The state of a Sun-3 cache following a Power-On reset, User Switch reset, or Watchdog reset may be indeterminate. The reset forces the Enable External Cache bit (above) to be inactive. To initialize the cache, the Valid tags for all cache blocks must be reset by issueing "Write Cache Tags" commands in Control Space before the Enable External Cache bit is set active (see Control Space, below).

9.3.7. Read and Write Cycles: Cache Hit Operation

For the Sun-3 cache, read and write cycles begin by determining whether the cache access "hits" or "misses" the cache. If the access "hits" the cache, then a protection check is made. If a protection violation is found, a protection error terminates the bus cycle; no valid data are read (read cycle) or written (write cycle), and no block tags are updated.

If no protection violation is found, the cache cycle is valid. On a valid read cycle, up to a longword of cache data is read from the cache. Data alignment is implementation dependent. On a valid write cycle, if the Modified block tag is active, then up to a longword of access data is written into the block. The sizes of data updates and alignment requirements are implementation dependent. No block tags are updated on either of these cycles.

If the Modified tag is inactive on a valid write cycle, then both the Modified block tag bit and the Modified bit in the MMU Page Map must be updated. The cache data may be rewritten from memory data during the tag and MMU update, depending on the implementation. Cache data will be updated by writing up to a longword of access data into the block. Again, the sizes of data updates and alignment requirements are implementation dependent.

9.3.8. Read and Write Cycles: Cache Miss Operation

A cache "miss" on read and write cycles in Device Space may be caused by either an access to a cacheable Type 0 page whose data are missing from the cache, or by an access to a "Don't Cache" page or a non-Type 0 page. Reading the MMU Page Map determines whether the access data should be in the cache.

If the MMU translates the access address to a valid Type 0 page whose "Don't Cache" bit is inactive, then the cache will be updated at the block corresponding to the access address on both read and write cycles. The cache tags will be updated to show the access context and virtual address, the Valid bit will be set active, and the Modified bit will be set on a write cycle.

9.3.9. Cache Misses and Data Consistency

The source of a valid block of data to update the cache following a cache "miss" may be either Type 0 memory or the old cache block data. The selection depends on the contents of the old cache block to be replaced. If the cache block contains an invalid entry, then the source of the data update is memory. If the cache block is valid, then data consistency issues determine the source.

In the section "Data Consistency through Modulo 128K Addressing" above, it was stated that cache data consistency would be guaranteed if all alias addresses were to map to a common cache location modulo 128K. With modulo 128K addressing, it is possible that the access virtual address and cache tag virtual address both translate to the same physical block. The Sun-3 cache architecture requires a physical address comparator to determine whether the translated block addresses are identical. The number of address bits compared in the physical address comparator is implementation dependent.

If the physical addresses compare, then the source of valid data to update the cache is the old cache block data. If the old cache block has not been modified, then the cache data and memory data are identical. Either can serve as the data source, depending on the implementation. If the old cache block has been modified, then the old cache data must be the source for the cache update.

Once the source of valid block data has been determined, the access cycle can complete. On read cycles, up to a longword of data may be read directly from the source or from the cache following the cache update, depending on the implementation. On write cycles, up to a longword of access data may be written into the cache. The size of data updates and cache data alignment are implementation dependent.

9.3.10. Write Back Cycles

The Sun-3 cache is a write-back cache: modified data are held in the cache until a cache block is either replaced (on a cache miss) or flushed (see the Control Space section, below). Memory cycles which write modified cache blocks back into memory are called Write Back cycles.

Following a cache "miss" on a read or write cycle, the cache block addressed on the access will be updated, provided the access address translates to a cacheable page. The sources for the update may be either the old cache block data or main memory (see above). If the source is the old cache block, cache data need not be rewritten to memory. If the source is the main memory, then an old cache block which is valid and modified must be written back to memory. This cache data may be temporarily buffered; the architecture does not require that the Write Back cycle to memory complete prior to updating the cache with new data.

Similarly, during a Flush Cache Set operation in Control Space, valid and modified cache blocks must be written back to memory if they satisfy the "flush match" criteria.

During a Write Back cycle, the virtual address for the block being rewritten must be translated in the MMU. Any translation error is signalled as a "Write Back" error interrupt to the CPU. No protection checking is performed.

9.3.11. Cache Error Conditions

There are two types of errors unique to systems with Sun-3 caches. These are Asynchronous Time Out errors and Write Back errors.

In Sun-3 systems with caches, a time out error may be reported differently, depending on when the condition is detected. If the condition is detected while either a CPU or DVMA bus cycle is in process, then the time out is reported as a bus error to the CPU or DVMA master. On CPU bus cycles, the TIMEOUT bit (D<5>) in the Bus Error register records the cause of the error. (See also the Bus Error register description.)

If the time out error is detected during a Write Back cycle (above), then it is reported, if enabled, as an interrupt to the processor. The TIMEOUT bit (D<3>) in the Memory Error Control register records this error.

A Write Back error results whenever a translation exception is detected for the address of a valid and modified cache block which must be written back to memory during a Write Back cycle (above). This error, if enabled, is also reported to the processor as an interrupt and is recorded as the WBACKERR bit (D<2>) in the Memory Error register. Note that no protection check is performed during the Write Back address translation.

9.4. The Sun-3 Cache and the MMU

9.4.1. The MMU Accessed Bit

In Sun-3 systems with no cache, an MMU Accessed bit is updated on every bus cycle to memory. In Sun-3 systems with a cache, no MMU update is made if the memory read or write access "hits" the cache. If the operating system resets an MMU Accessed bit, this means that the bit will not be set again in cache systems until there is a cache miss for data contained in that page. Note that the Control Space commands Block Copy (Read) and Block Copy (Write) may also update the MMU Accessed bit.

As a consequence, MMU Accessed bits may be somewhat inaccurate in reflecting page useage within Sun-3 systems with caches.

9.4.2. Modified Bits for the Cache and MMU

The cache and MMU Modified bits are coordinated as follows. If a Write access misses the cache, then the MMU Access and Modified bits plus the cache Valid and Modified bits are all set active when data are returned from main memory. For subsequent Writes to the same cache block, no tag updates are required.

If a Read access misses the cache, then the MMU Access bit and the cache Valid bits are set active when data are returned from main memory. If a subsequent Write access to this same block occurs, then both the cache Modified and MMU Modified bits must be set active. Again, subsequent Writes do not affect the cache tags.

9.4.3. Write Back Cycles, Control Space Operations, and the MMU

Write Back cycles and Control Space operations (see below) require special mention regarding their use of the MMU. MMU translations are required during Write Back cycles, flushes, and Block Copy operations. During Write Back cycles and flushes, NO protection checking is performed; checking is done for Block Copy operations.

The MMU is never updated on Write Back cycles or flushes. On Block Copy (Read) operations, the MMU Accessed bit of the block read is updated if the block is not in the cache. On Block Copy (Write) operations, the Accessed and Modified bits are both updated.

9.5. Control Space Operations for the Sun-3 Cache

9.5.1. Control Space Operations: Overview

There are four Control Space operations for the Sun-3 cache. These are Read/Write Cache Tags; Read/Write Cache Data; Flush; and Block Copy.

The sizes of data transfers supported for cache Control Space operations are implementation dependent.

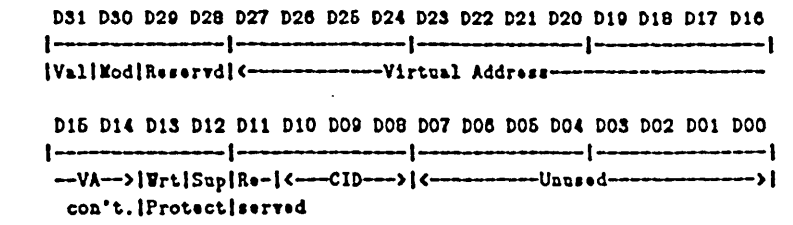
Detailed descriptions of the cache control operations are given below. The following notation is used: a data bit value of "d" indicates "Don't Care".

9.5.2. Read/Write Cache Tags

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
CACHE TAGS	0x8		LONG	R/W	A<16..4>*

* Address field for 128KB cache; A<13..4> for 16KB cache

A Read or Write command in Control Space.
At the addressed cache block, Read or Write the cache tags as
data in a 32 bit format.
Data fields:



Data format:
32 bit (logical) data, D<31..0>.
Unused bits D<7..0> are undefined.
Sizes of data transfers supported are implementation dependent.
Note: Writing all Valid bits with 0's initializes the cache.
Bus Error Conditions: None.
Interrupts Generated: None.

9.5.3. Read/Write Cache Data

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
CACHE DATA	0x9		LONG	R/W	A<16..2>*

* Address field for 128KB cache; A<13..2> for 16KB cache
Address note: A<3..2> address a Longword within a cache block.

A Read or Write command in Control Space.
Read or Write the cache data at the addressed cache block.
Cache tags are not checked for this operation.
Data format:
32 bit data, D<31..0>.
Sizes of data transfers supported are implementation dependent.
Bus Error Conditions: None.
Interrupts Generated: None.

9.5.4. Flush Cache Operations: Overview

Three Flush Cache operations are defined, depending on the match criteria: the context match flush, the page match flush, and the segment match flush. All operations operate over a set of cache blocks. A set is defined as one or more cache blocks, the number of blocks being a power of two and implementation dependent. The starting address for a set is specified in the command, with the assumption that lower order address bits are 0's. By incrementing the set address, a sequence of Flush Cache Set commands will flush the entire cache.

Each cache flush command may change the cache in two ways. First, each flush command causes all Valid and Modified blocks from the set which satisfy the flush match criteria to be written back to main memory. Second, each flush command causes all Valid blocks within the set which satisfy the match criteria to be invalidated.

The match criteria for flush commands are defined below.

9.5.5. Flush Cache Set [Context Match]

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
FLUSH CACHE SET (CONTEXT)	0xA	'01'	N/A	WRITE	CX<2..0>, A<16..4>**

** A<16..4> corresponds to 128KB cache (max) with 1 block Set (min)
Upper bound of A<16..4> varies with cache size;
Lower bound of A<16..4> varies with Set size for flush
Min. address field = CX<2..0> (with Set size = cache size)

A Write command in Control Space.
Operation summary: Flush from the cache all cache blocks within a specified context which are from User protected pages.
Flush match criteria: The Supervisor protection tag must be reset (User space), and the Context ID tags must = the Context ID register (3 bits).
Data format:
(D<1..0> = '01') identifies this flush command as a Context Match flush.
MMU notes: No MMU protection check is done, and no update is performed on MMU Accessed or Modified bits.
Error interrupt condition:
Write Back error if the address translation for a Valid and Modified cache block is invalid.
Notes:
A Context Flush is used to ensure cache addressing consistency whenever a new active context replaces an old context in the MMU. The Context Flush must be performed before the old context references are removed from the MMU, since the MMU is required to translate the cache blocks' virtual addresses.

9.5.6. Flush Cache Set [Page Match]

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
FLUSH CACHE SET (PAGE)	0xA	'10'	N/A	WRITE	CX<2..0>, A<27..13>, A<12..4>**

** A<12..4> corresponds to 1 block Set (min)
Lower bound of A<12..4> varies with Set size for flush
Min. address field = CX<2..0>, A<27..13> (with Set size >= 8KB)

A Write command in Control Space.
Operation summary: Flush from the cache all cache blocks within a specified page, regardless of the page protection.
Flush match criteria: Two criteria must be satisfied.
First, the cache block's virtual page address A<27..13> must match the access page address A<27..13>. One (or more) bits from the cache block's page address A<27..13> form part of the cache address; the others are tag bits. The boundary between the tag bits and the cache address field is implementation dependent. For example, in a 16KB cache, A<13> is part of the cache address, and A<27..14> are tags.
The second criterion is that either the cache block's Supervisor protection tag be active or that the cache block's Context ID field matches the Context ID register.
Data format:
(D<1..0> = '10') identifies this flush command as a Page Match flush.
MMU notes: No MMU protection check is done, and no update is performed on MMU Accessed or Modified bits.
Error interrupt condition:
Write Back error if the address translation for a Valid and Modified cache block is invalid.
Notes:
The Page Flush is used during page management to purge all references to a virtual page from the cache. It must be performed before the MMU is updated to remove the page, since the MMU is required to translate the cache blocks' virtual addresses.

9.5.7. Flush Cache Set [Segment Match]

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
FLUSH CACHE SET (SEGMENT)	0xA	'11'	N/A	WRITE	CX<2..0>, A<27..17>, A<16..4>**

** A<16..4> corresponds to 128KB cache (max) with 1 block Set (min)
Upper bound of A<16..4> varies with cache size;
Lower bound of A<16..4> varies with Set size for flush
Min. address field = CX<2..0>, A<27..17> (with Set size = cache)

A Write command in Control Space.
Operation summary: Flush from the cache all cache blocks within a specified segment, regardless of the page protection.
Flush match criteria: Two criteria must be satisfied.
First, the segment address A<27..17> from the cache block's virtual address tags must match the access segment address A<27..17>.
The second criterion is that either the cache block's Supervisor protection tag be active or that the cache block's Context ID field matches the Context ID register.
Data format:
(D<1..0> = '11') identifies this flush command as a Segment Match flush.
MMU notes: No MMU protection check is done, and no update is performed on MMU Accessed or Modified bits.
Error interrupt condition:
Write Back error if the address translation for a Valid and Modified cache block is invalid.
Notes:
The Segment Flush is used during page management to purge all references to a virtual segment from the cache. It is required whenever an active Page Map Entry Group (PMEG) must be replaced. It must be performed before the MMU is updated to remove the PMEG, since the MMU is required to translate the cache blocks' virtual addresses.

9.5.8. Block Copy: Introduction

Block Copy (Read) and Block Copy (Write) are commands which, executed in sequence, cause a block of data to be moved from one block address to another while maintaining cache data integrity and avoiding the displacement of any valid blocks from the cache. The source block for a Block Copy sequence may be either a cache block or main memory; the destination is always a block in main memory.

Block Copy (Read) and (Write) must maintain cache data consistency. For both Block Copy (Read) and (Write), the virtual address hit criteria are the same as those used in cache read and write accesses. An additional comparison is required for both Block Copy (Read) and (Write) if the command address "misses" the cache. In this case, the translated physical address for the command must be compared with the translated cache block address. If they match, the command address and cache block address are synonyms, and the command must be handled as a cache "hit".

In what follows, the term "hit" indicates either a virtual address "hit" or physical address match. If a Block Copy (Read) command "hits" the cache, the source block for the copy operation is taken from the cache; otherwise the source is in main memory. No protection violation can occur, since Block Copy (Read) has Supervisor read access. The MMU Accessed bit is updated if the Block Copy (Read) command "misses" the cache.

The destination of the Block Copy (Write) command is always main memory. If the Block Copy (Write) command "hits" the cache, then the cache block must be invalidated. A protection violation for the Write command will result if the page for the destination block prohibits writes. A protection violation prevents both writing the block and updating the MMU. Otherwise, both the MMU Accessed and Modified bits will be updated by a valid Block Copy (Write) command.

Certain restrictions apply to the instruction sequence which may be executed between a Block Copy (Read) and Block Copy (Write). First, if any changes are made to the source block following the execution of a Block Copy (Read) but before a Block Copy (Write), the results of those changes are unpredictable in the Block Copy (Write) block. Second, if any other Block Copy (Read) or (Write) commands are executed during the instruction sequence, the results of the original Block Copy commands are unpredictable.

Implementations of Block Copy commands may vary. Command formats are given below.

9.5.9. Block Copy [Read]

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
BLOCK COPY (READ)	0xB		N/A	READ	CX<2..0>, A<27..4>

A Read command in Control Space.
Operation: See above.
Data format: N/A.
Error interrupt condition: ECC error (implementation dependent).

9.5.10. Block Copy [Write]

REGISTER/MAP	A<31..28>	D<1..0>	SIZE	TYPE	ADDRESS FIELD
BLOCK COPY (WRITE)	0xB		N/A	WRITE	CX<2..0>, A<27..4>

A Write command in Control Space.
Operation: See above.
Data format: N/A.
Error interrupt condition: ECC error (implementation dependent).

10. The Sun-3 ECC Memory Architecture

10.1. The Sun-3 ECC Memory: Overview

ECC memory is the standard memory for Sun-3 systems with caches. However, the ECC memory architecture, as specified below, is not restricted to cache based systems alone.

ECC memory requires three control registers on each memory board mapped into the Device Space. One of these registers initializes the base address of the memory board; the base address of each board is restricted to be a multiple of the size of the board. (For example, the base address of an 8 MB board must be located on an 8 MB address boundary.) The second register captures the failing address and syndrome on single bit errors, and the third defines the mode and operation of the ECC chips.

The contents of these registers and how they are accessed are discussed in the following sections.

In addition, the Memory Error Control and Memory Error Address registers on the processor board control and record ECC error interrupts to the processor. See the section "Memory Error Registers" above for a discussion of these registers.

10.1.1. ECC Memory Operations

The following operations are supported by ECC Memory:

- Bus Memory Cycles: Read and write main memory in response to CPU, DVMA, or cache Write Back bus cycles.
- Register Access Cycles: Read and write memory control registers. All registers are addressable through a single Type 1 page in Device Space; see below.
- Refresh Scrub Cycles: If enabled, a data scrub will be performed during memory refresh.

10.1.2. Memory Error Conditions

There are two error conditions which may be detected in ECC memory: Correctable Errors (CE's) and Uncorrectable Errors (UE's). These may be detected during either memory access cycles or refresh scrub cycles.

How these errors are reported to the processor (or DVMA master) is discussed below.

10.2. Error Reporting for ECC Memory Systems

The following sections summarize first the controls to enable error reporting and second how the error reporting differs according to the type of memory cycle and error.

10.2.1. Enabling Error Checking and Reporting

A table listing the names and functions of enable bits for error reporting is given below.

Enable Bit/ Register/Bd	Applies on Memory Cycles	Function of Enable Bit
ECC ENABLE/ Mem Enab/Mem	Bus Memory cycles only	Enables Check Bit generation and both CE and UE reporting to Mem Error reg on CPU Bd
SCRUB ENABLE/ Mem Enab/Mem	Refresh Scrub cycles only	Enables Check Bit generation and ONLY CE reporting to Mem Error reg on CPU Bd (UE reporting is inhibited)
ENABLE CE/ Mem Error/CPU	All Bus Memory, Refresh Scrub	Enables CE bit in Mem Error Reg; CE is set if the Error Status bit (D0) in the Correctable Error Reg on ANY Mem Bd is active; CE bit is reset only when ALL Error Status bits are reset
(None)	Bus Memory cycles only	UE bit in Mem Error Reg is always enabled and is set on UE reported to CPU Bd
ENABLE INT/ Mem Error/CPU	All Bus Memory, Refresh Scrub	Enables interrupt to CPU if any error bit in the Memory Error Reg is active. The error bits, D0:D3, include the CE, UE, WBACKERR, and TIMEOUT bits. (See "Cache Error Conditions" in the Sun-3 Cache for WBACKERR, TIMEOUT.)

10.2.2. Reporting ECC Errors

Uncorretable and Correctable errors are reported to the CPU or DVMA Master as follows.

- On DVMA cycles, the ONLY error that is reported to the DVMA Master is an Uncorrectable error on the data requested during a DVMA Read cycle. This UE is reported to the Master as a bus error. Errors resulting from DVMA Write cycles or cache Write Back cycles resulting from DVMA are not reported to the DVMA Master.

- Correctable and Uncorrectable errors, if enabled, are reported to the processor as interrupts if they result from CPU, DVMA or Write Back bus cycles. Correctable errors resulting from refresh scrub cycles, if enabled, are also reported as interrupts. Uncorrectable errors resulting from refresh scrub cycles are not reported.

- Note that the Memory Error Address register on the processor board does NOT capture the failing address of Correctable errors. The Correctable Error Reg on each memory board saves this address. The Memory Address Error Reg saves the virtual addresses for Uncorrectable errors, Write Back errors, and those Timeout errors reported as interrupts.

10.3. Device Space Registers for ECC Memory

In this section, the registers defined in the Device Space (Type 1 access) for ECC control which are on memory boards are examined in detail. The Memory Error Control and Memory Error Address registers for ECC error reporting are discussed in the section "Memory Error Registers".

10.4. Addressing Registers on ECC Memory Boards

Each memory board on the Sirius bus contains three different Device Space registers. All of these registers are addressable through a single page in the Device Space.

Initialization: See the chapter on CPU Reset

TYPE	DEVICE ADDR (Decode A<20..17>)	DEVICE	PHYS. SPACE
1	0x001E0000	ECC Memory	A<7..0>

Within this page, address bits A7:A6 address the memory board number (set by jumpers). Address bits A3:A0 address each of the three registers, as shown below.

ADDR	REGISTER	DATA	TYPE
A<7..6>=Bd#			
A<3..0>=0x0	ECC MEM ENABLE	WORD	READ-WRITE
A<3..0>=0x4	CORRECTABLE ERR	WORD	READ-ONLY
A<3..0>=0x8	ECC CHIP DIAG	64 Bits	WRITE-ONLY

The ECC Chip Diagnostic register may only be written by word or longword writes on word (longword) boundaries (see below).

10.4.1. The ECC Memory Enable Register

ECC memory uses 28 bit physical addressing. The ECC Memory Enable register supports 28 bit physical addressing on the memory bus. The contents of this 16 bit register are shown below.

ECC MEMORY ENABLE REGISTER

BIT	NAME	TYPE	MEANING
D<5..0>	BASE ADDRESS	read-write	Base Address; compare with bus address A<27..22>
D<6>	BOARD ENABLE	read-write	Overall memory board enable
D<7>	BUS ECC ENAB	read-write	ECC generation/check enable on memory bus cycles
D<8>	SCRUB ENABLE	read-write	Enable refresh scrub cycle with Correctable Err report
D<10..9>	BOARD SIZE	read-only	Board size encoding: 00 = 4 MB 01 = 8 MB 10 = 16 MB 11 = 32 MB
D<11>	ENABLE DM0	read-write	Enable 2960 pin DM0
D<12>	ENABLE DM1	read-write	Enable 2960 pin DM1
D<15..13>	BOARD TYPE	read-only	Board type identifier

The register contents are used as follows:

- The Base Address (D<5..0>) for each memory board is compared with the memory bus address (A<27..22>) during each bus memory cycle. (Additional information, not identified in the architecture, is required to identify a bus cycle as a memory cycle.) If the Board Enable bit (D<6>) is active, then a memory bus cycle is addressed to this memory board if A<27..22> is greater than or equal to the Base Address and less than the Base plus the board size (4 MB to 32 MB, encoded in D<10..9>).

- The Board Enable bit is initialized to zero when reset, as indicated in the chapter on CPU Reset; it must be active to read or write the memory board.

- The Bus ECC Enable bit enables check bit generation and error reporting for all bus memory cycles. All single bit and uncorrectable errors from bus memory cycles are reported to the Memory Error Control register.

- The Scrub Enable bit enables a background data scrub during refresh cycles. It also enables reporting Correctable errors detected during the refresh scrub.

- The Board Size field is a Read Only field encoding the board size, 4 MB to 32 MB.

- Enable DM0 and Enable DM1 are used to control the mode of operation for the 2960 ECC chip by driving 2960 pins DM0 and DM1.

- The Board Type field can be used to identify particular versions of the Memory board. This field will be coded as 0's for the initial version of ECC memory.

10.4.2. The ECC Memory Enable Register: Initialization

The ECC memory architecture requires that memory Base Address registers be initialized with board address boundaries which are multiples of the board size. As examples, an 8 MB board must be initialized with bit D<0> = 0; a 16 MB board with D<1..0> = 00; and a 32 MB board with D<2..0> = 000. Memory logic comparing A<27..22> with the Base Address field (D<5..0>) may assume that lower order Base Address bits meet this restriction

10.4.3. The Correctable Error Register

The 32-bit Correctable Error Register captures the syndrome and physical address of the first single bit error on a memory board. It is reset on a write to the high order byte of the CE register.

CORRECTABLE ERROR REGISTER			
BIT	NAME	TYPE	MEANING
D<0>	ERROR STATUS	read-only	Active status = CE error
D<23..1>	CE ADDRESS	read-only	Real Address bits A<25..3> for first CE
D<31..24>	SYNDROME	read-only	Syndrome for first CE

The register contents are used as follows:

- The Error Status bit is set active when the first correctable error is detected and reset by a Write to the (read only) CE register. The address and syndrome are frozen while the bit is set.

- The CE Address field holds physical address bits A<25..3> of the first CE. Real address bits A<27..26> may be read from the BASE ADDRESS field (bits D<5..4>) in the ECC Memory Enable register for the board.

- The Syndrome field holds the syndrome of the first CE.

10.4.4. The ECC Chip Diagnostic Register

Any implementation of the Sun-3 ECC memory must include control registers to initialize and test the ECC generation and check logic. For the first implementation of ECC memory, the AMD 2960A ECC chips are selected for this logic. It is not intended that all implementations of Sun-3 ECC memory be restricted to use these AMD chips.

The first ECC memory utilizes four AMD 2960A ECC chips per memory board. A 16 bit register internal to each of these chips controls its initialization and testing. This register may be written but not read.

The ECC Chip Diagnostic Register is a logical 64 bit register consisting of the four 16 bit registers inside the 2960A chip. This register must be written as 16 bit words (or 32 bit longwords) on word (longword) boundaries.

See the AMD Data Book for a complete description.

11. References

AMD 7990 Ethernet Interface: AMD Order # 03378D

AMD 9513 Timer Technical Manual: AMD Order # 03402C.

AMD 8068 DCP Technical Manual: AMD Order # 04862A.

Intel 82586 EDLC Technical Manual: Intel LAN Manual

Intersil 7170 Data Sheet: Intersil 1985 Data Book

Motorola 68020 CPU Manual: Motorola MC68020UM(ADI).

Zilog 8530 SCC Technical Manual: Zilog Order # 00-2057-02.