KA46 System Board Specification

Order Number KA46-0-DBF

Revision/Update Information: X.02

John Kirk
MLO5-2/B6
223-4690
LEWS4::KIRK

10-OCT-1990 13:54:58.19

This document was prepared using VAX DOCUMENT, Version 1.2

Chapter 1
Introduction

This Engineering Specification defines the functional characteristics of the KA46 used in the VS4000-400 desktop workstation.

The KA46 is made up of the following major component parts

- DC595 Processor Chip
- DC598 Clock Chip
- DC596 Floating Point Accelerator Chip
- 256 Kbyte second level write-through cache memory
- Gate Array—DC7203
- Gate Array—DC7201
- 8-104 Megabyte RAM memory with parity checking
- 256 Kbyte ROM memory (socketed)
- 32-byte network address ROM (socketed)
- time-of-year clock
- four asynchronous serial ports (300 - 19,200 baud) for:
  - keyboard
  - mouse or tablet
  - printer
  - communications (full modem control on this port only)
- single asynchronous/synchronous SCSI port
- controller for single/multiple plane bitmap display
- controller for ThinWire or Transceiver Ethernet network
- controller for audio I/O

The KA46 carries connectors to allow the attachment of one of several bitmap modules for high performance monochrome or color graphics and for an optional synchronous communications adaptor.

Components that together with the KA46 make up the engineering workstation are : one of a selection of monochrome or color high resolution video monitors: an LK401 keyboard : a VSXXX-AA mouse or VSXXX-AB tablet : one or more storage devices.

The VS4000-400 is housed in a desktop enclosure which contains the KA46 system board, an H7819 power supply and the storage devices.

1.1 Applicable Documents

The following documents describe the principal LSI chips which are used on the KA46:

- DC595 Processor chip spec
- DC598 Clock Chip spec
- DC596 Floating Point Chip spec
- A-PS-2132064-0-0 DC7203 Cache control Gate Array spec.
- A-PS-21-34159-0-0 DC7201 Video/IO/Memory Gate Array spec.
- A-PS-21-21672-0-0 LANCE Ethernet Controller chip spec
- A-PS-21-31889-0-0 53C94 SCSI Controller Chip spec.
- A-PS-21-32756-0-0 79C30 Sound Generator Chip spec.
- A-PS-21-32423-0-0 MC146818BM Watch chip spec.
- A-PS-23-365A1-0-0 Ethernet Network Address ROM pattern spec.

Chapter 2
CENTRAL PROCESSOR

The system central processor is a single CMOS chip fabricated in Digital's CMOS-3 process. This chip is fully described in its specification document which is cited in Section 1.1. This section covers the manner in which the chip is used by the VS4000-400; it does not attempt to reproduce all the material in the chip specification.

2.1 CPU Chip

The central processor of the KA46 supports the MicroVAX Chip subset (plus six additional string instructions) of the VAX instruction set and datatypes and full VAX memory management. It supports both 30-bit and 32-bit physical addressing as defined in the VAX SRM. Physical address mode is selected by a Mode Bit in the Internal Processor Register ACCS - see Section 2.1.9. Only the 30-bit addressing mode is supported in the use of this chip in the VS4000-400. The CPU is implemented as a single VLSI chip, the DC595, internally called the MARIAH.

2.1.1 Addressing

In 30-bit mode, 0.5 GBytes of program space and 0.5 GBytes of I/O space are addressable. The actual physical address output by the CPU has address bits<31:29> = 000 for program space references and address bits<31:29> = 111 for I/O space references.

2.1.2 Processor State

The processor state consists of that portion of the state of a process which is stored in processor registers rather than in memory. The processor state is composed of sixteen General Purpose Registers (GPR's), the Processor Status Longword (PSL), and the Internal Processor Registers (IPRs).

Non-privileged software can access the GPR's and the Processor Status Word (bits <15:00> of the PSL). The IPRs and bits <31:16> of the PSL can only be accessed by privileged software. The IPRs are explicitly accessible only by the the Move To Processor Register (MTPR) and Move From Processor Register (MFPR) instructions which can be executed only while running in kernel mode.

2.1.2.1 General Purpose Registers

The CPU implements 16 General Purpose Registers per DEC STD 032. These registers are used for temporary storage, accumulators, and base and index registers for addressing. These registers are denoted R0 - R15. The bits of a register are numbered from the right <0> through <31>.

Figure 2–1: General Purpose Register

```
 3                                                             0
 1                                                             0
+--------------------------------------------------------------+
|                                                              |
+--------------------------------------------------------------+
```

Certain of these registers have been assigned special meaning by the VAX-11 architecture:

- R15 is the Program Counter (PC). The PC contains the address of the next instruction byte of the program.
- R14 is the Stack Pointer (SP). The SP contains the address of the top of the processor defined stack.
- R13 is the Frame Pointer (FP). The VAX-11 procedure call convention builds a data structure on the stack called a stack frame. The FP contains the address of the base of this data structure.
- R12 is the Argument Pointer (AP). The VAX-11 procedure call convention uses a data structure termed an argument list. The AP contains the address of the base of this data structure.

DEC STD 032 should be consulted for more information on the operation and use of these registers.

2.1.2.2 Processor Status Longword

The KA46 Processor Status Longword (PSL) is implemented per DEC STD 032, which should be consulted for a detailed description of the operation of this register. The PSL is saved on the stack when an exception or interrupt occurs and is saved in the Process Control Block (PCB) on a process context switch. Bits <15:00> may be accessed by non-privileged software, while bits <31:16> may only be accessed by privileged software. Processor initialization sets the PSL to 041F 0000 (hex).

Figure 2–2: Processor Status Longword

```
 3 3 2 2 2 2 2 2 2 2 2 2       1 1
 1 0 9 8 7 6 5 4 3 2 1 0       6 5             8 7 6 5 4 3 2 1 0
+-+-+-+-+-+-+-+---+---+-+---------+---------------+-+-+-+-+-+-+-+-+
| | | |M|F| |   |   |M|         |               | | | | | | | | |
|C|T|V|B|P|I|CUR|PRV|B|         |               |D|F|I| | | | | |
|M|P|M|Z|D|S|MOD|MOD|Z|   IPL   |      MBZ      |V|U|V|T|N|Z|V|C|
+-+-+---+-+-+---+---+-+---------+---------------+-+-+-+-+-+-+-+-+
```

PSL<31> (CM) Compatibility Mode. This bit always reads as ZERO, attempting to load a ONE into this bit has no effect.

Note

VAX Compatibility Mode instructions can be emulated by macrocode, but the emulation software runs in native mode, so the CM bit is never set.

PSL<30> (TP) Trace Pending.

PSL<29> (VM) Virtual Machine Mode

PSL<28> (MBZ) Must Be written as Zero.

PSL<27> (FPD) First Part Done.

PSL<26> (IS) Interrupt Stack.

PSL<25:24> (CUR) Current Mode.

PSL<23:22> (PRV) Previous Mode.

PSL<21> (MBZ) Must Be written as Zero.

PSL<20:16> (IPL) Interrupt Priority Level.

PSL<15:8> (MBZ) Must Be written as Zero.

PSL<7> (DV) Decimal Overflow Trap Enable. This read/write bit has no effect on the KA46 hardware; it can be used by macrocode which emulates VAX decimal instructions.

PSL<6> (FU) Floating Underflow Fault Enable.

PSL<5> (IV) Integer Overflow Trap Enable.

PSL<4> (T) Trace Trap Enable.

PSL<3> (N) Negative Condition Code.

PSL<2> (Z) Zero Condition Code.

PSL<1> (V) Overflow Condition Code.

PSL<0> (C) Carry Condition Code.

2.1.2.3 Internal Processor Registers

The CPU Internal Processor Registers (IPRs) can be accessed by using the MFPR and MTPR privileged instructions. Each IPR falls into one of the following five categories:

1. Implemented as specified in the VAX Architecture Standard (DEC STD 032).
2. CPU-specific implementation which is unique or different from the VAX Architecture Standard (DEC STD 032).
3. Not implemented by this CPU, read as ZERO, NOP on writes.
4. Not implemented by this CPU, access causes RESERVED OPERAND FAULT.
5. Not fully implemented by this CPU, access causes UNPREDICTABLE results.

A table listing each of the IPRs with its mnemonic, access type (read or write), scope (CPU-wide or per-process), which chip it is implemented in, whether it is initialized by hardware reset and its category number, appears below.

Table 2–1: KA46 Internal Processor Registers

		Number						
Register Name	**Mnemonic**	**Decimal**	**Hex**	**Type**	**Scope**	**Impl**	**Init**	**Category**
Kernel Stack Pointer	KSP	0	0	RW	PROC	DC595		1
Executive Stack Pointer	ESP	1	1	RW	PROC	DC595		1
Supervisor Stack Pointer	SSP	2	2	RW	PROC	DC595		1
User Stack Pointer	USP	3	3	RW	PROC	DC595		1
Interrupt Stack Pointer	ISP	4	4	RW	CPU	DC595		1
Reserved		5-7	5-7					3
P0 Base Register	P0BR	8	8	RW	PROC	DC595		1
P0 Length Register	P0LR	9	9	RW	PROC	DC595		1
P1 Base Register	P1BR	10	A	RW	PROC	DC595		1
P1 Length Register	P1LR	11	B	RW	PROC	DC595		1
System Base Register	SBR	12	C	RW	CPU	DC595		1
System Length Register	SLR	13	D	RW	CPU	DC595		1
Reserved		14-15	E-F					3
Process Control Block Base	PCBB	16	10	RW	PROC	DC595		1
System Control Block Base	SCBB	17	11	RW	CPU	DC595		1
Interrupt Priority Level	IPL	18	12	RW	CPU	DC595	Yes	1
AST Level	ASTLVL	19	13	RW	PROC	DC595	Yes	1

Table 2–1 (Cont.): KA46 Internal Processor Registers

		Number						
Register Name	**Mnemonic**	**Decimal**	**Hex**	**Type**	**Scope**	**Impl**	**Init**	**Category**
Software Interrupt Request Register	SIRR	20	14	W	CPU	DC595		1
Software Interrupt Summary Register	SISR	21	15	RW	CPU	DC595	Yes	1
Reserved		22-23	16-17					3
Interval Counter Control Status	ICCS	24	18	RW	CPU	DC595		2
Reserved		25-26	19-1A					3
Time of Year Register	TODR	27	1B	RW	CPU			1
Console Storage Receiver Status	CSRS	28	1C	RW	CPU			5
Console Storage Receiver Data	CSRD	29	1D	R	CPU			5
Console Storage Transmitter Status	CSTS	30	1E	RW	CPU			5
Console Storage Transmitter Data	CSTD	31	1F	W	CPU			5
Console Receiver Control/Status	RXCS	32	20	RW	CPU			2
Console Receiver Data Buffer	RXDB	33	21	R	CPU			2
Console Transr Control/Status	TXCS	34	22	RW	CPU			2
Console Transr Data Buffer	TXD	35	23	W				2
Reserved		36-37	24-25					3
Machine Check Error Register	MCESR	38	26	W	CPU	DC595		2
Reserved		39	27					3
Accelerator Control and Status Register	ACCS	40	28	RW	CPU	DC595	Yes	2
Reserved		41	29					3
Console Saved PC	SAVPC	42	2A	R	CPU	DC595		2
Console Saved PSL	SAVPSL	43	2B	R	CPU	DC595		2
Reserved		44-46	2C-2E					3
Translation Buffer Tag	TBTAG	47	2F	W	CPU	DC595		2
Reserved		48-54	30-36					3
I/O System Reset Register	IORESET	55	37	W	CPU			2

Table 2–1 (Cont.): KA46 Internal Processor Registers

		Number						
Register Name	**Mnemonic**	**Decimal**	**Hex**	**Type**	**Scope**	**Impl**	**Init**	**Category**
Memory Management Enable	MAPEN	56	38	RW	CPU	DC595	Yes	1
Translation Buffer Invalidate All	TBIA	57	39	W	CPU	DC595		1
Translation Buffer Invalidate Single	TBIS	58	3A	W	CPU	DC595		1
Translation Buffer Data	TBDATA	59	3B	W	CPU	DC595		2
Reserved		60-61	3C-3D					3
System Identification	SID	62	3E	R	CPU	DC595		1
Translation Buffer Check	TBCHK	63	3F	W	CPU	DC595		1
Reserved		64-122	40-7A					3
Vector Interface Error Status Register	VINTSR	123	7B	RW	CPU			2
Primary Cache Tag Store	PCTAG	124	7C	RW	CPU	DC595		2
Primary Cache Index Register	PCIDX	125	7D	RW	CPU	DC595		2
Primary Cache Error Address Register	PCERR	126	7E	RW	CPU	DC595		2
Primary Cache Status Register	PCSTS	127	7F	RW	CPU	DC595	Yes	2
Reserved		128-255	80-FF					3
Reserved		>255	>FF					4

Table Headings Meaning

Decimal—Decimal number of the Processor Register
Hex —Hex Number of the Processor Register
Type —Read or Write Access

R —Read-Only Register
W —Write-Only Register
RW—Read-Write Register

Scope —Processor Register's Scope

CPU —CPU-Wide Register
PROC—Per-Process Register

Impl —Chip in which the Processor Register is Implemented

DC595 —Implemented in the CPU chip

Init —Initialized on Module RESET (Power-up, Negation of DCOK)
Category—Processor Register Category as previously defined

2.1.2.3.1 KA46 VAX Standard Internal Processor Registers

Internal Processor Registers (IPRs) which are implemented per DEC STD 032 are classified as Category One IPRs. DEC STD 032 should be consulted for details on the operation and use of these registers.

The following category one registers are also referenced in other sections of this specification:

Table 2–2: Category One Internal Processor Registers

Number			
Decimal	**Hex**	**Register Name**	**Mnemonic**
18	12	Interrupt Priority Level	IPL
20	14	Software Interrupt Request	SIRR
21	15	Software Interrupt Summary	SISR
27	1B	Time Of Year Clock	TODR
56	38	Memory Management Enable	MAPEN
57	39	Translation Buffer Invalidate All	TBIA
58	3A	Translation Buffer Invalidate Single	TBIS
62	3E	System Identification	SID
63	3F	Translation Buffer Check	TBCHK

2.1.2.3.2 KA46 Unique Internal Processor Registers

Internal Processor Registers (IPRs) which are implemented uniquely on the KA46 (i.e., they are not contained in, or do not fully conform to, DEC STD 032) are classified as Category Two IPRs and are described in detail in this specification. Refer to the following sections for a description of these registers:

Table 2–3: Category Two Internal Processor Registers

Number			
Decimal	**Hex**	**Register Name**	**Mnemonic**
24	18	Interval Clock Control/Status	ICCS
32	20	Console Receiver Control/Status	RXCS
33	21	Console Receiver Data Buffer	RXDB
34	22	Console Transmit Control/Status	TXCS
35	23	Console Transmit Data Buffer	TXDB
38	26	Machine Check Error Register	MCESR
40	28	Accelerator Control and Status	ACCS
42	2A	Console Saved PC	SAVPC
43	2B	Console Saved PSL	SAVPSL
47	2F	Translation Buffer Tag	TBTAG
55	37	I/O System Reset Register	IORESET
59	3B	Translation Buffer Data	TBDATA
123	7B	Vector Interface Error Status Reg	VINTSR
124	7C	Primary Cache Tag Store	PCTAG
125	7D	Primary Cache Index Register	PCIDX
126	7E	Primary Cache Error Address Register	PCERR
127	7F	Primary Cache Status Register	PCSTS

2.1.3 Process Structure

A process is a single thread of execution. The context of the current process is contained in the Process Control Block (PCB) which is pointed to by the Process Control Block Base Register (PCBB). The KA46 implements these structures as defined in DEC STD 032, which should be referenced for a description of the PCB and the PCBB.

2.1.4 Data Types

The KA46 CPU supports the following subset of the VAX data types:

1. Byte
2. Word
3. Longword
4. Quadword

5. Character string
6. Variable length bit field
7. Absolute queues
8. Self-relative queues
9. Character string
10. F-floating
11. G-floating
12. D-floating

Support for the remaining VAX data types can be provided via macrocode emulation.

2.1.5 Instruction Set

The CPU implements the following subset of the VAX instruction set types in microcode.

1. Integer arithmetic and logical
2. Address
3. Variable length bit field
4. Control
5. Procedure call
6. Miscellaneous
7. Queue
8. Character string (MOVC3, MOVC5, CMPC3, CMPC5, LOCC, SCANC, SKPC, SPANC)
9. Operating system support
10. F_floating
11. G_floating
12. D_floating

The CPU chip provides special microcode assistance to aid the macrocode emulation of the following instruction groups:

1. Character string (except MOVC3, MOVC5, CMPC3, CMPC5, LOCC, SCANC, SKPC, SPANC)
2. Decimal string
3. CRC
4. EDITPC

The following instruction groups are not implemented, but may be emulated by macrocode:

1. Octaword
2. Compatibility mode instructions

2.1.6 Memory Management

The CPU implements full VAX Memory Management as defined in DEC STD 032. System space addresses are virtually mapped through single-level page tables, and process space addresses are virtually mapped through two-level page tables. See DEC STD 032 for descriptions of the virtual to physical address translation process, and the format for VAX Page Table Entries (PTEs).

2.1.6.1 Translation Buffer

To reduce the overhead associated with translating virtual addresses to physical addresses, the CPU employs a 64-entry, fully associative translation buffer for caching VAX PTE's. Each entry can store a PTE for translating virtual addresses in either the VAX Process or System Space. The translation buffer is flushed whenever memory management is enabled or disabled (i.e., by writes to IPR 56), any page table base or length registers are modified (i.e. by writes to IPRs 13:8) and by writing to IPR 57 (TBIA) or IPR 58 (TBIS).

Each entry is divided into two parts: a 24-bit Tag Register and a 27-bit PTE Register. The Tag Register is used to store the Virtual Page Number (VPN) of the virtual page that the corresponding PTE Register maps, and a valid bit (TB.V) that indicates that the tag contains a valid VPN. The PTE register stores the 21-bit PFN field, the PTE.V bit, the PTE.M bit and the 4-bit PROT field from the corresponding VAX PTE.

During virtual-to-physical address translation, the contents of the 64 Tag Registers are compared with the Virtual Page Number Field (bits <31:9>) of the virtual address of the reference. If there is a match with one of the Tag Registers and the TB.V bit indicates the entry is valid, then a translation buffer "hit" has occurred and the contents of the corresponding PTE register are used for the translation.

If there is no match, the translation buffer does not contain the necessary VAX PTE information to translate the address of the reference and the PTE must be fetched from memory. Upon fetching the PTE, the translation buffer is updated by replacing the entry that is selected by the replacement pointer. Since this pointer is moved to the next sequential translation buffer entry whenever it is pointing to an entry that is accessed, the replacement algorithm is Not Last Used (NLU). This pointer is called the NLU pointer.

2.1.6.2 Memory Management Control Registers

There are four IPRs that control the Memory Management Unit (MMU):

IPR 56 - MAPEN
IPR 57 - TBIA
IPR 58 - TBIS
IPR 63 - TBCHK

Memory management can be enabled/disabled via IPR 56 (MAPEN). Writing ZERO bit<0> of MAPEN with a MTPR instruction disables memory management, writing a ONE to bit<0> of this register with a MTPR instruction enables memory management. Writes to this register flush the translation buffer. To determine whether or not memory management is enabled, IPR 56 is read using the MFPR instruction. Translation buffer entries that map a particular virtual address can be invalidated by writing the virtual address to IPR 58 (TBIS) using the MTPR instruction.

Note

Whenever software changes a valid Page Table Entry for the system or current process region, or a System Page Table Entry that maps any part of the current process page table, all process pages mapped by the Page Table Entry must be invalidated in the translation buffer.

The entire translation buffer can be invalidated by writing a ZERO to IPR 57 (TBIA) using the MTPR instruction.

The translation buffer can be checked to see if it contains a valid translation for a particular virtual page by writing a virtual address within that page to IPR 63 (TBCHK) using the MTPR instruction. If the translation buffer contains a valid translation for the page, the condition code V bit (bit<1> of the PSL) is set.

Note

The TBIS, TBIA, and TBCHK IPRs are write only. The operation of a MFPR from any of these register is UNDEFINED.

There are three pairs of base and length registers which specify the base and length of P0, P1, and S0 space:

IPR 8 - P0BR
IPR 9 - P0LR
IPR 10 - P1BR
IPR 11 - P1LR
IPR 12 - SBR
IPR 13 - SLR

The base and length of the P0, P1, and S0 page tables may be changed by writing the appropriate address or length to IPR 8 (P0BR), IPR 9 (P0LR), IPR 10 (P1BR), IPR 11 (P1LR), IPR 12 (SBR), and IPR 13 (SLR).

Note

Whenever the location or size of the system map is changed by changing the SBR (IPR 12) or SLR (IPR 13), the entire translation buffer must be cleared. The CPU accomplishes this by flushing the TB on any change to SBR, SLR, or to P0BR, P1BR, P0LR, and P1LR.

When a process context is loaded with LDPCTX, all TB entries that map process-space pages are automatically cleared. System-space mappings are preserved.

There are two IPRs which are used by diagnostic software to test the translation buffer: IPR 47 (TBTAG) and IPR 59 (TBDATA).

Figure 2–3: Translation Buffer Tag (TBTAG) - (IPR 47)

```
 3                                         0 0               0
 1                                         9 8               0
+-------------------------------------------+-----------------+
|      VIRTUAL PAGE NUMBER (Write Only)     |       MBZ       |:TBTAG
+-------------------------------------------+-----------------+
```

Figure 2–4: Translation Buffer Data (TBDATA) - (IPR 59)

```
 3 3     2 2 2       2 2                                    0
 1 0     7 6 5       1 0                                    0
+-+-------+-+---------+---------------------------------------+
| |       | |   MBZ   |         PTE.PFN (Write Only)          |:TBDATA
+-+-------+-+---------+---------------------------------------+
 |    |    |
 |    |    |
 |    |    +-- PTE.M    (Write Only)
 |    +------- PTE.PROT (Write Only)
 +------------ PTE.V    (Write Only)
```

Diagnostic software may use IPR 47 (TBTAG) and IPR 59 (TBDATA) to test the operation of the TB. A write to TBTAG writes bits <31:9> of the source data into the VPN field of the current tag location and clears the TB.V bit. A subsequent write to TBDATA interprets the source data as a PTE and writes PTE.V, PTE.M, PTE.PROT, and PTE.PFN into the current PTE location, sets the TB.V bit, and increments the NLU pointer.

These registers are provided for diagnostic purposes only and should not be written during normal operation. Writes to these registers must be done under very controlled conditions to achieve the desired results. Specifically, the following restrictions apply:

- The NLU pointer must be in a known state. A TBIA will initialize it to the first location in the array.
- Memory management must be enabled during the use of TBTAG and TBDATA, because writing to MAPEN implicitly does a TBIA and resets the NLU pointer.
- Data- and instruction-stream references during the use of TBTAG and TBDATA must not be allowed to change the NLU pointer.

Note

The TBIS, TBIA, TBCHK, TBTAG, and TBDATA IPRs are write only. An MFPR used to read any of these registers will cause the CPU to initiate a reserved operand fault.

2.1.7 Interrupts And Exceptions

Both interrupts and exceptions divert execution from the normal flow of control. An interrupt is caused by some activity outside the current process and typically transfers control outside the process *e.g.*, an interrupt from an external hardware device, while an exception is caused by the execution of the current instruction and is typically handled by the current process *e.g.*, an arithmetic overflow.

2.1.7.1 Interrupts

Interrupts can be divided into two classes: non-maskable and maskable.

Non-maskable interrupts cause a halt via the hardware halt procedure which saves the PC, PSL, MAPEN<0> and a halt code in IPRs, raises the processor IPL to 1F and then passes control to the resident firmware. The firmware dispatches the interrupt to the appropriate service routine based on the halt code and hardware event indicators. Non-maskable interrupts cannot be blocked by raising the processor IPL, but can be blocked by running out of the Halt Protected Address Space (except those non-maskable interrupts that generate a halt code of 3). Non-maskable interrupts with a halt code of 3 cannot be blocked since this halt code is generated after a hardware reset).

Maskable interrupts cause the PC and PSL to be saved, the processor IPL to be raised to the priority level of the interrupt (except for Mass Storage and Network Interface interrupts where the processor IPL is set to 14 independent of the level at which the interrupt was received) and the interrupt to be dispatched to the appropriate service routine through the SCB.

The various interrupt conditions for the KA46 are listed below along with their associated priority levels and SCB offsets.

Table 2–4: Interrupt Priority Levels

Priority Level	Interrupt Condition	SCB Offset
	BREAK Generated by console device	*
1F:1E	Unused	
1D	Second Level Cache Tag Parity Error	60
1C:17	Unused	
16	Interval Timer Interrupt	C0
15:14	Refer to Chapter 6	

The interrupt system is controlled by three IPRs: IPR 18, the Interrupt Priority Level Register (IPLR), IPR 20, the Software Interrupt Request Register (SIRR), and IPR 21, the Software Interrupt Summary Register (SISR). The IPLR is used for loading the processor priority field in the PSL (bits<20:16>). The SIRR is used for creating software interrupt requests. The SISR records pending software interrupt requests at levels 1 through 15. The format of these registers is presented below, refer to DEC STD 032 for more information on these registers.

Figure 2–5: Interrupt Priority Level Register (IPLR) - (IPR 18(DEC) 12(HEX))

```
 3                                            0 0         0
 1                                            5 4         0
+--------------------------------------------+----------+
|              Ignored, Returns 0            |PSL<20:16>| :IPLR
+--------------------------------------------+----------+
```

Figure 2–6: Software Interrupt Request Register (SIRR) - (IPL 20(DEC) 14(HEX))

```
 3                                                0 0     0
 1                                                4 3     0
+------------------------------------------------+-------+
|                      Ignored                   |Request| :SIRR
+------------------------------------------------+-------+
```

Figure 2–7: Software Interrupt Summary Register (SISR) - (IPL 21(DEC) 15(HEX))

```
 3                         1 1                            0
 1                         6 5                            0
+---------------------------+---------------------------+-+
|                           | Pending Software Interrupts |M|
|                           |                           |B| :SISR
|                           |F E D C B A 9 8 7 6 5 4 3 2 1|Z|
+---------------------------+---------------------------+-+
```

2.1.7.2 Exceptions

Exceptions can be divided into 3 types: trap, fault and abort.

A trap is an exception that occurs at the end of the instruction that caused the exception. After an instruction traps, the PC saved on the stack is the address of the next instruction that would have normally been executed and the instruction can be restarted.

A fault is an exception that occurs during an instruction, and that leaves the registers and memory in a consistent state such that the elimination of the fault condition and restarting the instruction will give correct results. After an instruction faults, the PC saved on the stack points to the instruction that faulted.

An abort is an exception the occurs during an instruction, leaving the value of registers and memory UNPREDICTABLE, such that the instruction cannot necessarily be correctly restarted, completed, simulated or undone. After an instruction aborts, the the PC saved on the stack points to the instruction that was aborted, (which may or may not be the instruction that caused the abort) and the instruction may or may not be restarted depending on the class of the exception and the contents of the parameters that were saved.

Exceptions can also be divided into six classes. A list of exceptions, grouped by class, is given in the table below. Exceptions save the PC and PSL and in some cases, one or more parameters, on the stack. Most exceptions do not change the IPL of the processor (except the exceptions in serious system failures class, which set the processor IPL to 1F) and cause the exception to be dispatched to the appropriate service routine through the SCB (except for the Interrupt stack not valid exception, and exceptions that occur while an interrupt or another exception are being serviced, which cause the exception to be dispatched to the appropriate service routine by the resident firmware). The exceptions listed below (except machine check) are described in greater detail in DEC STD 032.

Table 2–5: Exception Classes

Exception Class	Type	SCB Offset
Arithmetic Exceptions		
Integer overflow	trap	34
Integer divide by zero	trap	34
Subscript range	trap	34
Floating overflow	fault	34
Floating divide by zero	fault	34
Floating underflow	fault	34
Memory Management Exceptions		
Access control violation	fault	20
Translation not valid	fault	24
Operand Reference Exceptions		
Reserved addressing mode	fault	1C
Reserved operand fault or abort		18
Instruction Execution Exceptions		
Reserved/Privileged instr.	fault	10
Emulated instruction	fault	C8,CC
Change mode	trap	40-4C
Breakpoint	fault	2C
Tracing Exception		
Trace	fault	28
System Failure Exceptions		
Console Error Halt	abort	*

* —Dispatched by ROM resident firmware rather than through the SCB

Table 2–5 (Cont.): Exception Classes

Exception Class	Type	SCB Offset
Interrupt stack not valid	abort	*
Kernel stack not valid	abort	08
Machine check including: • DAL Read or Write Error • CP_Bus Parity Errors • Primary Cache Read Error • CP_Bus Timeout Errors • Main memory NXM Errors • Main Memory Uncorrectable Errors • FPU Chip Status Error • Translation Buffer Status Error • Internally Detected Inconsistency	abort	04

* —Dispatched by ROM resident firmware rather than through the SCB

2.1.7.3 Information Saved On A Machine Check Exception

In response to a machine check exception the PSL, PC, eight parameters, and a Byte Count are pushed on the stack as shown in the diagram below:

Figure 2–8: Information Saved On A Machine Check Exception

```
 3                             1 1
 1                             6 5                             0
+---------------------------------------------------------------+
|                Parameter Byte Count  (18 Hex)                 |  :SP
+-+-----------------------------+-------------------------------+
|R|                             |      Machine Check Code       |
+-+-----------------------------+-------------------------------+
|                   Contents of VA Register                     |
+---------------------------------------------------------------+
|                  Contents of VIBA Register                    |
+-------------------------------+-------------------------------+
| ICCS Register bit<6> Contents |        SISR <15:0> |
+-------------------------------+-------------------------------+
|                  Internal State Information                   |
+---------------------------------------------------------------+
|         Contents of the Shift Count (SC) Register             |
+---------------------------------------------------------------+
|            Contents of the Program Counter (PC)               |
+---------------------------------------------------------------+
|        Contents of the Process Status Longword (PSL)          |
+---------------------------------------------------------------+
```

The meaning of this information and how it effects the recovery procedure is described in the following sections.

2.1.7.3.1 Byte Count

Byte Count<31:0> - 24. This value indicates the number of bytes of information that follow on the stack (not including the PC and PSL).

2.1.7.3.2 "R" - Vax Restart Bit

Bit <31> of the Longword at (SP)+4 after a Machine Check is the Vax Restart Bit (R). If R is ONE , no state has been changed by the instruction which was executing when the error was detected. If R is ZERO, state has been changed by the instruction.

2.1.7.3.3 Machine Check Code Parameter

Bits <15:0> of the Longword at (SP)+4 after a Machine Check contents the Machine Check Parameter Code. This code value indicates the type of Machine check that occurred. A list of the possible machine check codes (in hex) and their associated causes follows:

Floating Point Errors - These codes indicate that the Floating Point Accelerator (FPA or CPU) chip detected an error while communicating with the CPU/FPA chip during the execution of a floating point instruction. The most likely cause(s) of these types of machine checks are: a problem internal to the CPU chip, a problem internal to the FPA or a problem with the interconnect between the two chips. Machine checks due to floating point errors MAY BE RECOVERABLE, depending on the state of the VAX RESTART bit (captured in bit tag(31) of the Longword at (SP)+4, the FIRST PART DONE flag (captured in PSL <27>). The error is recoverable only if the VAX RESTART bit is set and the FIRST PART DONE flag is cleared. Otherwise, the error is unrecoverable and depending on the current mode, either the current process or the operating system should be terminated or disable the FPA. The information pushed on the stack by this type of machine check is from the instruction that caused the machine check.

Table 2–6: Floating Point Error Machine Checks

Hex Code	Error Description
01	A protocol error was detected by the FPA chip during an F-Chip operand/result transfer.
02	An illegal opcode was detected by the FPA chip.
03	The FPA chip detected an operand parity error.
04	Unknown status was returned by the FPA chip.
05	The returned FPA chip result had an parity error.

Memory Management Errors - These codes indicate that CPU microcode detected an impossible situation (as described below) while performing functions associated with memory management. The most likely cause of this type of a machine check is a problem internal to the CPU. Machine checks due to memory management errors *may be recoverable*. Depending on the current mode, either the current process or the operating system should be terminated. The state of the P0BR, P0LR, P1BR, P1LR, SBR and SLR should be logged.

Table 2–7: Memory Management Error Machine Checks

Hex Code	Error Description
08	Memory Management Error occurred while the CPU was handling an Access Control Violation/Translation Not-Valid fault. The Read/Write that caused the error **missed** the translation buffer. This error is recoverable if the VAX **restart** bit or the **first part done** flag is set.
09	Memory Management Error occurred while the CPU was handling an Access Control Violation/Translation Not-Valid fault. The Read/Write that caused the error hit the translation buffer. This error is recoverable if the VAX **restart** bit or the **first part done** flag is set. The fact that the errant reference hit the the Translation Buffer means that the CPU is the most likely cause of the error.

Interrupt Errors - This code indicates that the interrupt controller within the CPU requested a hardware interrupt at an unused hardware IPL. The most likely cause of this type of a machine check is a problem internal to the CPU chip. Machine checks due to unused IPL errors may be recoverable. A non-vectored interrupt generated by a serious error condition (memory error, power fail or processor halt) has probably been lost. The operating system should be terminated.

Table 2–8: Interrupt Error Machine Checks

Hex Code	Error Description
0A	A hardware interrupt was requested at an unused Interrupt Priority Level (IPL). This error is RECOVERABLE if the VAX RESTART bit or the FIRST PART DONE flag is set.

Microcode Errors - This code indicates that an impossible situation was detected by the microcode during instruction execution. Note that most erroneous branches in the CPU microcode will cause random microinstructions to be executed. The most likely cause of this type of machine check is a problem internal to the CPU chip. Machine checks due to microcode errors may be RECOVERABLE. Depending on the current mode, either the current process or the operating system should be terminated.

Table 2–9: Microcode Error Machine Checks

Hex Code	Error Description
0B	An impossible state (ie. an undefined state bit combination in the microsequencer) was detected during a MOVC3 or MOVC5 instruction (not move forward, move backward, or fill). This error is RECOVERABLE if the FIRST PART DONE flag is set.
0C	An undefined TRAP code was produced by the CPU I-BOX. This error is recoverable if the VAX **restart** bit is set, **first part done** is cleared.
0D	An undefined Control Store address was reached by the microsequencer. This error is recoverable if either the VAX **restart** bit or **first part done** is set.

Read Errors - These codes indicate that an error was detected while the CPU was attempting to read from either the primary cache, the backup cache, or main memory. The most likely cause of this type of machine check must be determined from the state of the PCSTS, PCERR, BCERR

DSER, MEMCSR32, MEMCSR33 and MEMCSR34. Machine checks due to read errors MAY BE RECOVERABLE, depending on the state of the VAX RESTART flag, the FIRST PART DONE flag and the PCSTS<TRAP2> double error bit. If either the FIRST PART DONE flag or VAX RESTART is set, and PCSTS<TRAP2> are cleared then the error is recoverable. Otherwise, the error is unrecoverable and depending on the current mode, either the current process or the operating system should be terminated. The information pushed on the stack by this type of machine check is from the instruction that caused the machine check.

Table 2–10: Read Error Machine Checks

Hex Code	Error Description
10	A Primary Cache tag or data parity error occurred during a read.
11	A DAL bus or data parity error occurred during a read.

Write Errors - These codes indicate that an error was detected while the CPU was attempting to write to either the primary cache, the backup cache, or main memory. The most likely cause of this type of machine check must be determined from the state of the PCSTS, PCERR, BCERR DSER, MEMCSR32, MEMCSR33, MEMCSR34 and CBTCR. Machine checks due to write errors are most likely NON-RECOVERABLE because the CPU is capable of performing many read operations out of the primary cache before a write operation completes. For this reason, the information that is pushed onto the stack by this type of machine check cannot be guaranteed to be from the instruction that caused the machine check.

Table 2–11: Write Error Machine Checks

Hex Code	Error Description
12	A DAL bus error (ie. ERR L terminated cycle) occurred on a write or clear write buffer.

DAL Bus Errors - This codes indicate that the CPU detected that the DAL bus was in an UNDEFINED state. This Machine check is **non-recoverable**.

Table 2–12: DAL Bus Error Machine Check

Hex Code	Error Description
13	An Undefined DAL bus state was detected by the CPU.

2.1.7.3.4 Contents of the CPU's internal VA Register

After a Machine Check the location at (SP)+8 captures the contents of the CPU's VA Register at the time of the machine check. After a machine check of 10 or 11 this field represents the virtual address of the memory location that was being read when the error occurred. On machine checks of 12 this field represents the virtual address of a location that was being referenced either when the error occurred, or sometime after the error occurred. In other words, if the machine check occurred on a write operation, the contents of this field cannot be used for error recovery.

2.1.7.3.5 Contents of the CPU's internal VIBA Register

After a Machine Check the location at (SP)+12 captures the contents of the CPU's VIBA Register at the time of the machine check. After a machine check this field represents the virtual address of the last I-Stream fetch plus four.

2.1.7.3.6 ICCS Register Bit<6> Contents

After a Machine Check the location at (SP)+16 Bit<22> captures the contents of the CPU's ICCS (Interval Clock Control and Status) Register Bit<6> the Interrupt Enable (IE) at the time of the machine check.

2.1.7.3.7 SISR Register Bits<15:0> Contents

After a Machine Check the location at (SP)+16 Bits<15:0> captures the contents of the CPU's SISR (Software Interrupt Summary Register) Bits<15:0> at the time of the machine check.

2.1.7.3.8 Internal State Information

The Internal State Information Field is divided into five sub-fields. The contents of these sub-fields are described below:

Table 2–13: Internal State Information Field Description

Bits	Description
<31:24>	Delta PC (PC - backup PC)
<20:18>	AT - The Access Type at Machine Check time where: <000> —Read Access <001> —Write Access <010> —Modify Access <101> —Address Access <110> —Variable Bit Access <111> —Branch Access
<17:16>	DL - The Data Length at Machine Check time where: <00> —Byte Long <01> —Word Long <10> —Longword Long <11> —Quadword Long
<15:8>	Opcode - This field captures the opcode of the instruction that was being read or executed at the time of the machine check.
<3:0>	RN - This field captures the Register Number of the register that was the destination of the instruction that was being executed at the time of the machine check.

2.1.7.3.9 Contents of the Shift Count (SC) Register

After a Machine Check the location at (SP)+24 captures the contents of the CPU's Shift Count (SC) Register at the time of the machine check. The CPU uses this register is different ways dependent the instruction being executed. For a more complete description of the SC register please refer to the CPU chip Specification.

2.1.7.3.10 Contents of the Program Counter (PC)

PC<31:0> Captures the virtual address of the start of the instruction being executed at the time of the machine check.

2.1.7.3.11 Contents of the Process Status Longword (PSL)

PSL<31:0> Captures the contents of the PSL at the time of the machine check.

Note

Machine Checks must be acknowledged by the software by writing a ZERO to the MCESR (IPR 38)

2.1.7.4 Machine Check Error Register (MCESR) IPR 38

The Machine Check Error Register (IPR 38, MCESR) provides the mechanism by which software acknowledges receipt of a machine check. MCESR is a write-only register, and has the following format:

Figure 2–9: Machine Check Error Register (MCESR) - (IPR 38(DEC) 26(HEX))

```
 3
 1                                                        0
+----------------------------------------------------------+
|                         Write Only                        |
+----------------------------------------------------------+
```

When the CPU microcode invokes the software machine check handler, it sets a *machine check in progress* flag. If a machine check or memory management exception occurs while this flag is set, the microcode initiates a console double error halt.

Software should clear the *machine check in progress* flag as soon as possible in the machine check handler by writing a ZERO to IPR MCESR. Doing so re-enables normal machine check and memory management exception reporting.

2.1.7.5 System Control Block (SCB)

The System Control Block (SCB) consists of two pages in main memory that contain the vectors by which interrupts and exceptions are dispatched to the appropriate service routines. The SCB is pointed to by IPR 17, the System Control Block Base Register (SCBB).

Figure 2–10: System Control Block Base Register (SCBB) - (IPR 17(DEC) 11(HEX))

```
 3 3 2                                   0 0               0
 1 0 9                                   9 8               0
+---+---------------------------------------+-----------------+
|MBZ|   Physical Longword Address of SCB    |       MBZ       |
+---+---------------------------------------+-----------------+
```

Table 2–14: The System Control Block format

SCB Offset	Interrupt/Exception Name	Type	# Params	Notes
00	Passive Release	Interrupt	0	IPL is raised to request IPL
04	Machine Check	Abort	6	Parameters reflect machine state
08	Kernel Stack Not Valid	Abort	0	Must be serviced on interrupt stack
0C	Power Fail	Interrupt	0	IPL is raised to 1E
10	Reserved/Privileged Instruction	Fault	0	
14	Customer Reserved Instruction	Fault	0	XFC instruction
18	Reserved Operand	Fault/Abort	0	Not always recoverable
1C	Reserved Addressing Mode	Fault	0	
20	Access Control Violation/Vector Alignment Fault	Fault	2	Parameters are virtual address, status code
24	Translation Not Valid	Fault		2 Parameters are virtual address, status code
28	Trace Pending (TP)	Fault	0	

Table 2–14 (Cont.): The System Control Block format

SCB Offset	Interrupt/Exception Name	Type	# Params	Notes
2C	Breakpoint Instruction	Fault	0	
30	Unused	-	-	Compatibility mode in other VAXes
34	Arithmetic	Trap/Fault	1	Parameter is type code
38:3C	Unused	-	-	
40	CHMK	Trap	1	Parameter is sign-extended operand word
44	CHME	Trap	1	Parameter is sign-extended operand word
48	CHMS	Trap	1	Parameter is sign-extended operand word
4C	CHMU	Trap	1	Parameter is sign-extended operand word
50	Unused	-	-	
54	Memory Soft Error Notification		IPL is 1A	
58-5C	Unused	-	-	
60	Memory Hard Error Notification	Interrupt	0	IPL is 1D
64:80	Unused	-	-	
84	Software Level 1	Interrupt	0	
88	Software Level 2	Interrupt	0	Ordinarily used for AST delivery
8C	Software Level 3	Interrupt	0	Ordinarily used for process scheduling
90:BC	Software Levels 4-15	Interrupt	0	
C0	Interval Timer	Interrupt	0	IPL is 16
C4	Unused	-	-	

Table 2–14 (Cont.): The System Control Block format

SCB Offset	Interrupt/Exception Name	Type	# Params	Notes
C8	Emulation Start	Fault	10	Same mode exception, FPD = 0; parameters are opcode, PC, specifiers
CC	Emulation Continue	Fault	0	Same mode exception, FPD = 1: no parameters
D0:FC				Unused

The SCBB vector index is determined from bits <15:2> of the value supplied by external hardware. The new PSL priority level is determined by either the external interrupt request level that caused the interrupt or by bit <0> of the value supplied by external hardware. If bit<0> is 0, the new IPL level is determined by the interrupt request level being serviced (i.e. IRQ<0> 14 (hex)). Software must determine the level of the device that was serviced and set the IPL to the correct value.

Only device vectors in the range of 100 to 400 (hex) should be used, except by devices emulating console storage and terminal hardware.

2.1.7.6 The Hardware Halt Procedure

The Hardware Halt Procedure is the mechanism by which the hardware assists the firmware in emulating a processor halt. The hardware Halt Procedure saves the current value of the PC in IPR 42 (SAVPC), and the current value of the PSL, MAPEN<0>, a halt code and VALID bit in IPR 43 (SAVPSL). SAVPSL<14> (VALID BIT) is set to ZERO if the PSL is valid and to ONE if it is not. The VALID BIT is undefined after a halt due to a system reset).

Figure 2–11: Console Saved PC (SAVPC) - (IPR 42(DEC) 2A(HEX))

```
 3                                                           0
 1                                                           0
+-------------------------------------------------------------+
|                  Saved PC  (Read Only)                      |:SAVPC
+-------------------------------------------------------------+
```

Figure 2–12: Console Saved PSL (SAVPSL) - (IPR 43(DEC) 2B(HEX))

```
 3                             1 1 1 1         0 0           0
 1                             6 5 4 3         8 7           0
+-------------------------------+-+-+-----------+-------------+
|          PSL<31:16>           | | | HALT CODE |   PSL<7:0>  |:SAVPSL
+-------------------------------+-+-+-----------+-------------+
                                 | |
           MAPEN<0> -----------+ |
Valid Bit (Valid if ZERO) -------------+
```

The current stack pointer is saved in the appropriate internal register. The PSL is set to 041F 0000 (hex) (IPL=1F, kernel mode, using the interrupt stack) and the current stack pointer is loaded from the interrupt stack pointer. Control is then passed to the resident firmware at physical address E004 0000 (hex) with the state of the CPU as follows:

Table 2–15: CPU State After a HALT

Register	New Contents
SAVPC	Saved PC
SAVPSL<31:16,7:0>	Saved PSL<31:16,7:0>
SAVPSL<15>	Saved MAPEN<0>
SAVPSL<14>	Valid PSL flag (unknown for halt code of 3)
SAVPSL<13:8>	Saved HALT code
SP	Current Interrupt Stack (IPR 4)
PSL	041F 0000 (hex)
PC	E004 0000 (hex)
MAPEN	0
ICCS	0 (for a halt code of 3)
SISR	0 (for a halt code of 3)
ASTLVL	0 (for a halt code of 3)
All else	Undefined

The firmware uses the halt code in combination with any hardware event indicators to dispatch the exception or interrupt that caused the halt to the appropriate firmware routine (either console emulation, power-up, reboot, or restart). A list of the interrupts and exceptions that can cause a halt along with their corresponding halt codes and event indicators follows:

Table 2–16: HALT Codes for UnMaskable Interrupts

Halt Code	Interrupt Condition	Event Indicator
2	External Halt - CPU HALT_L pin asserted	Halt Button Pressed
3	Hardware Reset - CPU RESET pin negated	Power-up only

Table 2–17: HALT Codes for Exceptions

Halt Code	Interrupt Condition
6	HALT instruction executed in Kernel Mode

Table 2–18: HALT Codes for Exceptions that occurred while Servicing an Interrupt or Exception

Halt Code	Interrupt Condition
4	Interrupt stack not valid during exception
5	Machine check during normal exception
7	SCB vector bits<1:0>= 11
8	SCB vector bits<1:0>= 10
A	CHMx executed while on interrupt stack
10	ACV or TNV during machine check exception
11	ACV or TNV during kernel stack not valid exception
12	Machine check during machine check exception
13	Machine check during kernel stack not valid exception
19	PSL<26:24>= 101 during interrupt or exception
1A	PSL<26:24>= 110 during interrupt or exception
1B	PSL<26:24>= 111 during interrupt or exception
1D	PSL<26:24>= 101 during REI
1E	PSL<26:24>= 110 during REI
1F	PSL<26:24>= 111 during REI
3F	CPU Powerup selftest failed (Microcoded)

2.1.8 System Identification

2.1.8.1 System Identification Register (SID) IPR 62

The System Identification Register (SID), IPR 62, is a Read Only register implemented per the DEC STD 032 in the CPU chip. This 32-bit, Read-Only register is used to identify the processor type and its microcode revision level.

Figure 2–13: System Identification Register (SID) - (IPR 62(DEC) 3E(HEX))

```
 3            2 2                        0 0              0
 1            4 3                        8 7              0
+--------------+-------------------------+----------------+
|     TYPE     |        RESERVED         | MICROCODE REV. |
+--------------+-------------------------+----------------+
```

SID<31:24> (TYPE) Processor type. This field always reads as 12 (hex) indicating the processor is implemented using the DC595 CPU chip.

SID<23:8> Reserved for future use.

SID<7:0> (MICROCODE REV). Microcode Revision. This field reflects the microcode revision level of the CPU chip.

2.1.8.2 System Type Register (SYS_TYPE)

In order to distinguish between different CPU implementations that use the same CPU chip, the KA46 as must all VAX processors which use the DC595 CPU chip, implements a MicroVAX System Type Register (SYS_TYPE) at physical address 2004.0004 (hex). This 32-bit Read-Only register is implemented in the KA46 ROM. The format of this register appears below:

Figure 2–14: System Type Register (SYS_TYPE)

```
 3             2 2             1 1             0 0        0 0 0
 1             4 3             6 5             8 7        2 1 0
+---------------+---------------+---------------+----------+---+
|   SYS_TYPE    |   REV LEVEL   | SYS_SUB_TYPE  |   MBZ    |S|M|
|               |               |               |          |U|U|
+---------------+---------------+---------------+----------+---+
```

SYS_TYPE<31:24> (SYS_TYPE) System Type Code. 4

SYS_TYPE<23:16> (REV LEVEL) Revision Level. This field reflects the revision level of the KA46 firmware. The upper nibble (bits <15-12>) is the MAJOR revision level and the lower nibble (bits <12-8>) is the MINOR revision level.

SYS_TYPE<15:8> (SYS_SUB_TYPE) System Sub-Type Code. This field reads as 0 (zero) for the KA46.

SYS_TYPE<7:2> Reserved. Zero.

SYS_TYPE<1> (SU) Single User. 1.

SYS_TYPE<0> (MU) Multi User. 0.

2.1.9 Accelerator Control and Status Register (ACCS) IPR 40

The Accelerator Control and Status Register (IPR 40, ACCS) provides control for the F-chip, an optional vector unit, selects 30 or 32-bit physical addressing mode and provides the ability to generate bad data parity on write operations. The format of ACCS is as follows:

Figure 2–15: Accelerator Control and Status Register (ACCS) - (IPR 40(DEC) 28(HEX))

```
 3 3                                                    0 0 0 0
 1 0                                                    3 2 1 0
+-+------------------------------------------------------+-+-+-+
| |                        MBZ                           | | |0|
+-+------------------------------------------------------+-+-+-+
 |                                                        | | |
 |                                                        | | |
 +--- Write Even Parity        Physical Addressing Mode --+ | |
                                         F_Chip Present ----+ |
                               Vector Processor Present ------+
```

<31>	Write Even Parity. Write Only. This bit enables the generation of bad data parity for write operations. If Write Even Parity is set to a ONE, all subsequent cache or memory writes and F-chip operand transfers will be done with bad data parity on all bits. This bit is used for diagnostics only, and should never be set during normal operations. Write Even Parity is automatically cleared if ACCS is read with an MFPR instruction. Also, the write-even-parity state is cleared at the start of any interrupt, exception, or console halt to prevent an exception stack frame from being written with bad data parity. The Write Even Parity bit is cleared by Power-on.

Note

The M bit should be set in any PTE that maps pages that are to be written while WRITE_EVEN_PARITY is enabled. Failure to do so may result in a PTE being written with bad parity during an M bit update.

<30:3>	Reserved. Must be written as ZERO. Read as zero.
<2>	Physical Address Mode. Read/Write. This bit controls whether the DC595 uses 30-bit (this bit cleared) or 32-bit (this bit set) physical addressing mode. This bit is cleared by Power-on and may only be changed when memory management is disabled. For the application of the DC595 in the VS4000-400 this bit **MUST** remain cleared.
<1>	F_Chip Present. Read/Write. This bit enables use of the F-chip. If F_Chip Present is a ONE, floating point and longword-length integer multiply instructions are passed to the F-chip for execution. If F_Chip Present is a ZERO, the execution of a floating point instruction results in a reserved instruction fault. As an F-chip is included on every KA46 module, F_CHIP PRESENT should be set during normal operation. If an F-chip error is detected, the F-chip may be disabled if the operating system emulation software is loaded. This bit is cleared during reset.
<0>	Vector Processor Present. Read/Write. If this bit is a ONE, vector instructions are decoded and passed to an optional vector unit for execution. If VECTOR_PRESENT is a ZERO, the execution of a vector instruction results in a reserved instruction fault. This bit is cleared at Power-on. For this application of the DC595 in the VS4000-400, the optional vector unit is never present and this bit **MUST** remain cleared.

2.2 Primary Cache

The DC595 has on-chip a small (2K Bytes) single set, direct mapped, write through cache. This cache is controlled and reports status via four IPRs.

2.2.1 Primary Cache Status Register

This IPR is used for controlling the operation of the primary cache, flushing it and maintaining status on errors.

Figure 2–16: Primary Cache Status Register (PCSTS) - (IPR 127(DEC) 7F(HEX))

```
 3                                  1 1 1 0 0 0 0 0 0 0 0 0 0 0
 1                                  3 2 1 0 9 8 7 6 5 4 3 2 1 0
+------------------------------------+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  IGN               | | | | | | | | | | | | |0|
+------------------------------------+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                      | | | | | | | | | | | | |
                       B_Cache_Hit ---+ | | | | | | | | | | | |
                         Bus_Error -----+ | | | | | | | | | | |
               P_Data_Parity_error -------+ | | | | | | | | | |
             DAL_Data_Parity_Error ---------+ | | | | | | | | |
                  TAG_Parity_Error -----------+ | | | | | | | |
                             TRAP1 -------------+ | | | | | | |
                             TRAP2 ---------------+ | | | | | |
                         INTERRUPT -----------------+ | | | | |
                       P_Cache_Hit -------------------+ | | | |
                             SPARE ---------------------+ | | |
                       FLUSH_Cache -----------------------+ | |
                      ENABLE_Cache -------------------------+ |
                         FORCE_Hit ---------------------------+
```

31:13	Unused, ignored on Write, Read as zero.
12	B_Cache_Hit. Read ONLY - this bit is updated for any reference that has an error associated with it and the Status Register is not locked. The Status Register is considered locked if INTERRUPT or TRAP1 is set and the error results in an interrupt or if TRAP1 is set and the error results in a trap. B_Cache_hit indicates that the condition that resulted in the Status Register being locked was a reference that hit in the secondary cache. For DAL parity errors, this bit can be used to identify the source of that error. This bit is cleared by power-on.
11	Bus_Error. Read ONLY - this bit is updated for any reference that has an error associated with it and the Status Register is not locked. The Status Register is considered locked if INTERRUPT or TRAP1 is set and the error results in an interrupt or if TRAP1 is set and the error results in a trap. Bus_Error is never set for the use of the DC595 in the VS4000-400.

10	P_Data_Parity_Error. Read ONLY - this bit is updated for any reference that has an error associated with it and the Status Register is not locked. The Status Register is considered locked if INTERRUPT or TRAP1 is set and the error results in an interrupt or if TRAP1 is set and the error results in a trap. This bit is set when a read hits in the primary cache and the requested data has a parity error. If the operation was an I-stream read, INTERRUPT is also set and the error is reported as an IPL 1Ah soft error interrupt. If the operation was a D-stream read, TRAP1 (or TRAP2) is also set and the error is reported as a microtrap with subsequent machine check. This bit is cleared by power-on.
09	DAL_Data_Parity_Error. Read ONLY - this bit is updated for any reference that has an error associated with it and the Status Register is not locked. The Status Register is considered locked if INTERRUPT or TRAP1 is set and the error results in an interrupt or if TRAP1 is set and the error results in a trap. This bit is seet when the data returned in response to a non-I/O space read has a parity error. If the error is detected on the non-requested longword of a quadword D-stream read, or on either longword of an I-stream read, INTERRUPT is also set and the error is reported as an IPL 1Ah soft error interrupt. If the error is detected on the requested longword of a D-stream read, TRAP1 (or TRAP2) is also set and the error is reported as a microtrap with subsequent machine check. This bit is cleared by power-on.
08	Tag_Parity_Error. Read ONLY - this bit is updated for any reference that has an error associated with it and the Status Register is not locked. The Status Register is considered locked if INTERRUPT or TRAP1 is set and the error results in an interrupt or if TRAP1 is set and the error results in a trap. This bit is set if a tag parity error is detected during a read, write or invalidate cycle. If this bit is set, INTERRUPT is also set and the error is always reported as an IPL 1Ah soft error interrupt. If the reference was a D-stream read that hit, TRAP1 (or TRAP2) is also set and the error is reported as a microtrap with subsequent machine check. This bit is cleared by Power-on.
07	TRAP1. Read/Write to clear. This bit is set when an error is detected and is to be reported as a microtrap with subsequent machine check. Bits <12:08> of the Status Register and the Error Address Register are latched unless TRAP1 is already set and remain latched until TRAP1 is cleared. If this bit is set, the primary cache is automatically disabled (the state of the Enable Bit is *NOT* changed). TRAP1 is cleared by Power-on and by writing to this IPR with bit<07> set.

Note

If INTERRUPT is already set when another error sets TRAP1, the information about the error reported as an interrupt is lost. Because errors reported as interrupts are non-fatal, software should assume that bits<11:8> are all set and perform the error recovery procedures for all four possible errors.

06	TRAP2. Read/Write to clear. This bit is set when an error is detected that is to be reported as a microtrap with subsequent machine check and TRAP1 is already set. If TRAP2 is found set in the Status Register, it indiactes that a nested error occurred and that bits<12:8> of the Status Register and the Error Address Register contain information about the first error only. This should be considered a fatal error. When this bit is set, the primary cache is automatically disabled (the state of the Enable Bit is *NOT* changed). TRAP2 is cleared by Power-on and by writing to this IPR with bit<06> set.
05	INTERRUPT. Read/Write to clear. This bit is set when an error has been detected that is to be reported as a soft erroor interrupt at IPL 1Ah. When the error occurs, bits<12:8> are latched unless INTERRUPT or TRAP1 is already set. These bits then remain latched until INTERRUPT is cleared or until another error sets TRAP1. When this bit is set, the primary cache is automatically disabled (the state of the Enable Bit is *NOT* changed). INTERRUPT is cleared by Power-on and by writing to this IPR with bit<05> set.
04	P_Cache_Hit. Read ONLY. This bit is for diagnostic purposes only. It is the latched value of the Primary Cache Tag comparator output and is updated for all D-stream reads, writes and invalidate cycles. This bit is cleared by Power-on.
03	SPARE. This bit had significance for the CPU chip previous to the DC595. It now has none and may be used as a general purpose read/write bit with no hardware implications.
02	Flush_Cache. Read as zero/Write. This bit is written with a ONE to clear all Valid bits in the Primary Cache Tag array. When the flush is complete, this bit is automatically cleared by hardware. This bit always reads as a zero. **Note** To correctly initialize the primary cache, it is also necessary to write all TAG Array locations with valid parity - see below Section 2.2.4. Each TAG Array entry whould be written with the value of 80000000h.
01	Enable_Cache. Read/write. This bit controls overall operation of the Primary Cache. When it is written with a ONE, I-stream and D-stream references are cached in the Primary Cache and both Cache Data and Tag errors are reported. When this bit is a zero, all references result in a Primary Cache Miss and no allocates occur. This bit is cleared by Power-on.
00	Force_Hit. Read/write. This bit is for diagnostic purposes only. When this bit is set, the Primary Cache control forces a Hit for all non I/O space references and all Primary Cache error reporting is disabled; DAL and Bus error reporting is not affected. This bit is cleared by Power-on.

2.2.2 Error Address Register

For read commnads, the Error Address Register latches and holds the physical address which causes TRAP1 to set in the Status Register. As write errors are asynchronous to the instruction pipeline, the address latched on a write is not the error address. For write commands no address is available. The contents of this register remain locked until TRAP1 is cleared. The contents of this register only have significance if TRAP1 is set.

Figure 2–17: Primary Cache Error Address Register (PCSTS) - (IPR 126(DEC) 7E(HEX))

```
 3                                                           0
 1                                                           0
+-------------------------------------------------------------+
|                         READ ONLY                           |
+-------------------------------------------------------------+
```

2.2.3 Primary Cache Tag Array Register (PCTAG) - (IPR 124(DEC) 7C(HEX))

The Tag Array Register, together with the Index Register (see below) provides a mechanism for reading and writing the Primary Cache Tag Array. Format is as below :

Figure 2–18: Primary Cache Tag Address Register

```
         3 3                                  1 1                   0 0
         1 0                                  1 0                   1 0
        +-+-------------------------------------+--------------------+-+
        | |                                     |       ZERO         | |
        +-+-------------------------------------+--------------------+-+
         | |                                   |                      |
 PARITY -+ |                                   |                      |
   TAG  ---+-----------------------------------+                      |
  VALID --------------------------------------------------------------+
```

2.2.4 Primary Cache Index Register (PCIDX) - (IPR 125(DEC) 7D(HEX))

The Index provides a mechanism for reading and writing the Primary Cache Tag array.

Figure 2–19: Primary Cache Tag Address Register

```
 3                                    1 1              0 0   0
 1                                    1 0              3 2   0
+-------------------------------------+----------------+-----+
|              0 ... 0                |      INDEX     |  0  |
+-------------------------------------+----------------+-----+
```

2.2.4.1 Writing the Primary Cache Tag Array

First the Index of the entry to be accessed is written into IPR PCIDX in the format shown in Section 2.2.4, then the Tag value, Valid and Parity bits are written to IPR PCTAG, format as in Figure 2–19 above.

2.2.4.2 Reading the Primary Cache Tag Array

First PCIDX is loaded with the desired Index, then PCTAG is read and the Tag, Valid and Parity bits are returned in the bits positions shown in Figure 2–19 above.

2.3 Chip Interconnect

The DC595 connects to the DC596, the DC598, the DC7203 and to the secondary cache RAMs only. All connections to the remainder of the system are made from the DC7203. See Chapter 3 for more details of the cache operation and Chapter 4 for more details of memory and I/O operations.

Chapter 3

Secondary Cache

The CPU and Floating Point Accelerator connect to a secondary cache used to reduce the load on the main memory system and to improve system performance. This cache is organized as single set, quadword line size, direct mapped, write-though. The cache controller supports a maximum main memory size of 128 MBytes and a cache capacity of 256 KBytes. The Tag Store is thus 32K x 9, plus Valid and (Odd) Parity Bits; the Data Store 32K x 64 plus eight parity bits, one per byte. Data Store parity is generated and checked by the CPU; Tag Store parity is generated and checked by the DC7203 chip.

3.1 Cache Organisation

Each cache *set* consists of an array of *entries*. Each entry contains a *valid* flag, a *tag*, and a *data* block.

Figure 3–1: Cache Organization

```
                                               b
                V    TAG: t bits       DATA: 2  bytes
              +---+---------------+------------------------+
            / |   |               |                        |
            | |   |               |                        |
            | |   |               |                        |
 e          | |   |               |                        |
2  entries <  |   |               |                        |
            | |   |               |                        |
            | |___|_______________|________________________|
            | |___|_______________|________________________|<-+
            \ |   |               |                        |  |
              +---+---------------+------------------------+  |
                                                              |
                                             one cache entry---+
```

The data block holds 8 bytes (*b*=3) which are a copy of an aligned quadword in main memory. There is just one entry in which a copy of a given main memory quadword can be found. That entry is selected by an *e*-bit entry index which is extracted from the quadword's physical address. However, there are numerous quadwords which could occupy one cache entry, so they are distinguished by a *t*-bit tag which is also extracted from the quadword's physical address. The breakdown of the 30-bit VAX physical address is shown below:

Figure 3–2: Cache Address Mapping

```
 3 3 2 2  2 2           1 1                    0 0      0
 1 0 9 8  7 6           8 7                    3 2      0
+---+-+----+-------------+---------------------+--------+
|IGN|0|IGN |    t bits   |        e bits       | b bits |
+---+-+----+-------------+---------------------+--------+
```

Starting from bit 0, the value of *b* depends on the size of the data block, which in this system is 8 bytes, so *b* is 3 (2**3 = 8). The values of *e* and *t* depend upon how many entries are in the set. For this cache, there are 32,768 entries and a maximum supported memory size of 128 Mbytes, so *e*=15 and *t*=8. Bits <31:30, 28:27> of the address are ignored by the cache controller. If Bit<29>=1, the address is ignored by the cache controller, data for addresses in I/O space are never cached.

Each entry's valid flag *V* indicates whether the entry contains a valid copy of data from main memory. The hardware automatically sets the valid flag whenever a processor read cycle stores data in the entry. All the valid flags in the cache must be cleared under program control before the cache is enabled after a period of processor operation with the cache disabled, since main memory data may have been changed during this period.

3.2 Cache Operation

The secondary cache controller section of the DC7203 accepts a 27-bit address from the CPU. Bits <2:0> are ignored as the cache line size is quadword. All read accesses from the cache are of 64 bits, CPU initiated writes of less than a full quadword are controlled by the eight CPU-generated Byte Mask control lines.

On a Read, bits <17:03> are used as an address for both the Tag Store and the Data Store. The output of the Tag Store is compared to address bits <26:18> for a possible match. If the Tag Store output matches the Tag Field of the address and the Valid bit is set and the secondary cache is enabled and there is no parity error indicated, the data requested by the CPU is supplied from the secondary cache and no cycle to main memory is started.

On a Write, the same address lookup and comparison are made, but regardless of whether the data are contained in the cache or not, a cycle to update main memory is started. If the address does match, the Valid bit is set and no tag parity error indicated, the specified bytes are also updated in the secondary cache.

3.2.1 Secondary Cache Control

A longword register at address 2300.0000 controls the operation of the secondary cache. Its writeable bits are all cleared to zero on power-on.

Figure 3–3: Secondary Cache Control Register

```
 3                                    1 0 0 0           0 0
 1                                    0 9 8 7           1 0
+------------------------------------+-+-+-+-+---------+-+-+
|              ... IGN ...           | | | | |         | | |
+------------------------------------+-+-+-+-+---------+-+-+
                                      | | | |           | |
REV    -------------------------------+ | | |           | |
SPECIO ---------------------------------+ | |           | |
FONFUL -----------------------------------+ |           | |
MOD    -------------------------------------+           | |
TERR   -------------------------------------------------+ |
CENA   ---------------------------------------------------+
```

<0> - CENA	Cache Enable, read/write. When this bit is set, the secondary cache is enabled.
<1> - TERR	Tag Error, read only. This bit is set on detection of a Tag Parity Error. When this bit sets, it unconditionally clears CENA. This bit is cleared by any write to the secondary cache control register or by power-on.
<7> - MOD	Mode, read/write. This bit sets the operation mode of the cache controller write buffer : 0 - default, write buffer is emptied when it is full. 1 - write buffer is emptied when it contains any data - equivalent to no write bufferring.
<8> - FONFUL	Flush on Full, read/write. This bit sets the flush behaviour of the write buffer. When this bit is clear and the write buffer is filled, a subsequent write to memory space will cause the oldest entry only in the write buffer to be written to the DC7201, the buffer remains full after the write has completed. When this bit is set and the write buffer is filled, a subsequent write to memory space will cause all the entries in the write buffer including the last one to be sent to the DC7201, the buffer is empty at the conclusion of this write. This bit is normally cleared.
<9> - SPECIO	Special IO, read/write. This bit sets the action of the DC7203 write buffer for two special IO address ranges. When this bit is cleared, address ranges 2100.0000:21FF.FFFF and 2018.0000:201F.FFFF are treated on writes as memory addresses are treated, i.e. they are write buffered. As they are IO addresses, however, no longword to quadword packing is performed if sequential writes are to successive longwords as happens with memory space addresses. In this way, when such an address finally exits the write buffer to the DC7201, it will be as a longword write only. If this bit is set, writes to these address ranges are treated as are normal IO address writes and cause a write buffer flush. Regardless of the setting of this bit, reads to these two address ranges are treated as are normal IO addresses, they cause a write buffer flush before the IO read is executed.

<10> - REV	Revision of the DC7203. This bit is READ ONLY, a ZERO indicates Revision A of the chip, a ONE indicates Revision B.

3.2.2 Diagnostic Mode

The Secondary Cache Tag and Data Stores can be accessed via two ranges of fixed addresses for diagnostic purposes.

3.2.2.1 Tag Store Diagnostic Access

The address range allocated for diagnostic access to the Tag Store is 2D00.0000 - 2D03.FFFF. Tag Store entries are accessed at sequential *quadword* addresses (address bits<02:00> = 000) with the following format:

Figure 3–4: Tag Store Diagnostic Access

```
 3                                         1          0 0 0
 1                                         0          2 1 0
+-----------------------------------------+------------+-+-+
|                ... IGN ...              |     TAG    |P|V|
+-----------------------------------------+------------+-+-+
```

V	Tag Valid Bit - read/write. When accessed via this address range, the Valid bit is directly accessible. Reads of the Tag Store do not cause a Tag comparison as does a normal access.
P	Tag Parity Bit - read/write. When accessed via this address range, the Parity bit is directly accessible. Writing wrong parity will not cause a parity error if the subsequent read also occurs via the diagnostic address range. Correct parity is ODD, i.e. an odd number of ONES in the TAG field requires that the parity bit also be a ONE.
TAG	Cache Tag - read/write.

3.2.2.2 Data Store Diagnostic Access

The address range allocated for diagnostic access to the Secondary Cache Data Store is 0800.0000 - 0803.FFFF. Accesses using this address range perform no Tag lookup, so the entire Data Store is accessible as 32,768 sequential quadwords. Parity checking is performed by the CPU as for any normal access.

3.3 Secondary Cache Initialization

Before the Secondary Cache is enabled, see Figure 3–3, it must be initialized. This is done by writing to the Tag Store using the diagnostic address range and clearing all the Valid bits to zero. The parity and data written may be anything. The Secondary Cache Data Store need not be initialized.

3.3.1 Error Recovery

Read parity errors will be detected by the CPU. On detection of such an error, it is recommended that the CPU disable the secondary cache and retry the read. The data this time will come from main memory and indicate whether the initial error really was a Secondary Cache error.

Secondary Cache Tag Store errors will result in the assertion of the Hard Error Interrupt line HERR_IRQ_L. This is a CPU interrupt at IPL 1Dh, SCB vector 60h. The Secondary Cache will also be automatically disabled.

Chapter 4

MAIN MEMORY

Main memory consists of from 8 to 104 Mbytes of DRAM installed on the KA46 system board. The base system board always has 8 MBytes installed, expansion memory is added as pairs of SIMM assemblies, using either 4 MByte or 16 MByte SIMMs. A maximum of six SIMM assemblies may be installed. Subject to the following constraints :

- memory must be expanded as pairs of SIMMS.
- both SIMMs of any pair must be the same, i.e. *both* either 4 MByte or *both* 16 MByte.
- The six SIMM sockets have an implied address sequence. Sockets 0 and 1 are always the lowest address of expansion memory, sockets 4 and 5 always the highest. The address ranges for each SIMM SOCKET pair are a function of the type of SIMMs installed.
- Memory addressing is adjusted automatically according to the type of SIMM installed in each socket to always present a contiguous address range to the CPU, starting at physical address 0 and extending upwards towards 67F.FFFF
- the SIMMs may only be intermixed in combinations that put 4 MByte assemblies at lower physical addresses than 16 MByte assemblies.

The following are the possible memory configurations. The six SIMM sockets are designated SOCKET<5:0> where SOCKET<0,1> are the first to be populated. 4,4 in a SOCKET column says that that pair of SIMM sockets both have 4 MByte SIMMs installed; 16,16 that both have 16 MByte assemblies installed. Total Memory Capacity includes the 8 Mbytes always on the system board.

Table 4–1: Memory Configurations

Socket	0,1	2,3	4,5	Total Memory Capacity
	4,4	–	–	16
	4,4	4,4	–	24
	4,4	4,4	4,4	32
	16,16	–	–	40
	4,4	16,16	—	48
	4,4	4,4	16,16	56
	16,16	16,16	–	72
	4,4	16,16	16,16	80
	16,16	16,16	16,16	104

Each SIMM assembly is organised 32-bits wide, so has just one DRAM per bit for either the 4 MByte or 16 MByte variations. Four parity bits are added to each SIMM, one per byte.

The amount of memory installed in a system can be read by accessing the Configuration register, see Section 7.2, this data is then used to initialize the MEM_CONFIG register if necessary - see below.

4.1 LCG Specified Registers

Certain control registers essential to the memory and video control are fully specified in the Graphics Controller Specification. For convenience, the definition of bits relevant to memory operation in one of these registers is repeated here.

4.1.1 MEM_CONFIG Register

Prior to being able to access any memory location, this register must be written to specify the physical memory configuration if this differs from the default case. This is a longword aligned register at location 2010.18000h. Memory is considered as four Banks, Bank 0 is the base system board 8 MBytes, so is always present, each of Banks<3:1> being either 8 MBytes or 32 Mbytes depending on whether the two SIMM slots associated with each Bank are populated with 4 MByte or 16 MByte SIMM memory modules (see Memory Configurations above). Bits <2:0> set the SIMM size for Banks<3:1> respectively. A ZERO in a bit position specifies 4 MByte SIMMs, a ONE 16 MByte SIMMs.

On Power-on, these Bank bits are initialized for 4 Mbyte SIMMs, as this is the normal configuration. For First Pass machines, normal mmeory operation is possible without altering the contents of this register.

Other bits in this register specify Video Configuration Options - they are fully described in the Graphics Controller Specification.

The DC7201 arbitrates between and services requests for main memory cycles from several sources; the CPU Secondary Cache Controller, the Bus Adaptor controller, the Ethernet Controller (NI), the Storage Controller (SCSI), the Invalidate Filter and the Graphics Controller (GC) section of the memory controller. To optimize use of the available RAM bandwidth, data to/from these several requestors is buffered within the DC7201. To further minimize interaction between the requestors, the DC7201 has three data busses; the CDAL which connects it to the CPU Secondary Cache Controller; the EDAL which connects it to the NI, SCSI and Invalidate Filter; the MDAL which connects to the memory system, including the Video RAMS which make up the PV21X-GB or PV21X-GD option module.

The memory controller is capable of performing several types of RAM cycles; longword, quadword and octaword. Buffering between the several requestors and the memory allows these various cycles to be used in a way that makes best use of the available memory bandwidth. The sections that follow will discuss each requestor and the cycles generated.

The NI controller and SCSI controller are DMA devices; the GC can generate addresses independently. Thus all three of these devices may attempt to write to memory locations that are currently cached. To maintain cache coherency it is necessary that for all writes requested by any of these devices, the CPU cache be checked and, if necessary, that entry invalidated. This could impose a significant load on the CPU to check the potential invalidates. For this reason the DC7201 controls a separate invalidate filter which maintains a copy of the CPU's Secondary Cache tags (a superset of the Primary Cache tags). In this way only those writes that do require a cache invalidate will disturb the CPU/Secondary Cache Controller.

To allow greater flexibility, DMA devices access memory via a translation map contained in memory - see Section 4.6. The GC address generator performs virtual access to memory via the page tables - refer to the PVAX2 Graphics Controller Specification for further information.

4.2 Main Memory Requests

Main memory requests have a fixed priority, as follows (highest to lowest) :

- GC - shift register load
- GC - cursor buffer load
- Refresh
- NI Controller
- DC7203
- Bus Adaptor Controller
- SCSI Controller
- GC Address Generator

Refer to the PVAX2 Graphics Controller Specification for details of all GC functions.

4.2.1 NI Controller

The NI Controller is one of the requestors connected to the EDAL bus. It contends with the SCSI controller and the Invalidate Filter for access. The DC7201 maintains input and output buffers for the 16-bit wide data received from and to be sent to the NI Controller.

For NI controller data transfers to the DC7201, the 16-bit data words are accumulated until a quadword address boundary is crossed or until the address of the next received data is seen to be no longer in sequence. This accumulated data is then written to memory in a single operation. For transfers that start at a quadword aligned address, the write to memory will be a page-mode quadword cycle. For transfers that start at a longword aligned address, the first write to memory will be a longword write of all four bytes. For transfers that start at a non-longword aligned word address, the first write to memory will be a longword write to bytes <3:2> only. In either of these two non-quadword aligned cases, after this initial alignment write has occurred, all subsequent sequential transfers will cause page-mode quadword cycles. An interrupt from the NI Controller or a timeout - no data transfer from the NI Controller to the S_Chip within <<<tbd microseconds>>> will cause any accumulated data to be written to memory.

Requests from the NI Controller for data from memory that have a quadword aligned address will cause the DC7201 to retrieve a quadword of data from memory. The three additional 16-bit words read in addition to the word actually requested by the NI Controller remain buffered within the DC7201 to satisfy the (assumed) next three read requests from the NI controller. For non-quadword aligned address requests the DC7201 retrieves the longword of data that includes that actual word requested. Depending on the actual address alignment, the second word of this data may be used to fill the next NI Controller request or may be discarded.

This buffer/lookahead read mechanism is totally transparent to software; the operating system driver need have no knowledge that this is happening, but should be aware that program loops that rely on exact timings may show inconsistencies.

All memory write requests that originate from the NI Controller are passed to the invalidate filter logic as the write occurs to memory - see Section 4.5.

4.2.2 SCSI Controller

The SCSI Controller is one of the requestors connected to the EDAL bus. It contends with the NI controller and the Invalidate Filter for access. The DC7201 maintains input and output buffers for the 16-bit wide data received from and to be sent to the SCSI Controller.

For SCSI controller data transfers to the DC7201, the 16-bit data words are accumulated until a quadword address boundary is crossed or until the address of the next received data is seen to be no longer in sequence. This accumulated data is then written to memory in a single operation. For transfers that start at a quadword aligned address, the write to memory will be a page-mode quadword cycle. For transfers that start at a longword aligned address, the first write to memory will be a longword write of all four bytes. For transfers that start at a non-longword aligned word address, the first write to memory will be a longword write to bytes <3:2> only. For transfers that start at a non-word aligned byte address the first write to memory will be a longword write to byte<3> only. For all of these non-quadword aligned cases, after this initial alignment write has occurred, all subsequent sequential transfers will cause page-mode quadword cycles. An interrupt from the SCSI Controller or a timeout - no data transfer from the SCSI Controller to the S_Chip within <<<tbd microseconds>>> will cause any accumulated data to be written to memory.

Requests from the SCSI Controller for data from memory that have a quadword aligned address will cause the DC7201 to retrieve a quadword of data from memory. The three additional words read in addition to the word actually requested by the SCSI Controller remain buffered within the DC7201 to satisfy the (assumed) next three read requests from the SCSI controller. For non-quadword aligned address requests the DC7201 retrieves the longword of data that includes that actual word requested. Depending on the actual address alignment, the remainder of this data may be used to fill the next SCSI Controller request(s) or may be discarded.

This buffer/lookahead read mechanism is totally transparent to software, the operating system driver need have no knowledge that this is happening.

All memory write requests that originate from the SCSI Controller are passed to the invalidate filter logic as the write occurs to memory - see Section 4.5.

4.2.3 CPU Memory Requests

The secondary cache controller can request longword or quadword read cycles and longword or quadword write cycles. As the CPU's internal cache and the secondary cache are both write-through, all CPU generated writes appear immediately as write requests to the DC7203. The DC7203 contains a four quadword write buffer for CPU write requests so that longword write requests to sequential longword addresses may be performed as quadword writes to the DC7201. The DC7201, in turn has its own two quadword write buffer. Writes from the DC7201 are, wherever possible sent to memory as page mode quadword writes to take advantage of the faster page-mode cycles possible when writes are performed to sequential addresses within the same page.

Note

depending on available gates within the DC7201, the write buffering may be extended to octawords for the seond pass of the DC7201.

4.3 Write Buffer and Read Requests

When the CPU issues a non-I/O space read request that misses in the secondary cache and so must be filled by data from main memory and there is data currently in the DC7203 write buffer waiting to be written, the read request is executed without flushing the write buffer. On any CPU generated read request when the write buffer is not empty, the DC7203 first compares the requested address with the address(es) of the data in the write buffer; if a match occurs, the data is returned to the CPU from the write buffer. If only a part of the requested data is found in the write buffer, a memory read is requested and the data returned supplies the missing bytes. If the requested data is not found in the write buffer, a memory read is requested and all the data is supplied from memory to the CPU. In no cases are the contents of the write buffer altered by a read operation.

4.3.1 Write Buffer Flushing

Certain operations require that data held in the DC7203 write buffer be written back to memory before these operations are allowed to occur. The DC7203 makes this determination and ignores any CPU-generated *Clear Write Buffer* cycles.

The DC7203 will write back all data contained in its write buffer when it receives a request for a read or write to any address in I/O space (address bit <29> = 1), when it recognises an INTACK Cycle or when a write is directed to the Bus Adaptor (address range 30000000:3DFFFFFFh). In these cases, the data in the write buffer is first written to memory, then the I/O read or write or the ROM read (INTACK) occurs. The result at the end of either operation is that the write buffer is empty. For the duration of this operation, the CPU is stalled waiting for completion of the cycle that initiated the write buffer flush.

Similar sequences of events occurs within the DC7201; read requests coming from the DC7203 first perform a lookup in the DC7201 write buffer before starting a memory read cycle. I/O read/write and INTACK cycles cause a DC7201 write buffer flush.

The result of the two serial write buffers is that on occasions, I/O cycles may take some considerable time to execute, operating system software should be aware of the possibility of such delays.

4.4 Alternate Address Range

The DC595 CPU chip always checks for parity errors on data returned to it on reads from addresses in memory space. To allow easier diagnosis and testing of the parity checking logic, main memory may be accessed via an alternate address range in I/O space : 2400.0000 - 2BFF.FFFF. The CPU does not check parity on read references to addresses in I/O space. Using this address range for read accesses to memory, in conjunction with the Write_Even_Parity control bit in IPR 28h, see Section 2.1.9 in Chapter 2 allows verification of the memory error checking capabilities by diagnostic software. Note that using this alternate address range, in common with all other I/O addresses, only longword (or shorter) references can be made in any given cycle.

4.5 Invalidate Filter

The DC7201 controls three 64K x 4 SRAMs where a copy of the Secondary Cache Tag Store contents is maintained. On CPU cache allocates, the tag value is written into this RAM. On any write generated by the NI, SCSI or GC, the tag copy is checked for a possible invalidate request to be passed back to the CPU. Only if a match is found will be CPU be disturbed and the entry in the invalidate filter SRAMs be cleared. CPU invalidates are requested by assertion of the CPU DMA Request line by the DC7201 which then executes a quadword or octaword invalidate cycle. The DC7203 converts any octaword invalidate requests into two sequential quadword invalidates.

Refer to the DC7201 Specification for details of data storage in the SRAMs of the Invalidate Filter.

4.5.1 Invalidate Filter Initialization

Prior to enabling the CPU cache, the Invalidate Filter must be initialized so that it represents a copy of the initial CPU cache state, i.e. no entries cached. This is done by accessing the Invalidate Filter RAMs via a range of I/O addresses, 2020.0000:2021.FFFF; the 32,768 filter entries are accessed as bits<10:00> of each of the 32,768 longwords in this range. Sequential longword writes to this address range with data of zero will initialize the filter RAMs. Bits <31:11> are UNPREDICTABLE on reads of this address range. All bits of the Invalidate Filter RAM are writeable when accessed via this address range.

4.6 DMA Mapping

The NI and SCSI controllers access memory via a translation table which is contained in main memory. A Map Base Address Register - MAP_BASE - within the DC7201 points to the beginning of this reserved region of memory. A possible 32,768 longwords extending upwards from MAP_BASE provide translations for the page address supplied by either DMA device. Each DMA device is provided with a two-entry cache of current translations (one for read, one for write) kept within the DC7201. It is the responsibility of the operating system to allocate entries for each DMA device in this common translation table. See below, Section 4.6.1 for details of the contents of a translation table entry.

Figure 4–1: Map Base Register MAP_BASE - Address 2008.0008

```
 3         2 2               1 1                              0
 1         7 6               7 6                              0
+-----------+-----------------+--------------------------------+
|    IGN    |      BASE       |             IGN                |
+-----------+-----------------+--------------------------------+
```

<31:27>	IGN. Ignored on write, read as zero.
<26:17>	BASE<9:0>. Specifies the start of a 32 K longword region in main memory. The longwords specified by this address and up to 32,767 longwords that follow contain translation entries for DMA devices.
<16:00>	IGN. Ignored on write, always regarded as zero. Read as zero.

4.6.1 Translation Table Entry Format

Although the space allocated for the DMA translation table is 32,768 longwords, the actual space used is a function of the starting addresses supplied to each of the DMA devices and the maximum transfer length allowed.

Any of the 32,768 longwords allocated for DMA translation entries must be written in the following format prior to initiating any DMA transfer :

Figure 4–2: Translation Entry Format

```
 3 3                       1 1                            0
 1 0                       8 7                            0
+-+------------------------+------------------------------+
|V|          MBZ           |          PAGE NUMBER         |
+-+------------------------+------------------------------+
```

<31>	Valid Bit. Indicates to the DMA translation hardware within the DC7201 that this location contains a valid page number for use in translation. The operating system must set this bit for all translation table entries that it expects to be used.
<30:18>	IGNORED. Thes bits MUST BE ZERO.
<17:00>	Page Number, the longword aligned start address of a 512 byte region of memory allocated as a part of a buffer for one of the DMA devices. The eighteen bits specified in this field allow for a maximum memory size of 128 MBytes.

4.6.2 Translation

Each DMA controller has its own 24-bit address counter for DMA transfers that has a page field (15-bits) and an address-within-page field (9-bits). When a DMA controller presents an address to the DC7201 to perform a DMA cycle to/from main memory, the DC7201 translates the address supplied, using MAP_BASE and the translation table contained in main memory as follows. Bits <23:09> of the address supplied by the DMA controller - the page field - are compared, as appropriate, to the address value held in either the read or write translation cache for that DMA device. If the addresses match and if that entry is marked as Valid, the associated page address held in a field of that cache entry is concatenated with the address-within-page field of the supplied address to form the actual address to be used and the DMA cycle proceeds.

If the address match fails, indicating that this DMA transfer is to or from an address on a different page from the last DMA transfer that this device initiated, bits <23:09> of the DMA address supplied are concatenated with bits <26:17> of MAP_BASE to form a new 25-bit longword aligned map register address. This address is then used to retrieve data from the translation table. Bits <17:00> of the data returned from memory are then concatenated with the original address-within-page bits supplied by the controller to form a 27-bit address that is the actual address to be accessed. Bits<17:00> that were retrieved from the translation table are stored as a new value in the appropriate translation cache associated with that DMA device and the Valid bit set for that entry.

Figure 4–3: DMA Address Translation - Translation Cache Hit

```
                                                       Byte
                        2 <-- Unmapped Page -->  0 0  within  0
                        3     Address           9 8   Page   0
                       +--------------------------+------------+ DMA
                       |        15 bits           |  9 bits    | Device
                       +--------------------------+------------+ Address
                       |                          |            |
    __________________/                          /|            |
   /              ______________________________/ |            |
  |              /                                |            |
  |             |                                 |            |
 3|3           1|1 <---- Mapped Page ---> 0       |            |
 3|2           8|7       Address          0       |            |
+-+-------------+-------------------------+       |            | Cached
|V|   15 bits   |        18 bits          |   <--------------  | Entry
+-+-------------+-------------------------+       |            |
 V=1 and these  |                         |       |            |
 Fields match   |                          \_____ |            |
                |                                \|            |
                |                                 |            |
                +------------------------+--------+------------+
                                         |
                                         V
                                Address to memory
```

Figure 4–4: DMA Address Translation - Translation Cache Miss

```
          2         1 1                  0
          6         7 6                  0
+--------+-----------+--------------------+
|        |  10 bits  |                    |  MAP_BASE
+--------+-----------+--------------------+
         |           |     2                     0 0          0
         |           |     3                     9 8          0
         |           |    +-----------------------+------------+ DMA
         |           |    |       15 bits         |  9 bits    | Device
         |           |    +-----------------------+------------+ Address
         |           |    |  Save addr. in cache  |            |
         |           | ___/             _________/|            |
         |           |/                /          |            |
         |2         1|1               |           |            |
         |6         7|6              2|1 0        |            |
+--------+-----------+----------------+---+       |            |
|        |           |                |0 0|       |            |
+--------+-----------+----------------+---+       |            |
         |                            |           |            |
         +----------------+-----------+           |            |
     Map Register Address |                       |            |
  +-----------------------+                       |            |
  |     3 3          1 1                      0|            |
  |     1 0          8 7                      0|            |
  |     +-+--------------+------------------------+            |
  +---->|V|     IGN      |         18 bits        |            |
Lookup  +-+--------------+------------------------+            |
New Address              |  Save this value in    |            |
                         |  cache, Set Valid      |            |
                         |                        |            |
                         +-----------------+------+------------+
                                           |
                                           V
                                  Address to memory
```

4.6.3 Example

As noted above, the actual memory space that need be allocated for translation entries can be up to 32,768 longwords. In many cases, less than this may be used. Consider, for example, that the two DMA devices of the system are restricted to maximum transfers of 64 KBytes, with only a single buffer available for each device. This requires that only 104 pages be allocated for each device.

The defined bit positions in MAP_BASE imply that the translation buffer must be aligned on a 104 KByte address boundary, and indeed, if the full 24-bit addressing capability of the DMA devices is to used, this is necessary. However, in the example being considered, we need only to define 104 contiguous entries for each device, the translation table can be offset from a 104 KByte address boundary by suitably modifying the start address supplied to each DMA controller; remember that it is the concatenation of the page address of the DMA controller address and MAP_BASE that forms the address from which the actual page address will be retrieved.

e.g. MAP_BASE = 40000h; SCSI Controller Start address = 1000h

Extracting the page address from the SCSI Controller start address yields 1000 binary. According to the translation algorithm above, this is then shifted two places higher in significance and merged with MAP_BASE. This yields an initial translation table address of 40020h. The 104 translation entries now should be placed at sequential longword addresses 40020:401F0h.

In this way, for transfer lengths of less than that allowed by the full 24-bit addressing of the DMA devices, the translation table can be placed more or less anywhere in memory, indeed the table can consist of two non-contiguous segments, one for the SCSI controller, the other for the NI Controller, simply by supplying the appropriate starting address to each controller.

4.7 Alternate Access to Map Registers

The translation map may be referenced without knowledge of the contents of MAP_BASE via a reserved range of I/O addresses. This address range is 2000.0000 - 2001.FFFF, the 32,768 map entries are referenced as longwords within this address range. Note that although the translation map is being referenced by I/O addresses, which do not normally have parity appended, correct parity will be written into any entry written using this alternate access mode.

4.8 Parity

The CPU generates and checks parity on all operations that reference memory (Address bit<29> = <0>). Parity is added as an additional bit per byte in each 4 or 16 MByte bank of RAM, thus each RAM bank is 36 DRAMs. Refer to Chapter 7, Section 7.5.

Note

As the Floating Point Accelerator unit checks parity on *all* read data that it accepts, care should be taken to ensure that floating point operands are **NEVER** placed in I/O address space.

Chapter 5

ROM MEMORY

The system board ROM contains processor restart, diagnostic and console code and the primary bootstrap program. (The contents of this ROM is detailed in the KA46 System Firmware specification.) Another small ROM is uniquely programmed for each system with its network address.

5.1 System Board ROM

The system board contains two 40-pin sockets for 64K by 16 EPROM chips which collectively hold 256 Kbytes of data. ROM data appears at physical addresses 2004.0000h through 2007.FFFFh. The data path to this ROM is 32 bits wide. Certain physical addresses in the ROM have fixed uses. These are:

2004.0000h	Processor restart address. The processor begins execution at this address in non-mapped mode when a processor restart occurs. (See section 2–15)
2004.0004h	System type register SYS_TYPE. The contents of this longword supplement the internal processor SID register to identify the processor and system type. (See section 2.1.8.2)
2004.0020	Interrupt vector numbers. Eleven consecutive longwords starting at this address are automatically referenced by the hardware to supply the interrupt vector numbers for the interrupt sources connected to the interrupt controller. (See section 6.7)

5.2 Network Address ROM

A 32-byte ROM on the system board contains a unique network address for each system. Data from this ROM is read in the low-order bytes of 32 consecutive longwords at physical addresses 2009.0000 through 2009.007C. The network address occupies the first six bytes (addresses 2009.0000 through 2009.0014). The byte at 2009.0000 is the first byte to be transmitted or received in an address field of an Ethernet packet; its low-order bit (bit 0) is transmitted or received first in the serial bit stream. Digital purchase specification A-PS-23365A1-0-0 describes this ROM in detail and discusses the checksum bytes which follow the six address bytes.

This ROM is installed in a socket so it can be moved in the event that a system's system board is replaced.

Chapter 6

INTERRUPT CONTROLLER

The interrupt controller is a section of the DC7201 and performs two separate functions.

- It receives up to eight interrupt request signals from the system's I/O devices, synchronises and latches them. The latched requests are then masked by individual Enable bits and the results ORed to form a single interrupt, presented to CPU on IRQ1 L, Interrupt Priority Level 15h.
- It recognises CPU Interrupt Acknowledge Cycles for IPL 14h, 15h, 16h and 17h and generates a ROM cycle to retrieve an appropriate vector..

6.1 IPL15h Interrupt

The eight interrupt request sources that result in an interrupt at IPL15h have associated with them three registers :-

INT_REQ	holds the latched interrupt requests received from I/O devices (read-only).
INT_MSK	contains a mask which determines which interrupt requests will generate a processor interrupt (read/write).
INT_CLR	enables a program to selectively reset interrupt request bits in the INT_REQ register (write-only).

The bits in each of these registers are uniformly associated with interrupt sources according to the numbering given in Section 6.3. For example, bit 0 in each register is associated with the SCSI controller interrupt.

The eight latches which comprise the INT_REQ register are edge-triggered: each latch records an interrupt request from an I/O device when the device's interrupt request signal changes from false to true. The contents of the INT_REQ register are ANDed with those of the INT_MSK register, the results ORed to form an interrupt request to the CPU. The resulting interrupt request is presented to the CPU as IRQ1 L.

When the CPU acknowledges an interrupt request, the interrupt controller sends to it the interrupt vector associated with the highest-numbered bit which is set in both the INT_REQ and INT_MSK registers and clears that bit in the INT_REQ register (the INT_MSK bit is not affected). The interrupt vector values are listed in Section 6.7.

Any set bit in the INT_REQ register can also be cleared by a writing a byte to the INT_CLR register which has a one in the corresponding bit position. (This is a transient operation; the data written is not stored and does not prevent the INT_REQ bit from being set in the future). This enables a program to handle a device in non-interrupt mode by clearing the device's INT_MSK bit, polling its requests by reading the INT_REQ register, and clearing its requests by writing to the INT_CLR register.

Since the bits of the INT_REQ register are edge sensitive rather than level sensitive, a program which responds to a request from a device (either in an interrupt service routine or by polling the INT_REQ register) and clears its bit in INT_REQ must do whatever is necessary to return that device's interrupt request signal to its false state in order that a subsequent request from the device will be able to set its bit in INT_REQ again.

Upon power-on, all bits in the INT_REQ and INT_MSK registers are cleared.

6.2 INTACK Cycles

6.2.1 IPL 15h

The DC7201 generates a three bit number from the eight possible interrupt requests, using a fixed priority encoding, shifts this number 2 places to the left to form a number in the range <00:1C>h, then ORs this with the value 2004.0020h and presents the result - a number now in the range <2004.0020:2004.003C>h to the system ROM as a longword aligned address. The data which is retrieved from ROM is returned to the CPU to be added to the SCCB to form the interrupt vector specific to the highest priority interrupt currently active.

6.2.2 IPL14, 16, 17h

For IPL14h, 16h and 17h, a fixed address is presented to the system ROM from which a single interrupt vector for each level is retrieved.

IPL	ROM Address (HEX)
14	2004.0040
16	2004.0048
17	2004.004C

6.3 Interrupt Sources and Ranking

The eight interrupt sources that result in an IPL15h interrupt are listed in the following table. The interrupt numbers 7:0 indicate their bit positions in the INT_xxx registers and their relative priority when more than one request is pending; 7 is the highest priority.

Interrupts 0, 1, 4, 5 and 6 are dedicated to devices on the system board. Interrupts 2, 3 and 7 come from optional devices.

Table 6–1: Interrupt Signal Sources

Num	Name	Source
7	SY	PV21X-DA controller request for service
6	SO	79C30 controller request for service
5	AR	Async. line receiver done or silo full
4	AT	Async. line transmit done
3	G1	Graphics
2	G0	Graphics
1	NI	Network Controller
0	SC	Storage controller

6.4 Interrupt Request Register (INT_REQ)

The interrupt request register is an 8-bit read-only register at physical address 2008.000F each of whose bits reflects the state of the interrupt request latch for one interrupt source. Bits<7:0> correspond to interrupt numbers <7:0> as listed in Section 6.3.

Figure 6–1: Interrupt Request Register (INT_REQ)

```
INT    7       6       5       4       3       2       1       0
   +-------+-------+-------+-------+-------+-------+-------+-------+
   |  SY   |  SO   |  AR   |  AT   |  G1   |  G0   |  NI   |  SC   |
   +-------+-------+-------+-------+-------+-------+-------+-------+
```

A bit in the INT_REQ register is set only by an active transition on the corresponding device’s interrupt request line. The bit will be set by an active transition regardless of the state of the corresponding bit in the interrupt mask register, INT_MSK. However, an interrupt request is sent to the CPU only when the corresponding bits in both INT_REQ and INT_MSK are set.

A bit in the INT_REQ register is cleared either by a program which writes to the INT_CLR register with a one in the corresponding bit position, or by a CPU interrupt acknowledge cycle during which the bit is the highest-numbered bit in INT_REQ which is set and whose corresponding bit in the interrupt mask register INT_MSK is also set.

INT_REQ may be read at any time; reading it does not alter the state of the system in any way.

Upon power-on, the interrupt request register is cleared to zero.

6.5 Interrupt Clear Register (INT_CLR)

The interrupt clear register is an 8-bit write-only register at physical address 2008.000F which is used to selectively clear bits in the interrupt request register INT_REQ. Bits 7:0 correspond to interrupt numbers 7:0 as listed in Section 6.3.

Figure 6–2: Interrupt Clear Register (INT_CLR)

```
INT    7       6       5       4       3       2       1       0
   +-------+-------+-------+-------+-------+-------+-------+-------+
   |  SY   |  SO   |  AR   |  AT   |  G1   |  G0   |  NI   |  SC   |
   +-------+-------+-------+-------+-------+-------+-------+-------+
```

For each bit of INT_CLR which is a one, the corresponding bit of INT_REQ is cleared. For each bit of INT_CLR which is a zero, the corresponding bit of INT_REQ is not changed. The effect of writing to INT_CLR is transient; its contents are not stored and writing to it does not prevent any INTREQ bits from being set in the future.

WARNING

As INT_REQ and INT_CLR share the same address, use of a Read/Modify/Write instruction when accessing the INT_CLR register could cause loss of interrupts.

6.6 Interrupt Mask Register (INT_MSK)

The interrupt mask register is an 8-bit read/write register at physical address 2008.000C each of whose bits is a mask for one interrupt source. Bits 7:0 correspond to interrupt numbers 7:0 as listed in Section 6.3. Each mask bit is ANDed with the corresponding bit of the INT_REQ register before being input to a priority encoder, the output of which determines which bit in INT_REQ wil be cleared (if more than one bit is set) when the CPU executes an Interrupt Acknowledge Cycle.

Figure 6–3: Interrupt Mask Register (INT_MSK)

```
MSK    7       6       5       4       3       2       1       0
   +-------+-------+-------+-------+-------+-------+-------+-------+
   |  SY   |  SO   |  AR   |  AT   |  G1   |  G0   |  NI   |  SC   |
   +-------+-------+-------+-------+-------+-------+-------+-------+
```

Note that a zero in a mask register bit does not prevent the corresponding device from setting its interrupt request register bit. If a request bit is set whose corresponding mask bit is zero, a CPU interrupt is not requested until the mask bit is subsequently set to one (assuming that the request bit has not meanwhile been cleared by writing to INT_CLR). A program which is changing from polled to interrupt servicing of a device should be sure to clear the device's bit in INT_REQ prior to setting its bit in INT_MSK in order to avoid a possible false interrupt signal to the CPU.

Upon power-on, the interrupt mask register is cleared to zero.

6.7 Interrupt Vector Generation

When the CPU acknowledges an interrupt from the interrupt controller, the CDAL Bus Adaptor or for IPL16h or IPL17h, the interrupt controller causes a vector to be placed on the CDAL bus corresponding either to the highest priority pending interrupt if the INTACK cycle is for IPL 15h, or a fixed vector if the cycle is for IPL 14, 16 or 17h. It obtains these vectors from reserved locations in the System Board ROM, Chapter 5, Section 5.1.
The vector presented to the CPU in an INTACK cycle has the following format

Figure 6–4: Interrupt Vector Longword

```
 3                                   1 0                   0 0 0
 1                                   0 9                   2 1 0
+-------------------------------------+-+-------------------+-+-+
|              ...0...                |1|      VNUM         |0|P|
+-------------------------------------+-+-------------------+-+-+
```

<31:10>	Zero.
<09>	Must be one.
<08:02>	VNUM. Interrupt vector number which is multiplied by 4 to form an offset to a vector position in the current SCB. Since only vectors in the range 200h through 3FCh should be used for I/O devices, bits 15:10 are zero and bit 9 is a one.
<01>	Must be zero.
<00>	P. Priority level flag which selects the IPL to which the processor is raised when it acknowledges the interrupt. If this bit is zero, the IPL will be the interrupting IPL; if it is one, the IPL will be 17h. The normal setting is zero.

The conventional vector values established by the system ROM firmware for the eight devices that interrupt at IPL 15h, the CDAL Bus Adaptor and the other two IPLs are as follows (the value in the Vect column represents bits 15:0 of the longword; the value in the P column is then placed in bit 0 of the longword):

Table 6–2: Interrupt Vectors

IPL	No	Name	Vect	P	Source
15	7	SR	02C0	0	PV21X-DA controller request for service
15	6	ST	02C4	0	79C30 request for service
15	5	AR	0250	0	Asynchronous serial line controller receiver done or silo full
15	4	AT	0254	0	Asynchronous serial line controller transmit done
15	3	G1	0244	0	Graphics interrupt 1
15	2	G0	0248	0	Graphics interrupt 0
15	1	NI	03F8	0	Network controller
15	0	SC	03FC	0	Storage controller
14			3F4	0	CDAL Bus Adaptor
16			3F0	0	Unassigned
17			3EC	0	Unassigned

Note

The vectors listed above are subject to change; the System ROM Specification should be consulted for current values.

6.7.1 Passive Release

A special case occurs when the CPU executes an Interrupt Acknowledge Cycle for IPL 15h and no interrupt is pending (INT_REQ<7:0> = 00000000). This case can only occur when for some reason a write has occurred to INT_CLR, with the interrupt system enabled, between the time that the interrupt controller has asserted IRQ1 L and the CPU initiates the Interrupt Acknowledge cycle. For this case, the interrupt controller does not access the System Board ROM, but internally generates a vector of all zeros.

Chapter 7

MISCELLANEOUS I/O REGISTERS

This chapter describes several miscellaneous I/O registers. Their names and physical addresses are listed in the table below.

Table 7–1: Miscellaneous I/O Registers

Name	Address	Function
IORESET	2002.0000	I/O Reset register. Used to generate a reset signal to certain I/O controllers.
CFGTST	2002.0000	Configuration and Test register. Indicates which options are present in a system.
HLTCOD1	2008.0000	Halt Code register 1. Used as a temporary storage location by system firmware during a processor restart.
HLTCOD2	2008.0004	Halt Code register 2. Used as a temporary storage location by system firmware during a processor restart.
BWF0	2008.0014	Miscellaneous Control register 0. This controls many miscellaneous functions for the system.
DIAGDISP	2008.0010	Diagnostic Display register. Controls diagnostic display LEDs on the system board.
DIAGTIMU	2008.001C	High resolution Diagnostic Timer register. A one microsecond time counter.
DIAGTIMM	2008.001E	Low resolution Diagnostic Timer register. A one millisecond time counter.

7.1 IO Reset Register (IORESET)

The IO reset register (IORESET) is a write-only byte register at physical address 2002.0000. Any write access to this register (the data value is ignored) generates a reset signal to the following:

1. Storage Controller, Chapter 10
2. Network Controller, Chapter 9
3. Video option connector, Chapter 15
4. Synchronous Communications Adaptor connector, Chapter 15

The indicated chapters should be consulted for details of the effects of writing to IORESET. Note that the processor, FPA, interrupt controller, and serial line controller are *not* affected by IORESET.

The minimum duration of the reset signal is 700 nsec.

7.2 Configuration and Test Register (CFGTST)

The configuration and test register (CFGTST) is a 16-bit read-only register at physical address 2002.0000.

Figure 7–1: Configuration and Test Register (CFGTST)

```
    15      14      13      12      11      10       9       8
  +------+----------------------------------------------+-------+--
  | RES  |                    MTYPE                     | ALCON |
  +------+----------------------------------------------+-------+-
      7       6       5       4       3       2       1       0
 -+---------------+---------------------------------------------+
  |    OPPRES     |                   MSIZE                     |
--+---------------+---------------------------------------------+
```

<05:00>	MSIZE. Each bit represents the presence or absence of one of the six possible SIMMs that may be added to the machine for additional memory. The six SIMM sockets have fixed physical addresses assigned. The SIMMs should be installed in adjacent positions starting with the socket whose populated bit is read as bit<00> of this register. Each bit that reads a a one indicates the presence of an additional 4 or 16 MBytes of RAM above the base 8 MBytes - see MTYPE below.
<08>	ALCON. Alternate Console. When a special jumper is installed in JNN on the VS4000-400 System Board, allowing serial line 3, the printer port, to be used as the diagnostic console, this bit is reported as a one. When the jumper is not installed, this bit is reported as a zero.

<07:06>	OPPRES. Indicates that Option Modules are plugged into either or both of the video option connector and the communications adaptor connectors. 07 06 0 0 – No Option Modules present. 0 1 – Video Option module present. 1 0 – Communications Option present. 1 1 – Both Video and Communications Options present. ROM Self-test then has the responsibility for identifying and verifying the type and status of the options indicated.
<14:09>	MTYPE. For any bit in the field MSIZE that reads as a ONE, the corresponding bit in this field specifies the type of SIMM installed in that position. A bit that reads as ONE indicates that the SIMM in that position is a 4 MByte SIMM, a bit that reads as a ZERO that that SIMM is a 16 MByte module. The addressing range for each SIMM socket is set by the type of SIMM installed, e.g. if MSIZE = 001111b and MTYPE = 000011b, there are four SIMMs installed, the first two are 4 MBYte, the second two are 16 MByte, giving a machine total memory capacity of 48 Mbytes.
<15>	Reserved. Read as one.

WARNING

The CFGTST register shares its physical address with the IORESET register (section 7.1, above). Programs must take care not to attempt to write to the CFGTST register, since this will generate an IORESET signal.

7.3 Halt Code Registers (HLTCOD1/2)

The halt code registers (HLTCOD1 and HLTCOD2) are a pair of read/write longword registers at physical addresses 2008.0000 and 2008.0004. They are intended for use by the ROM-resident program when handling a processor restart.

Figure 7–2: **Halt Code Registers (HLTCOD1/2)**

```
 3
 1                                                             0
+---------------------------------------------------------------+
|                                                               |
+---------------------------------------------------------------+
```

7.4 Diagnostic Display Register (DIAGDISP)

The diagnostic display register is a 16-bit write-only register at physical address 2008.0010. Its low-order 8 bits control the eight diagnostic display LEDs on the system board. This display is used by the system firmware for diagnostic displays during system test and bootstrap. Upon power-on this register is cleared to zero, which illuminates all the LEDs. The register is not affected by an I/O reset.

Figure 7–3: **Diagnostic Display Register (DIAGDISP)**

```
 1
 5                                       8 7                   0
+-----------------------------------------+---------------------+
|                 reserved                |       LEDDISP       |
+-----------------------------------------+---------------------+
```

<15:8>	Reserved. Must be written as zeros.
LEDDISP	LED Display (bits 7:0). Each bit controls one LED: zero lights a LED and one extinguishes it. Bit 0 corresponds to the rightmost of the eight LEDs when viewed from the front with the cpu module mounted in the system enclosure.

7.5 Miscellaneous Control Register 0 (BWF0)

This register is a read/write register at address 2008.0014. It controls what action is taken by the NI and SCSI controllers when a parity error is detected on a memory read and enables or disables the Invalidate Filter mechanism. It also controls certain function of the graphics address generator.

Figure 7–4: Miscellaneous Control Register 0 (BWF0)

```
 3 3     2 2 2        1 1 1 1 1    1 1 0 0 0                0
 1 0     5 4 3        9 8 7 6 5    1 0 9 8 7                0
+-+-----+-+-+-----------+-+-+-+------+-+-+-+------------------+
|A|     |G|F|           |M|P|P|      |M|P|P|                  |
|D| 0.0 |M|E|   0...0   |A|E|E| 0.0  |A|E|E|      0...0       |
|P|     |D|N|           |P| |N|      |P| |N|                  |
+-+-----+-+-+-----------+-+-+-+------+-+-+-+------------------+
         FILTER           SCSI         NI
```

<31>	ADP. Read ONLY. Reads as a ONE if the Bus Adaptor is present in the system. See Chapter 14.
<30:26>	Reserved. Return UNPREDICTABLE data upon reading. Ignored on write.
<25>	GMD. Controls access by the Graphics Controller to main memory. This bit is normally cleared and graphics drawing operations are allowed unrestricted access to main memory. When this bit is set, graphics drawing operations to main memory are restricted when the bus traffic is high to an extent that could cause the Network Controller to experience data over or underrun conditions. This bit is cleared by Power-on and is unaffected by IORESET.
<24>	FEN, Invalidate Filter Enable. Read/write, if this bit is set, the Invalidate Filter will store Cache tag values on CPU allocates and perform invalidate lookups on DMA writes. If this bit is clear, the invalidate filter logic is totally disabled. See Chapter 4, Section 4.5.
<23:19>	Reserved. Return UNPREDICTABLE data upon reading. Ignored on write.
<18>	SCSI_MAP, MAP Error. Read-only, this bit is set when the SCSI control logic accesses the translation map area of memory and finds an entry with the Valid bit clear. The setting of this bit causes a SCSI Interrupt, see Chapter 6. This bit is cleared by any write to this register - write data ignored - and by power-on.
<17>	SCSI_PE, Parity Error. Read-only, this bit is set when the SCSI memory control section of the DC7201 detects a memory parity error on a read from main memory if bit<16> is also set. This condition causes an SCSI interrupt This bit is cleared by any write to this register - write data ignored - and by power-on.
<16>	SCSI_PEN, SCSI Parity enable. Read/write, when this bit is one, the SCSI control section of the DC7201 checks the parity of data read from main memory during DMA cycles; when the bit is zero, parity is not checked. Cleared to zero upon power-on. See bit<17> above.
<15:11>	Reserved. Return UNPREDICTABLE data upon reading. Ignored on write.
<10>	NI_MAP, MAP Error. Read-only, this bit is set when the NI control logic accesses the translation map area of memory and finds an entry with the Valid bit clear. The setting of this bit causes an NI Interrupt, see chapter 6. This bit is cleared by any write to this register - write data ignored - and by power-on.

<09>	NI_PE, Parity Error. Read-only, this bit is set when the NI memory control section of the DC7201 detects a memory parity error on a read from main memory if bit<08> is also set. This condition causes an NI interrupt by allowing the LANCE chip to time out on the read. The LANCE timeout will cause an NI Interrupt with the MERR bit set in the LANCE status register, see chapter 9, Section 9.3.3. This bit is cleared by any write to this register - write data ignored - and by power-on.
<08>	NI_PEN, NI Parity enable. Read/write, when this bit is one, the Ethernet control section of the DC7201 checks the parity of data read from main memory during DMA cycles; when the bit is zero, parity is not checked. Cleared to zero upon power-on. See bit<09> above.
<07:00>	Reserved. Return UNPREDICTABLE data upon reading. Ignored on write.

7.6 Diagnostic Timer Registers (DIAGTIMU/M)

There are two diagnostic timer registers which may be read as fields of the same longword address - 2008.001C. Not all bits of the longword have meaning, hence the definition of two timers. The High resolution timer (DIAGTIMU) has a resolution of 1 microsecond (see note 1 below), the low resolution timer has a resolution of 1 millisecond (see note 2 below).

The high resolution timer may be read as bits<09:00> of address 2008.001C, i.e. it may be regarded as a 16-bit register at address 2008.001C whose maximum count is 3FFh - equivalent to a time of 1 millisecond (see note 1 below) - wrapping to 000 after this count. This timer's transition from 3FFh to 0 increments the low resolution timer.

The low resolution timer may be read as bits<31:16> of address 2008.001C. It has a maximum value of FFFFh, equivalent to a time of approx. 65 seconds (see note 2 below). It wraps to 000 after this highest count.

Reads of either register are synchronised to the timer count source so that a count cannot occur during a read and so data read will always be correct.

Writes to address 2008.001C affect DIAGTIMU/M according to the type of write. A write to either byte of the low word of this address will clear DIAGTIMU; a write to either of the high bytes will clear DIAGTIMM; a longword write will clear both timers.

Since the count source for these timers is asynchronous to the system clock, the time between clearing either timer with a write instruction and the first increment is uncertain.

DIAGTIMM/U are cleared to zero upon power-on. They are not affected by an I/O reset.

Figure 7–5: Diagnostic Timer Register (DIAGTIMM/U)

```
     3                                                           1
     1                                                           6
    +-------------------------------------------------------------+
    |                real time count (milliseconds)               |
    +-------------------------------------------------------------+ ...

     1            1 0
     5            0 9                                            0
... +--------------+----------------------------------------------+
    | ..  ZERO ..  |    real time count (microseconds )           |
    +--------------+----------------------------------------------+
```

Note 1 : the actual count source for the high resolution timer is 1.017 µs, thus the maximum count will overflow every 1.042 milliseconds.

Note 2 : the resolution of the low resolution timer is 1.042 milliseconds (see note 1 above), thus the maximum count will overflow every 68.26 seconds.

Chapter 8

GRAPHICS CONTROLLER

8.1 Introduction

The graphics controller for the VS4000-400 is a part of the memory control section of the DC7201. It is one of the devices that compete for memory cycles - see chapter 4, Section 4.2 and following for details of the other devices. Graphics operations have the lowest priority of all requests for memory cycles. The graphics controller will be referred to from now on as the GC.

The GC supports both 8 plane and single plane 2D graphics and is optimized to execute the most commonly used primitives of DEC-Windows. Drawing is accomplished with linear addressing. The types of operations supported are: lines; one, two, and three operand rasterops; text. All rasterops can be solid colored, tiled, stippled, color expanded and plane compacted. All operations can be performed to both the frame buffer and non-displayable main system memory using virtual addressing with multiple clipping rectangles for overlapping window hardware support.

The KA46 has no video frame buffer; one of several video option modules can be added to the KA46 system module to provide the frame buffer from which pixels are output to the monitor. These option modules carry video RAMS, pixel timing and output digital to analog conversion/level shifters. The interface to the KA46 is at the "nibble" level, i.e. one quarter of the pixel rate.

8.2 Screen Formats

The GC supports several screen formats, which may or may not be supported by individual video frame buffer modules, see below :

Table 8–1: Supported Screen Formats

Format	Color	Mono	Refresh Rate:	Status
1280x1024	yes	yes	66 Hz	Supported
1024x 768	yes	yes	66 Hz	Supported
800x 600	yes	no	60 Hz	TBD

A 2 plane 64x64 cursor is supported and is stored in off-screen memory - see Section 8.6.

The virtual drawing feature makes use of the operating system paging mechanism to allocate space in main memory as the GC address generator requires it. Hardware support is provided within the DC7201 to track CPU write references to the page table entries used by the GC.

8.3 Graphics Primitives

The following graphics primitives are supported:

— Lines
— 1, 2, or 3 Operand RasterOps with Tiling, Stipples
— Text: Variable Sized Glyphs, Solid, Stenciled
— Fill Spans with Tiling, Stipples for Implementing Polygons
— Light Pixel for Implementing Complex Primitives
— Multiple Clipping Rectangle Support for all Primitives
— All Operands can be Virtual

This section of the VS4000-400 specification will give only an overview of the GC, for complete details, the PVAX2 Graphics Controller Specification should be consulted.

8.4 CPU - GC Communication

Commands are passed from the CPU to the GC in the form of variable length packets. The CPU sends such a packet by writing to an address in I/O space. The write data is passed through the normal write buffers of the DC7201, see Section 4.2.3 in chapter Chapter 4. The GC accepts the command packet data and if it is not currently busy and if the Clip List feature is not enabled, loads the command directly into its registers for execution. If the GC is busy, or if the Clip List feature is enabled, the command packet data is written by the GC into a circular buffer area in main memory - the LCG Command FIFO. The size and location of this FIFO are two of the setup parameters required by the GC.

Figure 8–1: General Format of Command Packets

```
 3       2|2      1|1              0
 1       4|3      6|5              0
+---------+--------+----------------+
| OPCODE  | FLAGS  | OPCODE Specific |
+---------+--------+----------------+
|
| ...  More Longwords, specified by <23:22> in FLAGS above
```

The first longword of any command packet has, within an eight bit flag field, two bits which specify how many additional longwords are contained in this packet.

The function that the GC performs is determined by the OPCODE field of the command packet. This is the most significant byte of the first longword of the packet.

8.5 Virtual Drawing

As the GC may be instructed to draw into main memory and because it is generating addresses which are unknown to the CPU when it sends the command packet, drawing to main memory is done to virtual addresses so that the operating system paging mechansim may be used to validate memory as the GC address generator requires. To control virtual memory operations, three translation buffer pointers (PTEs) are maintained within the GC, one for each of the three data structures the GC may read or write - Source, Mask and Destination. For every CPU write cycle, the address being referenced is compared to each of the PTEs, if a match occurs, the GC is halted and a CPU interrupt occurs - the assumption being the the CPU is accessing the PTE with the intention of clearing the Valid bit.

8.6 Cursor

The cursor data is stored in off-screen memory - main memory only. Cursor data is loaded into the GC from the appropriate memory area for each line of the active cursor region.

Chapter 9

NETWORK CONTROLLER

The network controller enables the connection of a VS4000-400 system to an Ethernet network via a ThinWire connection using RG-58 coax cable or via a transceiver cable. Selection of which of these is active is made by a two position switch acessible from the rear of the machine. One of two LEDs will be lit, indicating the selected connector. The controller is a part of the system board and consists of a LANCE Ethernet controller chip, a Serial Interface Adapter, an Ethernet transceiver chip, a BNC connector for the RG-58 cable to the Ethernet and a 15 pin D-sub connector for the transceiver cable.

9.1 Ethernet Implementation

This option supports the Ethernet Data Link Layer which is specified in DEC Standard 134: *The Digital Ethernet Specification*.

9.1.1 Packet Format

Data is passed over the Ethernet at a serial data rate of 10 million bits per second in variable-length packets. Each packet has the following format:

Figure 9–1: Ethernet Packet Format

```
                +---------------------------+
       6 bytes  |    destination address    |
                +---------------------------+
       6 bytes  |      source address       |
                +---------------------------+
       2 bytes  |           type            |
                +---------------------------+
                |                           |
                |                           |
46..1500 bytes  |           data            |
                |                           |
                |                           |
                +---------------------------+
       4 bytes  |      CRC check code       |
                +---------------------------+
```

The minimum size of a packet is 64 bytes, which implies a minimum data length of 46 bytes. Packets shorter than this are called "runt packets" and are treated as erroneous when received by the network controller.

9.1.2 Network Addresses

Network addresses are 48 bits (6 bytes) long and are of two types:

Physical address: The unique address associated with a particular station on the Ethernet, which should be distinct from the physical address of any other station on any other Ethernet.

Multicast address: A multi-destination address associated with one or more stations on a given Ethernet (sometimes called a logical address). There are two kinds of multicast addresses:

Multicast-group address: An address associated by higher-level convention with a group of logically related stations.

Broadcast address: A predefined multicast address which denotes the set of *all* the stations on the Ethernet.

Bit 0 (the least significant bit of the first byte) of an address denotes the type: it is 0 for physical addresses and 1 for multicast addresses. In either case the remaining 47 bits form the address value. A value of 48 ones is always treated as the broadcast address.

The physical address of each VS4000-400 system is determined at the time of manufacture and is stored in the Ethernet Address ROM on the system board (see section 5.2.

9.2 Lance Chip Overview

The Lance chip is a microprogrammed controller which can conduct extensive operations independently of the central processor. There are four control and status registers (CSRs) within the Lance chip which are programmed by the central processor (i.e. the CVAX CPU chip) to initialize the Lance chip and start its independent operation. Once started, the Lance uses its built-in DMA controller to directly access RAM memory to get additional operating parameters and to manage the buffers it uses to transfer packets to and from the Ethernet. The Lance uses three structures in memory:

1. Initialization Block—24 bytes of contiguous memory starting on a word boundary. The initialization block is set up by the central processor and is read by the Lance when the processor starts the Lance's initialization process. The initialization block contains the system's network address and pointers to the receive and transmit descriptor rings; it is described in section 9.6 below.
2. Descriptor Rings—two logically circular rings of buffer descriptors, one ring used by the chip receiver for incoming data and one ring used by the chip transmitter for outgoing data. Each buffer descriptor in a ring is 8 bytes long and starts on a quadword boundary. It points to a data buffer elsewhere in memory, contains the size of that buffer, and holds various status information about the buffer's contents. Buffer descriptors are described in section 9.7 below.
3. Data Buffers—contiguous portions of memory to buffer incoming or outgoing packets. Data buffers must be at least 64 bytes long (100 bytes for the first buffer of a packet to be transmitted) and may begin on any byte boundary. They are discussed in section 9.7 below.

When the system is ready to begin network operation, the central processor sets up the initialization block, the receive descriptor ring, the transmit descriptor ring, and their data buffers in memory, and then starts the Lance by writing to its CSR's. The Lance performs its initialization process and then enters its polling loop. In this loop, it listens to the network for packets whose destination addresses are of interest and it scans the transmit descriptor ring for descriptors which have been marked by the central processor to indicate that they contain outgoing data packets. When it detects a network packet of interest, it receives and stores that packet in one or more receive buffers and marks their descriptors accordingly. When it finds a packet to be transmitted, it transmits it to the network and marks its descriptor when transmission is complete. Whenever it completes a reception or transmission (or encounters an error condition), the Lance chip sets flags in its control and status register 0 to signal the central processor (usually by an interrupt) that it has done something of interest.

9.3 Program Control of the Lance

Program control of the Lance chip is via two 16-bit read/write ports, each of which appears as the low-order word of a longword address. These ports are:

Address	Name	Description
200E.0000	NI_RDP	Register data port
200E.0004	NI_RAP	Register address port

These ports provide access to four 16-bit control and status registers which are named NI_CSR0 through NI_CSR3. A CSR is accessed by first writing its number into the register address port NI_RAP after which the contents of the CSR are read or written by accesses to the register data port NI_RDP. Note that registers other than NI_CSR0 may be accessed only while the STOP bit of NI_CSR0 is set.

9.3.1 Register Address Port (NI_RAP)

The register address port is a 16-bit read/write port at physical address 200E.0004. It selects which of the four CSR's is accessed via the register data port.

Figure 9–2: Lance Register Address Port(NIRAP)

```
     15      14      13      12      11      10       9       8
   +-------+-------+-------+-------+-------+-------+-------+-------+--
   |                           reserved
   +-------+-------+-------+-------+-------+-------+-------+-------+-

      7       6       5       4       3       2       1       0
  -+-------+-------+-------+-------+-------+-------+-------+-------+
                   reserved                        |     CSRNO     |
 --+-------+-------+-------+-------+-------+-------+-------+-------+
```

<15:2>	Reserved. Ignored on write; read as zeros.
CSRNO	CSR select (bits 1:0). These read/write bits select which of the four CSR's is accessible via the register data port. They are cleared to zero upon power-on. Values are:

```
Bits 1:0     Register
--------     --------
  0 0        NI_CSR0
  0 1        NI_CSR1
  1 0        NI_CSR2
  1 1        NI_CSR3
```

9.3.2 Register Data Port (NI_RDP)

The register data port at physical address 200E.0000 is a 16-bit window through which the CPU can read and write the CSR designated by the register address port NI_RAP.

Note that registers NI_CSR1, NI_CSR2, and NI_CSR3 are accessible only while the STOP bit in NI_CSR0 is set. If that STOP bit is clear (i.e. the Lance chip is active), attempts to read from those CSR's will return UNDEFINED data and attempts to write to them will be ignored. Accesses to a CSR via NI_RDP do not alter the register address pointer NI_RAP. In normal operation only NI_CSR0 can be accessed, so NI_RAP should be set to point to NI_CSR0 and left that way.

9.3.3 Control and Status Register 0 (NI_CSR0)

This register is used by the controlling program to start and stop the operation of the Lance chip and to monitor its status. It is accessible to the processor via port NI_RDP when bits 1:0 of NI_RAP are set to 00. All of its bits can be read at any time and none of its bits is affected by reading the register. The effects of a write operation are described individually for each bit. When power is applied to the system, all the bits in this register are cleared except the STOP bit which is set.

Figure 9–3: Lance Control and Status Register 0 (NICSR0)

```
      15      14      13      12      11      10       9       8
   +-------+-------+-------+-------+-------+-------+-------+-------+--
   |  ERR  |  BABL |  CERR |  MISS |  MERR |  RINT |  TINT |  IDON |
   +-------+-------+-------+-------+-------+-------+-------+-------+-

       7       6       5       4       3       2       1       0
  -+-------+-------+-------+-------+-------+-------+-------+-------+
   |  INTR |  INEA |  RXON |  TXON |  TDMD |  STOP |  STRT |  INIT |
 --+-------+-------+-------+-------+-------+-------+-------+-------+
```

ERR	Error summary (bit 15). This read-only bit is one whenever any of the bits BABL, CERR, MISS, or MERR in this register are ones. Writing to this bit has no effect. It is cleared when all of the bits which set it are zero or when the STOP bit is set.
BABL	Transmitter timeout error (bit 14). This bit is set when the transmitter has been on the channel longer than the time required to send the maximum length packet. It will be set after 1519 data bytes have been transmitted (the chip will continue to transmit until the whole packet is transmitted or until there is a failure before the whole packet is transmitted). This bit is cleared when a one is written to it (writing a zero has no effect) or when the STOP bit is set. When this bit is one, the ERR and INTR bits are also ones.
CERR	Collision error (bit 13). This bit is set when the collision input to the chip failed to activate within 2 microsec after a chip-initiated transmission is completed. This collision-after-transmission is a transceiver test feature. This function is also known as heartbeat or SQE (signal quality error) test. This bit is cleared when a one is written to it (writing a zero has no effect) or when the STOP bit is set. When this bit is one, the ERR bit is also one.
MISS	Missed packet (bit 12). This bit is set when the receiver loses a packet because it does not own a receive buffer. The MISS bit is not valid in internal loopback mode. This bit is cleared when a one is written to it (writing a zero has no effect) or when the STOP bit is set. When this bit is one, the ERR and INTR bits are also ones.
MERR	Memory error (bit 11). This bit is set when the chip attempts a DMA transfer and does not receive a ready response from the memory within 25.6 microsec after beginning the memory cycle. This condition occurs when a parity error occurred on an immediately preceding DMA bus read cycle. When MERR is set, the receiver and transmitter are turned off (bits RXON and TXON of this register are cleared to zero). This bit is cleared when a one is written to it (writing a zero has no effect) or when the STOP bit is set. When this bit is one, the ERR and INTR bits are also ones.
RINT	Receive interrupt (bit 10). This bit is set when the chip updates an entry in the receive descriptor ring for the last buffer received or when reception is stopped due to a failure. This bit is cleared when a one is written to it (writing a zero has no effect) or when the STOP bit is set. When this bit is one, the INTR bit is also one.
TINT	Transmitter interrupt (bit 9). This bit is set when the chip updates an entry in the transmit descriptor ring for the last buffer sent or when transmission is stopped due to a failure. This bit is cleared when a one is written to it (writing a zero has no effect) or when the STOP bit is set. When this bit is one, the INTR bit is also one.

IDON	Initialization done (bit 8). This bit is set when the chip completes the initialization process which was started by setting the INIT bit in this register. When IDON is set, the chip has read the initialization block from memory and stored the new parameters. This bit is cleared when a one is written to it (writing a zero has no effect) or when the STOP bit is set. When this bit is one, the INTR bit is also one.
INTR	Interrupt request (bit 7). This read-only bit is one whenever any of the bits BABL, MISS, MERR, RINT, TINT, or IDON in this register are ones. Writing to this bit has no effect. It is cleared when all of the bits which set it are zero or when the STOP bit is set. When both the INTR and INEA bits in this register are set, an interrupt request is sent to the system interrupt controller (see section 9.4).
INEA	Interrupt enable (bit 6). This read/write bit controls whether the setting of the INTR bit generates an interrupt request. When both the INTR and INEA bits in this register are set, an interrupt request is sent to the system interrupt controller (see section 9.4). This bit is set when a one is written to it. It is cleared when a zero is written to it or when the STOP bit is set.
RXON	Receiver on (bit 5). This read-only bit indicates (when it is one) that the receiver is enabled. RXON is set when initialization is completed (i.e. when IDON is set, unless the DRX bit of the initialization block MODE register was one) and then the STRT bit in this register is set. Writing to this bit has no effect. RXON is cleared when either the MERR or STOP bits of this register are set.
TXON	Transmitter on (bit 4). This read-only bit indicates (when it is one) that the transmitter is enabled. TXON is set when initialization is completed (i.e. when IDON is set, unless the DTX bit of the initialization block MODE register was one) and then the STRT bit in this register is set. Writing to this bit has no effect. TXON is cleared when either the MERR or STOP bits of this register are set or when any of bits UFLO, BUFF, or RTRY in a Transmit Buffer Descriptor (see section 9–14) are set.
TDMD	Transmit demand (bit 3). Setting this bit signals the chip to access the transmit descriptor ring without waiting for the polltime interval to elapse. This bit need not be set to transmit a packet; setting it merely hastens the chip's response to the insertion of a transmit descriptor ring entry by the host program. This bit is set by writing a one to it (writing a zero has no effect) and is cleared by the chip when it recognizes the bit (the bit may read as one for a short time after it is set, depending upon the level of activity in the chip). TDMD is also cleared when the STOP bit is set.
STOP	Stop external activity (bit 2). Setting this bit stops all external activity and clears the internal logic of the chip; this has the same effect as the electrical reset signalled upon power-on. The chip remains inactive and STOP remains set until the STRT or INIT bits in this register are set. This bit is set by writing a one to it (writing a zero has no effect) or upon power-on. It is cleared when either INIT or STRT is set. If the processor writes ones to STOP, INIT, and STRT at the same time, STOP takes precedence and neither STRT nor INIT is set. Setting STOP clears all the other bits in this register. After STOP has been set, the other three CSR's (NI_CSR1, NI_CSR2, and NI_CSR3) must be reloaded before setting INIT or STRT (note that those three registers may be accessed *only* while STOP is set).

STRT	Start operation (bit 1). Setting this bit enables the chip to send and receive packets, perform DMA and do buffer management. The STOP bit must be set prior to setting the STRT bit (setting STRT then clears STOP). STRT is set by writing a one to it (writing a zero has no effect). It is cleared when the STOP bit is set.
INIT	Initialize (bit 0). Setting this bit causes the chip to perform its initialization process, which reads the initialization block from the memory addressed by the contents of NI_CSR1 and NI_CSR2 using DMA accesses. The STOP bit must be set prior to setting the INIT bit (setting INIT then clears STOP). INIT is set by writing a one to it (writing a zero has no effect). It is cleared when the STOP bit is set.

NOTE

The INIT and STRT bits must not be set at the same time. The proper initialization procedure is as follows:

1. set STOP in NI_CSR0
2. set up the initialization block in memory
3. load NI_CSR1 and NI_CSR2 with the starting address of the initialization block
4. set INIT in NI_CSR0
5. wait for IDON in NI_CSR0 to become set
6. set STRT in NI_CSR0 to begin operation

9.3.4 Control and Status Register 1 (NI_CSR1)

This read/write register is used in conjunction with NI_CSR2 to supply the 24-bit physical memory address of the initialization block which the chip reads when it performs its initialization process. The register is accessible to the processor via NI_RDP when bits 1:0 of NI_RAP are 01 and the STOP bit of NI_CSR0 is set. Its contents upon power-on are UNPREDICTABLE.

Figure 9–4: Lance Control and Status Register 1 (NICSR1)

```
      15      14      13      12      11      10       9       8
   +-------+-------+-------+-------+-------+-------+-------+-------+--
   |                           IADR 15:8
   +-------+-------+-------+-------+-------+-------+-------+-------+-

       7       6       5       4       3       2       1       0
  -+-------+-------+-------+-------+-------+-------+-------+-------+
                               IADR 7:1                    |   0   |
 --+-------+-------+-------+-------+-------+-------+-------+-------+
```

IADR	Initialization block address (bits 15:0). These are the low-order sixteen bits of the (24-bit physical) byte address of the first byte of the initialization block. Note that since the block must be word-aligned, bit 0 must be zero.

9.3.5 Control and Status Register 2 (NI_CSR2)

This read/write register is used in conjunction with NI_CSR1 to supply the 24-bit physical memory address of the initialization block which the chip reads when it performs its initialization process. The register is accessible to the processor via NI_RDP when bits 1:0 of NI_RAP are 10 and the STOP bit of NI_CSR0 is set. Its contents upon power-on are UNPREDICTABLE.

Figure 9–5: Lance Control and Status Register 2(NICSR2)

```
      15      14      13      12      11      10       9       8
   +-------+-------+-------+-------+-------+-------+-------+-------+--
   |                           reserved                            |
   +-------+-------+-------+-------+-------+-------+-------+-------+-

       7       6       5       4       3       2       1       0
  -+-------+-------+-------+-------+-------+-------+-------+-------+
   |                           IADR 23:16                          |
 --+-------+-------+-------+-------+-------+-------+-------+-------+
```

<15:8>	Reserved. Write with zeros.
IADR	Initialization block address (bits 7:0). These are the high-order eight bits of the (24-bit physical) byte address of the first byte of the initialization block.

9.3.6 Control and Status Register 3 (NI_CSR3)

This read/write register controls certain aspects of the electrical interface between the Lance chip and the system. It *must* be set as indicated for each bit. The register is accessible to the processor via NI_RDP when bits 1:0 of NI_RAP are 11 and the STOP bit of NI_CSR0 is set. Its contents upon power-on are entirely zeros.

Figure 9–6: Lance Control and Status Register 3(NICSR3)

```
     15      14      13      12      11      10       9       8
   +-------+-------+-------+-------+-------+-------+-------+-------+--
   |                              reserved
   +-------+-------+-------+-------+-------+-------+-------+-------+-

      7       6       5       4       3       2       1       0
  -+-------+-------+-------+-------+-------+-------+-------+-------+
                reserved                   |  BSWP |  ACON |  BCON |
 --+-------+-------+-------+-------+-------+-------+-------+-------+
```

<15:3>	Reserved. Ignored on write; read as zeros.
BSWP	Byte swap (bit 2). When this bit is set, the chip will swap the high and low bytes for DMA data transfers between the silo and bus memory in order to accomodate processors which consider bus bits 15:08 to be the least significant byte of data. This bit is read/write; it is cleared when the STOP bit in NI_CSR0 is set. For this system, this bit *must be ZERO*.
ACON	ALE control (bit 1). This bit controls the polarity of the signal emitted on the chip's ALE/AS pin during DMA operation. This bit is read/write; it is cleared when the STOP bit in NI_CSR0 is set. For this system, this bit *must be ZERO*.
BCON	Byte control (bit 0). This bit controls the configuration of the byte mask and hold signals on the chip's pins during DMA operation. This bit is read/write; it is cleared when the STOP bit in NI_CSR0 is set. For this system, this bit *must be ZERO*.

9.4 Interrupts

The Lance chip asserts an interrupt request signal whenever the INTR and INEA bits bit in its control and status register 0 (NI_CSR0) are both ones. This signal is presented to the system interrupt controller as interrupt number 5, the "network controller primary" source, whose vector number is 250h. The change of the interrupt signal from false to true will set bit NP in the interrupt request register INT_REQ (which will generate a CPU interrupt when the corresponding bit in the interrupt mask register INT_MSK is also set). Note that since the input to INT_REG is transition sensitive rather than level sensitive, a program which services an interrupt request from the Lance must either service all the conditions which contributed to the setting of the INTR bit in NI_CSR0 so that INTR will become zero, or must generate another transition of the interrupt signal by setting the INEA bit of NI_CSR0 to zero and then back to one again. (The interrupt controller is described in chapter 6. Interrupt number 4, the "network controller secondary" source, is not used by this option.)

9.5 DMA Operation

The Lance chip contains a built-in DMA controller which can transfer data directly between the chip and main memory in the address range 0000.0000 through 07FF.FFFF. (In the VS4000-400 only system board RAM and option board RAM appear in this address range.) The LANCE is actually connected to the DC7201 which contains data buffers to minimize the use of the RAM bus by the LANCE. The LANCE contains a 48-byte FIFO buffer to allow for DMA service latency and to minimize the number of request-grant arbitration cycles. When transferring large amounts of data in burst mode, the chip transfers 16 bytes per DMA request. Each longword transfer requires <TBS>, so a 16-byte burst will require either

<TBS> or <TBS> depending upon whether or not the data block is longword aligned. This DMA controller is used to read the initialization block, to read and write the descriptor rings, and to read and write data buffers. Note that all the memory addresses handled by the chip are physical addresses. Programs which operate with CPU memory management enabled must translate their addresses from virtual to physical form before presenting them to the Lance chip.

If the DPEN bit of the PAR_CTL register is set, then parity is checked during DMA read cycles. When a parity error is detected, the MERR bit of the NI_CSR0 register will be set (with the consequences described in section 9.3.3 but no CPU Machine Check will occur.

9.6 Initialization Block

When the Lance chip is initialized (by setting the INIT bit in NI_CSR0), it reads a 24-byte block of data called the initialization block from main memory using DMA accesses. The physical address of the initialization block (IADR) is taken from NI_CSR1 and NI_CSR2. Since the data must be word-aligned, the low-order bit of the address must be zero. The initialization block comprises 12 16-bit words arranged as follows:

Figure 9–7: Lance Initialization Block Layout

```
             +-------------------------------+
IADR +  0    |             MODE              |
             +-------------------------------+
IADR +  2    |         PADR <15:00>          |
             +-------------------------------+
IADR +  4    |         PADR <31:16>          |
             +-------------------------------+
IADR +  6    |         PADR <47:32>          |
             +-------------------------------+
IADR +  8    |        LADRF <15:00>          |
             +-------------------------------+
IADR + 10    |        LADRF <31:16>          |
             +-------------------------------+
IADR + 12    |        LADRF <47:32>          |
             +-------------------------------+
IADR + 14    |        LADRF <63:48>          |
             +-------------------------------+
IADR + 16    |         RDRA <15:00>          |
             +------+------------------------+
IADR + 18    | RLEN |  RDRA <23:16>          |
             +------+------------------------+
IADR + 20    |         TDRA <15:00>          |
             +------+------------------------+
IADR + 22    | TLEN |  TDRA <23:16>          |
             +------+------------------------+
```

9.6.1 Initialization Block MODE Word (NIB_MODE)

The MODE word of the initialization block allows alteration of the Lance chip's normal operation for testing and special applications. For normal operation the MODE word is entirely zero.

Figure 9–8: Initialization Block MODE Word(NIBMODE)

```
     15      14      13      12      11      10       9       8
   +-------+-------+-------+-------+-------+-------+-------+-------+--
   |  PROM |                       reserved
   +-------+-------+-------+-------+-------+-------+-------+-------+-

      7       6       5       4       3       2       1       0
  -+-------+-------+-------+-------+-------+-------+-------+-------+
    resv   |  INTL |  DRTY |  COLL |  DTCR |  LOOP |  DTX  |  DRX  |
 --+-------+-------+-------+-------+-------+-------+-------+-------+
```

PROM	Promiscuous mode (bit 15). When this bit is set, all incoming packets are accepted regardless of their destination addresses.
<14:7>	Reserved. Should be written with zeros.
INTL	Internal loopback (bit 6). This bit is used in conjunction with the LOOP bit in this word to control loopback operation. See the description of the LOOP bit, below.
DRTY	Disable retry (bit 5). When this bit is set, the chip will attempt only one transmission of a packet. If there is a collision on the first transmission attempt, a retry error (RTRY) will be reported in the transmit buffer descriptor (section 9–14).
COLL	Force collision (bit 4). Setting this bit allows the collision logic to be tested. The chip must be in internal loopback mode for COLL to be used. When COLL is one a collision will be forced during the subsequent transmission attempt. This will result in 16 total transmission attempts with a retry error reported in NI_TMD3.
DTCR	Disable transmit CRC (bit 3). When DTCR is zero the transmitter will generate and append a 4-byte CRC to each transmitted packet (normal operation). When DTCR is one the CRC logic is allocated instead to the receiver and no CRC is sent with a transmitted packet. During loopback, setting DTCR to zero will cause a CRC to be generated and sent with the transmitted packet, but no CRC check can be done by the receiver since the CRC logic is shared and cannot both generate and check a CRC at the same time. The CRC transmitted with the packet will be received and written into memory following the data where it can be checked by software. If DTCR is set to one during loopback, the driving software must compute and append a CRC value to the data to be transmitted. The receiver will check this CRC upon reception and report any error.

LOOP	Loopback control (bit 2). Loopback allows the Lance chip to operate in full duplex mode for test purposes. The maximum packet size is limited to 32 data bytes (in addition to which 4 CRC bytes may be appended). During loopback, the runt packet filter is disabled because the maximum packet is forced to be smaller than the minimum size Ethernet packet (64 bytes). Setting LOOP to one allows simultaneous transmission and reception for a packet constrained to fit within the silo. The chip waits until the entire packet is in the silo before beginning serial transmission. The incoming data stream fills the silo from behind as it is being emptied. Moving the received packet out of the silo into memory does not begin until reception has ceased. In loopback mode, transmit data chaining is not possible. Receive data chaining is allowed regardless of the receive buffer length. (In normal operation, the receive buffers must be 64 bytes long, to allow time for buffer lookahead.) Valid loopback bit settings are:

```
LOOP  INTL  Operation
----  ----  ---------
  0    x  Normal on-line operation
  1    0  External loopback
  1    1  Internal loopback
```

Internal loopback allows the chip to receive its own transmitted packet without disturbing the network. The chip will not receive any packets from the network while it is in internal loopback mode. External loopback allows the chip to transmit a packet through the transceiver out to the network cable to check the operability of all circuits and connections between the Lance chip and the network cable. Multicast addressing in external loopback is valid only when DTCR is one (user needs to append the 4 CRC bytes). In external loopback, the chip also receives packets from other nodes.

DTX	Disable transmitter (bit 1). If this bit is set, the chip will not set the TXON bit in NI_CSR0 at the completion of initialization. This will prevent the Lance chip from attempting to access the transmit descriptor ring, hence no transmissions will be attempted.
DRX	Disable receiver (bit 0). If this bit is set, the chip will not set the RXON bit in NI_CSR0 at the completion of initialization. This will cause the chip to reject all incoming packets and to not attempt to access the receive descriptor ring.

9.6.2 Network Physical Address (NIB_PADR)

The 48-bit physical Ethernet network node address is contained in bytes 2:7 of the initialization block. (This is a network address; it has no relationship to any memory address.)

Figure 9–9: Network Physical Address(NIBPADR)

```
|<---IADR+6--->|<---IADR+4--->|<---IADR+2--->|

 47          32 31          16 15           0
+-------------+-------------+-------------+
|                                        |0|
+-------------+-------------+-------------+
```

The contents of NIB_PADR identify this station to the network and must be unique within the domain of the network. Its value is normally taken from the Network Address ROM (see section 5.2. The low-order bit (bit 0) of this address must be zero since it is a physical address.

9.6.3 Multicast Address Filter Mask (NIB_LADRF)

Bytes 8:15 of the initialization block contain the 64-bit multicast address filter mask. The multicast address filter is a partial filter which assists the network controller driver program to selectively receive packets which contain multicast network addresses.

Figure 9–10: Multicast Address Filter Mask(NIB LADRF)

```
|<---IADR+14--->|<---IADR+12--->|<---IADR+10--->|<---IADR+8---->|

 63           48 47           32 31           16 15            0
+--------------+--------------+--------------+--------------+
|                                                            |
+--------------+--------------+--------------+--------------+
```

Multicast Ethernet addresses are distinguished from physical network addresses by the presence of a one in bit 0 of the 48-bit address field. If an incoming packet contains a physical destination address (bit 0 is zero), then its entire 48 bits are compared with the contents of NIB_PADR and the packet is ignored if they are not equal. If the packet contains a multicast destination address which is all ones (the broadcast address), it is always accepted and stored regardless of the contents of the multicast address filter mask.

All other multicast addresses are processed through the multicast address filter to determine whether the incoming packet will be stored in a receive buffer. This filtering is performed by passing the multicast address field through the CRC generator. The high-order 6 bits of the resulting 32-bit CRC are used to select one of the 64 bits of NIB_LADRF. (These high-order six bits represent in binary the number of the bit in NIB_LADRF, according to the labelling in figure 9–10.) If the bit selected from NIB_LADRF is one, the packet is stored in a receive buffer; otherwise it is ignored. This mechanism effectively splits the entire domain of 2**47 multicast addresses into 64 parts, and multicast addresses falling into each part will be accepted or ignored according to the value of the corresponding bit in NIB_LADRF. The driver program must examine the addresses of the packets accepted by this partial filtering to complete the filtering task.

9.6.4 Receive Descriptor Ring Pointer (NIB_RDRP)

Bytes 16:19 of the initialization block describe the starting address and extent of the receive descriptor ring.

Figure 9–11: Receive Descriptor Ring Pointer(NIBRDRP)

```
|<-------IADR+18------->|<-------IADR+16------->|

 31  29 28  24 23     16 15                    0
+------+------+---------+----------------------+
| RLEN | resv |             RDRA            000|
+------+------+---------+----------------------+
```

RLEN Receive ring length (bits 31:29). This field gives the number of entries in the receive descriptor ring, expressed as a power of 2:

RLEN	*Entries*
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

<28:24> Reserved; should be zeros.

RDRA Receive descriptor ring address (bits 23:0). This is the physical address in system memory of the first element in the ring. Since each 8-byte element must be aligned on a quadword boundary, bits 2:0 of this address must be zero.

9.6.5 Transmit Descriptor Ring Pointer (NIB_TDRP)

Bytes 20:23 of the initialization block describe the starting address and extent of the Transmit descriptor ring.

Figure 9–12: Transmit Descriptor Ring Pointer(NIBTDRP)

```
|<-------IADR+22------->|<-------IADR+20------->|

 31  29 28  24 23     16 15                    0
+------+------+---------+----------------------+
| TLEN | resv |            TDRA             000|
+------+------+---------+----------------------+
```

TLEN Transmit ring length (bits 31:29). This field gives the number of entries in the transmit descriptor ring, expressed as a power of 2:

TLEN	*Entries*
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

<28:24> Reserved; should be zeros.

TDRA Transmit descriptor ring address (bits 23:0). This is the physical address in system memory of the first element in the ring. Since each 8-byte element must be aligned on a quadword boundary, bits 2:0 of this address must be zero.

9.7 Buffer Management

The Lance chip manages its data buffers by using two rings of buffer descriptors which are stored in memory: the receive descriptor ring and the transmit descriptor ring. Each buffer descriptor points to a data buffer elsewhere in memory, contains the size of that buffer, and contains status information about that buffer's contents.

The starting location in memory of each ring and the number of descriptors in it are given to the Lance chip via the initialization block (see sections 9–11 and 9–12) during the chip initialization process. Each descriptor is 8 bytes long and must be aligned on a quadword boundary (the three low-order bits of its address must be zero). The descriptors in a ring are physically contiguous in memory and the number of descriptors must be a power of 2. The Lance keeps an internal index to its current position in each ring which it increments modulo the number of descriptors in the ring as it advances around each ring.

Once started, the Lance polls each ring to find descriptors for buffers in which to receive incoming packets and from which to transmit outgoing packets, and revises the status information in buffer descriptors as it processes their associated buffers. When polling, the Lance is limited to looking only one ahead of the descriptor with which it is currently working. The high speed of the data stream requires that each buffer be at least 64 bytes long to allow time to chain buffers for packets which are larger than one buffer. (The first buffer of a packet to be transmitted should be at least 100 bytes to avoid problems in case a late collision is detected.)

Each descriptor in a ring is "owned" either by the Lance chip or by the host processor; this status is indicated by the OWN bit in each descriptor. Mutual exclusion is accomplished by the rule that each device can only relinquish ownership of a descriptor to the other device, it can never take ownership; and that each device cannot change any field in a descriptor or its associated buffer after it has relinquished ownership. When the host processor sets up the rings of descriptors before starting the Lance, it sets the OWN bits such that the Lance will own all the descriptors in the receive descriptor ring (to be used by the Lance to receive packets from the network) and the host will own all the descriptors in the transmit descriptor ring (to be used by the host to set up packets to be transmitted to the network).

9.7.1 Receive Buffer Descriptor

A receive buffer descriptor comprises 4 words aligned in memory on a quadword address boundary.

Figure 9–13: Receive Buffer Descriptor

```
 Memory
 Offset
          1 1 1 1 1 1
          5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0
         +---------------+---------------+
         |                               |
  +0:    |             LADR              |
         |                               |
         +---------------+---------------+
         |O E F O C B S E|               |
         |W R R F R U T N|               |
  +2:    |N R A L C F P P|     HADR      |
         |    M O   F    |               |
         +---------------+---------------+
         |       |                       |
  +4:    |1 1 1 1|         BCNT          |
         |       |                       |
         +---------------+---------------+
         |       |                       |
  +6:    |0 0 0 0|         MCNT          |
         |       |                       |
         +---------------+---------------+
```

LADR	Low-order buffer address (offset 0, bits 15:0). These are the low-order 16 bits of the 24-bit physical memory address of the start of the buffer associated with this descriptor. Written by the host; unchanged by the Lance.
HADR	High-order buffer address (offset 2, bits 7:0). These are the high-order 8 bits of the 24-bit physical memory address of the start of the buffer associated with this descriptor. Written by the host; unchanged by the Lance.

OWN	Owned flag (offset 2, bit 15). This bit indicates whether the descriptor is owned by the host (OWN = 0) or by the Lance (OWN = 1). The Lance clears OWN after filling the buffer associated with the descriptor with an incoming packet. The host sets OWN after emptying the buffer. In each case, this must be the last bit changed by the current owner, since changing OWN passes ownership to the other party and the relinquishing party must not thereafter alter anything in the descriptor or its buffer.
ERR	Error summary (offset 2, bit 14). This is the logical OR of the FRAM, OFLO, CRC and BUFF bits in this word. Set by the Lance and cleared by the host.
FRAM	Framing error (offset 2, bit 13). This bit is set by the Lance to indicate that the incoming packet stored in the buffer had both a non-integral multiple of eight bits and a CRC error. It is cleared by the host.
OFLO	Overflow error (offset 2, bit 12). This bit is set by the Lance to indicate that the receiver has lost part or all of an incoming packet because it could not store it in the buffer before the chip's silo overflowed. Cleared by the host.
CRC	Checksum error (offset 2, bit 11). This bit is set by the Lance to indicate that the received packet has an invalid CRC checksum. Cleared by the host.
BUFF	Buffer error (offset 2, bit 10). This bit is set by the Lance when it has used all its owned receive descriptors or when it could not get the next descriptor in time while attempting to chain to a new buffer in the midst of a packet. When a buffer error occurs, an overflow error (bit OFLO) also occurs because the Lance continues to attempt to get the next buffer until its silo overflows. BUFF is cleared by the host.
STP	Start of packet (offset 2, bit 9). This bit is set by the Lance to indicate that this is the first buffer used for this packet. Cleared by the host.
ENP	End of packet (offset 2, bit 8). This bit is set by the Lance to indicate that this is the last buffer used for this packet. When both STP and ENP are set in a descriptor, its buffer contains an entire packet; otherwise two or more buffers have been chained together to hold the packet. ENP is cleared by the host.
1111	Offset 4, bits 15:12 must be set by the host to ones. Unchanged by the Lance.
BCNT	Buffer size (offset 4, bits 11:0). This is the number of bytes in the buffer (whose starting address is in HADR and LADR) in two's complement form. Note that the minimum buffer size is 64 bytes and that the maximum required for a legal packet is 1518 bytes. Written by the host; unchanged by the Lance.
0000	Offset 6, bits 15:12 are reserved; they should be set to zeros by the host when it constructs the descriptor.
MCNT	Byte count (offset 6, bits 11:0). This is the length in bytes of the received packet for which this is the last or only descriptor. MCNT is valid only in a descriptor in which ENP is set (last buffer) and ERR is clear (no error). Set by the Lance and cleared by the host.

9.7.2 Transmit Buffer Descriptor

A transmit buffer descriptor comprises 4 words aligned in memory on a quadword address boundary.

Figure 9–14: Transmit Buffer Descriptor

```
Memory
Offset
         1 1 1 1 1 1
         5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0
        +---------------+---------------+
        |                               |
 +0:    |             LADR              |
        |                               |
        +---------------+---------------+
        |O E r M O D S E|               |
        |W R e O N E T N|               |
 +2:    |N R s R E F P P|     HADR      |
        |    v E        |               |
        +---------------+---------------+
        |       |                       |
 +4:    |1 1 1 1|         BCNT          |
        |       |                       |
        +---------------+---------------+
        |B U r L L R|                   |
 +6:    |U F e C C T|                   |
        |F L s O A R|       TDR         |
        |F O v L R Y|                   |
        +---------------+---------------+
```

LADR	Low-order buffer address (offset 0, bits 15:0). These are the low-order 16 bits of the 24-bit physical memory address of the start of the buffer associated with this descriptor. Written by the host; unchanged by the Lance.
HADR	High-order buffer address (offset 2, bits 7:0). These are the high-order 8 bits of the 24-bit physical memory address of the start of the buffer associated with this descriptor. Written by the host; unchanged by the Lance.
OWN	Owned flag (offset 2, bit 15). This bit indicates whether the descriptor is owned by the host (OWN = 0) or by the Lance (OWN = 1). The host sets OWN after filling the buffer with a packet to be transmitted. The Lance clears OWN after transmitting the contents of the buffer. In each case, this must be the last bit changed by the current owner, since changing OWN passes ownership to the other party and the relinquishing party must not thereafter alter anything in the descriptor or its buffer.
ERR	Error summary (offset 2, bit 14). This is the logical OR of the LCOL, LCAR, UFLO and RTRY bits in this descriptor. Set by the Lance and cleared by the host.
resv	Offset 2, bit 13 is reserved. The Lance will write a zero in this bit.
MORE	More retries (offset 2, bit 12). The Lance sets this bit when more than one retry was required to transmit the packet. Cleared by the host.

ONE	One retry (offset 2, bit 11). The Lance sets this bit when exactly one retry was required to transmit the packet. Cleared by the host.
DEF	Deferred (offset 2, bit 10). The Lance sets this bit when it had to defer while trying to transmit the packet. This occurs when the network is busy when the Lance is ready to transmit. Cleared by the host.
STP	Start of packet (offset 2, bit 9). This bit is set by the host to indicate that this is the first buffer used for this packet. STP is not changed by the Lance.
ENP	End of packet (offset 2, bit 8). This bit is set by the host to indicate that this is the last buffer used for this packet. When both STP and ENP are set in a descriptor, its buffer contains an entire packet; otherwise two or more buffers have been chained together to hold the packet. ENP is not changed by the Lance.
1111	Offset 4, bits 15:12 must be set by the host to ones. Unchanged by the Lance.
BCNT	Byte count (offset 4, bits 11:0). This is the number of bytes, in two's complement form, which the Lance will transmit from this buffer. Note that for any buffer which is not the last of a packet, at least 64 bytes (100 bytes if it is the start of the packet) must be transmitted to allow adequate time for the Lance to acquire the next buffer. Written by the host; unchanged by the Lance.
	NOTE: The remaining fields of the descriptor (which make up its entire fourth word) are valid only when the ERR bit in the second word has been set by the Lance.
BUFF	Buffer error (offset 6, bit 15). This bit is set by the Lance during transmission when it does not find the ENP bit set in the current descriptor and it does not own the next descriptor. When BUFF is set, the UFLO bit (below) is also set because the Lance continues to transmit until its silo becomes empty. BUFF is cleared by the host.
UFLO	Underflow (offset 6, bit 14). This bit is set by the Lance when it truncates a packet being transmitted because it has drained its silo before it was able to obtain additional data from a buffer in memory. UFLO is cleared by the host.
RESV	Offset 6, bit 13 is reserved. The Lance will write a zero in this bit.
LCOL	Late collision (offset 6, bit 12). This bit is set by the Lance to indicate that a collision has occured after the slot time of the network channel has elapsed. The Lance does not retry after a late collision. LCOL is cleared by the host.
LCAR	Loss of carrier (offset 6, bit 11). This bit is set by the Lance when the carrier-present input to the chip becomes false during a transmission initiated by the Lance. The Lance does not retry after such a failure. LCAR is cleared by the host.
RTRY	Retries exhausted (offset 6, bit 10). This bit is set by the Lance after 16 attempts to transmit a packet have failed due to repeated collisions on the network. (If the DRTY bit of the initialization block MODE word (section 9–8) is set, RTRY will instead be set after only only one failed transmission attempt.) RTRY is cleared by the host.
TDR	Time domain reflectometer (offset 6, bits 9:0). These bits are the value of an internal counter which is set by the Lance to count system clocks from the start of a transmission to the occurrence of a collision. This value is useful in determing the approximate distance to a cable fault; it is valid only when the RTRY bit in this word is set.

9.8 Lance Operation

The Lance chip operates independently of the host under control of its own internal microprogram. This section is a simplified description of the operation of the Lance in terms of its principal microcode routines (these should not be confused with device driver programming in the host, which is not a part of this specification). These microcode routines make use of numerous temporary storage cells within the Lance chip; most of these are not accessible from outside the chip but they are mentioned here when necessary to clarify the operation of the microcode.

Two such (conceptual) internal variables are of central importance: the pointers to the "current" entry in the receive descriptor ring and in the transmit descriptor ring, which are referred to below as TXP and RXP. Each of these designates the descriptor which the Lance will use for the next operation of that type. If the descriptor designated by one of these pointers is not owned by the Lance (the OWN bit is 0), then the Lance can neither perform activity of that type nor advance the pointer. For the transmit ring, the Lance will do nothing until the host sets up a packet in the buffer and sets the OWN bit in the descriptor designated by the Lance's TXP. (The host must keep track of the position of the TXP, since setting up a packet in some other descriptor will not be detected by the Lance). For the receive ring, if the Lance does not own the descriptor designated by RXP, it cannot receive a packet. In both rings, when the Lance finishes with a descriptor and relinquishes it to the host by clearing OWN, it then advances the ring pointer (modulo the number of entries in the ring).

When the Lance begins activity using the current descriptor (i.e. begins receiving or transmitting a packet), it may look ahead at the next descriptor and attempt to read its first three words in advance so it can chain to the next buffer in mid-packet without losing data. However, it does not actually advance its RXP or TXP until it has cleared the OWN bit in the current descriptor.

The Lance is a very complex chip and this system specification does not attempt to cover all the details of its operation. The chip purchase specification cited in section 1.1 and the chip vendor's literature should also be consulted.

9.8.1 Switch Routine

Upon power on, the STOP bit is set and the INIT and STRT bits are cleared in NI_CSR0. The Lance microprogram begins execution in the switch routine, which tests the INIT, STRT, and STOP bits. When the host sets either INIT or STRT, STOP is cleared. While STOP is set, if the host writes to NI_CSR1 and NI_CSR2, that data is stored for use by the initialization routine.

When the microprogram sees STOP cleared, it tests first the INIT bit and then the STRT bit. If INIT is set, it performs the initialization routine (section 9.8.2). Then if STRT is set, it begins active chip operation by jumping to the look-for-work routine (section 9.8.3). Control returns to the switch routine whenever the host again sets the STOP bit (which also clears the INIT and STRT bits). Note that the ring pointers RXP and TXP are not altered by the setting of either STOP or START; they are reset to the start of their rings only when INIT is set.

9.8.2 Initialization Routine

The initialization routine is called from the switch routine when the latter finds the INIT bit set. It reads the initialization block (section 9.6) from the memory addressed by NI_CSR1 and NI_CSR2 and stores its data within the Lance chip. This routine also sets the ring pointers RXP and TXP to the start of their rings (i.e. to point to the descriptor at the lowest memory address in the ring).

9.8.3 Look-for-work Routine

The look-for-work routine is executed while the Lance is active and looking for work. It is entered from the switch routine when the STRT bit is set, and is returned to from the receive and transmit routines after they have received or transmitted a packet.

This routine begins by testing whether the receiver is enabled (bit RXON of NI_CSR0 is set). If so, it tries to have a receive buffer available for immediate use when a packet addressed to this system arrives. It tests its internal registers to see whether it has already found a receive descriptor owned by the Lance and, if not, calls the receive poll routine (section 9.8.4) to attempt to get a receive buffer.

Next the routine tests whether the transmitter is enabled (bit TXON of NI_CSR0 is set). If so, it calls the transmit poll routine (section 9.8.7) to see whether there is a packet to be transmitted and to transmit it if so.

If there was no transmission and the TDMD bit of NI_CSR0 is not set, the microprogram delays 1.6 milliseconds and then goes to check the receive descriptor status again. If a packet was transmitted or the host has set TDMD, the delay is omitted so that multiple packets will be transmitted as quickly as possible.

If at any point in this routine the receiver detects an incoming packet whose destination address matches the station's physical address, is the broadcast address, or passes the multicast address filter (or if the PROM bit of NIB_MODE is set), the receive routine (section 9.8.5) is called.

9.8.4 Receive Poll Routine

The receive poll routine is called whenever the receiver is enabled and the Lance needs a free buffer from the receive descriptor ring. The routine reads the second word of the descriptor designated by RXP and, if the OWN bit in it is set, reads the first and third words also.

9.8.5 Receive Routine

The receive routine is called when the receiver is enabled and an incoming packet's destination address field matches one of the criteria described in 9.8.3 above. The routine has three sections: initialization, lookahead, and descriptor update.

In initialization, the routine checks whether a receive ring descriptor has already been acquired by the receive poll routine. If not, it makes one attempt to get the descriptor designated by RXP (if OWN is not set in it, MISS and ERR are set in NI_CSR0 and the packet is lost). The buffer thus acquired is used by the receive DMA routine to empty the silo.

In lookahead, the routine reads the second word of the next descriptor in the receive ring and, if the OWN bit is set, reads the rest of the descriptor and holds it in readiness for possible data chaining.

The descriptor update section is performed when either the current buffer is filled or the packet ends. If the packet ends but its total length is less than 64 bytes, it is an erroneous "runt packet" and is ignored: no status is posted in the descriptor, RXP is not moved, and the buffer will be reused for the next incoming packet (this is why a receive buffer must be at least 64 bytes long; otherwise the runt might be detected after advancing RXP).

If the packet ends (with or without error), the routine writes the packet length into MCNT, sets ENP and other appropriate status bits and clears OWN in the current descriptor, and sets RINT in NI_CSR0 to signal the host that a complete packet has been received. Then it advances RXP and returns to the look-for-work routine.

If the buffer is full and the packet has not ended, chaining is required. The routine releases the current buffer by writing status bits into its descriptor (clearing OWN and ENP, in particular), makes current the next descriptor data acquired in the lookahead section, advances RXP, and goes to the lookahead section to prepare for possible additional chaining. Note that RINT is not set in NI_CSR0, although the host would find OWN cleared if it looked at the descriptor, and it could begin work on that section of the packet, since the mutual exclusion rule prevents the Lance from going back and altering it.

9.8.6 Receive DMA Routine

The receive DMA routine is invoked asynchronously by the chip hardware during execution of the receive routine whenever the silo contains 16 or more bytes of incoming data or when the packet ends and the silo is not empty. It executes DMA cycles to drain data from the silo into the buffer designated by the current descriptor.

9.8.7 Transmit Poll Routine

The transmit poll routine is called by the look-for-work routine (section 9.8.3) to see whether a packet is ready for transmission. It reads the second word of the descriptor designated by TXP and tests the OWN bit. If OWN is zero, the Lance does not own the buffer and this routine returns to its caller. If OWN is set, the routine tests the STP bit, which should be set to indicate the start of a packet. If STP is clear, this is an invalid packet; the Lance sets its OWN bit to return it to the host, sets TINT in NI_CSR0 to notify the host, and advances TXP to the next transmit descriptor. If both OWN and STP are set, this is the beginning of a packet, so the transmit poll routine reads the rest of the descriptor and then calls the transmit routine (section 9.8.8) to transmit the packet. During this time the chip is still watching for incoming packets from the network and it will abort the transmit operation if one arrives.

9.8.8 Transmit Routine

The transmit routine is called from the transmit poll routine (section 9.8.7) when the latter finds the start of a packet to be transmitted. This routine has three sections: initialization, lookahead, and descriptor update. In initialization, the routine sets the chip's internal buffer address and byte count from the transmit descriptor, enables the transmit DMA engine, and starts transmission of the packet preamble. It then waits until the transmitter is actually sending the bit stream (including possible backoff-and-retry actions in case of collisions).

In lookahead, the transmit routine test the current descriptor to see whether it is the last in the packet (the ENP bit is set). If so, no additional buffer is required so the routine waits until all the bytes from the current packet have been transmitted. If not, the routine attempts to get the next descriptor and hold it in readiness for data chaining, and then waits until all the bytes from the current buffer have been transmitted.

Descriptor update is entered when all the bytes from a buffer have been transmitted or an error has occured. If there is no error and the buffer was not the last of the packet, the pre-fetched descriptor for the next buffer is made current for use by the transmit DMA routine. The routine writes the appropriate status bits and clears the OWN bits in the current descriptor and advances TXP. If this was the last buffer in the packet, the routine sets the TINT bit in NI_CSR0 to notify the host and returns to the look-for-work routine (section 9.8.3); otherwise it goes back to the lookahead section in this routine.

9.8.9 Transmit DMA Routine

The transmit DMA routine is invoked asynchronously by the chip hardware during execution of the transmit routine whenever the silo has 16 or more empty bytes. It executes DMA cycles to fill the silo with data from the buffer designated by the current descriptor.

9.8.10 Collision Detect Routine

This routine is invoked asynchronously by the chip hardware during execution of the transmit routine when a collision is detected on the network. It ensures that the "jam" sequence is transmitted, then backs up the chip's internal buffer address and byte count registers, waits for a pseudo-random backoff time, and then attempts the transmission again. If 15 retransmission attempts fail (a total of 16 attempts), it sends the microcode to the descriptor update routine to report an error in the current transmit descriptor (bits RTRY and ERR are set).

9.9 Lance Programming Notes

1. The interrupt signal is simply the OR of the interrupt-causing conditions. If another such condition occurs while the interrupt signal is already asserted, there will not be another active transition of the interrupt signal and the interrupt request bit in INT_REQ will not be set again. An interrupt service routine should use logic similar to the following to avoid losing interrupts:

   i Read NI_CSR0 and save the results in a register, say R0.

   ii Clear the interrupt enable bit INEA in the saved data in R0.

   iii Write NI_CSR0 with the saved data in R0. This will make the interrupt signal false because INEA is clear and will clear all the write-one-to-reset bits such as RINT, TINT and the error bits; it will not alter the STRT, INIT or STOP bits nor any interrupt-cause bits which came true after NI_CSR0 was read.

   iv Write NI_CSR0 with only INEA to enable interrupts again.

   v Service all the interrupt and error conditions indicated by the flags in the data in R0.

vi Exit from the interrupt service routine.

Note

Be sure to access NI_CSR0 only with instructions which do a single access, such as MOVE. Instructions such as BIS which do a read-modify-write operation can have unintended side effects.

2. An interrupt is signalled to the host only when the *last* buffer of a multibuffer (chained) packet is received or transmitted. However, the OWN bit in each descriptor is cleared as soon as the Lance has finished with that portion of the packet, and the mutual exclusion rule makes it safe for the host to process such a descriptor and its buffer.
3. When a transmitter underflow occurs (UFLO is set in a transmit descriptor because the silo is not filled fast enough), the Lance will turn off its transmitter and the Lance must be restarted to turn the transmitter back on again. This can be done by setting STOP in NI_CSR0 and then setting STRT in NI_CSR0 (DTX will still be clear in the chip's internal copy of NIB_MODE). It is necessary to set INIT to reread the initialization block.

Note

Note that setting STOP will immediately terminate any reception which is in progress. If the status of a receive descriptor has been updated and its OWN bit is now clear, then the contents of its buffer are valid. If the incoming packet was chained into more than one buffer, however, the packet is only valid if its last buffer has been completed (the one with the ENP bit set).

4. The network controller hardware requires up to five seconds after power on to become stable. Self-test routines must delay at least this time before attempting to use the controller for either internal or external testing.
5. The LCAR flag (loss of carrier) may be set in the transmit descriptor when a packet is sent in internal loopback mode. When the Lance is operating in internal loopback mode and a transmission is attempted with a non-matching address, the Lance will correctly reject that packet. If the next operation is an internal loopback transmission without first resetting the Lance, the packet will not be sent and LCAR will be set in the transmit descriptor for that packet. The receive descriptor will still be owned by the Lance. To avoid this problem, the Lance should be reinitialized after each internal loopback packet.
6. The ONE flag is occasionally set in a transmit descriptor after a late collision. The Lance does not attempt a retransmission even though ONE may be set. The host should disregard ONE if the LCOL flag is also set.
7. The chip's internal copy of NI_CSR1 may become invalid when the chip is stopped. The NI_CSR1 and NI_CSR2 registers should always be loaded prior to setting INIT to initialize the Lance chip.
8. Attempting an external loopback test on a busy network can cause a silo pointer misalignment if a transmit abort occurs while the chip was preparing to transmit the loopback packet. The resulting retransmission may cause the transmitter enable circuit to hang, and the resulting illegal length transmission must be terminated by the jabber timer in the transceiver. It is unlikely that there will be a corrupted receive buffer because the reception that caused the transmit abort will usually not pass address recognition.

Since external loopback is a controlled situation it is possible to implement a software procedure to detect a silo pointer misalignment problem and prevent continuous transmissions. Since the test is being done in loopback the exact length and contents of the receive packet are known; thus the software can determine whether the data in the receive buffer has been corrupted.

On transmission the diagnostic software should allow up to 32 retries before a hard error is flagged. This is not to say that 32 errors are allowed for each condition; the sum of all errors encountered in the test should not exceed 32. The diagnostic software should expect to get a transmit done interrupt with 1 millisecond of passing the transmit packet to the Lance. If this does not occur, it should reset the Lance and retry the test. This prevents a continuous transmission (babble) longer than the longest legal packet in case the Lance has become hung.

9. When the chip is in internal loopback mode and a CRC error is forced, a framing error will also be indicated along with the CRC error. In external loopback, when a CRC error is forced only that error is indicated; a framing error is indicated only if the Lance actually receives extra bits.

10. When transmit data chaining, a BUFF error will be set in the current transmit descriptor if a late collision or retry error occurred while the Lance was still transmitting data from the previous buffer. The BUFF error in this case is an invalid error indication and should be ignored. BUFF is valid only when UFLO is also set.

11. When the host program sets up a packet for transmission in chained buffers, it should set the OWN bits in all the transmit buffers *except the first one* (i.e. the one containing the STP bit), and then as its last act set the OWN bit in the first descriptor. Once that bit is set, the Lance will start packet transmission and may encounter an underflow error if the subsequent descriptors for the packet are not available.

12. Do not set INIT and STRT in NI_CSR0 at the same time. After stopping the chip, first set INIT and wait for IDON, then set STRT. If both are set at once, corrupt transmit or receive packets can be generated if RENA becomes true during the initialization process.

13. When a missed packet error occurs, the Lance chip must be halted using the stop bit and the re-loaded with it's initialization block. The stop bit must be set within 75 microseconds of the missed packet error or it may result in a silo pointer misalignment, which can cause transmission of the next packet to be bad.

Chapter 10

SCSI CONTROLLER

This chapter describes the SCSI controller portion of the VS4000-400 system board.

The controller conforms to the ANSI Small Computer System Interface (SCSI) specification. It has a single port, connecting both to devices within the VS4000-400 system box and allowing for expansion externally.

10.1 SCSI Overview

The SCSI electrical and logical interface and operation is described in detail in the ANSI draft standard issued by ANSI task group X3T9.2, and the particular subset of that standard used by the devices attached to the VS4000-400 system board are described in each device's specification. The programmer must use all of those documents in conjunction with this specification as a guide. This section reviews a few important features of the ANSI document to set the context for the following discussion of the VS4000-400 implementation of the SCSI interface.

The SCSI interface is a single-ended bi-directional 8-bit-wide bus to which up to eight devices can be attached. The KA46 System Module itself is one of those devices, so up to seven additional devices can be attached. Devices may play one of two roles: initiator or target. An initiator originates an operation by sending a command to a specific target. A target performs an operation which was requested by an initiator. In this specification it is assumed that the VS4000-400 is always an initiator and that all other SCSI devices attached to it are targets. (There is, however, no *hardware* feature of the VS4000-400 SCSI interface which prevents its sharing the bus with a second, or third or . . . initiator or assuming the role of a target.)

Each device attached to the SCSI bus is identified by a unique device ID number in the range 0 through 7. During the arbitration, selection, and reselection bus phases in which an initiator and a target establish a connection, the device IDs of the initiator and target are both placed on the data bus by asserting the data bits corresponding to the device ID numbers. By convention, the ID number of the VS4000-400 system is six (this is controlled by the programs which drive the SCSI interface; it is not fixed in VS4000-400 hardware). The KA46 System Firmware gives details of the allocation of device IDs.

The electrical interface consists of 18 signal lines. Some of these lines are driven only by initiators, some only by targets, and some by either. These lines are summarized below, using the signal names by which they are called in this specification and in the ANSI specification.

In this specification and in all the registers of the SCSI interface chip, the "true" or "asserted" value of a signal appears as a "one", and the "false" or "negated" value of a signal appears as a "zero". The bus electrical signals are all low true and are driven by open-collector drivers.

The SCSI bus is always terminated at each end. The bus is permanently terminated at the controller (near end). Far end termination can take place in one of two locations:

At the expansion connector on the rear of the system enclosure.
At the second expansion connector on a storage expansion unit.

Note that none of the DEC supported SCSI storage devices provide SCSI bus termination.

10.2 SCSI Bus Signals

DB7..0 and DBP	comprise an 8-bit parallel data bus with an associated odd parity bit. The use of the parity bit is optional but strongly encouraged. These lines may be driven by either an initiator or a target, depending upon the direction of data transfer.
RST	signals all devices on the SCSI bus to reset to their initial power-on states. Thereafter, it should be asserted only as a last resort during error recovery since it indiscriminately affects all devices on the bus. An RST signal generated by some other device on the bus causes an internal reset of the 53C94 chip used in this controller and sets the interrupt request bit (INT in register SCS_STATUS).
BSY and SEL	are used by initiators and targets during the arbitration, selection, and reselection bus phases to establish or resume a logical connection between an initiator and a target. Once the connection is established, the target asserts BSY and the SEL signal is not driven by anyone.
C/D, I/O and MSG	collectively indicate one of six possible information transfer phases, according to the following table. These signals are always driven by the target device.

MSG	C/D	I/O	Phase name	Transfer direction
0	0	0	Data out	to Target
0	0	1	Data in	to Initiator
0	1	0	Command	to Target
0	1	1	Status	to Initiator
1	0	0		(reserved)
1	0	1		(reserved)

MSG	C/D	I/O	Phase name	Transfer direction
1	1	0	Message out	to Target
1	1	1	Message in	to Initiator

ATN	is used by an initiator to signal a target that it has a message ready. The target can receive the message by entering the "message out" phase. ATN is always driven by an initiator.
REQ and ACK	are used to synchronize information transfers over the data bus during any of the six information transfer phases. REQ is always driven by the target, ACK is always driven by the initiator.

10.3 Controller Overview

At the heart of the VS4000-400 storage controller is a single 53C94 SCSI controller chip through which a host program can examine and manipulate all the SCSI Bus signals. Associated with this chip is DMA logic which can transfer data between the SCSI bus and the DC7201. DMA data transfers are 16-bits wide, CPU initiated transfers are 8-bits wide using the 8 low order bits of the same 16-bit bus.

The 53C94 chip can be used by both initiator and target devices. In this specification, only its use as an initiator is described.

The 53C94 chip reduces CPU overhead by performing common SCSI commands as sequences in reponse to a single CPU command. It has an on-chip FIFO which may be accessed by the SCSI bus, the CPU directly or the DC7201 when performing DMA transfers to/from system memory. All Command, Data, Status and Message bytes pass through the FIFO going to/from the SCSI bus. As an example, the CPU can load a Command Descriptor Block and (optionally) one or three Message Bytes into the FIFO, issue one of several Selection Commands and then just wait for an interrupt. The 53C94 chip will wait for the SCSI bus to become free, Arbitrate for the bus until it acquires it, send the Message Bytes followed by the Command Descriptor Block and then interrupt the CPU.

The 53C94 SCSI Controller chip has thirteen addressable 8-bit internal registers that control its operation. Note that although data transfers to/from the 53C94 chip registers are eight bits wide; some registers actually are concatenated in use to form wider registers. Additionally some registers have different bit definitions for read operations than for write operations.

10.3.1 Register Definition

The thirteen internal registers of the 53C94 SCSI Contoller chip are accessed as bits<07:00> of thirteen sequential longwords. Writes to the remaining bits of each longword are ignored, reads of those bits return UNPREDICTABLE data. The address range used by the 53C94 chip register set is 200C.0080 to 200C.00B0. In the following register description, only bits<5:2> of the address will be referenced.

Register 0	Write : Transfer Count - LOW. The value loaded into this register is copied to the low 8-bits of the Transfer Counter Register at the start of a subsequent DMA transfer. This then becomes the low 8-bits of the count of bytes to be transferred during that DMA transfer. **Note** Some confusion may be possible here, be careful to distinguish between the *Transfer Count* and the *Transfer Counter* registers.
Register 1	Write : Transfer Count - HIGH. The value loaded into this register is copied to the high 8-bits of the Transfer Counter Register at the start of a subsequent DMA transfer. This then becomes the high 8-bits of the count of bytes to be transferred during that DMA transfer. Note: for sucessive DMA transfers having the same transfer length, Registers 0/1 need not be re-loaded between transfers. Loading Registers 0 and 1 with all zeroes specifies a maximum length transfer (65,536 bytes). Registers 0 and 1 are unaffected by any RESET, their contents are UNPREDICATABLE following power-on.
Register 0	Read : Transfer Counter - LOW. The value read is the low 8-bits of the Transfer Counter Register. This register may be read to help to determine the number of bytes remaining to be transferred if a transfer sequence terminates early.
Register 1	Read : Transfer Counter - HIGH. The value read is the high 8-bits of the Transfer Counter Register. This register may be read to help to determine the number of bytes remaining to be transferred if a transfer sequence terminates early.
Register 2	Read/Write : FIFO. The 53C94 chip FIFO is a 16-deep register through which all data to/from the SCSI bus flows. The SCSI bus may transfer 8 or 9 bit values into the FIFO depending on the setting of the parity check control bits in the Configuration 1 Register (see below). The last FIFO data element and associated flags are initialized to zero by any RESET, by a software RESET CHIP and by the beginning of a Bus initiated Selection or Re-selection. The other FIFO elements are not affected by any RESET, but when the flags are zero, successive FIFO reads will always return the contents of the bottom element.

Register 3	Read/Write : Command Register. This is actually a two deep FIFO in which up to two commands may be stacked. Once the first command has completed, the second, if present, will be executed immediately. The last command, or the currently executing command will be the value read from this register. If two commnads are stacked in the Command Register, subsequently two interrupts may result. If the first interrupt is not serviced before the second command completes, the second interrupt will be waiting. When the Interrupt Register is then read, servicing the first interrupt, the contents of the Status Register, Sequence Step Register and Interrupt Registers will then change to reflect the second (pending) interrupt status.

Bit 7 Enable DMA : When this bit is clear (0) and the 53C94 chip is sending data to the SCSI bus, data is transmitted to the bus until the FIFO is empty. Subsequent bytes, as needed, must be written to the FIFO register using the programmed I/O mode. The Transfer Counter Register is not modified when sending in this mode.

When this bit is clear (0) and the 53C94 chip is receiving data from the SCSI bus, a single byte of data will be transferred into the FIFO.

When this bit is set (1) and the 53C94 chip is sending data to the SCSI bus, data will be transferred from memory, via the DC7201 into the FIFO. The Transfer Counter Register will be decremented as data is loaded into the FIFO. Transmission will continue to the SCSI bus until the FIFO is empty and the Transfer Counter Register is zero.

When this bit is set (1) and the 53C94 chip is receiving data from the SCSI bus, the data received will be put into the FIFO and from there transferred via the DC7201 to memory. As data is put into the FIFO, the Transfer Counter Register is decremented. Data transfer from the SCSI bus into the FIFO continues until the Transfer Counter Register is zero. Data transfer out of the FIO to the DC7201 continues until the FIFO is empty and the Transfer Counter Register is zero.

Bits<6:0> Command Code : The 53C94 chip responds to 28 commands, specified by the command code.

Bit<6> = 1 - Disconnect Mode

Bit<5> = 1 - Target Mode

Bit<4> = 1 - Initiator Mode

Bits<6:4> all zero - Miscellaneous Mode

The Command Register is cleared by any of the following conditions :

- Hardware RESET
- Software RESET
- SCSI Bus RESET
- SCSI Bus Disconnect
- Bus initiated Selection or Re-selection
- Select Command
- Reconnect Command if ATN is set
- Select or Re-select Timeout
- Target Terminate Command
- Parity Error detected in Target Mode
- Assertion of ATN in Target Mode
- Any Phase change in Initiator Mode
- Illegal Command

Register 4	Write : Select/reselect Bus ID Register. This is a 3-bit write-only register that specifies the destination bus ID for a Select or Reselect command. Data is right justified in the byte. The other 5 bits are RESERVED. The destination ID is not affected by any RESET. These bits are UNPREDICTABLE following power-on.

Register 4

Read : Status Register, reading this register returns information on the status of the 53C94 chip and the reason for an interrupt having occurred.

Bit 7 Interrupt Bit. Any RESET or a read from the Interrupt Register will reset this bit.

Bit 6 Gross Error. This chip status bit is set when any one of the following has occurred.

– The top of the FIFO has been overwritten.

– The top of the Command Register has been overwritten.

– A DMA transfer has been attempted in the wrong direction.

– A phase change occurs in the middle of an initiator synchronous data transfer.

No interrupt is generated when a Gross Error occurs.

This bit is cleared by a hardware or software RESET, but not by a SCSI bus RESET, or by reading the Status Register if the Interrupt bit is set.

Bit 5 Parity Error. This bit is set when parity has been enabled in the Configuration 1 Register and the 53C94 chip subsequently detects a parity error on data received from the SCSI bus. This bit is cleared by a hardware or software RESET, but not by a SCSI bus RESET, or by reading the Status Register if the Interrupt bit is set.

Bit 4 Terminal Count. This bit sets when the Transfer Counter decrements to zero. It is reset when the Transfer Count is loaded. Reading the Interrupt Register *does not* clear this bit. This bit is cleared by a hardware or software RESET, but not by a SCSI bus RESET.

Bit 3 Valid Group Code. This bit is set during the target command sequence if the received command is one of the non-reserved groups (0, 1, 5, 6 or 7). It is reset when the Interrupt Register is read if the Interrupt bit is set. It is also cleared by a hardware or software RESET, but not by a SCSI bus RESET.

Bits<2:0> Phase Bits. These bits indicate the state of the SCSI MSG, C/D and I/O signals showing the current bus phase. These bits can be expected to be stable for any read following an interrupt.

Bits<2:0>	**SCSI Bus Phase**
000	Data OUT
001	Data IN
010	Command
011	Status
100	*** reserved ***
101	*** reserved ***

Bits<2:0>	SCSI Bus Phase
110	Message OUT
111	Messsage IN

Register 5

Read : Interrupt Register, this is an eight bit register used in conjunction with the Status Register and the Sequence Step Register to determine the cause of an interrupt. Reading this register when an interrupt has been posted by the 53C94 chip will clear all three of these registers.

Bit 7 - SCSI RESET Detected. This bit will be set if the SCSI Reset Reporting bit is set to 0 in the Configuration 1 Register AND the chip detects a Reset on the SCSI bus. If the interrupt is not serviced in <TBD>, the chip itself will assert Reset for <TBD>.

Bit 6 - Illegal Command. This bit will be set when the chip detects an unused code or an illegal command written to the Command Register.

Bit 5 - Disconnect. This bit will be set when the chip is operating in initiator mode and the current target disconnects from the SCSI bus or if a selct/reselect timeout occurs. In target mode, this bit is set when the Terminate Sequence or Command Complete Sequence command cuases the 53C94 chip to disconnect from the SCSI bus.

Bit 4 - Bus Service. When the chip is operating in target mode this bit is set when the Initiator requests service by asserting ATN. When the chip is operating in initiator mode this bit will be set whenever the target is requesting an Information Transfer Phase.

Bit 3 - Function Complete. When operating as a Target, this bit is set after completion of any command, When the chip is operating in initiator mode, the bit is set after a Target has been selected - by transferring any command phase bytes - or after a Transfer Information Command when the Target is requesting Message In Phase.

Bit 2 - Reselected. This bit is set to indicate that the chip has been reselected as an initiator.

Bit 1 - Selected with ATN. This bit is set during Selection Phase to indicate that the chip has been selected as a Target and that ATN was asserted on the SCSI bus.

Bit 0 - Selected. This bit is set during Selection Phase to indicate that the chip has been selected as a target and that the ATN line was false.

Reading the Interrupt Status Register with an interrupt pending will cause the Interrupt Request line to become de-asserted and the Interrupt Status and Sequence Step Registers to be cleared.

On receipt of an interrupt, the sequence of action should be always to read the Status Register and Sequence Step Registers before reading the Interrupt Status Register.

Register 5

Write : Select/reselect Timeout Register. This is an eight bit register that specifies the time to wait for a response during selection and reselection. More <TBS>.

Register 6	Read : Sequence Step. The three least signicant bits of this register, <2:0> indicate for certain sequences which sub-steps of the current sequence had been executed when an interrupt occurs. Bit <3>, when read as a one, indicates when the synchronous offset limit has been reached. The upper four bits are reserved. Sequences that utilize this register are as follows :

Table 10–1: Initiator Select with ATN and Stop

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00100000	Arbitration complete, Selection Time-out, disconnected
000	00011000	Arbitration and Selection Complete. Stopped because target did not assert Messsage Out Phase. ATN still asserted by the 53C94 chip.
001	00011000	Message Out Complete. Sent one message byte. ATN on.

Table 10–2: Initiator Select without ATN

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00100000	Arbitration complete, Selection Time-out, disconnected
010	00011000	Arbitration and Selection Complete. Stopped because target did not assert Command Phase.
011	00011000	Stopped during Command Transfer because target prematurely changed phase.
100	00011000	Select sequence complete.

Table 10–3: Initiator Select with ATN

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00100000	Arbitration complete, Selection Time-out, disconnected
000	00011000	Arbitration and Selection Complete. Stopped because target did not assert Message Out Phase. ATN still driven by 53C94 chip
010	00011000	Message Out complete. Sent one Message byte with ATN true, then released ATN. Stopped because Target did not assert Command Phase after Message byte was sent.

Table 10–3 (Cont.): Initiator Select with ATN

Sequence Step 210	Interrupt Register 76543210	Interpretation
011	00011000	Stopped during Command transfer due to premature phase change. Some CDB bytes may not have been sent; check FIFO flags.
100	00011000	Selection with ATN sequence complete.

Table 10–4: Initiator Select with ATN3

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00100000	Arbitration complete, Selection Time-out, disconnected
000	00011000	Arbitration and Selection Complete. Stopped because target did not assert Message Out Phase. ATN still driven by 53C94 chip
010	00011000	Sent 1, 2 or 3 Message bytes. Stopped because target prematurely changed from Message Out phase or did not assert Command phase after third Message byte. ATN released only if third Message byte *was* sent.
011	00011000	Stopped during Command transfer due to premature phase change. Some CDB bytes may not have been sent; check FIFO flags.
100	00011000	Selection with ATN3 sequence complete.

Table 10–5: Target Selected without ATN

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00000001	Selected, loaded Bus ID into FIFO, loaded null-byte message into FIFO
001	00000001	Stopped in Command Phase due to parity error. Some Command Descriptor Block bytes may not have been received. Check FIFO flags.
001	00010001	Same as above, Initiator asserted ATN in Command phase.

Table 10–5 (Cont.): Target Selected without ATN

Sequence Step 210	Interrupt Register 76543210	Interpretation
010	00000001	Selected, received entire Command Descriptor Block. Check Valid Group Status Bits.
010	00010001	same as above, Initiator asserted ATN in Command phase.

Table 10–6: Target Selected with ATN, SCSI-2 bit not set

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00000010	Selected with ATN, stored Bus ID and one Message byte. Stopped due to either parity error or invalid ID Message.
000	00010010	Selected with ATN, stored Bus ID and one Message byte. Stopped because ATN remained true after first Message byte.
001	00000010	Stopped in Commmand phase due to a parity error. Some CDB bytes not received. Check Valid Group Code bit and FIFO flags.
001	00010010	Stopped in Commnad phase. Parity Error and ATN true.
010	00000010	Selection complete. Received one Message byte and the entire Command Descriptor Block.
010	00010010	same as above, Initiator asserted ATN during Command phase.

Table 10–7: Target Receive Command

Sequence Step 210	Interrupt Register 76543210	Interpretation
001	00001000	Stopped during Commnad Transfer due to parity error. Check FIFO flags.
001	00011000	Stopped during Command Transfer due to parity error. ATN asserted by Initiator.
010	00001000	Received entire Command Descriptor Block.
010	00011000	Received entire CDB, Initiator asserted ATN.

Table 10–8: Target Disconnect Sequence

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00011000	Sent one Message byte. Stopped becasue Initiator set ATN.
001	00011000	Sent two Message bytes. Stopped because Initiator set ATN.
010	00101000	Disconnect Sequence complete. Disconnected, bus is free.

Table 10–9: Target Terminate Sequence

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00011000	Sent Status byte. Stopped because Initiator set ATN.
001	00011000	Sent Status and Message bytes. Stopped because Initiator set ATN.
010	00101000	Terminate Sequence complete. Disconnected, bus is free.

Table 10–10: Target Command Complete Sequence

Sequence Step 210	Interrupt Register 76543210	Interpretation
000	00011000	Sent Status byte. Stopped because Initiator set ATN.
001	00011000	Sent Status and Message bytes. Stopped because Initiator set ATN.
010	00101000	Command Complete Sequence complete.

Register 6

Write : Synchronous Transfer Period Register. The lower five bits of this register specify the minimum time between leading edges of successive REQ and ACK pulses when operating in synchronous mode. The high order 3 bits are reserved. The register is loaded after power on, or following a hardware RESET with a value of 00101. The time unit is the input clock period with the following relation between value loaded and transfer period :

Register Value	**Offset (clock Periods)**
00100	5
00101	5
00110	6
00111	7
.	
.	
.	
11111	31
00000	32
00001	33
00010	34
00011	35

Register 7

Write : Synchronous Offset Register. This four bit register specifies the maximum REQ/ACK offset allowed during synchronous operation. An offset of zero specifies asynchronous operation. When sending data out on the SCSI bus, the 53C94 chip will stop sending when th is offset is reached and from then on send one byte for each ACK it subsequently receives. When receiving data from the SCSI bus, the 53C94 chip will send an ACK every time a byte is removed from its FIFO.

Register 7

Read : FIFO Flags Register. Bits<4:0> of this register indicate the number of bytes remaining in the FIFO. Bits<7:5> are a copy of the Sequence Step Register bits when operating in normal mode.

Register 8	Read/write : Configuration-1 Register. This specifies different operating options for the 53C94 chip. Bit 7 - slow cable mode. When set, this bit increases the minimum data setup time by one cycle when transmitting to the SCSI bus. This bit should be clear for the use of the 53C94 chip in this application. Bit 6 - SCSI Reset Interrupt Disable. This bit controls the reporting of a SCSI Bus RESET condition. If a SCSI RESET occurs with this bit set, the 53C94 chip will disconnect from the SCSI bus and remain in the idle state without reporting an interrupt or timing out. Bit 5 - Parity Test Mode. When set, this bit causes the parity line of the SCSI bus to be driven from bit 7 of the transmitted data byte rather than from the output of the parity generator of the 53C94 chip. It is intended for use in diagnostic mode ONLY. This bit must be cleared in normal operation. Bit 4 - Parity Enable. When this bit is set, the 53C94 chip checks received data for correct parity on all bus transactions except during arbitration and when receiving pad bytes. Bit 3 - Chip Test Mode. This bit must be cleared in normal operation. It enables test circuitry used only during chip test. Bits <2:0> - My Bus ID. This three bit field specifies the SCSI bus ID to which the 53C94 chip responds.
Register 9	Write : Clock Conversion Register. This is a three bit register, using bits <2:0> that sets the chip timings relative the external clock frequency. Bits <7:3> are reserved. The value loaded into this register must never be one. For the clock frequency used in this application of the 53C94 chip, this register must be programmed with a value of 5.
Register A	Write : Test Register. This is a three bit register, using bits <2:0> that places the 53C94 chip in various test modes. For this register's settings to be enabled, the chip must first be placed in the Test Mode by setting bit <3> in Register 8. More tbs.

Register B

Read/write : Configuration 2 Register.

Bit 7 - Reserve FIFO Byte. This bit allows a 16-bit DMA transfer to begin on a non-word boundary for Initiator Synchronous Data In. When this bit is set, a single byte is reserved in the bottom of the FIFO when the phase changes to Synchronous Data In. The next byte written to register F will be placed in the bottom of the FIFO and will clear this bit. The first subsequent 16-bit DMA read will read this byte as the low byte of the first word; the first byte received across the SCSI bus will become the high byte of the first 16-bit word read in the DMA operation. The device driver should copy the low byte of the memory destination word into the FIFO in response to s Synchronous Data In interrupt before issuing the DMA Transfer Info. Command. This bit is cleared by a hardware or software reset but is unaffected by a SCSI bus reset.

Bit 6 - Enable Phase Latch. When this bit is cleared, the Pahse Bits, as reported in the Ststus Register, are live indicators of the state of the SCSI phase lines. When this bit is set, the phase is latched at the completion of any command. Reading the Interrupt Register resets the phase latch. This bit is cleared by a hardware or software reset but is unaffected by a SCSI bus reset.

Bit 5 - Enable Byte Control. This bit is unused for the operation mode of the 53C94 chip used here. This bit *MUST* be left clear. It is cleared by a hardware or software RESET.

Bit 4 - DREQ High Impedance. This bit *MUST* be clear. It is cleared by a hardware or software RESET.

Bit 3 - SCSI-2. When this bit is set, the 53C94 chip supports two new features that are adopted in SCSI-2; the three byte message exchange for Tagged-Queuing. and Group 2 Commands.

> Tagged-Queuing. When this bit is set and the 53C94 chip is selected with ATN, it will request either one or three Message Bytes depending on whether ATN remains true or goes false. If ATN is still true after the first byte has been received, the 53C94 chip may request two more Message bytes before switching to Command Phase. If ATN goes false, it will switch to Command Phase after only a single byte. When this bit is clear, the 53C94 chip will abort the Selection if the target does not switch to Command phase after the transfer of a single Message Byte.
>
> Group 2 Commands. Group 2 commands are 10-byte sequences. Receiving a Group 2 Commnad with the SCSI-2 bit set will set the Valid Group Code bit in the Status Register. If the SCSI-2 bit is not set, Group 2 Commands will be treated as reserved commands; only six will be requested in Command Phase and the Valid Group Code bits will not be set.

Bit 2 - Target Bad Parity Abort. When this bit is set, the 53C94 chip will abort a Receive Command or Receive Data sequence if it deteects bad parity.

Bit 1 - Register Parity Enable. This bit must be left clear. It is cleared by a hardware or software RESET, but is unaffected by a SCSI RESET.

Bit 0 - DMA Parity Enable. This bit must be left clear. It is cleared by a hardware or software RESET, but is unaffected by a SCSI RESET.

Register C

Configuration 3 Register. Only the low three bit sof this register are used, the high order five bits are RESERVED.

Bit 2 - Save Residual Byte. If a DMA transfer is initiated that requires that an odd number of bytes be transferred and this bit is set, the last byte will be left in the FIFO, or will remain in the output buffer of the DC7201 and must be transferred by CPU intervention. If this bit is *not* set, the odd byte will be transferred as the low byte of the last 16-bit word and, on a transfer from the 53C94 chip to the DC7201, the high byte will be set to all ones. This bit is cleared by a hardware or software RESET, but is unaffected by a SCSI RESET.

Bit 1 - Alternate DMA Mode. This bit **must** be left clear. It is cleared by a hardware or software RESET, but is unaffected by a SCSI RESET.

Bit 0 - Threshold Eight. This bit must be left clear. It is cleared by a hardware or software RESET, but is unaffected by a SCSI RESET.

10.4 DMA Control Registers

Two registers control the DMA operations of the 53C94 chip. The first is a 24-bit register that specifies a byte address for the start of a SCSI DMA transfer. The high 15 bits of this address are used as an index into the reserved Map region of memory - see Chapter 4, Section 4.6. The other is a direction register, specifying the direction of DMA data transfer.

Figure 10–1: SCSI DMA Address Register - Address 200C.0000

```
 3            2 2                                            0
 1            4 3                                            0
+--------------+----------------------------------------------+
|     IGN      |          DMA Start Address                   |
+--------------+----------------------------------------------+
```

Figure 10–2: SCSI DMA Direction Register - Address 200C.000C

```
 3                                                         0 0
 1                                                         1 0
+---------------------------------------------------------+-+
|                       IGN                               |D|
+---------------------------------------------------------+-+
```

D = 0, the DMA transfer is a READ - data from 53C94 chip to memory.

D = 1, the DMA transfer is a WRITE - data from memory to 53C94 chip.

Chapter 11

SERIAL LINE CONTROLLER

The VS4000-400 system board serial line controller handles four asynchronous serial lines. The controller and a 64 entry silo shared by all four receive lines are parts of the DC7201.

Note: This controller is similar to a VAXstation 3100 serial controller. The operation of the receivers and transmitters is virtually identical, but the arrangement of modem signals, interrupt controls, and the register addresses is **NOT** the same.

11.1 Line Usage

The four serial lines are numbered 0 through 3, and each has a particular primary use, as follows:

Table 11–1: Serial Line Useage

Line	Device
0	Keyboard - connected to a 15-pin D-sub connector[1] and to a 4-pin Modular Jack mounted on the system board. Data leads only. Supports the LK401 keyboard.
1	Pointer - connected to a 15-pin D-sub connector[1] and to a miniature DIN connector mounted on the System Board. Data leads only. Supports VSXXX-AA mouse or VSXXX-AB Tablet.
2	Communications - connected to a 25-pin D-sub connector mounted on the system board, RS423 compatible. Data leads plus Modem Control signals.
3	Printer - connected to a 6-pin Modified Moduler Jack mounted on the system board. DEC423, data leads only.

[1]Same connector

11.2 Diagnostic Terminal Connection

Line 3 is normally connected to a printer through a BC16E cable. If the special jumper is installed in JNN on the VS4000-400 System Board, a received break condition on this line will assert the CPU halt signal which will cause a processor restart with a restart code of 2.

11.3 Interrupts

The controller generates two interrupt requests, each with a separate vector and enable bit in the INT_REQ and INT_MSK registers. These are receiver done or silo alarm, and transmitter done.Chapter 6, Section 6.7 lists the vector values. In order for these interrupts to be signalled to the CPU, the appropriate bits in the interrupt mask register INT_MSK must be set (see Chapter 6, Section 6.6. Note that, unlike a DZQ11 board, there are no interrupt enable bits in the control and status register SER_CSR.

11.4 Register Summary

The controller contains eight addressable registers as follows:

Table 11–2: Serial Line Controller Register Addresses

Address	Name	Access	Description
200A.0000	SER_CSR	R/W	Control and status register
200A.0004	SER_RBUF	R	Receiver buffer - oldest data in silo
200A.0004	SER_LPR	W	Line parameter register
200A.0008	SER_TCR	R/W	Transmitter control register
200A.000C	SER_MSR	R	Modem status register
200A.000C	SER_TDR	W	Transmitter data register
200A.0010	DZ_LPR0	R	Line parameter register, line 0
200A.0014	DZ_LPR1	R	Line parameter register, line 1
200A.0018	DZ_LPR2	R	Line parameter register, line 2
200A.001C	DZ_LPR3	R	Line parameter register, line 3

11.4.1 Control and Status Register (SER_CSR)

The Control and Status register is a 16-bit register at address 200A.0000. It must be read on a word basis but can be written on either a word or byte basis. All bits in SER_CSR are cleared to zero by power-on or by setting the master clear bit SER_CSR<CLR>.

Figure 11–1: Serial Line Control and Status Register (SER_CSR)

```
     15      14      13      12      11      10       9       8
   +-------+-------+-------+-------+-------+-------+-------+-------+--
   | TRDY  |   0   |  SA   |  SAE  |   0   |   0   |     TLINE     |
   +-------+-------+-------+-------+-------+-------+-------+-------+-

      7       6       5       4       3       2       1       0
  -+-------+-------+-------+-------+-------+-------+-------+-------+
   | RDONE |   0   |  MSE  |  CLR  | MAINT |   0   |   0   |   0   |
 --+-------+-------+-------+-------+-------+-------+-------+-------+
```

TRDY	Transmitter Ready (bit 15). This read-only bit is set by the hardware when the transmitter scanner stops on a line whose transmitter buffer is ready to be loaded with another character and whose related transmitter control register bit SER_TCR<TXEN_x>is set. While the <TRDY> bit is one, and at no other time, the transmitter line number bits SER_CSR<TLINE> are valid. When <TRDY> changes from zero to one, the interrupt request register bit INT_REQ<ST> is set to one. If interrupt mask register bit INT_MSK<ST> is also one, a transmitter interrupt request to the CPU will be generated; otherwise SER_CSR<TRDY> can be polled by the host program. However, the interrupt request bit INT_REQ<ST> is not automatically cleared while interrupts are masked, so when changing from polled to interrupt operation there may be a spurious interrupt request to the CPU unless the host program clears INT_REQ<ST> by writing a one to INT_CLR<ST>. The <TRDY> bit is cleared when data is loaded into the transmitter for the line number indicated in SER_CSR<TLINE> by writing to register SER_TDR. If additional transmitter lines need service, <TRDY> is set again within 1.4 µs of the completion of the transmitter data load operation. The <TRDY> bit is also cleared when the master scan enable bit SER_CSR<MSE> is cleared, or when the related transmitter control register bit SER_TCR<TXEN_x> is cleared.
<14>	Not used, read as zero.
SA	Silo Alarm (bit 13). This read-only bit is set by the hardware when 16 characters have been entered into the FIFO silo buffer. While the silo alarm enable bit SER_CSR<SAE> is one, the transition of <SA> from zero to one sets interrupt request register bit INT_REQ<SR> to one. If interrupt mask register bit INT_MSK<SR> is also one, an interrupt is signalled to the CPU; otherwise the <SA> bit may be polled. However, the interrupt request register bit INT_REQ<SR> is not automatically cleared while that interrupt is masked, so when changing from polled to interrupt operation, there may be a spurious interrupt request to the CPU unless the host program clears INT_REQ<SR> by writing a one to INT_CLR<SR>.

Note

The <SA> bit is cleared by reading the receiver buffer register SER_RBUF. When responding to a silo alarm, the host program need not empty the silo, <SA> will be re-asserted and a new interrupt will ocuur when sufficient new characters have entered the silo such that there are again 16 characters in the silo. This operation is different from that of the VAXstation 3100 serial line controller SILO.

The <SA> bit is always zero while the silo alarm enable bit SER_CSR<SAE> is zero.

SAE	Silo Alarm Enable (bit 12). This read/write bit selects the source of the receive interrupt request signal which is sent to bit INT_REQ<SR> in the interrupt controller. If <SAE> is one, the silo alarm bit <SA> discussed above is used as the signal; if <SAE> is zero, the receiver done bit <RDONE> discussed below is used instead. Note that while <SAE> is zero, <SA> is also forced to be zero.
<11:10>	Not used, read as zero.
TLINE	Transmitter Line Number (bits 9:8). These read-only bits indicate the number of the line whose transmitter buffer needs servicing (bit 8 is the least significant bit). These bits are valid only while the transmitter ready bit SER_CSR<TRDY> is one. These bits are cleared when the master scan enable bit SER_CSR<MSE> is cleared.
RDONE	Receiver Done (bit 7). This read-only bit is set by the hardware when an incoming character appears at the output of the silo buffer. While the silo alarm enable bit SER_CSR<SAE> is zero, the transition of <RDONE> from zero to one sets interrupt request register bit INT_REQ<SR> to one. If interrupt mask register bit INT_MSK<SR> is also one, an interrupt is signalled to the CPU; otherwise the <RDONE> bit may be polled. However, the interrupt request register bit INT_REQ<SR> is not automatically cleared while that interrupt is masked, so when changing from polled to interrupt operation, there may be a spurious interrupt request to the CPU unless the host program clears INT_REQ<SR> by writing a one to INT_CLR<SR>. <RDONE> is cleared when the receiver buffer register SER_RBUF is read. If another character is available in the silo, <RDONE> will be set again after a delay of between 0.1 and 1.0 microsecond. This bit is also cleared when the master scan enable bit SER_CSR<MSE> is cleared.
<6>	Not used, read as zero.
MSE	Master Scan Enable (bit 5). This read/write bit must be set to one to permit the receiver and transmitter control sections to scan the lines to service them. When this bit is zero, the transmitter ready bit SER_CSR<TRDY> is cleared and the receiver silo is cleared.

CLR	Master Clear (bit 4). When this bit is set by a program, the hardware performs an internal initialization process. At the conclusion of this process the hardware clears this bit. Reading this bit will always return a zero. This initialization clears all registers, the silo, and all UARTs with the following exceptions:

—In the receiver buffer register only bit SER_RBUF<DVAL> is cleared; the remaining bits are not.
—Bits <15:8> of the transmitter control register SER_TCR (the modem control outputs) are not cleared.
—The modem status register SER_MSR is not cleared.

Note

PROGRAMMING NOTE: In previous implementations of this "DZ11-like" serial line controller, after setting the master clear bit SER_CSR<CLR>, a program needed to repeatedly read SER_CSR until it found SER_CSR<CLR> equal to zero before attempting any other operations with the serial line controller. This is no longer needed, any read of SER_CSR will *always* return a zero in bit position SER_CSR<CLR>.

Neither of the interrupt controller registers INT_REQ and INT_MSK are altered when <CLR> is set. The host program must clear bits <SR> and <ST> of INT_MSK to zero and must clear those bits in INT_REQ by writing ones to the corresponding bits of INT_CLR to complete the initialization process.

MAINT	Maintenance (bit 3). This read/write bit, when set, loops the serial output connections of the transmitters to the corresponding serial input connections of the receivers. This feature is intended for hardware diagnostic use.
<2:0>	Not used, read as zero.

11.4.2 Receiver Buffer Register (SER_RBUF)

The Receiver Buffer register is a 16-bit read-only register at address 200A.0004 which must be read as a word. It contains the received character at the bottom of the silo buffer (that is, the oldest character in the silo). Reading this register removes the character from the silo buffer, and all the other characters in the silo then shift down to the lowest unoccupied location. When this register is read (or when the master clear bit SER_CSR<CLR> is set or after a power-on reset), the data valid bit SER_RBUF<DVAL> is cleared and the remaining bits of the register (although not cleared) are invalid.

Figure 11–2: Serial Line Receiver Buffer Register (SER_RBUF)

```
         15     14     13     12     11     10      9      8
      +------+------+------+------+------+------+------+------+--
      | DVAL | OERR | FERR | PERR |  1   |  1   |    RLINE    |
      +------+------+------+------+------+------+------+------+--

          7      6      5      4      3      2      1      0
    --+------+------+------+------+------+------+------+------+
      |                        RCHAR                          |
   ---+------+------+------+------+------+------+------+------+
```

DVAL	Data Valid (bit 15). This bit, when one, indicates that the data in bits <14:0> of the register is valid. This permits an interrupt handling program to read the receiver buffer register repeatedly and store each character until this bit is read as zero, which indicates that the silo is empty.
OERR	Overrun Error (bit 14). This bit is one when a received character is overwritten in a UART buffer by a following character before the first character was transferred to the silo. This condition indicates that the program is not emptying the silo fast enough.
FERR	Framing Error (bit 13). This bit is one if the received character did not have a stop bit present at the correct time. The combination of <FERR> set and <RCHAR> entirely zero is usually interpreted as indicating that a BREAK has been received. The receipt of a framing error on line 3 (the printer port) is a special case, the line controller hardware asserts a signal whose effect is described under Diagnostic Terminal Connection - Section 11.2, if JNN is inserted.
PERR	Parity Error (bit 12). This bit is one if the sense of the parity of the accompanying character does not agree with the parity which was defined for the line when its line parameter register SER_LPR was last loaded.
<11:10>	Not used, read as ones.
RLINE	Receiver Line Number (bits 9:8). These bits indicate the number of the line from which the character was received (bit 8 is the least significant bit).
RCHAR	Received Character (bits 7:0). Characters with a width of fewer than 8 bits (as defined when the line's line parameter register was last loaded) are right justified with the unused bit positions cleared. The parity bit is not included in the received character.

11.4.3 Line Parameter Register (SER_LPR) and DZ_LPR<3:0>

The Line Parameter register, SER_LPR is a 16-bit register at address 200A.0004 which controls the operating parameters of each line.

This register is write-only and must be written as a 16-bit word. The parameters for each line must be reloaded after each power-on reset or setting of the master clear bit SER_CSR<CLR>. The operating parameters should not be modified for a line while data transmission or reception is in progress on that line.

The parameters for each line may be read individually by accessing the four read only registers DZ_LPR<3:0>.

Figure 11–3: Serial Line Parameter Register (SER_LPR) and DZ_LPR3:0

```
      15     14     13     12     11     10      9      8
   +------+------+------+------+------+------+------+------+---
   |  0   |  0   |  0   |RXENAB|         SPEED             |
   +------+------+------+------+------+------+------+------+--

         7      6      5      4      3      2      1      0
   --+------+------+-------------+------+------+------+------+
     |ODDPAR|PARENB|     IGN     |CHARW | IGN  |    PLINE    |
  ---+------+------+-------------+------+------+------+------+
```

<15:13> Not used.

<12> RXENAB Receiver. This bit must be set in order for the UART for this line to receive bits and assemble them into characters.

<11:08> SPEED Speed Code. These bits select the data bit rate for the receiver and transmitter for the line. The bits are encoded as follows:

11	10	9	8	Data rate (bits/second)	
0	0	0	0	300	
0	0	0	1	300	
0	0	1	0	300	
0	0	1	1	300	
0	1	0	0	300	
0	1	0	1	300	*
0	1	1	0	600	
0	1	1	1	1200	
1	0	0	0	2400	
1	0	0	1	2400	
1	0	1	0	2400	*
1	0	1	1	4800	
1	1	0	0	4800	*
1	1	0	1	9600	
1	1	1	0	9600	*
1	1	1	1	19200	*

Note 1 : only those speeds marked with an asterisk are supported, attempts to set other non-supported speeds will default to the nearest supported speed as shown. The asterisks represent the VS3100 compatible settings for the supported speeds.

Note 2 : the highest speed supported is 19,200 baud - this is different from the VS3100.

<07>	ODDPAR	Odd Parity. If this bit is set and the parity enable bit SER_LPR<PARENB> is also set, then characters with odd parity are transmitted to the line and characters received from the line are expected to have odd parity. If this bit is clear and the parity enable bit SER_LPR<PARENB> is set, then characters with even parity are transmitted to the line and characters received from the line are expected to have even parity. If the parity enable bit SER_LPR<PARENB> is clear, then the setting of this bit is immaterial.
<06>	PARENB	Parity Enable. If this bit is set, characters transmitted to the line have a parity bit appended and characters received from the line have their parity checked. The sense of the parity is according to the setting of the odd parity bit SER_LPR<ODDPAR>.
<05:04>	IGN	These two bits are ignored.
<03>	CHARW	Character width. Character widths of 7 and 8 bits only are supported. Bit<03> = 1 makes transmitted characters have 8 data elements and expects that received characters will have 8 data elements; bit<03> = 0 sets 7 data elements for transmit and receive.
<02>		Ignored on write, reads as zero.
<01:00>	PLINE	Parameter Line Number. These bits specify the number of the line to which the parameters in the rest of the register apply. Bit 0 is the least significant bit.

11.4.4 Transmitter Control Register (SER_TCR)

The Transmitter Control register is a 16-bit register at address 200A.0008 which must be read on a word basis and can be written on either a word or byte basis.

Figure 11–4: Serial Line Transmitter Control Register (SER_TCR)

```
      15     14     13     12     11     10      9      8
    +------+------+------+------+------+------+------+------+---
    |        ... 0 ...          |LLBK_2|DTR_2 |DSRS_2|RTS_2 |
    +------+------+------+------+------+------+------+------+--


       7      6      5      4      3      2      1      0
  --+------+------+------+------+------+------+------+------+
    |        ... 0 ...          |TXEN_3|TXEN_2|TXEN_1|TXEN_0|
 ---+------+------+------+------+------+------+------+------+
```

<15:12>	Not used, read as zero.
LLBK_2	Local Loopback (bit 11). This read/write bit controls the state of the Local Loopback Modem Control signal (CCITT Circuit 141) for line 2. Setting this bit asserts the ON state of the LLBK signal. This bit is cleared by power-on; it is *not* cleared when the master clear bit SER_CSR<CLR> is set - see Section 11.4.1.

DTR_2	Data Terminal Ready (bit 10). This read/write bit controls the state of the Data Terminal Ready modem control signal (CCITT circuit 108/2) for line 2. Setting the bit asserts the ON state of the DTR signal. This bit is cleared by a power-on reset; it is *not* cleared when the master clear bit SER_CSR<CLR> is set - see Section 11.4.1.
DSRS_2	Data Rate Signalling Rate Selector (bit 9). This read/write bit controls the state of the Data Signalling Rate Selector modem control signal (CCITT circuit 111) for line 2. Setting the bit asserts the ON state of the signal. This bit is cleared by power-on; it is *not* cleared when the master clear bit SER_CSR<CLR> is set - see Section 11.4.1.
RTS_2	Request to Send (bit 8). This read/write bit controls the state of the Request to Send modem control signal (CCITT circuit 105) for line 2. Setting this bit asserts the ON state of this signal. This bit is cleared by power-on; it is *not* cleared when the master clear bit SER_CSR<CLR> is set - see Section 11.4.1.
<7:4>	Not used, read as zero.
TXEN_x	Transmitter Line Enable (bits 3:0). These read/write bits enable the transmitter logic for lines 3, 2, 1, and 0 respectively. Setting each of these bits causes the transmitter scanner to stop and assert the transmitter ready bit SER_CSR<TRDY> if the UART for that line has a transmitter buffer empty condition. The transmitter scanner resumes scanning when either the transmitter data register for the line at which the scanner stopped is loaded with another character, or when that line's transmitter line enable bit is cleared. A transmitter line enable bit should only be cleared while the scanner is not running (i.e. when the transmitter ready bit SER_CSR<TRDY> is set or the master scan enable bit SER_CSR<MSE> is clear). The transmitter line enable bits are cleared by a power-on reset and whenever the master clear bit SER_CSR<CLR> is set.

11.4.5 Modem Status Register (SER_MSR)

The Modem Status register is a 16-bit read-only register at address 200A.000C which contains the status of modem input signals for line 2. The ON condition of a modem signal is presented as the set state of the corresponding bit.

Figure 11–5: Serial Line Modem Status Register (SER_MSR)

```
      15     14     13     12     11     10      9      8
   +------+------+------+------+------+------+------+------+---
   |                           |SPDI_2| CD_2 |DSR_2 |CTS_2 |
   +------+------+------+------+------+------+------+------+--


       7      6      5      4      3      2      1      0
 --+------+------+------+------+------+------+------+------+
   |                           |  0   | RI_2 |   0  |TMI_2 |
---+------+------+------+------+------+------+------+------+
```

<15:11>	Not used; read values undefined.
SPDI_2	Speed Mode Indicate (bit 11). This bit reflects the state of the Speed Mode Indicate signal from a modem (CCITT circuit 112) connected to line 2.
CD_2	Carrier Detect (bit 10). This bit reflects the state of the Carrier Detect signal from a modem (CCITT circuit 109) connected to line 2.The set state corresponds to the ON state of this signal.
DSR_2	Data Set Ready (bit 9). This bit reflects the state of the Data Set Ready signal from an external modem (CCITT circuit 107) on line 2. The set state corresponds to the ON state of this signal.
CTS_2	Clear to Send (bit 8). This bit reflects the state of the Clear to Send signal from an external modem (CCITT circuit 106) on line 2. The set state corresponds to the ON state of this signal.
<7:4>	Not used; read values undefined.
<3>	Reserved, reads as 0.
RI_2	Ring Indicator (bit 2). This bit reflects the state of the Ring Indicator signal from an external modem (CCITT circuit 125) on line 2. The set state corresponds to the ON state of this signal.
<1>	Reserved, reads as 0.
TMI_2	Test Mode Indicate (bit 0). This bit reflects the state of the Test Mode Indicate signal from an external modem (CCITT circuit 142) on line 2. The set state corresponds to the ON state of this signal.

11.4.6 Transmitter Data Register (SER_TDR)

The Transmitter Data register is a 16-bit write-only register at address 200A.000C. It can be written on either a word or byte basis.

Figure 11–6: Serial Line Transmitter Data Register(SER_TDR)

```
      15     14     13     12     11     10      9      8
   +------+------+------+------+------+------+------+------+---
   |                           |BRK_3 |BRK_2 |BRK_1 |BRK_0 |
   +------+------+------+------+------+------+------+------+--

       7      6      5      4      3      2      1      0
 --+------+------+------+------+------+------+------+------+
   |                        TXDATA                         |
---+------+------+------+------+------+------+------+------+
```

<15:12>	Not used.
<11:08>	BRK_x Break Control. These write-only bits control the assertion of a BREAK condition on lines 3, 2, 1, and 0, respectively. Setting a bit immediately forces the transmitter output for the corresponding line to the SPACE condition. This condition will persist until the break control bit is cleared. These bits are cleared by a power-on reset and when the master clear bit SER_CSR<CLR> is set.
<07:00>	TXDATA Transmitter Buffer. Data to be transmitted by a line's UART is loaded into these 8 bits. If the character width is 7, the unused bit is the high-order (bit 7) of the byte. This register may be written to only while the transmitter ready bit SER_CSR<TRDY> is set. The line to which the character is sent is indicated by the transmitter line number bits SER_CSR<TLINE>.

Chapter 12

SOUND GENERATOR

This chapter decribes the sound output capability of the VS4000-400. Sound output uses the DTMF tone generation capability of the 79C30 chip. Two tone generators may be individually programmed for frequency and amplitude; their outputs appear summed using either the loudspeaker integral to the system unit, or to headphones or an external loudspeaker if plugged in to the jack at the front of the machine. The resolution of the frequency generators is eight bits, giving a frequency range of 8 Hz. to approx. 2 kHz.

12.1 Sound Chip Register Access

The 79C30 has some registers that are directly accessible and some that are accessed indirectly. These indirectly accessible registers are accessed by selecting one of several groups of registers and a register within that group with a write to the Command Register - address 200D.0000h. This register is eight bits wide and takes data from bits<7:0> of the longword specified. Following selection of a register within a group, one or more bytes of data are then transferred to load the selected register(s) - a function of which Group and which register within a Group. The data is transferred by writing one or more times to the Data Register - address 200D.0004h.

The Command Register format for selecting a register is as shown below :

Figure 12–1: Command Register Format - address 200D.0000h

```
   7    6    5    4    3    2    1    0
+----+----+----+----+----+----+----+----+
|    |    |    |    |    |    |    |    |
+----+----+----+----+----+----+----+----+
 \            / \                      /
  +----+-----+   +---------+----------+
       |                   |
    Selects          Selects Register
 Register Group      within Group
```

For tone generation there are three Register Groups that need to be accessed, the INIT, MUX and MAP groups. Refer to the 79C30 chip vendor specification for full details of all register groups.

12.1.1 INIT Register Group

This group is used to enable/disable the tone generators. The Register Group Select bits are set to <001> and the Register Select bits set to <00001>, i.e. a value of 21h is loaded into the Command Register.

The INIT register then is loaded with <00000001> to Enable the tone generators and <00000000> to Disable them.

12.1.2 MUX Registers

This Group is selected to set up the internal connections within the 79C30. The Register Group Select bits are set to <010> and the four MUX Control Registers (MCRs) selected in sequence by setting the Register Select bits to <00001> - MCR1, <00010> - MCR2, <00011> - MCR3 and <00100> - MCR4. Each register requires a single byte of data, zero for MCR1, 2 and 4 and 33h for MCR3. Thus the load sequence for the MCR registers is eight writes :

Table 12–1: MUX Register Load Sequence

Write Address	Value
200D.0000h	41h
200D.0004h	0
200D.0000h	42h
200D.0004h	0
200D.0000h	43h
200D.0004h	33h
200D.0000h	44h
200D.0004h	0

12.1.3 MAP Registers

Of the ten registers in the MAP Register Group - Main Audio Processor, some need only be initialixed, some need not be programmed for this mode of operation of the chip and four control the frequency and amplitude of the tones generated.

12.1.3.1 MAP Registers - Initialization

Table 12–2: MAP Register Initialization

Write Address	Value	Bytes
200D.0000h	66h	2
200D.0004h	08h	
200D.0004h	08h	
200D.0000h	69h	1
200D.0004h	48h	
200D.0000h	6Ah	1
200D.0004h	06h	

12.1.4 MAP Registers - Frequency and Amplitude Selection

Four of the MAP group registers select frequency (FTGR1 and 2) and amplitude (ATGR1 and 2) of the generated tones. The tone duration is set by how often the FTGR registers are re-loaded. There are no interrupts associated with this mode of operation of the 79C30.

The 79C30 has two frequency generators. When the FTGR registers have been selected (a write to 200D.0000h with a data value of 67h), the frequency produced by each tone generator when the FTGR register is loaded with a value of "i" is 1000*i/128 Hz. See Table 12–3 for a table of values that produce a chromatic scale. The frequencies of two tone generators are loaded by two successive writes to the Data Register following selection of the FTGR registers.

The ATGR registers allows the amplitude of the tones generated to be varied by 18 dB in 2 dB steps. See Table 12–4 for a list of values v amplitude. The ATGR registers are selected by loading the Command Register with a value of 68h, then writing the amplitude values by two successive writes to the Data Register.

12.2 FTGR Values v Frequency

The following table shows the value that must be loaded (in decimal) to each/either tone generator to produce a given frequency output. To obtain a single tone output, both generators should be loaded with the same value.

Table 12–3: Approximate Chromatic Scale

Note	Frequency Required	Actual Frequency	FTGR Value	Note
C	262	265.6	34	Middle C
C#	277	273.4	35	
D	294	296.9	38	
D#	311	312.5	40	
E	330	328.1	42	
F	349	351.6	45	
F#	370	367.2	47	
G	392	390.6	50	
G#	415	414	53	

Table 12–3 (Cont.): Approximate Chromatic Scale

Note	Frequency Required	Actual Frequency	FTGR Value	Note
A	440	437.5	56	
A#	466	468.8	60	
B	494	492.2	63	
C	523	523.4	67	
C#	554	554.7	71	
D	587	585.9	75	
D#	622	625	80	
E	659	656.3	84	
F	698	695.3	89	
F#	740	742.2	95	
G	784	781.3	100	
G#	831	828.1	106	
A	880	882.8	113	
A#	932	929.7	119	
B	988	984.4	126	

12.3 Amplitude Selection

The following table shows the value that must be loaded (in decimal) to set a particular amplitude of a generated tone.

Table 12–4: Amplitude v ATGR value

Gain (dB)	ATGR Value
-18	55
-16	50
-14	49
-12	39
-10	34
-8	33
-6	32
-4	18
-2	17
0	16

Chapter 13

TIME OF YEAR CLOCK

The time of year clock consists of an MC146818BM CMOS watch chip which keeps the date and time of day and contains 50 bytes of general purpose RAM storage. This chip includes a time base oscillator and a lithium battery on-chip. The battery powers the logic and oscillator while system power is off.

It is expected that data from the watch chip will be used by an operating system during its startup to determine the date and time, which the operating system will thereafter maintain by using interrupts from the interval timer. Therefore the watch chip's alarm and periodic interrupt features are not used in this system and the watch chip cannot generate a processor interrupt.

13.1 Battery Backup

A lithium battery within the watch chip supplies power to the watch chip and its time base oscillator while system power is off. The battery will maintain the clock operation and the data stored in the 50 bytes of RAM for a minimum of 10 years before it becomes exhausted.

13.2 Watch Chip Registers

The watch chip contains 64 8-bit registers. Ten of these contain date and time data, 4 are control and status registers, and the remaining 50 provide general purpose RAM storage. The registers occupy 64 consecutive longwords of address space as shown in the table below.

Each register is accessed as bits<9:2> of a longword (bits<31:10> and <1:0> are ignored on writing and undefined on reading).

WARNING

Because each register spans two bytes on the system bus, only *word* or *longword* access instructions may be used to manipulate these registers. The effects of using byte access instructions are undefined. In particular, instructions for modifying bits such as BBSS, BBSC, BBCC and BBCS cannot be used–they generate byte-access read-modify-write cycles which will corrupt the portion of the register which is not in the byte being accessed.

Table 13–1: **TOY Chip Register Addresses**

Address	Name	Description
200B.0000	WAT_SEC	Time seconds, 0 . . . 59
200B.0004	WAT_ALMS	Alarm seconds (not used)
200B.0008	WAT_MIN	Time minutes, 0 . . . 59
200B.000C	WAT_ALMM	Alarm minutes (not used)
200B.0010	WAT_HOUR	Time hours, 0 . . . 23
200B.0014	WAT_ALMH	Alarm hours (not used)
200B.0018	WAT_DOW	Day of week, 1 . . . 7
200B.001C	WAT_DAY	Day of month, 1 . . . 31
200B.0020	WAT_MON	Month of year, 1 . . . 12
200B.0024	WAT_YEAR	Year of century, 0 . . . 99
200B.0028	WAT_CSRA	Time base divisor
200B.002C	WAT_CSRB	Date mode and format
200B.0030	WAT_CSRC	Interrupt flags (not used)
200B.0034	WAT_CSRD	Valid RAM and time flag
200B.0038		First byte of RAM data
. . .		
200B.00FC		Last byte of RAM data

13.3 Control and Status Registers

Figure 13–1: Watch Time Base Divisor (WATCSRA)

```
       9       8       7       6       5       4       3       2
   +-------+-------+-------+-------+-------+-------+-------+-------+
   |  UIP  |          DVX          |              RSX              |
   +-------+-------+-------+-------+-------+-------+-------+-------+
```

UIP	Update in progress (bit 9). This read-only bit indicates when the date and time registers are being updated and are hence unstable. It is set to one 244 microseconds before the beginning of an update cycle and remains one until the cycle is complete.
DVX	Time base divisor (bits 8:6). These read/write bits set the amount by which the time base oscillator input to the watch chip is divided. These bits must be set to "010" to accomodate the 32.768 KHz time base in this system.
RSX	Rate select (bits 5:2). These read/write bits select the rate at which the watch chip generates periodic interrupts. Since this feature is not used, these bits must be set to "0000" to disable it.

Figure 13–2: Watch Date Mode and Format (WATCSRB)

```
       9       8       7       6       5       4       3       2
   +-------+-------+-------+-------+-------+-------+-------+-------+
   |  SET  |  PIE  |  AIE  |  UIE  | SQWE  |  DM   | 24/12 |  DSE  |
   +-------+-------+-------+-------+-------+-------+-------+-------+
```

SET	Set Time (bit 9). When this read/write bit is zero, the time and date registers are updated once per second. When this bit is one, any update cycle in progress is aborted and updates are inhibited so that a program can set new date and time values.
PIE	Periodic Interrupt Enable (bit 8). Not used; must be set to 0.
AIE	Alarm Interrupt Enable (bit 7). Not used; must be set to 0.
UIE	Update Interrupt Enable (bit 6). Not used; must be set to 0.
SQWE	Square-wave Enable (bit 5). Not used; must be set to 0.
DM	Data Mode (bit 4). This read/write bit selects the numeric representation in the time and date registers. If DM is one, the data format is binary; if DM is zero, the data format is two 4-bit decimal digits (BCD).
24/12	Hours Format (bit 3). This read/write bit selects the format of the WAT_HOUR AND WAT_ALMH registers. A value of one selects 24-hour mode; a value of zero selects 12-hour AM/PM mode. In the latter case, bit 7 of the hours registers is zero for AM and one for PM.
DSE	Daylight Saving Enable (bit 2). This read/write bit is zero for normal operation. If set to one, two special time updates occur: on the last Sunday in April the time increments from 01:59:59 AM to 03:00:00 AM, and on the last Sunday in October when the time first reaches 01:59:59 AM it changes to 01:00:00 AM. (This feature is obsolete, since it does not conform to the current start and end dates for Daylight Savings Time)

Figure 13–3: Watch Valid RAM and Time Flag (WATCSRD)

```
    9       8       7       6       5       4       3       2
+-------+-------+-------+-------+-------+-------+-------+-------+
|  VRT  |   0   |   0   |   0   |   0   |   0   |   0   |   0   |
+-------+-------+-------+-------+-------+-------+-------+-------+
```

VRT	Valid RAM and Time (bit 9). This bit indicates whether the contents of the time and RAM registers may have been corrupted by loss of power. This bit is set to zero whenever system power is off and the backup battery voltage drops below the value required for the watch chip to function properly. This bit is set to one after any read of this register (the register may not be written). Programming note: Since ANY read of this register to test the VRT bit will reset that bit to one, a program which finds VRT equal to zero must be prepared to either load valid data into the time and RAM registers or take other action to indicate that the contents of the watch chip are invalid.

<8:2> Not used. Always read as zeros.

13.4 Time of Year Data

The time of year is kept in six registers: WAT_SEC, WAT_MIN, WAT_HOUR, WAT_DAY, WAT_MON, and WAT_YEAR. A seventh register, WAT_DOW, indicates the day of the week (days are numbered from 1 (Sunday) through 7). The contents of each register may be in either binary form or BCD (two 4-bit decimal digits) as selected by register WAT_CSRB bit DM.

The time value is incremented once each second. Such an update requires 1948 microseconds, during which time the date and time register contents are unstable and should not be read by a program. Register WAT_CSRA bit UIP indicates when an update is in progress. This bit is one from 244 microseconds before the beginning of an update cycle until the cycle is complete. Therefore a program should read WAT_CSRA until it finds bit UIP zero, at which time it has at least 244 microseonds to read the date and time registers. The program should inhibit interrupts while reading the registers to ensure that an interrupt does not prolong its reading beyond the 244 microsecond window.

13.5 Non-volatile RAM Storage

The 50 bytes of RAM storage are used by system firmware. The use of these bytes is defined in the KA46 System Firmware specification.

13.6 Initialization

When a program finds the VRT bit equal to zero, it must assume that the contents of all other registers in the chip are invalid. To initialize the chip, a program should:

1. Load register WAT_CSRB with bit SET equal to one to inhibit time updates and bits PIE, AIE, UIE and SQWE equal to zero to disable unused features. Bits TM, 24/12 and DSE should be set for the desired date format.

2. Load the seven time registers with the current date and time.
3. Load register WAT_CSRA to set the proper time base divisor. The DVX bits should be set to "010" and the RSX bits to "0000".
4. Load register WAT_CSRB with the same value used in step 1 except that bit SET should now be zero to enable normal time updating.

As long as the backup battery voltage is sufficient, the contents and operation of the watch chip are not affected by system power-on and power-off events.

Chapter 14

Bus Adaptor Controller

This controller connects to the bus connecting the DC7203 to the DC7201, the CDAL and generates a <TBD> point to point private interconnect bus that runs from the VS4000-400 system enclosure to a second box containing the actual bus card cage.

The initial implementation is for a VME bus.

The DC7201 is signalled that such an adaptor is present on the system by the assertion of one of its inputs - EXIO<1> H - during power-on reset. (see Chapter 7, Section 7.5). When the DC7201 has been told that this adaptor exists, it assigns a range of addresses - <3000.0000 - 3FFF.FFFF> to this adaptor and assumes that the adaptor will respond to the CPU's accessing any addresses in this range. A gross timeout is implemented within the DC7201 for the case where the adaptor is indicated as not present, but an instruction references the assigned adaptor address range.

The DC7201 detects and responds to Interrupt Acknowledge cycles caused by interrupts from the Adaptor, refer to Chapter 6, Section 6.2.2.

The DC7201 can distinguish between cycles generated by the DC7203 and the bus adaptor controller by decoding the CSDP lines. A combination unused by the CPU is assigned to the Bus Adaptor. When the DC7201 detects such a cycle, it reverses its normal operation with respect to the Invalidate Filter, recognising that Adaptor-initiated cycles are equivalent to DMA cycles rather than CPU cycles. Thus an Adaptor initiated quadword read cycle does not cause the invalidate filter RAMs to be updated; an Adaptor initiated write cycle causes a lookup in the Invalidate Filter RAMs resulting in a possible CPU Invalidate request.

It is possible to envision deadlock situations where the CPU is attempting to reference some register within the adaptor or out on the bus at the same time that the bus is requesting control of the CDAL. In such situations it is the responsibility of the bus adaptor to recognise the deadlock and terminate the current CPU cycle with an error. The error is signalled to the CPU by the adaptor's assertion of the control line BARTRY_L prior to its (normal) assertion of CVRDY_L. This causes the DC7203 to terminate the current CPU cycle with a retry request back to the CPU (RTRY_L is asserted instead of MRDY_L). The CPU will subsequently retry this failing (IO) instruction.

Note

In order that retries may operate correctly, Writes to the Bus Adaptor do not use the write buffer mechanism of the DC7203, any such write will cause the CPU to stall until that write has completed. See Chapter 4, Section 4.3.1.

Refer to the Adaptor Specification for complete details.

Chapter 15

OTHER OPTIONS

This chapter discusses the configuration rules for options which are to be connected to a VS4000-400 system using the internal option connector.

15.1 General Rules for Options

In order to avoid hardware conflicts and to accommodate operating system configuration and device driver support constraints, each option must conform to a set of rules which specify:
—its firmware ROM address range;
—its control and status register address range;
—which interrupt(s) it uses.

Every option board must have a ROM to contain its device type identifier and diagnostic program. The device type ID values are specified in the VS4000-400 Power-On Initialization Specification. Whenever a new option is designed, a new device type ID value must be assigned and added to that specification.

An option's control and status registers (CSRs) must begin at the beginning of the address range specified by the rule; they may extend as far as necessary within that range and may occupy discontiguous subranges.

15.2 Interrupts

One interrupt is provided for use by options connected to the internal option connector: SC. This is the higest priority interrupt of the eight supported by the DC7201, see Chapter 6, Section 6.3.

15.3 Interface

The internal option connector data bus is only 16-bits wide, connecting physically to EDAL<15:00> - the low half of the I/O bus of the DC7201. For the address range allocated to this option slot, the DC7201 will perform two transfers for each CPU access to this address range, allowing arbtitrary length transfers up to a full longword in a single instruction. The DC7201 always reads or writes the low word of a longword address first. Note that although this is different from the operation of the VS3100, no programming changes are needed.

15.4 Specific Options

One option has been defined so far for the VS4000-400:

—PV21X-DA synchronous communications adaptor.

This option is described in a separate specification.

Chapter 16

Connectors

The VS4000-400 system board carries 22 connectors. Some of these are directly accessible at the rear of the machine, some are for connection of internal options only.

Figure 16–1: System Board connector Placement

```
                             R E A R

   J1   S1     J2          J3              J4       J5     J8    J7  J6
        |                                +---+
   -  +----+ +----+   +-----------+    +-|   |-+   +--+  +-----+ +-+ +-+
+-| |-|    |-|    |---|           |----|       |---|  |--|     |-| |-| |---+
| |_| +----+ +----+   +-----------+    +-------+   +--+  +-----+ +-+ +-+   |
|                                                                          |
|                                                                          |
|                           J30                                            |
|                      +-----------+                                       |
|                                                                          |
|  +------+ J10                                                            |
|                                                                          |
|                                    J31                                   |
|                               +-----------+                              |
|                           J14                                            |
|                 +-----------------------+          J15 +---------------+ |
|                                                    J16 +-------------+   |
|                                                                          |
|           J20  +------------------+                                     |
|           J21  +------------------+                                     |
|           J22  +------------------+                                     |
|           J23  +------------------+  J17                                |
|           J24  +------------------+   +                                 |
|           J25  +------------------+   |             J18                 |
|                                       +          +---------+            |
+--------------------------------------------------------------------------+

                            F R O N T
                     VIEW FROM COMPONENT SIDE
```

16.1 Power - J10

The power supply connects to the system module by this 18-pin connector. Power for the internal SCSI devices runs via etch on the system module to J17 - see below.

Table 16–1: Power Pinout

Pin	Signal
1	+3.3V
2	+3.3V
3	GND
4	GND
5	GND
6	+5V
7	+5V
8	+5V
9	DCOK
10	+12V
11	GND
12	+5V
13	+5V
14	GND
15	GND
16	-12V
17	-9V RET
18	-9V

16.2 Memory - J<20:25>

There are six, 80-pin SIMM connectors to allow memory to be added above the base 8 MBytes carried on the system module. The rules for population of these connectors are given at the start of Chapter 4. As memory is always added as pairs of SIMMs, there are two pinouts, J20, 22 and 24 carry the high 32-bits of memory data, J21, 23 and 25 the low 32-bits.

Table 16–2: Memory SIMM Pinout - J20, 22, 24

Pin	Signal	Comments
1	MEMSIZE	GND indicates SIMM Present
2	16MEG	GND indicates 16MByte SIMM
3	DATA>32>	IO
4	+5V	
5	DATA<48>	IO
6	DATA<33>	IO
7	DATA<56>	IO
8	+5V	
9	DATA<41>	IO
10	+5V	
11	DATA<49>	IO
12	DATA<57>	IO

Table 16–2 (Cont.): Memory SIMM Pinout - J20, 22, 24

Pin	Signal	Comments
13	DATA<34>	IO
14	GND	
15	DATA<42>	IO
16	DATA<50>	IO
17	DATA<58>	IO
18	GND	
19	DATA<35>	IO
20	DATA<43>	IO
21	DATA<51>	IO
22	+5V	
23	DATA<59>	IO
24	ADDR<5>	
25	ADDR<6>	
26	+5V	
2	ADDR<7>	
28	ADDR<8>	
29	GND	
30	CAS<6>	
31	GND	
32	RAS<1>	
33	GND	
34	ADDR<9>	
35	ADDR<10>	
36	CAS<4>	
37	GND	
38	CAS<5>	
39	+5V	
40	ADDR<0>	
41	ADDR<1>	
42	+5V	
43	WRITE	
44	GND	
45	CAS<7>	
46	GND	
47	ADDR<2>	
48	GND	
49	ADDR<3>	
50	GND	
51	ADDR<4>	
52	GND	
53	DATA<36>	IO
54	DATA<44>	IO
55	+5V	
56	DATA<52>	IO
57	DATA<60>	IO
58	DATA<37>	IO
59	+5V	
60	DATA<45>	IO

Table 16–2 (Cont.): Memory SIMM Pinout - J20, 22, 24

Pin	Signal	Comments
61	DATA<53>	IO
62	DATA<61>	IO
63	GND	
64	DATA<38>	IO
65	DATA<46>	IO
66	DATA<54>	IO
67	GND	
68	DATA<62>	IO
69	DATA<39>	IO
70	DATA<47>	IO
71	+5V	
72	DATA<55>	IO
73	DATA<63>	IO
74	PARITY<4>	IO
75	+5V	
76	PARITY<5>	IO
77	PARITY<6>	IO
78	PARITY<7>	IO
79	GND	
80	GND	

Table 16–3: Memory SIMM Pinout - J21, 23, 25

Pin	Signal	Comments
1	MEMSIZE	GND indicates SIMM Present
2	16MEG	GND indicates 16MByte SIMM
3	DATA>00>	IO
4	+5V	
5	DATA<08>	IO
6	DATA<16>	IO
7	DATA<24>	IO
8	+5V	
9	DATA<01>	IO
10	+5V	
11	DATA<09>	IO
12	DATA<17>	IO
13	DATA<25>	IO
14	GND	
15	DATA<02>	IO
16	DATA<10>	IO
17	DATA<18>	IO
18	GND	
19	DATA<03>	IO
20	DATA<11>	IO
21	DATA<19>	IO
22	+5V	
23	DATA<27>	IO

Table 16–3 (Cont.): Memory SIMM Pinout - J21, 23, 25

Pin	Signal	Comments
24	ADDR<5>	
25	ADDR<6>	
26	+5V	
2	ADDR<7>	
28	ADDR<8>	
29	GND	
30	CAS<2>	
31	GND	
32	RAS<1>	
33	GND	
34	ADDR<9>	
35	ADDR<10>	
36	CAS<0>	
37	GND	
38	CAS<1>	
39	+5V	
40	ADDR<0>	
41	ADDR<1>	
42	+5V	
43	WRITE	
44	GND	
45	CAS<3>	
46	GND	
47	ADDR<2>	
48	GND	
49	ADDR<3>	
50	GND	
51	ADDR<4>	
52	GND	
53	DATA<04>	IO
54	DATA<12>	IO
55	+5V	
56	DATA<20>	IO
57	DATA<28>	IO
58	DATA<05>	IO
59	+5V	
60	DATA<13>	IO
61	DATA<21>	IO
62	DATA<29>	IO
63	GND	
64	DATA<06>	IO
65	DATA<14>	IO
66	DATA<22>	IO
67	GND	
68	DATA<30>	IO
69	DATA<07>	IO
70	DATA<15>	IO
71	+5V	

Table 16–3 (Cont.): Memory SIMM Pinout - J21, 23, 25

Pin	Signal	Comments
72	DATA<23>	IO
73	DATA<31>	IO
74	PARITY<0>	IO
75	+5V	
76	PARITY<1>	IO
77	PARITY<2>	IO
78	PARITY<3>	IO
79	GND	
80	GND	

16.3 Video Module - J14

A single, 150-pin connector allows for one of several video bitmap/DAC modules to be added to the system module. Video modules currently proposed are :

Table 16–4: Video Controllers

Option #	Description	Resolution and Refresh Rate	Monitor
PV21X-GB	Gray scale, 4-plane	1280 x 1024 x 72Hz	VR319
PV21X-GC	Gray scale, 8-plane	640 x 480 x 60 Hz	PC4XV-AA or equivalent
PV21X-GC	Color, 8-plane	1024 x 768 x 60 Hz	VRT13
PV21X-GC	Gray scale, 8-plane	1024 x 864 x 60 Hz	VR260/262
PV21X-GC	Color, 8-plane	1024 x 864 x 60 Hz	VR290/VR297
PV21X-GD	Color, 8-plane	1280 x 1024 x 66 Hz	VRT16/ VRT19/VR320

Each module carries its own connections to the appropriate monitor which are accessible directly at the rear of the system enclosure.

Table 16–5: Video Bitmap Module Pinout - J14

Pin	Signal	Comments
A1	+3.3V	
A2	VSYNC	
A3	VBLANK	
A4	CUR<6>	
A5	CUR<4>	
A6	CUR<2>	
A7	CUR<0>	
A8	DMSEL/MSEL1	
A9	FBENA	

Table 16–5 (Cont.): Video Bitmap Module Pinout - J14

Pin	Signal	Comments
A10	BRESET	
A11	VMXSEL0	
A12	LCGDSF	
A13	MDAL<30>	
A14	MDAL<28>	
A15	MDAL<26>	
A16	MDAL<24>	
A17	MDAL<22>	
A18	MDAL<20>	
A19	MDAL<18>	
A20	MDAL<16>	
A21	MDAL<14>	
A22	MDAL<12>	
A23	MDAL<10>	
A24	MDAL<08>	
A25	MDAL<06>	
A26	MDAL<04>	
A27	MDAL<02>	
A28	MDAL<00>	
A29	VBAOE<1>	
A30	VBAOE<0>	
A31	VABOE	
A32	ADDR<10>	
A33	ADDR<08>	
A34	ADDR<06>	
A35	ADDR<04>	
A36	ADDR<02>	
A37	ADDR<00>	
A38	LCAS<6>	
A39	LCAS<4>	
A40	LCAS<2>	
A41	LCAS<0>	
A42	LRAS<2>	
A43	LRAS<0>	
A44	MCASEN	
A45	EDAL<6>	
A46	EDAL<4>	
A47	EDAL<2>	
A48	EDAL<0>	
A49	EADR<4>	
A50	EADR<2>	
B1	+5V	
B2	GND	
B3	GND	
B4	GND	
B5	GND	
B6	GND	
B7	GND	

Table 16–5 (Cont.): Video Bitmap Module Pinout - J14

Pin	Signal	Comments
B8	GND	
B9	GND	
B10	GND	
B11	GND	
B12	GND	
B13	GND	
B14	GND	
B15	+5V	
B16	GND	
B17	GND	
B18	GND	
B19	GND	
B20	GND	
B21	GND	
B22	GND	
B23	GND	
B24	GND	
B25	-12V	
B26	GND	
B27	GND	
B28	GND	
B29	GND	
B30	GND	
B31	GND	
B32	GND	
B33	GND	
B34	GND	
B35	+5V	
B36	GND	
B37	GND	
B38	GND	
B39	GND	
B40	GND	
B41	GND	
B42	GND	
B43	VID	
B44	GND	
B45	GND	
B46	GND	
B47	GND	
B48	GND	
B49	GND	
B50	+5V	
C1	+3.3V	
C2	VSHIFT	
C3	CUR<7>	
C4	CUR<5>	
C5	CUR<3>	

Table 16–5 (Cont.): Video Bitmap Module Pinout - J14

Pin	Signal	Comments
C6	CUR<1>	
C7	BTEN	
C8	EXVR	
C9	VROMCS	
C10	NIBCLK	
C11	LCGDT	
C12	MDAL<31>	
C13	MDAL<29>	
C14	MDAL<27>	
C15	+12V	
C16	MDAL<25>	
C17	MDAL<23>	
C18	MDAL<21>	
C19	MDAL<19>	
C20	MDAL<17>	
C21	MDAL<15>	
C22	MDAL<13>	
C23	MDAL<11>	
C24	MDAL<09>	
C25	MDAL<07>	
C26	MDAL<05>	
C27	MDAL<03>	
C28	MDAL<01>	
C29	LENA2B<1>	
C30	LENA2B<0>	
C31	LENB2A	
C32	WRITE	
C33	ADDR<09>	
C34	ADDR<07>	
C35	ADDR<05>	
C36	ADDR<03>	
C37	ADDR<01>	
C38	CAS<7>	
C39	CAS<5>	
C40	CAS<3>	
C41	CAS<1>	
C42	RAS<3>	
C43	RAS<1>	
C44	RASEN	
C45	EDAL<7>	
C46	EDAL<5>	
C47	EDAL<3>	
C48	EDAL<1>	
C49	EADR<5>	
C50	EADR<3>	

16.4 Option connector - J15

A single, 64-pin connector allows for the synchronous communications adaptor, see Chapter 15, to be added to the system module. This module has its own connector accessible at the rear of the system enclosure - see separate specificcation for full details of this connector.

Table 16–6: Option Connector Pinout - J14

Pin	Signal	Comments
1	OPT	Option Present
2	+12V	
3	GND	
4	-12V	
5	GND	
6	+5V	
7	GND	
8	+5V	
9	EXAS	
10	GND	
11	EXDS	
12	EXRDY	
13	GND	
14	EXWR	
15	OPTENA	
16	GND	
17	N.C.	
18	EADR<23>	
19	GND	
20	EADR<22>	
21	EADR<21>	
22	GND	
23	EADR<20>	
24	EXIO<3>	
25	GND	
26	EXIO<2>	
27	EXIO<1>	
28	GND	
29	EXIO<0>	
30	EDAL<15>	
31	GND	
32	EDAL<14>	
33	EDAL<13>	
34	GND	
35	EDAL<12>	
36	EDAL<11>	
37	GND	
38	EDAL<10>	
39	EDAL<09>	
40	GND	
41	EDAL<08>	
42	EDAL<07>	

Table 16–6 (Cont.): Option Connector Pinout - J14

Pin	Signal	Comments
43	GND	
44	EDAL<06>	
45	EDAL<05>	
46	GND	
47	EDAL<04>	
48	EDAL<03>	
49	GND	
50	EDAL<02>	
51	EDAL<01>	
52	GND	
53	EDAL<00>	
54	EXBM<1>	
55	GND	
56	EXBM<0>	
57	+5V	
58	GND	
59	+5V	
60	GND	
61	SYNIRQ	
62	GND	
63	BRESET	
64	GND	

16.5 Lights and Switches Panel - J18

The LED system status display, HALT switch, Alternate Console select switch, Audio IN and Audio OUT connectors are located on a small sub-assembly accessible at the front of the system enclosure. This connects to the system module via a 40-pin connector.

Viewed from the front of the system enclosure, the LEDs are at the left of this panel, with the Most Significant Bit at the extreme left; next is the Alternate Console Switch - UP for Alternate Console, DOWN for normal; next the HALT switch - momentary push to HALT; then the Audio IN connector - a $\frac{1}{4}$" jack and at the right of the panel, the Audio OUT connector - a $\frac{1}{8}$" jack.

16.5.1 Audio In/Out

With nothing plugged into the Audio OUT jack, the output of the sound chip is connected to an internal audio transducer. When a plug is inserted, this internal connection is broken. Only two pins of the jack are used, tip carries the signal, barrel is ground. Similarly, the Audio IN jack accepts audio on Tip with the barrel ground.

Table 16–7: Lights/Switches Connector Pinout - J18

Pin	Signal	Comments
1	+5V	

Table 16–7 (Cont.): Lights/Switches Connector Pinout - J18

Pin	Signal	Comments
2	GND	
3	LED<0>	
4	LED<1>	
5	LED<2>	
6	LED<3>	
7	LED<4>	
8	LED<5>	
9	LED<6>	
10	LED<7>	
11	GND	
12	GND	
13	+12V	
14	N.C.	
15	-12V	
16	GND	
17	FERENA	
18	PTRFER	
19	HALT	
20	GND	
21	GND	
22	GND	
23	N.C.	
24	GND	
25	GND	
26	GND	
27	GND	
28	GND	
29	AUDIOGND	
30	AUDIOGND	
31	AUDIOIN	
32	AUDIOGND	
33	AUDIOGND	
34	AUDIOGND	
35	AUDIOOUT	
36	AUDIOGND	
37	AUDIOGND	
38	AUDIOGND	
39	GND	
40	+5V	

16.6 Bus Adaptor - J3, 30, 31

Two 64-pin connectors allow for the attachment of the Bus Adaptor Controller to the system board. One of these (J31) supplies the CDAL bus signals to the controller, the other (J30) takes the external bus inputs/output of the controller and routes these to a third, 100-pin connector (J3) which is accessible at the rear of the system enclosure. The interconnect between J3 and J30 is given in the Bus Adaptor Controller Specification, it is not repeated here.

Table 16–8: Bus Adaptor Controller CDAL Pinout - J31

Pin	Signal	Comments
1	CDAL<31>	
2	CDAL<30>	
3	CDAL<29>	
4	CDAL<28>	
5	CDAL<27>	
6	CDAL<26>	
7	CDAL<25>	
8	CDAL<24>	
9	CDAL<23>	
10	CDAL<22>	
11	CDAL<21>	
12	CDAL<20>	
13	CDAL<19>	
14	CDAL<18>	
15	CDAL<17>	
16	CDAL<16>	
17	CDAL<15>	
18	CDAL<14>	
19	CDAL<13>	
20	CDAL<12>	
21	CDAL<11>	
22	CDAL<10>	
23	CDAL<09>	
24	CDAL<08>	
25	CDAL<07>	
26	CDAL<06>	
27	CDAL<05>	
28	CDAL<04>	
29	CDAL<03>	
30	CDAL<02>	
31	CDAL<01>	
32	CDAL<00>	
33	CVBM<3>	
34	CVBM<2>	
35	CVBM<1>	
36	CVBM<0>	
37	CSDP<3>	
38	CSDP<2>	
39	CSDP<1>	
40	CSDP<0>	
41	CPUCLKA	
42	N.C.	
43	CVAS	
44	CVDS	
45	CVWR	
46	CVREADY	
47	CVDMR	
48	CVDMG	

Table 16–8 (Cont.): Bus Adaptor Controller CDAL Pinout - J31

Pin	Signal	Comments
49	CVDMGO	
50	CPRESET	
51	ADAPT_IRQ	
52	ADP_RTY	
53	ADAPT_PRES	
54	GND	
55	GND	
56	GND	
57	GND	
58	GND	
59	GND	
60	GND	
61	GND	
62	GND	
63	GND	
64	GND	
65	GND	
66	GND	
67	GND	
68	GND	
69	GND	
70	N.C.	
71	GND	
72	GND	
73	GND	
74	GND	
75	GND	
76	GND	
77	GND	
78	GND	
79	GND	
80	N.C.	
81	GND	
82	GND	
83	GND	
84	GND	
85	GND	
86	GND	
87	GND	
88	GND	
89	GND	
90	N.C.	
91	GND	
92	GND	
93	GND	
94	GND	
95	GND	
96	GND	

16.7 SCSI - J16

A single, 50-pin connector connects to the SCSI controller of the system module. Termination of this end of the SCSI bus is made on the system module.

From this connector, daisy chain connections are made to the possible three SCSI devices within the system enclosure and then to a 50-pin CHAMP ^{tm} connector accessible at the rear of the system enclosure. A SCSI terminator plugs into this connector if no expansion devices are connected.

Table 16–9: SCSI Bus Pinout - J16

Pin	Signal	Comments
1	GND	
2	SCD<0>	See Chapter 10,Section 10.2
3	GND	
4	SCD<1>	
5	GND	
6	SCD<2>	
7	GND	
8	SCD<3>	
9	GND	
10	SCD<4>	
11	GND	
12	SCD<5>	
13	GND	
14	SCD<6>	
15	GND	
16	SCD<7>	
17	GND	
18	SCDIP	
19	GND	
20	GND	
21	GND	
22	GND	
23	GND	
24	GND	
25	N.C.	
26	TERMPWR	Protected by self-resetting fuse
27	GND	
28	GND	
29	GND	
30	GND	
31	GND	
32	SCATN	
33	GND	
34	GND	
35	GND	
36	SCBSY	
37	GND	
38	SCACK	

Table 16–9 (Cont.): SCSI Bus Pinout - J16

Pin	Signal	Comments
39	GND	
40	SCRST	
41	GND	
42	SCMSG	
43	GND	
44	SCSEL	
45	GND	
46	SCC/D	
47	GND	
48	SCREQ	
49	GND	
50	SCI/O	

16.8 SCSI Power - J17

A single 4-pin power connector supplies SCSI device power from the system module.

Table 16–10: SCSI Power Pinout - J17

Pin	Signal	Comments
1	+12V	
2	GND	
3	GND	
4	+5V	

16.9 Local Area Network - J1, J2

J1 is a BNC for Thinwire and J2 a 15-pin D-subminiature for Transceiver connect for the Ethernet ^{tm} LAN. The active connector is selected by S1, located between J1 and J2. The unselected connector does not require a terminator.

16.10 Communications - J4

J4 is a 25-pin D-subminiature connector allowing modem connect for serial line #2 of the four serial lines supported as a standard part of the system module. See Chapter 11 for signal definitions.

Table 16–11: Communications Connector

Pin	Signal	DIR	Comments
1	N.C.		May be connected to GND if W1 is installed
2	XMT	OUT	CCITT Circuit 103
3	RCV	IN	CCITT Circuit 104
4	RTS	IN	CCITT circuit 105

Table 16–11 (Cont.): Communications Connector

Pin	Signal	DIR	Comments
5	CTS	IN	CCITT Circuit 106
6	DSR	IN	CCITT Circuit 107
7	GND	NA	CCITT Circuit 102
8	DCD	IN	CCITT Cicruit 109
9	NC		
10	NC		
11	NC		
12	SPDI	IN	CCITT Cicruit 112
13	NC		
14	NC		
15	NC		
16	NC		
17	NC		
18	LLBK	OUT	CCITT Cicruit 141
19	NC		
20	DTR	OUT	CCITT Circuit 108/2
21	NC		
22	RI	IN	CCITT Circuit 125
23	DSRS	OUT	CCITT Circuit 111
24	NC		
25	TMI	IN	CCITT Cicruit 141

16.11 Printer - J5

J5 is a 6-pin MMJ jack supporting DEC423 data leads only connection. It is intended for use with a local printer and is connected to serial line #3 of the four serial lines supported as a standard part of the system module.

Table 16–12: DEC423 Connector

Pin	Signal	DIR	Comments
1	HIGH	OUT	150 Ω to +5V
2	XMT	OUT	
3	GND	NA	
4	RCV+	IN	
5	RCV-	IN	
6	TERM	IN	3K Ω to GND

16.12 Keyboard - J6

J6 is a 4-pin MJ jack intended for the connection of an LK401 keyboard. It is connected to serial line #0 of the four serial lines supported as a standard part of the system module.

Table 16–13: Keyboard Connector

Pin	Signal	DIR	Comments
1	RCV	IN	
2	+12V	OUT	Protected by self-resetting fuse
3	GND	NA	
4	XMT	OUT	

16.13 Mouse/Tablet - J7

J7 is an 8-pin DIN connector intended for connection of a VSXXX-AA or VSXXX-AB. It is connected to serial line #1 of the four serial lines supported as a standard part of the system module.

Table 16–14: Keyboard Connector

Pin	Signal	DIR	Comments
1	GND	NA	
2	XMT	OUT	
3	RCV	IN	
4	-12V	OUT	Protected by self-resetting fuse
5	+5V	OUT	Protected by self-resetting fuse
6	+12V	OUT	Protected by self-resetting fuse
7	NC		
8	NC		
9	NC		

16.14 Mouse/Tablet/Keyboard - J8

J8 is a 15-pin D-subminiature connector to which both the serial lines (#0 and #1) are connected to allow for a remote installation of both keyboard and pointing device (cable is 17-02640-01 - 10 feet).

Table 16–15: Keyboard Connector

Pin	Signal	DIR	Comments
1	GND	NA	
2	KBD_XMT	OUT	
3	KBD_RCV	IN	
4	+12V	OUT	Protected by self-resetting fuse
5	N.C.		
6	POINT_RCV	IN	Mouse/Tablet
7	POINT_XMT	OUT	Mouse/Tablet
8	GND	NA	
9	GND	NA	
10	N.C.		
11	N.C.		

Table 16–15 (Cont.): Keyboard Connector

Pin	Signal	DIR	Comments
12	N.C.		
13	+5V	OUT	Protected by self-resetting fuse
14	-12V	OUT	Protected by self-resetting fuse
15	GND	NA	

Appendix A
PHYSICAL ADDRESS MAP

A.1 System Module

The following addresses are defined for those sections of the VS4000-400 integral to the KA46 System Module.

Address(es)	**Name**	**R/W**	**Useage**
0000.0000 - 07FF.FFFF			Read/Write memory
2000.0000 - 2001.FFFF			DMA Translation Map Registers
0800.0000 - 0803.FFFF		(RW)	Secondary Cache Data Store diagnostic access range
2D00.0000 - 2D03.FFFF		(RW)	Secondary Cache Tag Store diagnostic access range
2C00.0000		(RW)	Secondary Cache Control Register
2400.0000 - 2BFF.FFFF		(RW)	Main Memory I/O space alternate access range
2002.0000	CFGTST	(R)	Configuration/Test Register
2002.0000	IORESET	(W)	I/O Reset
2020.0000 - 2021.FFFF			Invalidate Filter RAM
2004.0000 - 2007.FFFF			System ROM (256K Bytes)

Address(es)	Name	R/W	Useage
2008.0000	HLTCOD1		Halt Code Register 1
2008.0004	HLTCOD2		Halt Code Register 2
2008.0008	MAP_BASE		DMA Translation Map Base Register
2008.000C	INTMSK		Interrupt Mask Register
2008.000E			RESERVED
2008.000F	INTREQ	(R)	Interrupt Request Pending
2008.000F	INTREQ	(W)	Interrupt Request Clear
2008.0010	DIAG_DISP		Diagnostic Display Register
2008.0014	PAR_CTL		Parity Control Register
2008.0018			RESERVED
2008.001C	DIAGTIMU		Diagnostic Timer, High Resolution
2008.001E	DIAGTIMM		Diagnostic Timer, Low Resolution
2009.0000 - 2009.007F		(R)	Network Address ROM
200A.0000	SEER_CSR		DZ CSR
200A.0004	SER_RBUF	(R)	DZ Receive Buffer Register
200A.0004	SER_LPR	(W)	DZ Receive Line Parameter Register
200A.0008	SER_TCR		DZ Transmit Control Register
200A.000C	SER_MSR	(R)	DZ Modem Ststus Register
200A.000C	SER_TDR		DZ transmit Data Register
200A.0010	SER_LPR0	(R)	DZ Line Parameter Register 0
200A.0014	SER_LPR1	(R)	DZ Line Parameter Register 1
200A.0018	SER_LPR2	(R)	DZ Line Parameter Register 2

Address(es)	Name	R/W	Useage
200A.001C	SER_LPR3	(R)	DZ Line Parameter Register 3
200B.0000 - 200B.00FF			TOY Clock/NVR
200C.0000	SCSI_ADR		SCSI controller DMA Address Register
200C.000C	SCSI_DIR		SCSI DMA Direction Register
200C.0080 - 200C.00BF	SCSI_XXX		SCSI Controller Chip Registers
200D.0000 - 200D.00FF			Sound Chip Registers
200E.0000 - 200E.0007			LANCE Chip Registers
200F.0000 - 200F.00FF			SPARE
2010.0000 - 2017.FFFF			GC Registers and ROM
2018.0000 - 201F.FFFF			GC FIFO
2100.0000 - 22FF.FFFF			GC Frame Buffer
2C00.0000 - 2CFF.FFFF			Option Addresses, assigned to internal option slot.
3000.0000 - 3FFF.FFFF			CDAL Adaptor address range

A.2 Option Board Address Ranges

The following address ranges are defined for use by option boards connected to the synchronous communications controller option and video frame buffer connectors. This section lists the nominal ranges; subsequent sections show the actual ranges used by each option type.

2100.0000 - 22FF.FFFF	Video Frame Buffers
2014.0000 - 2017.FFFF	Video Frame Buffer Module ROM
2C00.0000 - 2CFF.FFFF	Synchronous Communications Adaptor

Appendix B

REVISION HISTORY

Revision X00—Initial review draft.

Revision X01—Corrected Invalidate Filter RAM init. address range. Many minor corrections.
Revision X02—Added Connector Chapter. Renamed PAR_CTL register to BWF0. Added note on initialization of CPU Primary Cache TAG Array. Added stall-on-write for writes to the Bus Adaptor to ensure that retries function correctly. Corrected Graphics allocated address range to include 2200.0000 - 22FF.FFFF and moved Secondary Cache Tag Store Diagnostic Address Range to 2D00.0000 - 2D03F.FFFF. Added Interrupt controller vector generation for IPL16 and 17. Added Sound Generation chapter.