# DECstation 3100 Desktop Workstation Functional Specification

## Revision 1.3

Workstation Systems Engineering
Digital Equipment Corporation
100 Hamilton Avenue
Palo Alto, CA 94301

August 28, 1990

**August 1990**

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibilty for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

Digital Equipment Coporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Digital or its affiliated companies.

© Digital Equipment Corporation 1990.
All Rights reserved
Printed in U.S.A.

The following are trademarks of Digital Equipment Corporation:

| | | | | |
|---|---|---|---|---|
| DEC | DECnet | DECstation | DECsystem | DECUS |
| MicroVAX | MicroVMS | PDP | TURBOchannel | ULTRIX |
| ULTRIX-32 | UNIBUS | VAX | VAXBI | VAXcluster |
| VAXstation | VMS | VT | | |

--

--

Blank Page

## Table of Contents


**1. DECstation 2100/3100 Desktop Workstation**

**2. External Interface**

**3. Power Requirements**

**4. Address Map**

**5. Interrupts**

**6. Subsystems**

6.1. Processor ..... 4

6.2. Memory ..... 5

6.3. ROM ..... 6

6.4. Serial Interface ..... 6

6.4.1. Control And Status Register ..... 6
6.4.2. Receiver Buffer Register ..... 8
6.4.3. Line Parameter Register ..... 9
6.4.4. Transmit Control Register ..... 11
6.4.5. Modem Status Register ..... 11
6.4.6. Transmit Data Register ..... 12

6.5. SCSI Interface ..... 13

6.5.1. SII Registers ..... 14

6.5.1.1. SII_SDB - SCSI Data Bus ..... 15
6.5.1.2. SII_SC1 - SCSI Control Signals One ..... 15
6.5.1.3. SII_SC2 - SCSI Control Signals Two ..... 16
6.5.1.4. SII_CSR - Control/Status Register ..... 16
6.5.1.5. SII_ID - Bus ID Register ..... 17
6.5.1.6. SII_SLCSR - Selector Control And Status Register ..... 17
6.5.1.7. SII_DESTAT - Selection Detector Status Register ..... 18
6.5.1.8. SII_DATA - Data Register ..... 18
6.5.1.9. SII_DMCTRL - DMA Control Register ..... 18
6.5.1.10. SII_DMLOTC - DMA Length Of Transfer Counter ..... 19

6.5.1.11. SII_DMADDR - DMA Address Registers ..... 19
6.5.1.12. SII_DMABYTE - DMA Initial Byte Register ..... 19
6.5.1.13. SII_CSTAT - Connection Status Register ..... 20
6.5.1.14. SII_DSTAT - Data Transfer Status Register ..... 22
6.5.1.15. SII_COMM - Command Register ..... 22
6.5.1.16. SII_DICTRL - Diagnostic Control Register ..... 24

6.5.2. Commands ..... 24

6.5.2.1. Immediate Commands ..... 24

6.5.2.1.1. Chip Reset ..... 24
6.5.2.1.2. Disconnect ..... 25

6.5.2.2. Complex Commands ..... 25

6.5.2.2.1. Request Data ..... 25
6.5.2.2.2. Select ..... 25
6.5.2.2.3. Information Transfer Command ..... 26

6.5.3. SCSI Operations ..... 27

6.5.3.1. Initiator Selection Of A Target ..... 27
6.5.3.2. Initiator Selection With ATN Of A Target ..... 28
6.5.3.3. Target Reselection Of An Initiator ..... 28
6.5.3.4. Information Transfers ..... 29

6.5.3.4.1. Initiator DMA Information Transfers ..... 29
6.5.3.4.2. Initiator Programmed I/O Transfers ..... 30
6.5.3.4.3. Target DMA Information Transfers ..... 31
6.5.3.4.4. Target Programmed I/O Transfers ..... 32

6.5.3.5. Initiator Setting ATN ..... 33
6.5.3.6. SII Setting RST ..... 33
6.5.3.7. Command Chaining ..... 33

6.6. Network Interface ..... 36

6.6.1. LANCE Chip Overview ..... 36
6.6.2. Programming of the LANCE ..... 37

6.6.2.1. Register Address Port (LANCE_RAP) ..... 37
6.6.2.2. Register Data Port (LANCE_RDP) ..... 38
6.6.2.3. Control and Status Register 0 (LANCE_CSR0) ..... 38
6.6.2.4. Control and Status Register 1 (LANCE_CSR1) ..... 41
6.6.2.5. Control and Status Register 2 (LANCE_CSR2) ..... 42
6.6.2.6. Control and Status Register 3 (LANCE_CSR3) ..... 42

6.6.3. Interrupts ..... 43
6.6.4. DMA Operation ..... 43
6.6.5. Initialization Block ..... 43

6.6.5.1. Initialization Block MODE Word (NIB_MODE) ..... 44
6.6.5.2. Network Physical Address (NIB_PADR) ..... 46
6.6.5.3. Multicast Address Filter Mask (NIB_LADRF) ..... 47
6.6.5.4. Receive ..... 47
6.6.5.5. Transmit Descriptor Ring Pointer (NIB_TDRP) ..... 48

6.6.6. Buffer Management ..... 49

6.6.6.1. Receive Buffer Descriptor ..... 50
6.6.6.2. Transmit Buffer Descriptor ..... 51

6.6.7. LANCE Operation ..... 54

6.6.7.1. Switch Routine ..... 54
6.6.7.2. Initialization Routine ..... 54
6.6.7.3. Look-for-work Routine ..... 55
6.6.7.4. Receive Poll Routine ..... 55
6.6.7.5. Receive Routine ..... 55
6.6.7.6. Receive DMA Routine ..... 56
6.6.7.7. Transmit Poll Routine ..... 56
6.6.7.8. Transmit Routine ..... 56
6.6.7.9. Transmit DMA Routine ..... 56
6.6.7.10. Collision Detect Routine ..... 57
6.6.7.11. LANCE Programming Notes ..... 57

6.6.8. Ethernet Station Address ROM ..... 59

6.7. System Clock/Battery-backed-up RAM ..... 60

6.7.1. RTC Registers ..... 61

6.7.1.1. Control Register A ..... 61
6.7.1.2. Control Register B ..... 62
6.7.1.3. Control Register C ..... 63
6.7.1.4. Control Register D ..... 63
6.7.1.5. Time of Year Registers ..... 64
6.7.1.6. Non-volatile RAM Storage ..... 64
6.7.1.7. Initialization ..... 64

6.8. System Control and Status Register ..... 64

6.9. Programmable Cursor ..... 68

6.9.1. Cursor Generation ..... 68
6.9.2. Cursor Coordinates ..... 68
6.9.3. PCC Registers ..... 69
6.9.4. Cursor Command Register (PCC_CMDR) ..... 69
6.9.5. Loading the Cursor Sprite Pattern ..... 70
6.9.6. Cursor Region Detectors ..... 71
6.9.7. Displaying a Sprite Cursor ..... 71
6.9.8. Displaying a Crosshair Cursor ..... 71
6.9.9. Controlling Cursor Plane Outputs ..... 71
6.9.10. Blanking the Display ..... 72
6.9.11. Cursor Chip Test ..... 72
6.9.12. Generating Video Interrupts ..... 72
6.9.13. Sampling the VDAC Outputs ..... 72

6.10. Color Map and Video DAC ..... 73

6.10.1. Updating the Color Map ..... 73

6.10.2. Updating the Overlay Map ..... 74
6.10.3. Grey-scale Displays ..... 74

6.11. Color Plane Mask ..... 74

6.12. Monochrome Frame Buffer ..... 74

6.13. Color Frame Buffer ..... 75

6.14. Video Timing ..... 75

6.15. Reset Switch ..... 76

## Revision History

| Date | Version | Content/Changes |
|---|---|---|
| 22 Aug 88 | 1.0 | First release |
| 3 Nov 88 | 1.1 | Added parity disable bit to system CSR<br>Specified VDAC output comparator resolution |
| 24 Mar 89 | 1.2 | Updated power supply requirements<br>Updated reliability data<br>Added video timing<br>Changed to company confidential distribution |
| 23 Aug 89 | 1.3 | Changed name to DS3100<br>Added references to DS2100<br>Eliminated reliability data<br>Changed to general distribution |

--

--

Blank Page

# 1. DECstation 2100/3100 Desktop Workstation

This is the functional specification for the DECstation 3100/2100 desktop workstations. The DS3100 has a processor with slightly more than 11 times the performance of a VAX 11/780, up to 24 Mbytes of memory, and integral disk, network, serial and monochrome or color bitmap graphics interfaces. The DS2100 is identical to the DS3100, except for running at 12MHz, only differing by the lower cost processor and cache chips the slower clock speed allows, with a performance slightly greater than eight times a VAX 11/780. Elsewhere in this document, DS3100 will be used to refer to either machines. This specification describes the workstation subsystems, and describes the software interface of VLSI devices on the module.

The DS3100 desktop workstation is designed for use in a diskless networked environment, in a networked environment with local SCSI peripherals, or as a stand-alone workstation. DS2100/3100 contains the following subsystems on the KN01 system module:

- R2000/R2010/R2020 12.0 or 16.67 MHz CPU/FPU/WB
- 64 KByte instruction cache
- 64 KByte data cache
- 4 to 24 Mbytes of parity protected memory in 4 MByte increments
- 256 KByte boot ROM
- Serial lines for keyboard, mouse, modem, and printer
- Disk/tape interface for SCSI peripherals
- Network interface for ThickWire Ethernet
- Integral ThinWire Ethernet transceiver
- System clock
- 50-byte battery-backed-up (BBU) RAM
- VFB01 1024-by-864 pixel, 1-plane, monochrome frame-buffer option
- VFB02 1024-by-864 pixel, 8-plane, color frame-buffer option
- 16 by 16 pixel, 2-plane graphics cursor
- 8-plane write mask for color frame buffer
- System control and status register
- Reset switch

The KN01 system module accepts up to 12 MS01 memory option modules, with two modules required per 4 MByte memory bank. The KN01 also accepts either a VFB01 monochrome frame buffer or a VFB02 color frame buffer option module. The monochrome option is compatible with the VR150 15-inch monochrome monitor or the VR262 19-inch monochrome monitor. The color option is compatible with the VR297 16-inch color monitor or the VR299 19-inch color monitor. The system module is designed for use with the LK201 or LK401 keyboard, and the VSXXX-AA mouse. The SCSI interface supports up to seven SCSI peripherals such as the RZ23 3.5-inch disk drive or the RZ55 5.25-inch disk drive. The KN01 is compatible with the VAXstation 3100 system enclosure.

The following sections describe the KN01 system module configured with MS01 memory options and VFB01/VFB02 frame buffer options. Further discussion of external peripherals or packaging options is beyond the scope of this document.

The memory SIM connectors are specified for a maximum of 25 insertion/removal cycles. Limiting insertion/removal cycles to 5 is recommended.

## 2. External Interface

The rear bulkhead has the following external interfaces:

- AC power switch
- AC line receptacle
- AC convenience receptacle for video monitor
- 8 red status LEDs
- Modem MMJ-6 female connector with DTR, TX, RX, and DSR DEC423/EIA-232-D compatible signals
- Printer MMJ-6 female connector with TX and RX DEC423/EIA-232-D compatible signals
- Video DB15 male connector with RS343A/RS170 compatible signals
- Mouse DIN7 female connector with TX and RX EIA-232-D compatible signals, +12 Volts at 300 mAmps, +5 Volts at 150 mAmps, and -12 Volts at 20 mAmps
- Keyboard MMJ-4 female connector with TX and RX EIA-232-D compatible signals and +12 Volts at 300 mAmps
- Momentary push-button reset switch
- ThinWire Ethernet BNC female connector to internal ThinWire transceiver
- ThinWire-selected green status LED
- Latching push-button selector switch for Thin/Thick Ethernet interface
- ThickWire-selected green status LED
- ThickWire Ethernet DB15 female connector for external ThickWire transceiver
- SCSI HONDA68 male connector

## 3. Power Requirements

DECstation 2100/3100 electronics has a theoretical maximum power dissipation of 81 Watts. Table 3-1 lists the theoretical worst case power supply requirements for a 24 MByte, color, diskless DECstation 3100. The worst case supply requirements include power provided to external LK201 keyboard, VSXXX-AA mouse, external ThickWire transceiver, and external SCSI terminator. The typical supply currents include power for a keyboard and mouse. All currents are for steady state conditions.

### Table 3-1: Supply Requirements

| Supply (Volts) | Maximum Current (Amps) | Typical Current (Amps) |
|---|---|---|
| +5 +/-5% | 15.6 | 10.0 |
| +12 +/-5% | 0.89 | 0.07 |
| -12 +/-5% | 0.11 | 0.04 |
| -9 +/-5% | 0.18 | 0.13 |

## 4. Address Map

The R2000 CPU has a 4 GByte virtual address space consisting of four regions. Table 4-1 lists these regions. Note that the KSEG0, KSEG1, and KSEG2 regions are only accessible while the R2000 is in kernel mode. Refer to the *R2000 Processor Architecture* for a detailed discussion of its virtual address space.

**Table 4-1: R2000 Virtual Address Space**

| Address Range | Size (GBytes) | Region | Properties |
|---|---|---|---|
| 0x00000000..0x7FFFFFFF | 2.0 | KUSEG | Mapped and cached |
| 0x80000000..0x9FFFFFFF | 0.5 | KSEG0 | Unmapped and cached |
| 0xA0000000..0xBFFFFFFF | 0.5 | KSEG1 | Unmapped and uncached |
| 0xC0000000..0xFFFFFFFF | 1.0 | KSEG2 | Mapped and cached |

The DS3100 workstation has a 512 MByte physical address space. Physical addresses beyond this range are reserved and must not be referenced. The first 256 MBytes of the physical address space are considered memory space. The second 256 MBytes of physical address space are considered I/O space with I/O subsystems decoded on 16 MByte boundaries. Table 4-2 summarizes the primary address space decoding of the workstation. Note that memory and the frame buffer are listed at both their KSEG0 and KSEG1 addresses. These are aliases of the same physical storage locations and cache data inconsistencies may result if they are referenced through both of the KSEG0 and KSEG1 regions.

**Table 4-2: System Address Map**

| Address Range | Subsystem |
|---|---|
| 0x00000000..0x7FFFFFFF | KUSEG |
| 0x80000000..0x817FFFFF | Memory |
| 0x81800000..0x8FBFFFFF | Reserved |
| 0x8FC00000..0x8FCFFFFF | Frame buffer |
| 0x8FD00000..0x9FFFFFFF | Reserved |
| 0xA0000000..0xA17FFFFF | Memory (uncached) |
| 0xA1800000..0xAFBFFFFF | Reserved |
| 0xAFC00000..0xAFCFFFFF | Frame buffer (uncache) |
| 0xAFD00000..0xAFFFFFFF | Reserved |
| 0xB0000000..0xB0FFFFFF | Color plane mask |
| 0xB1000000..0xB1FFFFFF | Cursor (PCC) |
| 0xB2000000..0xB2FFFFFF | Color map (VDAC) |
| 0xB3000000..0xB6FFFFFF | Reserved |
| 0xB7000000..0xB7FFFFFF | Write error address |
| 0xB8000000..0xB8FFFFFF | Network interface (LANCE) |
| 0xB9000000..0xB9FFFFFF | Network buffer (64 KBytes) |
| 0xBA000000..0xBAFFFFFF | SCSI interface (SII) |
| 0xBB000000..0xBBFFFFFF | SCSI buffer (128 KBytes) |
| 0xBC000000..0xBCFFFFFF | Serial interface (DZ) |
| 0xBD000000..0xBDFFFFFF | Real-time-clock/battery-backed-up-RAM (RTC) |
| 0xBE000000..0xBEFFFFFF | Control/status register (SYS_CSR) |
| 0xBF000000..0xBFFFFFFF | Self-test/bootstrap ROM |
| 0xC0000000..0xFFFFFFFF | KSEG2 |

I/O subsystems generally do not occupy their entire 16 MByte region. References to portions of I/O subsystem address space not explicitly defined in the following sections should not be issued. DS3100 contains a bus timer that aborts I/O space transactions that do not complete within 128 cycles (7.68 microseconds). Aborted read transactions result in a bus error exception. Aborted write transactions result in a memory error (MEMERR) interrupt.

Memory, the frame buffer, and the ROM are non-volatile (in the programming sense) and may be cached. The network buffer, SCSI buffer, color plane mask, cursor, color map, write error address, LANCE, SII, DZ, RTC, and SYS_CSR are volatile and must never be cached.

The R2000 CPU is configured for little-endian byte order. All address space descriptions in this

document are correspondingly little-endian.

## 5. Interrupts

Table 5-1 lists the I/O device connections to the R2000 interrupt inputs. The state of the interrupt signals is continually reflected in the R2000 CAUSE register at the bit position shown in the table. Note that a given interrupt signal only generates an R2000 exception if it is enabled in the STATUS register interrupt mask field, and interrupts are enabled by the STATUS<0> register bit. Note also that the interrupt signals are visible in the CAUSE register regardless of the state of the STATUS interrupt mask. That is, the operating system interrupt dispatcher must explicitly check that a given interrupt level, which is asserted in the CAUSE register, is enabled before activating that interrupt level's handler.

**Table 5-1: I/O Interrupt Levels**

| Level | CAUSE/STATUS | I/O Device |
|---|---|---|
| 5 | 15 | FPU |
| 4 | 14 | VINT or MEMERR |
| 3 | 13 | RTC |
| 2 | 12 | DZ |
| 1 | 11 | LANCE |
| 0 | 10 | SII |

## 6. Subsystems

### 6.1. Processor

The DS3100 processor is composed of the R2000 scalar processor, R2010 floating point coprocessor, and R2020 write buffers. The R2000 chipset operates at 16.67 MHz resulting in average processing performance of 12 million instructions per second (MIPS) over a range of applications.

The R2000 CPU implements the instruction set, processor registers, virtual memory, and interrupt system as defined by the R2000 architecture. The CPU maintains the direct-mapped instruction cache and the direct-mapped, write-through data cache. Each cache is 64 KBytes in capacity with a 4-byte line size. The tag and data stores of each cache are byte-parity protected, with cache parity errors transparently generating cache misses to reload the cache from memory. Operating system software should poll the CPU STATUS<PE> bit in its clock-interrupt handler to detect excessive rates of cache parity errors.

The R2010 FPU implements the IEEE arithmetic functions and coprocessor registers as defined by the R2000 architecture. The FPU is coprocessor 1 to the CPU. The FPU interrupts at level 5, which is visible in the R2000 CAUSE<15> register bit.

The R2020 WB implements a 4-stage write buffer for the CPU. The WB functions as part of coprocessor 0 to the CPU. Software may determine whether or not the WB is empty by conditionally branching on the coprocessor 0 condition (BC0F branches if WB is non-empty). If the CPU issues an I/O read immediately after a write, the I/O read will be completed before the write due to pipeline latency in the WB. Consequently, operating system device drivers must explicitly wait for the WB to drain after modifying I/O buffers or device registers, before reading device registers which have volatile state affected by the write. Note that this restriction does not apply to memory locations as the hardware automatically detects and corrects such conflicts.

In the event of a bus timeout on a memory write, the WB latches the write address, asserts the SYS_CSR<MEMERR> bit, and generates a level 4 interrupt, which is visible in the R2000 CAUSE<14>

register bit. The write error address may be read at address 0xB7000000. Note that in the event of multiple write errors, the WB error address latch records the address of the last error which occurred.

If a bus timeout occurs on an I/O read, or a parity error occurs on a memory read, then a bus-error exception results. Processor state is preserved and exception processing initiated as defined by the R2000 architecture. In the case of a memory parity error caused by a soft error, there is no direct record of which byte(s) of the word caused the parity error. To localize the error, the operating system handler may selectively rewrite bytes in the indicated word, and then reread the location until parity errors no longer occur.

Hard memory bit errors may be isolated by disabling parity checking via the system CSR<PARDIS> bit and memory test patterns.

### 6.2. Memory

The DS3100 system module supports 4 to 24 MBytes of byte-parity protected DRAM in 4 MByte increments. The memory system includes both the DRAM-based memory array and the VRAM-based video frame buffer. The frame buffer has the same memory access characteristics as memory, and may be cached if desired. The memory system supports byte (8-bit), half-word (16-bit), word (32-bit) writes, and word reads as per the R2000 instruction set.

The memory array starts at address 0x80000000 (in KSEG0) through the size of the array. The frame buffer starts at address 0xAFC00000 (in KSEG1) through the size of the frame buffer. Note that memory is only decoded with 32 MByte resolution, consequently the array is aliased eight times through the 256 MByte physical memory space. The memory array and frame buffer should only be referenced at the addresses defined above for future compatibility.

The memory subsystem serves only the processor, there is no I/O device DMA support. Operating system device drivers must explicitly copy network and SCSI data between buffers and user memory. Since there is no I/O DMA, the caches never need to be flushed.

The memory system control logic is optimized for minimum memory read latency, at a slight cost in memory write latency. On a memory read, the CPU incurs a 5 cycle stall in the absence of memory refresh contention. The memory system can sustain 5 cycle reads resulting in a peak read bandwidth of 13.3 MBytes/second.

Memory writes to an empty WB complete in 8 cycles, but do not stall the CPU. Successive memory writes complete at the rate of 6 cycles, with the CPU stalled whenever the write buffer is full. The memory system can sustain 6 cycle writes resulting in a peak write bandwidth of 11.1 MBytes/second.

Memory refresh occurs once each 18.5 microseconds and reloads the VRAM shift registers and runs two RAS-only DRAM refresh cycles. CPU memory access during refresh can result in up to 16 cycles of additional latency; a 5% overhead.

The DRAM memory array is implemented with MS01 single-inline memory modules (SIMMs), each implementing 1-M-by-18 bits of memory. Two SIMMs are required for a 4 MByte bank of memory. The system modules supports from 1 to 6 banks. The mono or color frame buffer is also implemented with a VFB01 or VBF02 SIMM.

If a soft error occurs in a text segment, the operating system should mark the page as swapped-out and reload from backing store. If a soft error occurs in a data or stack segment, the operating system should terminate the affected process.

Parity checking on memory reads is normally enabled. Consequently, after a restart exception, all

memory should be written before it is read to initialize the parity bits. For diagnostic isolation of memory errors, parity checking may be disabled by asserting the PARDIS bit in the system control and status register.

### 6.3. ROM

Workstation self-test and bootstrap software is resident in 256 KBytes of ROM starting at address 0xBFC00000. Reads to the ROM stall the CPU for 8 cycles. The ROM is in sockets.

Since the ROM does not contain parity, it is recommended that power-up-self-test software calculate and verify a checksum on the ROM contents.

### 6.4. Serial Interface

The quad asynchronous serial interface, based upon the DC7085 gate array, presents a DZ software interface and a DEC423/EIA-232-D electrical interface. The serial transmitters are double buffered, while the receivers share a 64-entry FIFO. The baud rate of each serial line is independently programmable to 50 through 9600 bits per second.

The keyboard, mouse, and printer lines are data leads only, while the modem line also supports DTR/DSR control signals. The keyboard line is available via a 4-pin MMJ connector. The mouse line is available via a 7-pin DIN connector. The printer and modem lines are available via 6-pin modular connectors. The keyboard and mouse connectors supply power (through current limit devices) to the external devices. Table 6-1 lists the binding of devices to serial lines.

**Table 6-1: Serial Device DZ Line Numbers**

| DZ Line | Serial Device |
|---|---|
| 0 | Keyboard |
| 1 | Mouse |
| 2 | Modem |
| 3 | Printer |

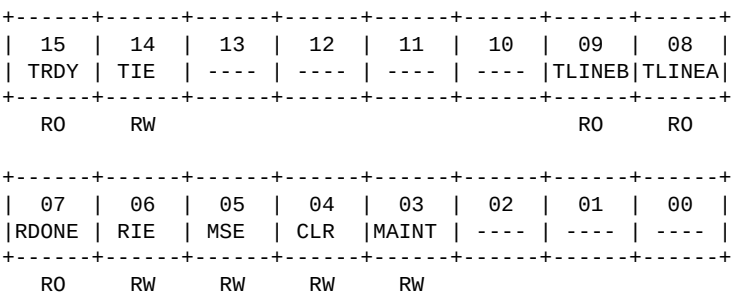
The DZ interrupts the CPU at level 2, which is visible in the R2000 CAUSE<12> register bit.

The DC7085 only supports half-word reads and writes. The DC7085 registers are 8-byte-aligned in the processor address space. Reads to the DZ nominally stall the CPU for 14 cycles. Writes to the DZ nominally complete in 19 cycles. Table 6-2 lists the addresses of the DZ registers.

**Table 6-2: DZ Register Addresses**

| Address | Access | Name | Function |
|---|---|---|---|
| 0xBC000000 | R/W | DZ_CSR | Control and status |
| 0xBC000008 | R | DZ_RBUF | Receiver buffer |
| 0xBC000008 | W | DZ_LPR | Line parameters |
| 0xBC000010 | R/W | DZ_TCR | Transmitter control |
| 0xBC000018 | R | DZ_MSR | Modem status |
| 0xBC000018 | W | DZ_TDR | Transmit data |

### 6.4.1. Control And Status Register




All bits in the DZ_CSR register are cleared by device reset or by asserting *master clear* (DZ_CSR<CLR>).

DZ_CSR<15> Transmitter Ready (TRDY)

This read-only bit is set when the transmitter scanner stops on a line whose transmit buffer may be loaded with another character and whose related DZ_TCR<LNENBX> bit is set. If the DZ_CSR<TIE> bit is also set, an interrupt request will be generated. When DZ_CSR<TRDY> is set, and at no other time, the *transmitter line number* (bits DZ_CSR<TLINEB:TLINEA>) is valid.

This bit is cleared when data is loaded into the transmitter for the line number indicated in DZ_CSR<TLINEB:TLINEA>. If additional transmit lines need service, DZ_CSR<TRDY> appears again within 1.4 microseconds of the completion of the transmitter data load operation. Note that since DZ_CSR<TRDY> requires 1.4 microseconds to update, software must not examine this bit during that period of time or it may erroneously interpret slow deassertion of the bit as a transmitter ready condition.

This bit is also cleared when *master scan enable* (DZ_CSR<MSE>) is cleared, or when the related DZ_TCR<LNENBX> bit is cleared.

DZ_CSR<14> Transmitter Interrupt Enable (TIE)

When this read/write bit is set, the setting of DZ_CSR<TRDY> will generate an interrupt request.

DZ_CSR<13:10> MBZ

DZ_CSR<09:08> Transmitter Line Number (TLINEB, TLINEA)

These read-only bits indicate the line number whose transmit buffer needs servicing. These bits are valid only when *transmitter ready* (DZ_CSR<TRDY)>) is set, and are cleared when *master scan enable* (DZ_CSR<MSE>) is cleared. Bit DZ_CSR<08> is the least significant bit.

DZ_CSR<07> Receiver Done (RDONE)

This is a read-only bit that is set when a character appears at the output of the silo. If the *receiver interrupt enable* (DZ_CSR<RIE>) is set, an interrupt request will be generated. If DZ_CSR<RIE> is clear, no interrupt request is generated, and the program may poll this bit to detect available characters.

This bit is cleared when the receiver buffer register (DZ_RBUF) is read. If another character is available in the silo, this bit will be cleared for a period between 100 nanoseconds and 1 microsecond, and will then be re-asserted.

This bit is also cleared when *master scan enable* (DZ_CSR<MSE>) is cleared.

DZ_CSR<06> Receiver Interrupt Enable (RIE)

This read/write bit permits the generation of an interrupt request when DZ_CSR<RDONE> is set.

DZ_CSR<05> Master Scan Enable (MSE)

This read/write bit must be set to permit the receiver and transmitter control sections to start the flag scanning process. When this bit is clear, *transmitter ready* (DZ_CSR<TRDY>) is inhibited from setting and the receiver silo is cleared.

DZ_CSR<04> Master Clear (CLR)

When written to a 1, this bit generates *initialize* within the chip. A read-back of this register with this bit set indicates that initialization is still in progress. This bit is self-clearing. All registers, silos, and UART functions are cleared with the following exceptions:

1. Only DZ_RBUF<DVAL> is cleared; the other bits are not
2. The modem control output bits are not cleared
3. The modem status register is not cleared

DZ_CSR<03> Maintenance (MAINT)

This is a read/write bit which, when set, loops the serial output connections of the transmitters to the corresponding serial input connections of the receivers. This feature is used only for maintenance.

Note that while in maintenance mode, the transmitter outputs are still active. To avoid sending internal loopback data to external serial devices, the EIA drivers may be disabled by asserting the system control and status register TXDIS bit during the loopback test.

DZ_CSR<02:00> MBZ

### 6.4.2. Receiver Buffer Register

| 15 | 14 | 13 | 12 | 11 | 10 | 09 | 08 |
|---|---|---|---|---|---|---|---|
| DVAL | OERR | FERR | PERR | ---- | ---- | RLINEB | RLINEA |
| RO | RO | RO | RO | | | RO | RO |

| 07 | 06 | 05 | 04 | 03 | 02 | 01 | 00 |
|---|---|---|---|---|---|---|---|
| RBUF07 | RBUF06 | RBUF05 | RBUF04 | RBUF03 | RBUF02 | RBUF01 | RBUF00 |
| RO | RO | RO | RO | RO | RO | RO | RO |

The receiver buffer register (DZ_RBUF) is a 16-bit read-only register that contains the received character as the output of the 64-location silo buffer. A read of the register causes the character entry to be removed from the buffer, and all other entries shift down to the lowest location that is not occupied. Only the *data valid bit* (DZ_RBUF<15>) is cleared by *master clear* (DZ_CSR<CLR>) or device reset. The other bits have **unpredictable** values.

DZ_RBUF<15> Data Valid (DVAL)

This bit, when set, indicates that the data in bits DZ_RBUF<14:00> is valid. This permits an interrupt handling program to read the silo repeatedly and test each entry (after it has been moved out of the silo) until the program finds an entry for which this bit is zero, indicating that the silo is now empty.

DZ_RBUF<14> Overrun Error (OERR)

This bit becomes set when a received character is overwritten in the UART buffer by a following character before it has been transferred to the silo by the scanner. This condition indicates that the program is not removing characters from the silo sufficiently quickly, resulting in silo full conditions.

DZ_RBUF<13> Framing Error (FERR)

This bit is set if the received character did not have a stop bit present at the correct time. This bit is usually interpreted as indicating that a BREAK has been received.

DZ_RBUF<12> Parity Error (PERR)

This bit is set if the sense of the parity of the received character does not agree with the parity defined for that line.

DZ_RBUF<11:10> RAZ

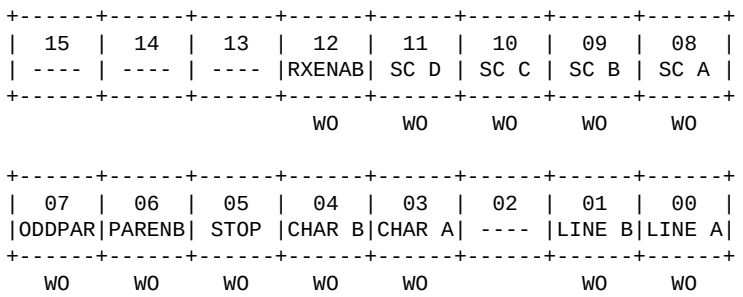
DZ_RBUF<09:08> Received Line Number (RLINEB, RLINEA)

These bits contain the line number upon which the received character arrived. Bit DZ_RBUF<08> is the least significant.

DZ_RBUF<07:00> Received Character (DZ_RBUF7 - DZ_RBUF0)

These bits contain the received character. Characters of less than eight bits in length are right justified with unused bit positions shown as zeroes. The least significant bit is bit DZ_RBUF<00>. The parity bit is not shown.

### 6.4.3. Line Parameter Register




The line parameter register (DZ_LPR) controls the operating parameters related to each line in the chip. The DZ_LPR must be addressed with a word address and is a write-only register. The line parameters for each line must be loaded again after the setting of *master clear* (DZ_CSR<CLR>) or the assertion of the device reset pin. This register should not be modified while data transmission or reception is in progress on the associated line.

DZ_LPR<15:13> MBZ

DZ_LPR<12> Receiver Enable (RXENAB)

This bit must be set before the UART receiver logic for this line can assemble characters from the serial input line.

DZ_LPR<11:08> Speed Code (SC D, SC C, SC B, SC A)

The state of these bits determines the operating speed for the transmitter and receiver of the selected line. Note that the non-standard 19800 baud rate is not supported.

| 11 SC D | 10 SC C | 09 SC B | 08 SC A | Speed |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 50 |
| 0 | 0 | 0 | 1 | 75 |
| 0 | 0 | 1 | 0 | 110 |
| 0 | 0 | 1 | 1 | 134.5 |
| 0 | 1 | 0 | 0 | 150 |
| 0 | 1 | 0 | 1 | 300 |
| 0 | 1 | 1 | 0 | 600 |
| 0 | 1 | 1 | 1 | 1200 |
| 1 | 0 | 0 | 0 | 1800 |
| 1 | 0 | 0 | 1 | 2000 |
| 1 | 0 | 1 | 0 | 2400 |
| 1 | 0 | 1 | 1 | 3600 |
| 1 | 1 | 0 | 0 | 4800 |
| 1 | 1 | 0 | 1 | 7200 |
| 1 | 1 | 1 | 0 | 9600 |
| 1 | 1 | 1 | 1 | 19800 |

DZ_LPR<07> Odd Parity (ODDPAR)

If this bit is set (and DZ_LPR<PARENB> is set), characters of odd parity are generated on the line and incoming characters are expected to have odd parity. If this bit is not set (and DZ_LPR<PARENB> is set), characters of even parity are generated and incoming characters are expected to have even parity. If DZ_LPR<PARENB> is not set, the state of this bit is immaterial.

DZ_LPR<06> Parity Enable (PARENB)

If this bit is set, characters transmitted on the line have an appropriate parity bit added, and characters received on the line have their parity checked.

DZ_LPR<05> Stop Code (STOP)

This bit sets the stop code length; 0 = 1 unit stop, 1 = 2 unit stop (or 1.5 unit stop if 5-bit character length is selected).

DZ_LPR<04:03> Character Length (CHAR B, CHAR A)

The bits control the length of the characters generated by the transmitter and expected by the receiver according to the table below:

| 04 CHAR B | 03 CHAR A | Character Length |
|---|---|---|
| 0 | 0 | 5 bits |
| 0 | 1 | 6 bits |
| 1 | 0 | 7 bits |
| 1 | 1 | 8 bits |

DZ_LPR<02> MBZ

DZ_LPR<01:00> Parameter Line Number (LINE B, LINE A)

These bits specify the line number for which the parameter information bits <12:03> is to apply. Bit DZ_LPR<00> is the least significant bit.

### 6.4.4. Transmit Control Register

| 15 | 14 | 13 | 12 | 11 | 10 | 09 | 08 |
|---|---|---|---|---|---|---|---|
| ---- | ---- | ---- | ---- | ---- | DTR 2 | ---- | ---- |
| | | | | | RW | | |

| 07 | 06 | 05 | 04 | 03 | 02 | 01 | 00 |
|---|---|---|---|---|---|---|---|
| ---- | ---- | ---- | ---- | LNENB3 | LNENB2 | LNENB1 | LNENB0 |
| | | | | RW | RW | RW | RW |

DZ_TCR<15:11> MBZ

DZ_TCR<10> Modem Control (DTR2)

This read/write bit controls the assertion of the DTR modem control signal for line 2. This bit is **not** cleared by the setting of *master clear* (DZ_CSR<CLR>). It is cleared by a device reset.

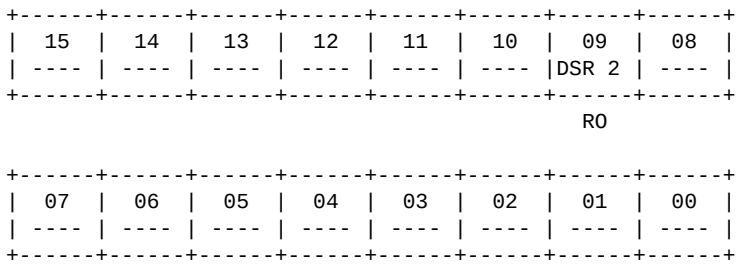
DZ_TCR<09:04> MBZ

DZ_TCR<03:00> Transmitter Line Enable (LNENB3, LNENB2, LNENB1, LNENB0)

These read/write bits enable the transmitter logic for lines 3, 2, 1, and 0 respectively. Setting one of these bits causes the transmitter scanner to stop if the UART for that line has a transmitter buffer empty condition. An interrupt is then generated if transmitter interrupts are enabled. The scanner restarts when either the transmit data register (DZ_TDR) is loaded with a character or when the DZ_TCR<LNENBX> bit is cleared for the line upon which the scanner stopped. DZ_TCR<LNENBX> bits should only be cleared while the scanner is not running, i.e. when *transmitter ready* (DZ_CSR<TRDY>) is set, or *master scan enable* (DZ_CSR<MSE>) is clear.

These bits are cleared by setting *master clear* (DZ_CSR<CLR>) or by asserting the device reset pin.

### 6.4.5. Modem Status Register




The modem status register (DZ_MSR) is a 16-bit read-only register. A read of this register gives the status of the DSR modem control signal. The ON condition of a modem control signal is interpreted as a logical one.

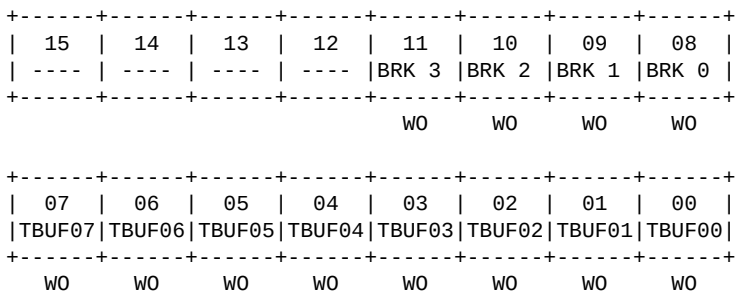
DZ_MSR<15:10> RAZ

DZ_MSR<09> Data Set Read (DSR)

This bit reflects the state of the *data set ready* signal from the modem on line 2.

DZ_MSR<08:00> RAZ

### 6.4.6. Transmit Data Register




DZ_TDR<15:12> MBZ

DZ_TDR<11:08> Break Control (BRK 3, BRK 2, BRK 1, BRK 0)

These bits control the assertion of BREAK on lines 3, 2, 1, and 0 respectively. Setting a break bit IMMEDIATELY forces the output of that line to space.

DZ_TDR<07:00> Transmitter Buffer

Characters for transmission are loaded into these bits. DZ_TDR<00> is the least significant bit. Loading of a character should occur only when *transmitter ready* (DZ_CSR<15>) is set. The character that is loaded into this register is routed to the line defined in DZ_CSR<TLINEB:TLINEA>.

This register is cleared by setting *master clear* (DZ_CSR<CLR>) or by asserting the device reset pin. This register can be used regardless of the state of the *maintenance* bit (DZ_CSR<MAINT>).

### 6.5. SCSI Interface

The SCSI interface is based upon the DC7061 SII gate array and a 64 K by 16-bit (128 KBytes) SCSI buffer. The SII manages the SCSI bus via selection, DMA data transfer, and disconnect commands. The SII supports command disconnect/reconnect and synchronous data transfers at 4.0 Mbytes/second on the SCSI bus; this is the highest performance mode of operation for the SCSI bus. The buffer is time-multiplexed between the SII and the processor. The SII can only access its buffer during DMA transfers; it has no access to memory.

An 83C11 transceiver drives the SCSI cable in the single-ended configuration that support a total cable length of up to six meters. The SCSI cable is always terminated on the DS3100 system module, which supplies termination power (through a current limit device) to the remote-end terminator. Up to seven SCSI peripherals may be connected to the cable.

The SII interrupts the CPU at level 0, which is visible in the R2000 CAUSE<10> register bit.

The SII only supports half-word reads and writes. The SII registers are word-aligned. Reads of the SII nominally stall the CPU for 21 cycles. Writes to the SII nominally complete in 20 cycles. The SCSI buffer only supports half-word reads and writes. The SCSI buffer is word-aligned in the processor address space. Reads of the buffer nominally stall the CPU for 12 cycles when the SII is idle; a peak read bandwidth of 2.8 MBytes/second. Writes to the buffer nominally complete in 14 cycles when the SII is idle; a peak write bandwidth of 2.4 MBytes/second. Buffer access during SII activity may increase the access latency by up to 5 additional cycles. Table 6-3 lists the SII register and SCSI buffer addresses.

**Table 6-3: SII Register and SCSI Buffer Addresses**

| Address | Name | Register |
|---|---|---|
| 0xBA000000 | SII_SDB | SCSI data bus and parity |
| 0xBA000004 | SII_SC1 | SCSI control signals 1 |
| 0xBA000008 | SII_SC2 | SCSI control signals 2 |
| 0xBA00000C | SII_CSR | Control and status register |
| 0xBA000010 | SII_ID | ID register |
| 0xBA000014 | SII_SLCSR | Selector control and status |
| 0xBA000018 | SII_DESTAT | Selection detector status |
| 0xBA00001C | SII_DSTMO | <Unsupported> |
| 0xBA000020 | SII_DATA | Data register |
| 0xBA000024 | SII_DMCTRL | DMA control |
| 0xBA000028 | SII_DMLOTC | DMA length of transfer |
| 0xBA00002C | SII_DMADDRL | DMA address pointer (low half) |
| 0xBA000030 | SII_DMADDRH | DMA address pointer (high half) |
| 0xBA000034 | SII_DMABYTE | DMA initial byte |
| 0xBA000038 | SII_STLP | <Unsupported> |
| 0xBA00003C | SII_LTLP | <Unsupported> |
| 0xBA000040 | SII_ILP | <Unsupported> |
| 0xBA000044 | SII_DSCTRL | <Unsupported> |
| 0xBA000048 | SII_CSTAT | Connection interrupt control |
| 0xBA00004C | SII_DSTAT | Data interrupt control |
| 0xBA000050 | SII_COMM | Command register |
| 0xBA000054 | SII_DICTRL | Diagnostic control register |
| | | |
| 0xBB000000 | Buffer base | SCSI buffer |

Since the buffer is word-aligned to the CPU, all addresses specified to the SII must be left-shifted by one when referencing the buffer. That is, from the CPU successive half-words of the buffer are at

0xBB000000, 0xBB000004, 0xBB000008, etc., while they are at 0x0000, 0x0002, 0x0004, etc., to the SII.

### 6.5.1. SII Registers

The registers of the SII can be divided into two categories according to usage. These categories can be described as:

- Diagnostic - those registers which are used ONLY to diagnose the functionality of the chip and nearby circuitry. Typically, these registers are accessed only during power-up testing.
- SCSI - those registers which are used when the SII is operating in SCSI mode.

Most registers in the SII are standard read/write registers. Some, however, do not fall into this class. These other classes are:

- R/W1TC - read/write 1 to clear. Several status registers contain bits which require that once a status bit has been set, it can only be cleared by writing a 1 to that bit position.
- R/O - read only. Several status registers contain only status and writing to them has no effect.
- R/W* - these registers are not true read/write in that under certain conditions they will not read back the value last written to them. These conditions will be noted in the description of the register.

Figure 6-1 shows the classification of each register.

| NAME | USAGE | CLASS |
|---|---|---|
| SII_SDB | DIAGNOSTIC | R/W* |
| SII_SC1 | DIAGNOSTIC | R/W* |
| SII_SC2 | DIAGNOSTIC | R/W* |
| SII_CSR | SCSI | R/W |
| SII_ID | SCSI | R/W* |
| SII_SLCSR | SCSI | R/W |
| SII_DESTAT | SCSI | R/O |
| SII_DATA | SCSI | R/W* |
| SII_DMCTRL | SCSI | R/W |
| SII_DMLOTC | SCSI | R/W |
| SII_DMADDRL | SCSI | R/W |
| SII_DMADDRH | SCSI | R/W |
| SII_DMABYTE | SCSI | R/W |
| SII_CSTAT | SCSI | R/W1TC |
| SII_DSTAT | SCSI | R/W1TC |
| SII_COMM | SCSI | R/W |
| SII_DICTRL | DIAGNOSTIC | R/W |

**Figure 6-1: SII Register Classification**

In the following register descriptions, all undefined bits will read as zero (0). Writing to any of these bits will have no effect.

#### 6.5.1.1. SII_SDB - SCSI Data Bus

The SII_SDB register is used only in diagnostic mode (see SII_DICTRL description) in conjunction with a loop back connector to test the SCSI port. It is also used in diagnostic internal loopback mode to effectively act like the SCSI bus. The fields in this register directly reflect the SCSI data bus ASSERTED HIGH. This register should NOT be used during normal operations. It should be noted that care must be taken to test this portion of the chip without any disturbance to the SCSI bus.

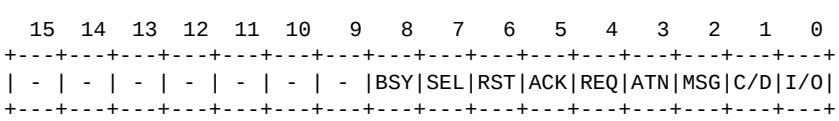
SII_SDB (0) -- READ/WRITE

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | PTY | SP DATA <7:0> | | | | | | | |

### 6.5.1.2. SII_SC1 - SCSI Control Signals One

This register is used in diagnostic mode (see SII_DICTRL description) in conjunction with a loop back connector to test the SCSI port or to effectively act as the SCSI bus in internal loopback mode. The bits in this register directly reflect some of the SCSI control lines ASSERTED HIGH. It should be noted that data written to this register may differ from that read back since only certain bits are driven while in the target or initiator mode.

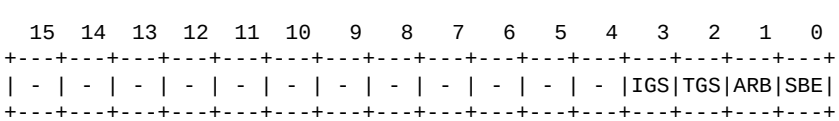
SII_SC1 (2) -- READ/WRITE




### 6.5.1.3. SII_SC2 - SCSI Control Signals Two

The SII_SC2 register is used only in diagnostic mode (see SII_DICTRL description) in conjunction with a loop back connector to test the SCSI port. These signals directly drive the four control signals on the NCR83C11 receiver/driver chip. Special care should be taken when writing this register to avoid disturbing the SCSI bus during power-up diagnostics. This register should only be accessed if an external loop-back connector is in place. It should not be used during normal operations.

SII_SC2 (4) -- READ/WRITE




IGS is set (1) to steer the SCSI drivers for the initiator role. READ/WRITE.

TGS is set (1) to steer the SCSI drivers for the target role. READ/WRITE.

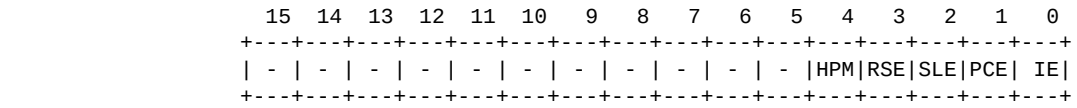
ARB is set (1) to enable the SCSI drivers for arbitration. READ/WRITE.

SBE is set (1) to drive the SCSI data bus and parity lines. READ/WRITE.

#### 6.5.1.4. SII_CSR - Control/Status Register

This register contains control and status information about the general operation of the SII in regard to the SCSI bus, including various enable bits.

SII_CSR (6) -- READ/WRITE




The fields in the SII_CSR are defined as follows:

HPM set to one (1) if the SII is operating on an arbitrated bus. In this mode, the SII will handle the arbitration. HP_RDY is used as a BUS_REQ, the SII returns HP_BUSGRANT to indicate that the external device has control of the bus. When clear(0), HP_RDY acts as an indicator that the current data transfer can be terminated and HP_BUSGRANT is not used. HPM must always be set to 1.

RSE set to one (1) if the SII is to respond to reselections. Clear (0) (default on reset) otherwise.

SLE set to one (1) if the SII is to respond to selections. Clear (0) (default on reset) otherwise.

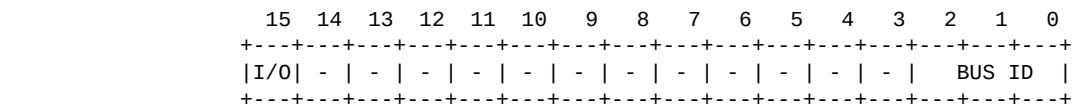
PCE set to one (1) if the SII is to check parity and report parity errors. When clear (0), the SII will continue to check parity but will not report any errors. In either case, the SII will continue to generate parity. The default value is zero (0).

IE set to one (1) if interrupts are to be enabled. Clear (0) (the default on reset) otherwise. If clear, all interrupts are disabled.

#### 6.5.1.5. SII_ID - Bus ID Register

This register contains the three bit ID number of this SII on the SCSI bus. This value is needed for selection and selection detection.

SII_ID (8) -- READ/WRITE



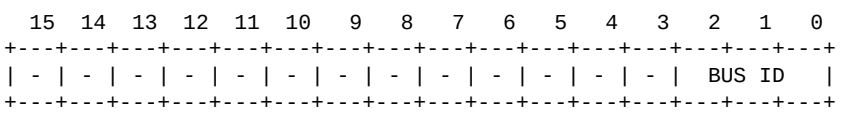

The bits is this register are defined as follows:

I/O when set to one, indicates that the 3 ID pins of the SII are outputs and the values presented in the SII_ID register are reflected (complemented) at these pins. It is expected that this register be written before the SII is enabled. When clear, the ID pins are inputs. The logical inversion of these pins will appear in the SII_ID register. Note that if this bit is cleared, writing this register has no effect. I/O must be set to 1.

BUS ID
the ID of the SII.

#### 6.5.1.6. SII_SLCSR - Selector Control And Status Register

SII_SLCSR (10) -- READ/WRITE



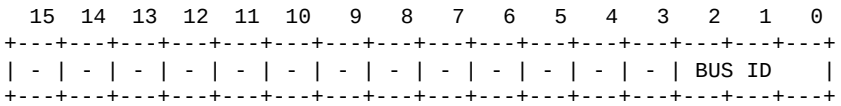

The bits in this register are defined as follows:

BUS ID
: the ID of the device to be selected or reselected (destination ID). This must be loaded before a SELECT command is issued.

#### 6.5.1.7. SII_DESTAT - Selection Detector Status Register

This register contains the bus ID of the device which has selected the SII. It is typically read after an interrupt is received to dispatch to the ID-dependent code.

SII_DESTAT (12) -- READ ONLY



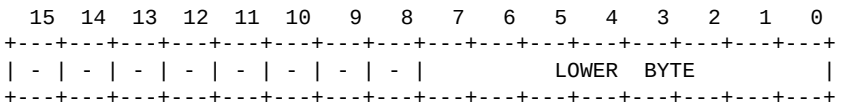

The bits in this register are defined as follows:

BUS ID
: the number of the device which selected the SII (source ID). This register is only updated by the SII after being selected.

#### 6.5.1.8. SII_DATA - Data Register

This register is used to load data to be sent out on the SCSI bus. It can also be used to read incoming information. Typically, it would be used for message and status phases. For all programmed I/O operations, only the lower byte is used. This register cannot be used for synchronous data transfers. This register will not reflect the data written to it.

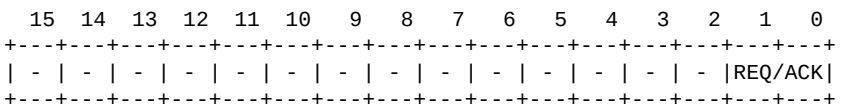
SII_DATA (16) -- READ/WRITE



#### 6.5.1.9. SII_DMCTRL - DMA Control Register

This register contains mode information concerning the current DMA activity. This consists of the req/ack offset used for synchronous data transfers. Note that this register must be written following detection of a selection or reselection to insure proper operation during synchronous data transfers.

SII_DMCTRL (18) -- READ/WRITE




These bits are defined as follows:

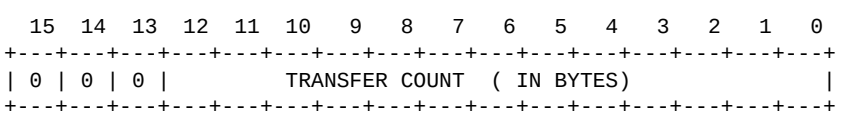
REQ/ACK OFFSET

the desired request/acknowledge offset for any synchronous data transfers occurring during this connection. A maximum of three (3) is implemented for data phase transfers. A zero (0) value implies SCSI asynchronous data transfers. This offset is only for data phase transfers; other information phases must be done asynchronously. It should be noted that there is no special command for synchronous data transfer. A non-zero value for the REQ/ACK offset implies all data transfers are done in synchronous mode.

#### 6.5.1.10. SII_DMLOTC - DMA Length Of Transfer Counter

This register contains the number of BYTES which are to be DMA'ed into/out of memory. This register will auto-decrement after each transaction and will reflect the number of bytes left to transfer. It will be implemented as a 13 bit counter. This register will contain the number of bytes sent across the SCSI bus during a read and the number of bytes deposited into RAM on a write. This is a true count and bytes currently in the FIFO are not considered transferred.

SII_DMLOTC (20) -- READ/WRITE

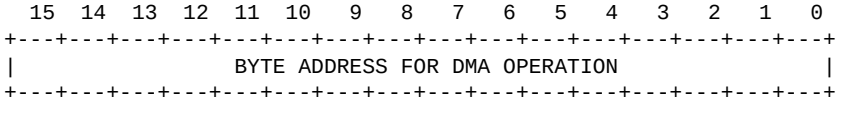



Note that data transfers in excess of 8 KBytes are possible by simply reloading the SII_DMADDR and SII_DMLOTC registers as appropriate for the next data fragment, and issuing a new DMA command transfer command after a DNE interrupt is received from the current transfer.

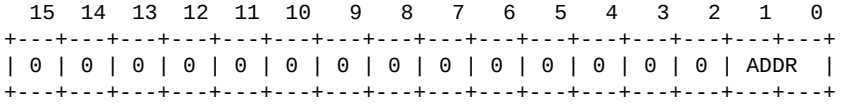
#### 6.5.1.11. SII_DMADDR - DMA Address Registers

This register contains the memory byte address from which the DMA operation will begin. Note that a "1" in the least significant bit position means that the first cycle will be done with an initial byte offset.

SII_DMADDRL (22) -- READ/WRITE




SII_DMADDRH (24) -- READ/WRITE

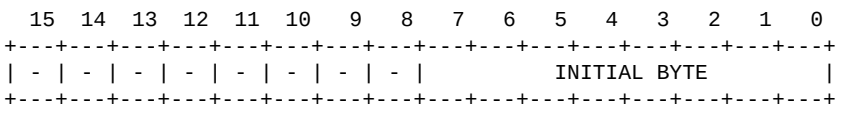


### 6.5.1.12. SII_DMABYTE - DMA Initial Byte Register

This register is used to load data to be sent to the memory bus. Typically, it would be used in the following scenario:

- The SII is operating as an initiator
- The target is sending data to this SII.
- The target changes phase on an odd boundary and requests the pointers be saved.
- The SII interrupts the processor indicating a phase change has occurred and the DMA transfer ended on an odd byte boundary.
- At some later time, the target reconnects to complete the transfer.
- The processor will load the "odd" byte from the previous transfer into this register.
- After receiving the next byte, the SII will transfer the whole word into memory.

SII_DMABYTE (26) -- READ/WRITE



### 6.5.1.13. SII_CSTAT - Connection Status Register

This register contains interrupt status related to SII connections.

SII_CSTAT (36) -- READ/SELECTIVE WRITE 1

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CI | DI | RST | BER | OBC | TZ | BUF | LDN | SCH | CON | DST | TGT | SWA | SIP | LST | 0 |
| | | * | * | * | | | * | * | * | | | | | | |

These bits are defined as follows:

CI SII_CSTAT Interrupt - composite error bit for the SII_CSTAT register. It is the logical 'or' of bits 13 through 11 and 9 through 7. Those bits marked by '*' will interrupt the processor when set.

DI SII_DSTAT Interrupt - composite error bit for the SII_DSTAT register.

RST RST asserted - set to one (1) if the RSTIN signal is asserted on the SCSI bus. The SII will automatically disconnect itself from the bus and interrupt the processor. This bit is write one (1) to clear.

BER Bus Error - this bit is set on any of the following conditions:

1. Fifo Overflow
2. Req/Ack Offset exceeded
3. Illegal Phase change

While this bit is asserted, the SII will not receive or transmit data in either DMA or programmed I/O mode. This bit is write one (1) to clear.

OBC Not used; ignore this bit.

TZ Not used; ignore this bit.

BUF Not used; ignore this bit.

LDN Not used; ignore this bit.

SCH State Change - this bit is set to one (1) if the state of the SII has changed. A change is considered to be any of the following:

1. Selected
2. Reselected
3. Disconnected
4. RST has occurred on the SCSI bus

Write one (1) to clear this bit.

CON Connected - this bit is set to one (1) if the SII is connected to another device. It is clear while the SII is idle. (Read only)

DST Destination - this bit is set to one (1) if the SII was the destination of the current transfer. In other words, this bit is set if the SII was selected or reselected by another device. (Read only)

TGT Target - this bit is set to one (1) if the SII is operating as a target during the current transfer. (Read only)

SWA Selected With ATN - this bit is set to one if the SII was selected with attention. Write one (1) to clear this bit.

SIP Selection In Progress - this bit is set if the SII is currently in a selection process. This is useful in determining if the desired destination is unavailable. (Read only)

LST Lost - this bit is set when the SII loses arbitration. It is cleared by the SII when it begins a

selection process. (Read only)

### 6.5.1.14. SII_DSTAT - Data Transfer Status Register

This register contains interrupt status related to data transfers.

SII_DSTAT (38) -- READ/SELECTIVE WRITE 1

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|CI |DI |DNE|TCZ|TBE|IBF|IPE|OBB| 0 | 0 | 0 |MIS|ATN|MSG|C/D|I/O|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
          *       *   *                       *
```

These bits are defined as follows:

CI SII_CSTAT Interrupt - composite error bit for the SII_CSTAT register.

DI SII_DSTAT Interrupt - composite error bit for the SII_DSTAT register. It is the logical 'or' of bits 13,11,10 and 4. Those bits marked with * will interrupt the processor when set.

DNE Xfer Done - this bit is set to one (1) when the DMA operation is completed (successfully or not). This bit is write one (1) to clear.

TCZ Transfer Counter Zero - this bit is set when the transfer counter has a value of zero. Cleared otherwise.

TBE Transmit Buffer Empty - This bit is set to one (1) if the attached target requests data from the SII while there is no command pending for the datamover (i.e. an Info Xfer command is not in progress). This bit is cleared when an Info Xfer command is started (either DMA or programmed I/O).

IBF Input Buffer Full - this bit is set to one (1) if the SII has received a byte while there is no command pending for the datamover (i.e. an Info Xfer command is not in progress). This bit is cleared when an Info Xfer command is started (either DMA or programmed I/O).

IPE Incoming Parity Error - this bit is set to one (1) if there was a parity error on the incoming data. It remains asserted until the next DMA operation begins.

OBB Odd Byte Boundary - this bit is set if the current transfer has ended on an odd byte boundary. It is automatically reset by the SII when the next DMA operation begins. This can be used in conjunction with the previous register to solve "odd byte disconnects".

MIS Phase Mismatch - this bit is set to one (1) if the phase currently on the bus does not match the expected phase (as described in the SII_COMM register) and a REQ has been issued by the target. This bit should only be asserted while acting in the initiator role. This bit is cleared by resolving the difference in phase by modifying the SII_COMM register.

ATN this bit is set to one (1) if, while the SII was in the target role, the initiator asserted ATN. Write one (1) to clear this bit.

C/D set to one (1) if the current bus state has the C/D signal asserted. This bit is read only.

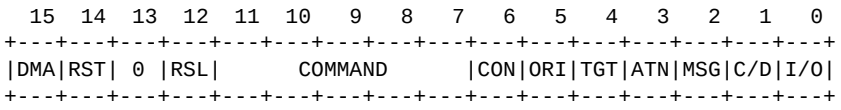
MSG set to one (1) if the current bus state has the MSG signal asserted. This bit is read only.

I/O set to one (1) if the current bus state has the I/O signal asserted. This bit is read only.

### 6.5.1.15. SII_COMM - Command Register

The SII uses this register to determine its actions while operating in SCSI mode. This register also contains information concerning use of DMA in the present command.

SII_COMM (40) -- READ/WRITE




The bits in this register are defined as follows:

DMA when asserted, data will be transferred to and from the memory area. When clear, data will be sent and received via the processor through the SII_DATA register.

RST when written to one (1), the SII will assert RST on the SCSI bus for 25 microseconds. This bit always reads zero.

RSL when asserted (1) along with a SELECT command, the SII will attempt to reselect the desired device. When clear (0), the SII will attempt a selection.

COMMAND<4:0>

these bits, collectively, control the actions of the SII. The roles in which these commands are valid are listed parenthetically.

I initiator role

T target role

D disconnected

These bits are defined in the following manner:

00001 Chip Reset - (I,T,D) - This command resets the entire chip in the same manner as a "hard" reset.

00010 Disconnect - (I,T,D) - This command forces the SII to release all signals it is driving on the SCSI bus (as a target). It is also used to "gracefully" abort a selection/reselection attempt.

The disconnect bit should not be written to one (1) if the SII is already disconnected. The command will remain in the COMM register and cause the SII to disconnect immediately following the next time it is selected.

00100 Request Data - (T) - This command forces the SII to issue a REQ on the SCSI bus. This command must be used only while a target receiving data. In order for this command to be executed, bits <6:3> of the SII_COMM register must match bits <6:3> of the SII_CSTAT and bits <2:0> of the SII_COMM register must match bits <2:0> of the SII_DSTAT register.

01000 Select - (D) - This command allows the SII to attempt to select or reselect another device on the bus.

10000 Information Transfer - (I,T) - This command allows the SII to transfer information to and from another device. This command is only valid while connected to another device. In order for this command to be executed, bits <6:3> of the SII_COMM register must match bits <6:3> of the SII_CSTAT and bits <2:0> of the SII_COMM register must match bits <2:0> of the SII_DSTAT register.

The information transfer bit is only cleared by clearing the command or a DONE interrupt. It is not cleared when the SII becomes disconnected from the bus or when RSTIN is asserted.

State lines

(D,I,T) - These three bits make up the expected state of the chip. These bits must match those in the SII_CSTAT register for a data transfer to take place. These bits are:

CON Connected

DST Destination

TGT Target

Control lines

(I,T) - This is used to directly drive several of the bus signals. While in SCSI mode and acting as a target, the values written to C/D,I/O, and MSG are driven onto the SCSI bus. While acting as an initiator, ATN is driven onto the SCSI bus. In either mode, the bits constitute the "expected phase". The three phase bits must match those on the SCSI bus or MIS is set in the SII_DSTAT register (following receipt of a REQ from the target).

#### 6.5.1.16. SII_DICTRL - Diagnostic Control Register

This register contains the various control bits used in diagnostic mode.

SII_DICTRL (42) -- READ/WRITE

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| - | - | - | - | - | - | - | - | - | - | - | - | - |LPB|PRE|DIA|TST|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

The bits is this register are defined as follows:

LPB set to one (1) if the values written to the diagnostic registers are to be looped back into the chip. This will enable the processor to insert test vectors into the chip during power-up diagnostics if desired. Note that the DIA bit must be deasserted for this test to be meaningful. Clear (0) (default on reset) otherwise.

PRE port enable. Set to one (1) to enable the off-chip drivers to the SCSI port. After a reset, the SII will be disconnected from the bus (this bit will be zero). The primary purpose of this bit is to allow chip diagnostics to run without affecting the rest of the SCSI bus.

DIA When this bit is asserted, the SII is in external loop-back mode. In this mode, the diagnostic registers directly control the SCSI data and control lines, as well as the bus steering signals. After a RESET condition, this bit is zero (0).

TST is set to one (1), when the chip is in test mode. This bit must always be zero.

### 6.5.2. Commands

The following is a description of the command set for the SII chip. Included will be operation of the chip during each command along with the various results due to execution. Registers used by the chip during the execution of the command will also be mentioned. Also included is a summary of the interrupts which may occur following the issuance of each command.

Commands will be divided into two groups, immediate and complex. All immediate commands are executed immediately, and return no status information. Complex commands are executed as soon as possible and interrupt the processor when the command has been completed. In addition, most will return status information regarding the execution of the operation.

#### 6.5.2.1. Immediate Commands

##### 6.5.2.1.1. Chip Reset

This command will stop any operation presently executing, and reset the chip. The registers will return to their default values and the chip will be left disconnected from the SCSI bus. This command may be executed in any mode ( disconnected, initiator, or target), although it is recommended that the chip is disconnected from the SCSI bus when this command is issued. Following the issuance of this command the following interrupts may occur:

<none>

the SII will reset its registers and IE will be disabled (as will selection and reselection attempts). This precludes interrupts from occurring.

##### 6.5.2.1.2. Disconnect

This command will cause the SII chip to immediately release all signals on the SCSI bus. As the target, this is typically used to end a transfer. As an initiator, this is used in the case of a firmware timeout during a selection. A disconnect command, in this case, will cause the SII to abort the selection in the way described by the SCSI specification. When disconnected already, this command will cause the SII to disconnect following the next time it is selected. Only the SII_CSTAT register is affected by this command. Following the issuance of this command the following interrupts may occur:

SCH acknowledgment of the disconnect or a device has selected or reselected the SII. The state bits can be used to determine what has happened.

RST a device has issued RST on the bus.

#### 6.5.2.2. Complex Commands

##### 6.5.2.2.1. Request Data

This command is only valid when in the target role. During a transfer directed towards the target (programmed I/O), the data must be requested first. The expected chip state written in the SII_COMM register must match the state of the chip for this command to be executed. Following the issuance of this command the following interrupts may occur:

IBF the initiator has sent a byte to the target. The following status bit may be set:

IPE the byte just received contained a parity error.

RST a device has asserted Reset on the SCSI bus.

#### 6.5.2.2.2. Select

This command instructs the SII to arbitrate for the SCSI bus and select a SCSI device. The ID of the device to be selected must be placed in the SII_SLCSR register. See SII_SLCSR definition for more detail. The SII chip will interrupt the processor for one of the following reasons:

SCH a state change has occurred for one of the following reasons.

- The selection has been accomplished. Some of the transfer status bits (SII_DSTAT) may be set:

  BER the target violated SCSI protocol

  MIS the target has issued a REQ in a phase other than that currently in the SII_COMM register.

  IBF the target has sent a byte to the SII.

  TBE the target has requested a byte from the SII.

- The SII has lost the arbitration and has been selected by another device. In this case, the ATN status bit may also be asserted.

- The SII has lost the arbitration and has been reselected by another device. In this case, the following bits may also be set:

  BER the target violated SCSI protocol

  MIS the target has issued a REQ in a phase other than that currently in the SII_COMM register.

  IBF the target has sent a byte to the SII.

  TBE the target has requested a byte from the SII.

RST A device has asserted RST on the bus.

#### 6.5.2.2.3. Information Transfer Command

This command allows the transfer of information to or from this SII. The direction of the transfer is contained in the bus phase, which can also be found in the SII_COMM register. The state of the chip must match the expected state written to the SII_COMM register. This command can be aborted by clearing bit <11> in the SII_COMM register. This command is not cleared when the SII disconnects from the bus, or when RSTIN is asserted. The SII will interrupt the processor for the following reasons:

DNE The transfer was completed. The transfer status bits can be used to determine if completed successfully. The following transfer status bits may be set:

TCZ The transfer counter has a zero count. In DMA mode, all bytes were received or transferred.

IPE The SII received a byte with a parity error. In the target role, the SII_DMLOTC register contains the number of bytes not sent (or received). In the initiator role, the SII must continue to accept data. However, the chip will automatically assert ATN on the bus.

OBB The DMA transfer has ended on an odd byte boundary. The SII_DSTAT will also indicate if the initiator has asserted ATN during the transfer.

IBF The SII has received a byte while the DataMover was idle (SII was in programmed I/O mode or the DMA was completed and more data arrived.)

TBE A byte has been requested from the SII while the DataMover was idle (SII was in programmed I/O mode or the DMA was completed and more data was requested.)

SCH When connected as an initiator, the attached target disconnects.

MIS When connected as an initiator, the target changes the phase and sends a REQ.

RST Reset is asserted on the SCSI bus.

BER A bus error occurs during the operation.

### 6.5.3. SCSI Operations

While the SII is operating in SCSI mode, it behaves quite similar to most industry-standard SCSI protocol controller chips.

The following is a detailed description explaining the procedure needed to execute many SCSI-type functions.

#### 6.5.3.1. Initiator Selection Of A Target

1. Begin by loading the ID of the device to be selected into the SII_SLCSR register.
2. Bits <11:7> of the SII_COMM register should be set to 01000B. This will instruct the SII to attempt to select the desired device. Bit 15 should be cleared since no DMA is involved. Bit 12 should be cleared since this is not a reselection. Should this SII win arbitration, IGS will be asserted. ATN and RST will both remain deasserted. The command code for selection would be 0400H. At this time, it is advisable to begin a firmware timer.
3. At the next detection of a bus free phase, the SII will arbitrate for the SCSI bus. Several actions may result from this:
   - The SII loses the arbitration and, before the software timeout, is selected by another device. In this case, the SII interrupts with the SCH bit set in the SII_CSTAT register. Since the SII is now connected to another device, the SELECT command is forgotten.
   - The SII wins the arbitration (before the timeout interval) and the selected device responds. In this case, the SII interrupts with the SCH bit set in the SII_CSTAT register.
   - No interrupt is generated before the software timer expires.

   The first two situations are straight forward. The last, however, is slightly more involved. In this case, the software must evaluate which of the following situations has occurred:
   - The SII has won the arbitration and the selected device did not respond.
   - The SII lost the arbitration. It continued to attempt the selection at each successive detection of a bus free phase. However, it was unsuccessful in gaining control of the SCSI bus before the software timer expired.

By reading the SII_CSTAT, the processor can tell which of these conditions exists. If the SII is currently attempting selection, the SIP bit (and only this bit) in the SII_CSTAT will be asserted. If this is true, the software may abort the selection by issuing a DISCONNECT command. The SII will abort the selection in the way explained in the SCSI specification and clear the SIP bit. If the SII hasn't been successful in gaining control of the SCSI bus, the LST bit in the SII_CSTAT register will be asserted. In this case, the processor may wish to restart the software timer.

#### 6.5.3.2. Initiator Selection With ATN Of A Target

This procedure is similar to that detailed above. Only the differences will be pointed out.

1. Begin by loading the ID on the device to be selected into the SII_SLCSR register.
2. Bits <11:7> of the SII_COMM register should be set to 01000B. This will instruct the SII to attempt to select the desired device. Bit 15 should be cleared since no DMA is involved. Bit 12 should be clear since this is not a reselection. Should this SII win arbitration, IGS will be asserted. ATN will be asserted by writing bit 3 to one. The command code for selection with ATN would be 0408H.
3. The rest follows the procedure explained for selection.

#### 6.5.3.3. Target Reselection Of An Initiator

This procedure is similar to that detailed for initiator selection. Only the differences will be pointed out.

1. Begin by loading the ID on the device to be reselected into the SII_SLCSR register.
2. Write 1400H to the SII_COMM register. This will instruct the SII, with RSL bit set, to attempt to reselect the desired device. This also causes the I/O line to be asserted once the SII gains control of the SCSI bus.
3. The rest follows the procedure explained for selection.

#### 6.5.3.4. Information Transfers

This will detail the steps needed to perform an information transfer in both DMA and programmed I/O mode.

#### 6.5.3.4.1. Initiator DMA Information Transfers

Information transfers require that the expected phase and state match those which exist currently or the transfer will not take place.

In this mode, the SII will automatically transfer information without processor intervention (excepting error conditions).

1. Typically, the SII will generate a MIS interrupt, signaling the processor that a bus phase change has occurred. Reading the SII_DSTAT register will inform the processor which phase the bus has transitioned to.
2. Load the starting byte address of the buffer area into the SII_DMADDRL and SII_DMADDRH registers. The SII will begin to read or deposit information from this address.
3. Load the SII_DMLOTC register with the numbers of bytes which are to be transferred during this operation.
4. If this is a write to memory and the starting address is odd, the SII_DMABYTE register can be loaded at this time. This will insure that a particular value is written into the low byte of the first word in the transfer.
5. Write the command code to the SII_COMM register. This will consist of bit 15 set since this operation involves DMA. along with bits <11:7> set to 10000B. The phase bits and state bits must match those in the SII_DSTAT and SII_CSTAT registers if the transfer is to take place. A COMMAND OUT using DMA would be encoded as 8862H by the initiator.
6. The SII will interrupt some time later for one of the following reasons:

   SCH The attached target disconnects.

   RST Reset has been asserted on the SCSI bus.

   BER A bus protocol error has occurred.

   DNE the transfer was completed.

   MIS the target has changed phase.

   TBE the target requested more bytes of information than indicated by the SII_DMLOTC.

   IBF The target changes the information phase and sends data or the target sends more data than indicated by the SII_DMLOTC.

   With the last four, the following status bits may be set:

   TCZ The transfer counter has reached zero, and all bytes were transferred.

   IPE While receiving data, a parity error occurs. In this case, it must continue to receive data. However, the chip will assert ATN on the bus.

   The assertion of both DNE and MIS indicates that the operation was successful and the target has now begun a new operation.

   The DMA operation can be aborted by the processor by clearing bit <11> in the SII_COMM register. The processor should then wait for a DNE interrupt before continuing.

#### 6.5.3.4.2. Initiator Programmed I/O Transfers

In this mode, the processor must read (or write) each byte that is transferred on the SCSI bus. This is only recommended for one byte transfers.

1. Typically, the SII will generate a MIS interrupt, signaling the processor that a bus phase change has occurred, along with either IBF or TBE status.
2. The processor reads (or writes) the SII_DATA register.
3. Write the command code to the SII_COMM register. This will consist of bit 15 cleared since this operation does not involve DMA. along with bits <11:7> set to 10000B. The phase bits must match those in the SII_DSTAT register if the transfer is to take place. A COMMAND OUT using programmed I/O would be encoded as 0862H by the initiator.

   Interrupts may also occur for the following reasons:

   SCH The attached target disconnects.

   RST Reset has been asserted on the SCSI bus.

   BER A bus protocol error has occurred.

   TBE The target requests another byte (a phase change need not occur).

   IBF The target sends another byte ( a phase change need not occur).

   MIS The target has changed phase.

   The following status bit may be set:

   IPE While receiving data, a parity error occurs. In this case, the chip will assert ATN on the bus.

   Programmed I/O is on a byte basis. In other words, a command to receive or send data in non-DMA mode is valid for only one byte. If multiple bytes are to be transferred in this mode, each new byte must be accompanied by a new command.

#### 6.5.3.4.3. Target DMA Information Transfers

- In this mode, the SII will automatically transfer information without processor intervention (excepting error conditions).

4. Typically, the SII will generate an interrupt, signaling that the previous command has been completed or the SII has been selected.
5. Load the starting byte address of the buffer area into the SII_DMADDRL and SII_DMADDRH registers. The SII will begin to read or deposit information from this address.
6. Load the SII_DMLOTC register with the numbers of bytes which are to be transferred during this operation.
7. If this is a write to memory and the starting address is odd, the SII_DMABYTE register can be loaded at this time. This will insure that a particular value is written into the low byte of the first word in the transfer.
8. Write the bus phase which this transfer will take place in the SII_COMM register. This should be written before the command is written to allow time for it to settle on the bus.
9. Write the command code to the SII_COMM register. This will consist of bit 15 set since this operation involves DMA, along with bits <11:7> set to 10000B. The desired phase should be encoded in bits <4:2>. For example, a target may issue a COMMAND OUT using DMA by writing 8852H to the SII_COMM register.
10. The SII will interrupt some time later for one of the following reasons:

    RST Reset has been asserted on the SCSI bus.

    BER A bus protocol error has occurred.

    DNE the operation is done.

    One or more of the following status bits may also be set:

    TCZ The transfer counter has reached zero, and the transfer has been completed.

    IPE While receiving data, a parity error occurs. In this case, it will stop asserting REQ (thus stopping the transfer). SII_DMLOTC will indicate the number of bytes not transferred.

    ATN The initiator asserts ATN.

The DMA operation can be aborted by the processor by clearing bit <11> in the SII_COMM register. The processor should then wait for a DNE interrupt before continuing.

#### 6.5.3.4.4. Target Programmed I/O Transfers

In this mode, the processor must read (or write) each byte that is transferred on the SCSI bus. This is only recommended for one byte transfers.

- Reads
  1. Typically, the SII will interrupt with a DNE interrupt.
  2. A REQDATA command must be issued. The command code for this would be 0252H to request data in COMMAND OUT phase.
  3. Wait for a IBF interrupt.
  4. Read the SII_DATA register.
  5. Write the command code to the SII_COMM register. This would be 0852H in this case. This clears the IBF signal.

     The following status bits may or may not be set:

     IPE While receiving data, a parity error occurs. In this case, the chip will stop asserting REQs on the bus (thus terminating the operation).

     ATN The initiator asserts ATN on the bus.
  6. A DNE interrupt will result. This signals the end of the operation.

- Writes
  1. Typically, the SII will interrupt with a DNE interrupt.
  2. Next, write the SII_DATA register with the value to be sent.
  3. Write the command code to the SII_COMM register. For the case of a DATA IN transfer, this code would be 0851H.
  4. A DNE interrupt will result some time later. This signals the end of the operation.

#### 6.5.3.5. Initiator Setting ATN

This can be done only in the initiator role. Typically this is used to allow the initiator to request a MSG OUT phase.

This is accomplished by reading the SII_COMM register, 'or'ing 0008H and writing this value back to the SII_COMM register. This allows the previous command to continue. Note that the SII will automatically assert ATN if it detects a parity error during an initiator transfer.

#### 6.5.3.6. SII Setting RST

This can be performed at any time by writing 4000H to the SII_COMM register. Note that in SCSI mode, this is equivalent to a hard reset of all devices.

### 6.5.3.7. Command Chaining

Since the Request Data and Transfer Info commands are only executed when the state is matched, command chaining is possible. For example, it might be desirable to select another device with ATN and if successful, expect a Message Out phase and do a DMA. This is possible with a single command. The DataMover registers (SII_DMLOTC, SII_DMADDRL, SII_DMADDRH, SII_DMABYTE, etc.) should be loaded first. Next the SII_SLCSR register should be loaded. Lastly, the command should be loaded. The command would include both a select and info transfer command. In addition, the expected state of connected, not destination and initiator must be loaded. The select command executes immediately. If the SII is successful in selecting the remote device, the chip's state will match the expected state and the DMA will occur. However, if the chip is selected or the phase does not match the expected phase, no data transfer will take place. There are several other variations of this, which are enumerated below.

- Wait for Select, then DMA. The command that must be loaded is: 1000100001010xxx. This indicates:
  - DMA enabled
  - Transfer Info command
  - Expected state: connected as target, destination
  - Phase: whatever desired

- Wait for Select, then Request Data. The command that must be loaded is: 0000001001010xxx. This indicates:
  - Request Data command
  - Expected state: connected as target, destination
  - Phase: whatever desired

- Wait for Select with ATN, then DMA. The command that must be loaded is: 1000100001011xxx. This indicates:
  - DMA enabled
  - Transfer Info command
  - Expected state: connected as target, destination and ATN set during selection
  - Phase: whatever desired

- Wait for Select with ATN, then Request Data. The command that must be loaded is: 0000001001011xxx. This indicates:
  - Request Data command
  - Expected state: connected as target, destination and ATN set during selection
  - Phase: whatever desired

- Select, then DMA. The command that must be loaded is: 1000110001100xxx. This indicates:
  * DMA enabled
  * Transfer Info command
  * Select command
  * Expected state: connected as initiator, origin
  * Phase: must match desired phase

- Select with ATN, then DMA. The command that must be loaded is: 1000110001101xxx. This indicates:
  * DMA enabled
  * Transfer Info command
  * Select command
  * Expected state: connected as initiator, origin
  * ATN set during selection
  * Phase: must match desired phase

- Reselect, then DMA. The command that must be loaded is: 1001110001110xxx. This indicates:
  * DMA enabled
  * Transfer Info command
  * Select command, RSL set (reselect)
  * Expected state: connected as target, origin
  * Phase: whatever desired

- Reselect, then Request Data. The command that must be loaded is: 1001011001110xxx. This indicates:
  * Request Data command
  * Select command, RSL set (reselect)
  * Expected state: connected as target, origin
  * Phase: whatever desired

- Wait for reselect, then DMA. The command that must be loaded is: 1000100001000xxx. This indicates:
  * DMA enabled
  * Transfer Info command
  * Expected state: connected as initiator, destination
  * Phase: must match desired phase

All commands using DMA can also be done in programmed I/O mode by clearing the DMA bit.

These are the only commands that can be chained.

### 6.6. Network Interface

The Ethernet network interface is based upon the LANCE chip and a 32 K by 16-bit network buffer. The LANCE manages transmission and reception of packets via ring descriptors and packet buffers located in the network buffer. The network buffer is time-multiplexed between the LANCE and the processor. The LANCE can only access its buffer during DMA transfers; it has no access to memory.

An AM7992 SIA implements the serial interface to a 15-pin DB connector for an external transceiver. The SIA is transformer coupled to the connector.

DS3100 also implements a ThinWire Ethernet transceiver. Use of the ThickWire connection to an external transceiver, or use of the integral ThinWire transceiver is selected by a push-button switch. A green LED adjacent to the ThickWire DB15 connector illuminates while the switch selects the ThickWire interface. A green LED adjacent to the ThinWire BNC connector illuminates while the switch selects the ThinWire interface.

The LANCE interrupts the CPU at level 1, which is visible in the R2000 CAUSE<11> register bit.

The LANCE only supports half-word reads and writes. The LANCE registers are word-aligned. Reads of the LANCE nominally stall the CPU for 12 cycles. Writes to the LANCE nominally complete in 16 cycles. The network buffer only supports half-word reads and writes. The network buffer is word-aligned in the processor address space. Reads of the buffer stall the CPU for 7 cycles when the LANCE is idle; a peak read bandwidth of 4.8 MBytes/second. Writes to the buffer complete in 10 cycles when the LANCE is idle; a peak write bandwidth of 3.3 MBytes/second. During LANCE DMA, CPU access to the LANCE registers or the network buffer may incur an additional latency of up to 5 microseconds. Table 6-4 lists the LANCE register and network buffer addresses.

**Table 6-4: LANCE Register and Network Buffer Addresses**

| Address | Register |
|---|---|
| 0xB8000000 | LANCE_RDP |
| 0xB8000004 | LANCE_RAP |
| | |
| 0xB9000000 | Buffer base |

Since the buffer is word-aligned to the CPU, all addresses specified to the LANCE must be left-shifted by one when referencing the buffer. That is, from the CPU successive half-words of the buffer are at 0xB9000000, 0xB9000004, 0xB9000008, etc., while they are at 0x0000, 0x0002, 0x0004, etc., to the LANCE.

#### 6.6.1. 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 (CSR's) within the LANCE chip which are programmed by a processor to initialize the LANCE chip and start its independent operation. Once started, The LANCE uses its builtin DMA controller to directly access the network buffer to get additional operating parameters and to manage the buffers it uses to transfer packets to and from the Ethernet.

All references to *memory* in the following LANCE descriptions refer to the network buffer. The LANCE generates 24-bit physical addresses. Since the network buffer is only 64 KBytes in size, the high order eight bits of the LANCE address have no effect. However, software should always zero those high-order address bits for future compatibility.

The LANCE uses three structures in memory:

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 below.

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 below.

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 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.

### 6.6.2. Programming of the LANCE

Program control of the LANCE chip is via two 16-bit read/write ports, LANCE_RAP and LANCE_RDP. These ports provide access to four 16-bit control and status registers which are named LANCE_CSR0 through LANCE_CSR3. A CSR is accessed by first writing its number into the register address port LANCE_RAP after which the contents of the CSR are read or written by accesses to the register data port LANCE_RDP. Note that registers other than LANCE_CSR0 may be accessed only while the STOP bit of LANCE_CSR0 is set.

#### 6.6.2.1. Register Address Port (LANCE_RAP)

The LANCE_RAP register selects which of the four CSR's is accessed via the register data port.

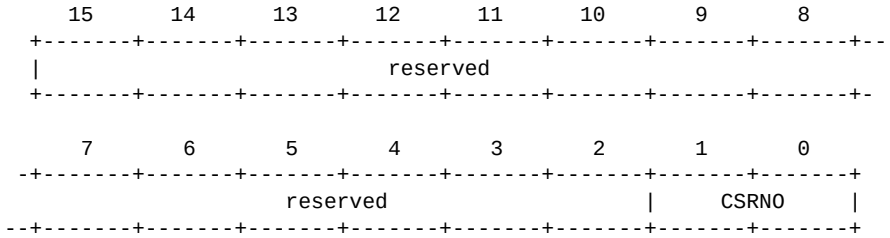



**Figure 6-2: LANCE Register Address Port (LANCE_RAP)**

<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 | LANCE_CSR0 |
| 0 1 | LANCE_CSR1 |
| 1 0 | LANCE_CSR2 |
| 1 1 | LANCE_CSR3 |

#### 6.6.2.2. Register Data Port (LANCE_RDP)

The register data port is a 16-bit window through which a processor can read and write the CSR designated by the register address port LANCE_RAP.

Note that registers LANCE_CSR1, LANCE_CSR2, and LANCE_CSR3 are accessible only while the STOP bit in LANCE_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 LANCE_RDP do not alter the register address pointer LANCE_RAP. In normal operation only LANCE_CSR0 can be accessed, so LANCE_RAP should be set to point to LANCE_CSR0 and left that way.

#### 6.6.2.3. Control and Status Register 0 (LANCE_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 LANCE_RDP when bits 1:0 of LANCE_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.

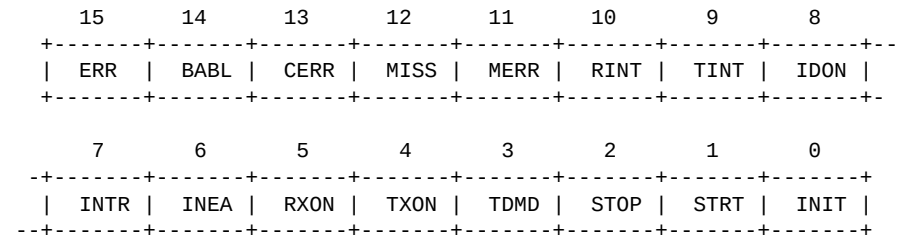



**Figure 6-3: LANCE Control and Status Register 0 (LANCE_CSR0)**

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 microseconds 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 microseconds after beginning the memory 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.

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 processor.

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 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 signaled 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 (LANCE_CSR1, LANCE_CSR2, and LANCE_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 LANCE_CSR1 and LANCE_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.

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

1. Set STOP in LANCE_CSR0
2. Set up the initialization block in memory
3. Load LANCE_CSR1 and LANCE_CSR2 with the starting address of the initialization block
4. Set INIT in LANCE_CSR0
5. Wait for IDON in LANCE_CSR0 to become set
6. Set STRT in LANCE_CSR0 to begin operation

#### 6.6.2.4. Control and Status Register 1 (LANCE_CSR1)

This read/write register is used in conjunction with LANCE_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 LANCE_RDP when bits 1:0 of LANCE_RAP are 01 and the STOP bit of LANCE_CSR0 is set. Its contents upon power-on are UNPREDICTABLE.

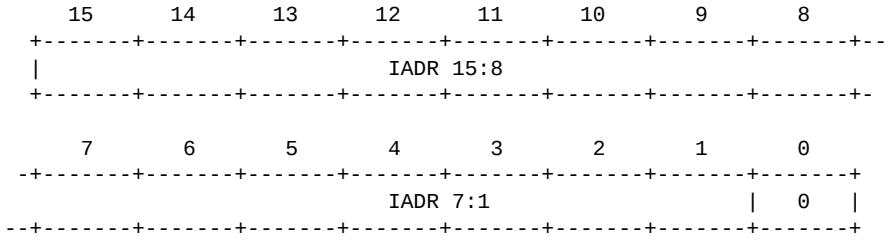



**Figure 6-4: LANCE Control and Status Register 1 (LANCE_CSR1)**

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.

#### 6.6.2.5. Control and Status Register 2 (LANCE_CSR2)

This read/write register is used in conjunction with LANCE_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 LANCE_RDP when bits 1:0 of LANCE_RAP are 10 and the STOP bit of LANCE_CSR0 is set. Its contents upon power-on are UNPREDICTABLE.

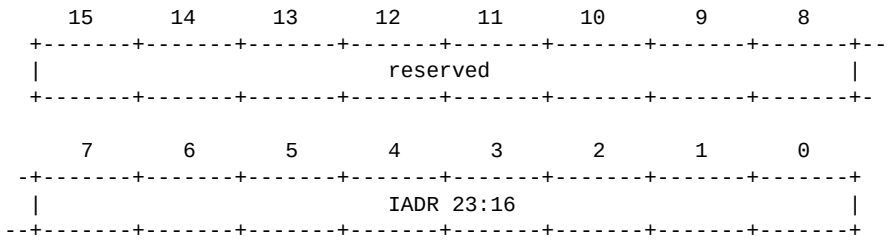



**Figure 6-5: LANCE Control and Status Register 2 (LANCE_CSR2)**

<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.

#### 6.6.2.6. Control and Status Register 3 (LANCE_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 LANCE_RDP when bits 1:0 of LANCE_RAP are 11 and the STOP bit of LANCE_CSR0 is set. Its contents upon power-on are entirely zeros.

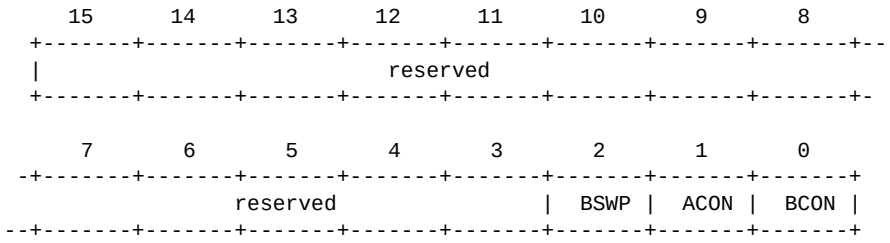



**Figure 6-6: LANCE Control and Status Register 3 (LANCE_CSR3)**

<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 accommodate 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 LANCE_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 LANCE_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 LANCE_CSR0 is set. For this system, this bit must be ZERO.

### 6.6.3. Interrupts

The LANCE chip asserts an interrupt request signal whenever the INTR and INEA bits in its control and status register 0 (LANCE_CSR0) are both ones. This signal is presented to the processor.

### 6.6.4. DMA Operation

The LANCE chip contains a built-in DMA controller which can transfer data directly between the chip and memory in the address range 0x0000 through 0xFFFF. The chip 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 word transfer requires 0.6 microseconds, so a 16-byte burst will require 4.8 microseconds.

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.

### 6.6.5. Initialization Block

When the LANCE chip is initialized (by setting the INIT bit in LANCE_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 LANCE_CSR1 and LANCE_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:

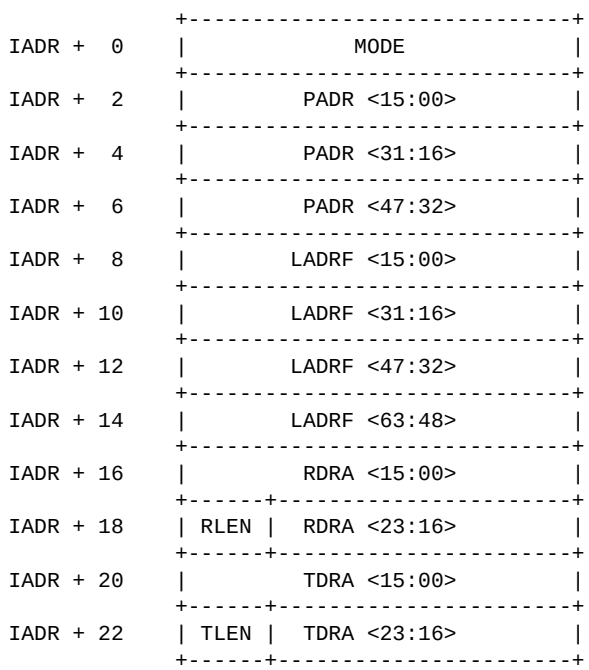



**Figure 6-7: LANCE Initialization Block Layout**

#### 6.6.5.1. 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.

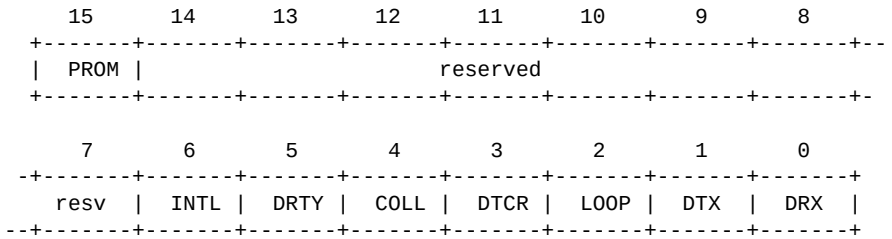



**Figure 6-8: Initialization Block MODE Word (NIB_MODE)**

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.

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 LANCE_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 LANCE_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 LANCE_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.

#### 6.6.5.2. 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.)

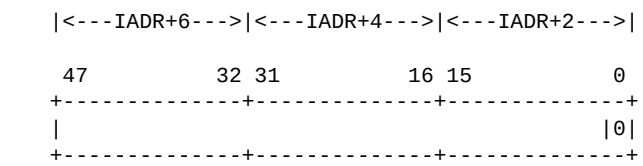



**Figure 6-9: Network Physical Address (NIB_PADR)**

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 Ethernet Station Address ROM. The low-order bit (bit 0) of this address must be zero since it is a physical address.

#### 6.6.5.3. 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.

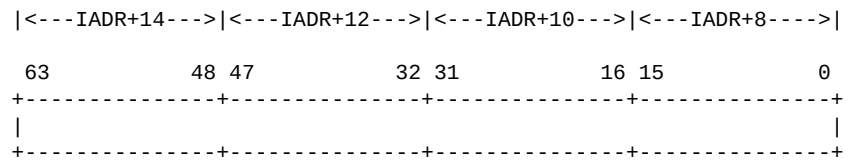



**Figure 6-10: Multicast Address Filter Mask (NIB_LADRF)**

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) 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.

#### 6.6.5.4. Receive

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

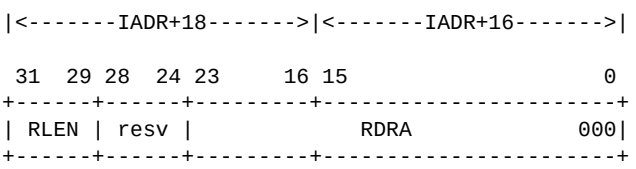



**Figure 6-11: Receive Descriptor Ring Pointer (NIB_RDRP)**

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.

#### 6.6.5.5. Transmit Descriptor Ring Pointer (NIB_TDRP)

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

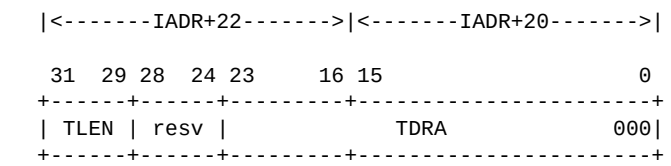



**Figure 6-12: Transmit Descriptor Ring Pointer (NIB_TDRP)**

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.

### 6.6.6. 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 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).

It is recommended that software configure all receive and transmit buffers to be the maximum packet size of 1518 bytes. The 64 KByte network buffer supports 32 such receive buffers, 8 such transmit buffers, plus the receive and transmit rings and the initialization block. Even with the theoretical worst case sequence of 32 back-to-back minimum length packets, the receiver buffers will not be exhausted for 2.157 milliseconds in this configuration.

#### 6.6.6.1. Receive Buffer Descriptor

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

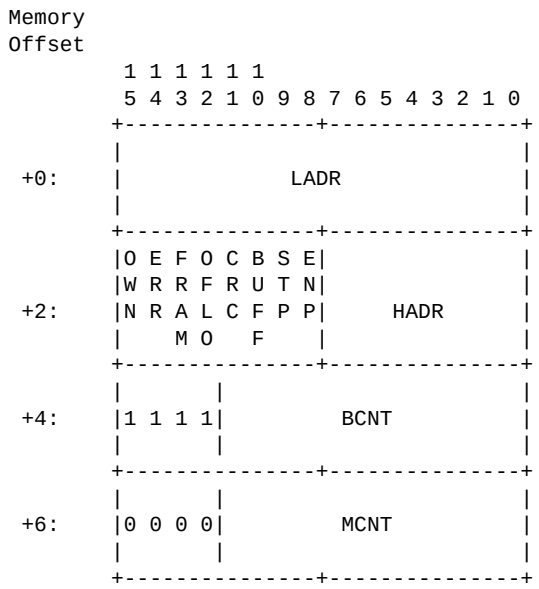



**Figure 6-13: Receive Buffer Descriptor**

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.

### 6.6.6.2. Transmit Buffer Descriptor

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

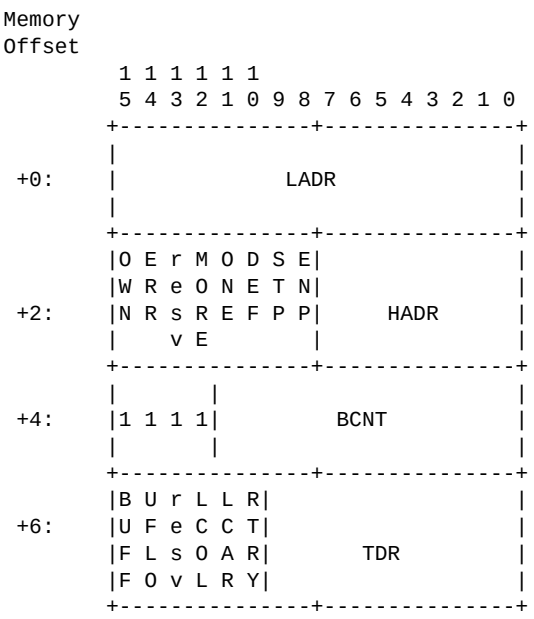



**Figure 6-14: Transmit Buffer Descriptor**

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 occurred 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 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 determining the approximate distance to a cable fault; it is valid only when the RTRY bit in this word is set.

### 6.6.7. 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 and the chip vendor's literature should also be consulted.

#### 6.6.7.1. Switch Routine

Upon power on, the STOP bit is set and the INIT and STRT bits are cleared in LANCE_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 LANCE_CSR1 and LANCE_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. Then if STRT is set, it begins active chip operation by jumping to the look-for-work routine. 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.

#### 6.6.7.2. Initialization Routine

The initialization routine is called from the switch routine when the latter finds the INIT bit set. It reads the initialization block from the memory addressed by LANCE_CSR1 and LANCE_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).

#### 6.6.7.3. 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 LANCE_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 to attempt to get a receive buffer.

Next the routine tests whether the transmitter is enabled (bit TXON of LANCE_CSR0 is set). If so, it calls the transmit poll routine 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 LANCE_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 is called.

#### 6.6.7.4. 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.

#### 6.6.7.5. 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 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 LANCE_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 LANCE_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 LANCE_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.

#### 6.6.7.6. 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.

#### 6.6.7.7. Transmit Poll Routine

The transmit poll routine is called by the look-for-work routine 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 LANCE_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 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.

#### 6.6.7.8. Transmit Routine

The transmit routine is called from the transmit poll routine 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 occurred. 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 LANCE_CSR0 to notify the host and returns to the look-for-work routine; otherwise it goes back to the lookahead section in this routine.

#### 6.6.7.9. 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.

#### 6.6.7.10. 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).

#### 6.6.7.11. 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:
   - Read LANCE_CSR0 and save the results.
   - Clear the interrupt enable bit INEA in the saved copy.
   - Write LANCE_CSR0 with the saved copy. 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 LANCE_CSR0 was read.
   - Write LANCE_CSR0 with only INEA to enable interrupts again.
   - Service all the interrupt and error conditions indicated by the flags in the saved copy.
   - Exit from the interrupt service routine.

1. An interrupt is signaled 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.

2. 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 LANCE_CSR0 and then setting STRT in LANCE_CSR0 (DTX will still be clear in the chip's internal copy of NIB_MODE). It is not necessary to set INIT to reread the initialization block.

   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).

3. The network transceiver 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.

4. 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.

5. 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.

6. The chip's internal copy of LANCE_CSR1 may become invalid when the chip is stopped. The LANCE_CSR1 and LANCE_CSR2 registers should always be loaded prior to setting INIT to initialize the LANCE chip.

7. 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.

8. 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.

9. 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.

10. 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.

11. Do not set INIT and STRT in LANCE_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.

12. Since neither the LANCE nor the network buffer support parity bits, it is recommended that operating system software always calculate and verify the software checksums present in the packets of higher-level network protocols.

### 6.6.8. Ethernet Station Address ROM

A 32 by 8-bit ROM contains the workstation's Ethernet station address (ESAR). The ROM can be read in bits 8 through 15 of the RTC address space. Reads of the ESAR stall the CPU for 14 cycles. The ESAR ROM is in a socket. Table 6-1 lists the addresses of the ESAR.

**Table 6-1: Ethernet Station Address ROM Addresses**

| Address | Content | | | Value |
|---|---|---|---|---|
| 0xBD000001 | Address | Octet | 0 | |
| 0xBD000005 | " | " | 1 | |
| 0xBD000009 | " | " | 2 | |
| 0xBD00000D | " | " | 3 | |
| 0xBD000011 | " | " | 4 | |
| 0xBD000015 | " | " | 5 | |
| 0xBD000019 | Chksum | Octet | 1 | |
| 0xBD00001D | " | " | 2 | |
| 0xBD000021 | " | " | 2 | |
| 0xBD000025 | " | " | 1 | |
| 0xBD000029 | Address | Octet | 5 | |
| 0xBD00002D | " | " | 4 | |
| 0xBD000031 | " | " | 3 | |
| 0xBD000035 | " | " | 2 | |
| 0xBD000039 | " | " | 1 | |
| 0xBD00003D | " | " | 0 | |
| 0xBD000041 | Address | Octet | 0 | |
| 0xBD000045 | " | " | 1 | |
| 0xBD000049 | " | " | 2 | |
| 0xBD00004D | " | " | 3 | |
| 0xBD000051 | " | " | 4 | |
| 0xBD000055 | " | " | 5 | |
| 0xBD000059 | Chksum | Octet | 1 | |
| 0xBD00005D | " | " | 2 | |
| 0xBD000061 | TEST | Pattern | 0 | FF |
| 0xBD000065 | " | " | 1 | 00 |
| 0xBD000069 | " | " | 2 | 55 |
| 0xBD00006D | " | " | 3 | AA |
| 0xBD000071 | " | " | 4 | FF |
| 0xBD000075 | " | " | 5 | 00 |
| 0xBD000079 | " | " | 6 | 55 |
| 0xBD00007D | " | " | 7 | AA |

## 6.7. System Clock/Battery-backed-up RAM

A MC146818 real-time-clock and RAM chip (RTC) supplies the system clock interrupt, a time-of-year clock, and 50 bytes of RAM. The time-of-year clock continues operation, and the 50 bytes of RAM preserve their contents, via a battery when the workstation is turned-off. The system clock interrupt rate is programmable from 122 microseconds to 500 milliseconds.

A battery supplies power to the RTC and its time base oscillator while system power is off. When starting from a fully charged condition, the battery will maintain valid time and RAM data in the RTC for a minimum of 100 hours. The battery is automatically recharged while system power is on.

The RTC interrupts the CPU at level 3, which is visible in the R2000 CAUSE<13> register bit.

The RTC only supports byte reads and writes. The RTC registers are word-aligned. Reads of the RTC stall the CPU for 14 cycles. Writes to the RTC complete in 17 cycles. Table 6-2 lists the RTC register addresses.

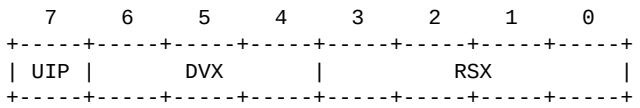
**Table 6-2: RTC Register Addresses**

| Address | Name | Register | Range |
|---|---|---|---|
| 0xBD000000 | RTC_SEC | Seconds | 0..59 |
| 0xBD000004 | RTC_ALMS | Seconds alarm | 0..59 |
| 0xBD000008 | RTC_MIN | Minutes | 0.59 |
| 0xBD00000C | RTC_ALMM | Minutes alarm | 0..59 |
| 0xBD000010 | RTC_HOUR | Hours | 0..23 |
| 0xBD000014 | RTC_ALMH | Hours alarm | 0..23 |
| 0xBD000018 | RTC_DOW | Day of week | 1..7 |
| 0xBD00001C | RTC_DAY | Date of month | 1..31 |
| 0xBD000020 | RTC_MON | Month | 1..12 |
| 0xBD000024 | RTC_YEAR | Year | 0..99 |
| 0xBD000028 | RTC_REGA | Register A | |
| 0xBD00002C | RTC_REGB | Register B | |
| 0xBD000030 | RTC_REGC | Register C | |
| 0xBD000034 | RTC_REGD | Register D | |
| 0xBD000038 | RTC_RAM | Base of BBU RAM | |

### 6.7.1. RTC Registers

The RTC contains 64 8-bit registers. Ten of these contain the date and time data, four are control and status registers, and the remaining 50 provide general purpose RAM storage. The alarm functions of the RTC are not supported and should not be enabled by software.

#### 6.7.1.1. Control Register A




RTC_REGA<7> Update In Progress (UIP)

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.

RTC_REGA<6:4> Timebase Divisor (DVX)

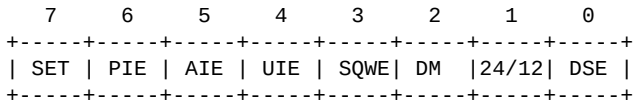
These read/write bits set the amount by which the time base oscillator input the the RTC is divided. These bits must be set to 010 to accommodate the 32.768 KHz time base in this system.

RTC_REGA<3:0> Rate Select (RSX)

These read/write bits select the rate at which the RTC generates periodic interrupts as shown in the following table. Software must also assert RTC_REGB<PIE> to enable the periodic interrupts.

| RSX | Rate | |
|---|---|---|
| 0x0 | none | |
| 0x1 | 3.90625 | ms |
| 0x2 | 7.8125 | ms |
| 0x3 | 122.070 | us |
| 0x4 | 244.141 | us |
| 0x5 | 488.281 | us |
| 0x6 | 976.562 | us |
| 0x7 | 1.953125 | ms |
| 0x8 | 3.90625 | ms |
| 0x9 | 7.8125 | ms |
| 0xA | 15.625 | ms |
| 0xB | 31.25 | ms |
| 0xC | 62.5 | ms |
| 0xD | 125 | ms |
| 0xE | 250 | ms |
| 0xF | 500 | ms |

#### 6.7.1.2. Control Register B




RTC_REGB<7> Set Time (SET)

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 software can set new date and time values.

RTC_REGB<6> Periodic Interrupt Enable (PIE)

When this bit is asserted, periodic interrupts occur as the rate specified by RTC_REGA<RSX>.

RTC_REGB<5> Alarm Interrupt Enable (AIE)

This bit is not used and must be set to zero.

RTC_REGB<4> Update Interrupt Enable (UIE)

When this bit is asserted, an interrupts occurs when the RTC_REGA<UIP> bit is asserted.

RTC_REGB<3> Square-wave Enable (SQWE)

This bit is not used and must be set to zero.

RTC_REGB<2> Date Mode (DM)

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).

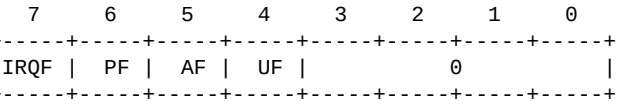
RTC_REGB<1> Hours Format (24/12)

This read/write bit selects the format of the RTC_HOUR and RTC_ALMH registers. A value of one selects 24-hour mode; a value of zero selects 12-hour mode. In the latter case, bit 7 of the hours registers is zero for AM and one for PM.

RTC_REGB<0> Daylight Savings Enable (DSE)

This read/write bit is obsolete and must be set to zero. Software must explicitly compensate the time to account for local daylight savings conventions.

#### 6.7.1.3. Control Register C




Note that the RTC_REGC register clears itself, and any pending interrupts, when read. If software enables more than one RTC interrupt source, it must save a copy of this register and dispatch to handlers from the saved copy.

RTC_REGC<7> Interrupt Request (IRQF)

When this read-only bit is set, it indicates that a RTC interrupt is pending.

RTC_REGC<6> Periodic Interrupt Flag (PF)

This read-only bit indicates that a RTC periodic interrupt is pending.

RTC_REGC<5> Alarm Interrupt Flag (AF)

This read-only bit indicates that a RTC alarm interrupt is pending.

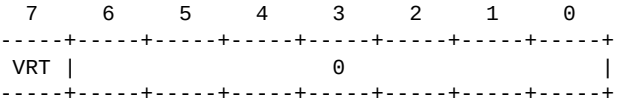
RTC_REGC<4> Update Interrupt Flag (UF)

This read-only bit indicates that a RTC update interrupt is pending.

RTC_REGC<3:0> RAZ

Not used; read as zero.

#### 6.7.1.4. Control Register D




RTC_REGD<7> Valid RAM/Time (VRT)

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 the system power is off and the backup battery voltage drops below the value required for the RTC to function properly. The bit is set to one after any read of this register (the register is read-only).

RTC_REGD<6:0> RAZ

Not used; read as zero.

#### 6.7.1.5. Time of Year Registers

The time of year is kept in six registers: RTC_SEC, RTC_MIN, RTC_HOUR, RTC_DAY, RTC_MON, and RTC_YEAR. A seventh register, RTC_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 as selected by the RTC_REGB<DM> bit.

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. The RTC_REGA<UIP> bit 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 RTC_REGA until it finds bit UIP zero, at which time is has a least 244 microseconds 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.

#### 6.7.1.6. Non-volatile RAM Storage

The 50 bytes of RAM storage are accessible at all times. They retain their value during power down provided the RTC_REGD<VRT> bit is asserted after power-up.

It is recommended that software implement a checksum on the non-volatile RAM contents as an additional safeguard against data corruption during battery-backed-up operation.

#### 6.7.1.7. Initialization

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

1. Load register RTC_REGB with bit SET equal to one to inhibit time updates and bits PIE, UIE, DM, and 24/12 set as desired.

2. Load the seven time registers with the current date and time.

3. Load register RTC_REGA to set the proper time base divisor and the desired periodic interrupt rate.

4. Load register RTC_REGB 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 fo the RTC are not affected by system power-on and power-off events.

### 6.8. System Control and Status Register

The Control and Status Register (SYS_CSR) is a 16-bit read/write register that controls the illumination of the LEDS and returns the state of various workstation options. Note that SYS_CSR is actually a 16-bit read-only status register, and a 16-bit write-only control register.

The SYS_CSR only supports half-word reads and writes. Reads of the SYS_CSR stall the CPU for 6 cycles. Writes to the SYS_CSR complete in 9 cycles. The SYS_CSR is word-aligned at address 0xBE000000. Figure 6-15 shows the format of the SYS_CSR during reads (STATUS).

Name: STATUS Address: BE000000 Access: R

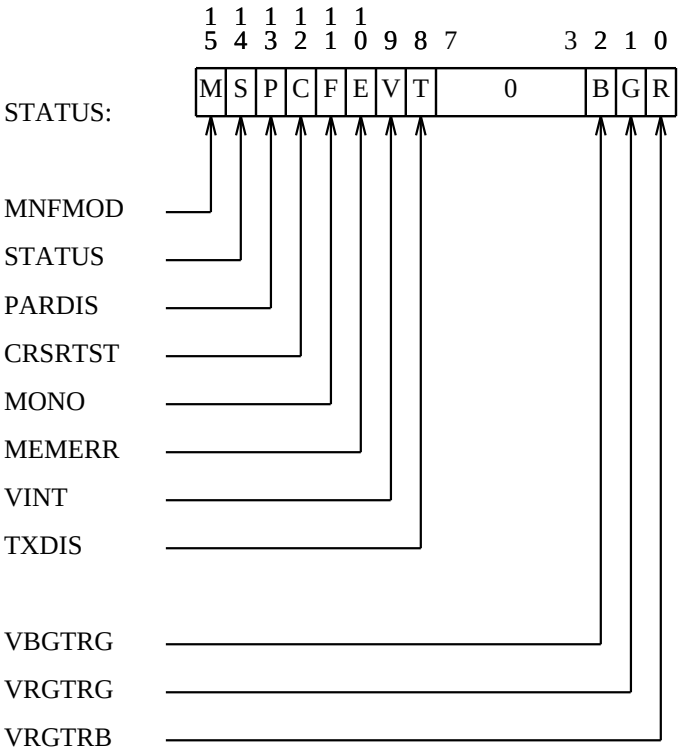



**Figure 6-15: Status Register**

VRGTRB

This bit is for use by self-test software. When asserted it indicates that the voltage level of the red video DAC is greater than the voltage level of the blue video DAC. The VRGTRB bit is only updated when the PCC programmable area detect 1 output asserts.

VRGTRG

This bit is for use by self-test software. When asserted it indicates that the voltage level of the red video DAC is greater than the voltage level of the green video DAC. The VRGTRG bit is only updated when the PCC programmable area detect 1 output asserts.

VBGTRG

This bit is for use by self-test software. When asserted it indicates that the voltage level of the blue video DAC is greater than the voltage level of the green video DAC. The VBGTRG bit is only updated when the PCC programmable area detect 1 output asserts.

TXDIS

This bit is for use by self-test software. TXDIS reflects the value of the TXDIS bit in the CONTROL register.

VINT

This bit asserts when the PCC programmable area detect 2 (PARD2) output asserts. Assertion of VINT also generates a level 1 interrupt in the CPU. Note that VINT and MEMERR shared the

same interrupt level. The video interrupt may be used to coordinate update of the color map with vertical retrace to avoid screen flicker from transient color map values during update. The PARD2 coordinate should be set to line 864. When color map updates are required, the PARD2 output should be enabled. On the subsequent interrupt, the color map should be updated and the PARD2 output disabled. The video interrupt must be serviced within the 685 microseconds of vertical retrace.

MEMERR

This bit asserts when a bus timeout occurs on a write transaction. Assertion of MEMERR also generates a level 1 interrupt in the CPU. Note that VINT and MEMERR shared the same interrupt level. When a MEMERR occurs, the address which caused the bus timeout may be obtained by reading the WB error address latch at location 0xB7000000. MEMERRs indicate that either software erroneously referenced a non-existent location, or that a hardware failure occurred.

MONO

When the MONO bit is one, a MSIM monochrome frame buffer is installed. When the MONO bit is zero, a CSIM color frame buffer is installed. Note that if no frame buffer is installed, the MONO bit will also be one. Consequently, software must probe the frame buffer memory to determine whether or not a monochrome frame buffer is installed.

CRSRTST

This bit is for use by self-test software. CSRSTST reflects the state of the PCC test output.

PARDIS

This bit is for use by memory diagnositc software. PARDIS reflects the state of the PARDIS bit in the CONTROL register.

STATUS

This bit is for use by self-test software. STATUS reflects the state of the STATUS bit in the CONTROL register.

MNFMOD

This bit is for use by self-test software in manufacturing. MNFMOD reflects the state of the MNFMOD jumper on the system module.

Figure 6-16 shows the format of the SYS_CSR during writes (CONTROL).

Name: CONTROL Address: BE000000 Access: W

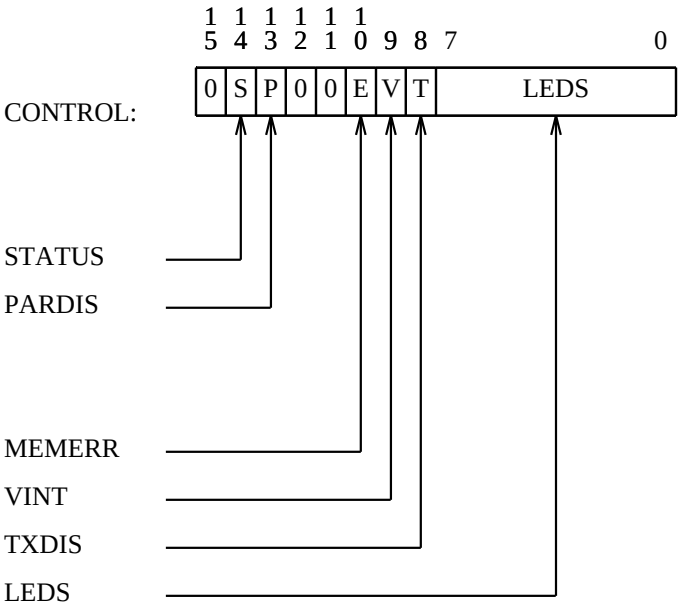



**Figure 6-16: Control Register**

LEDS

This field controls illumination of the system model LEDS. When a bit of this field is zero, the corresponding LED is illuminated. When a bit of this field is one, the corresponding LED is off. Since the control register is cleared by a reset or power-up, the LEDS are initially illuminated.

TXDIS

This bit is for use by self-test software. The EIA drivers are enabled by default when the SYS_CSR is cleared by a reset. Writing a one to TXDIS disables the EIA drivers so that data patterns present on the DZ serial outputs during internal loopback tests are not transmitted to the external serial devices.

VINT

Writing a 1 to VINT clears the VINT bit in the STATUS register. Writing a 0 to VINT has no effect. Refer to the STATUS register VINT description for more details.

MEMERR

Writing a 1 to MEMERR clears the MEMERR bit in the STATUS register. Writing a 0 to MEMERR has no effect. Refer to the STATUS register MEMERR description for more details.

PARDIS

This bit is for use by memory diagnostic software. Parity checking on memory reads may be disabled by asserting the PARDIS bit. This allows diagnostic software to isolate hard memory errors to the bit level via memory test patterns. The PARDIS bit must be zero during normal system operation.

STATUS

This bit is for use by self-test software. When self-test completes successfully, it should assert the STATUS bit. The value of STATUS is available on a system module header pin for use in

manufacturing.

### 6.9. Programmable Cursor

The graphics programmable cursor, based upon the DC503 Programmable Cursor Chip (PCC), supports a 16-by-16-pixel, 2-plane cursor. The PCC also contains two programmable area detect (PARD) circuits. PARD1 supports a self-test function that samples an analog voltage comparator on the video DAC outputs. PARD2 may be used to generate a video interrupt to coordinate video DAC color map updates with vertical retrace.

The PCC interrupts the CPU at level 4, which is visible in the R2000 CAUSE<14> register bit.

The PCC only supports half-word writes. The PCC registers are word-aligned in the processor address space. Writes to the PCC complete in 9 cycles. Table 6-3 lists the addresses of the PCC registers.

**Table 6-3: PCC Register Addresses**

| Address | Name | Register |
|---|---|---|
| 0xB1000000 | PCC_CMDR | Cursor command register |
| 0xB1000004 | PCC_XPOS | Cursor X position |
| 0xB1000008 | PCC_YPOS | Cursor Y position |
| 0xB100000C | PCC_XMIN1 | Region 1 left edge |
| 0xB1000010 | PCC_XMAX1 | Region 1 right edge |
| 0xB1000014 | PCC_YMIN1 | Region 1 top edge |
| 0xB1000018 | PCC_YMAX1 | Region 1 bottom edge |
| 0xB100002C | PCC_XMIN2 | Region 2 left edge |
| 0xB1000030 | PCC_XMAX2 | Region 2 right edge |
| 0xB1000034 | PCC_YMIN2 | Region 2 top edge |
| 0xB1000038 | PCC_YMAX2 | Region 2 bottom edge |
| 0xB100003C | PCC_MEMORY | Cursor sprite pattern load |

#### 6.9.1. Cursor Generation

The cursor can take two forms: a 16-by-16 bit pattern or a crosshair whose lines may extend to the edges of the visible raster or may be clipped to a programmed region. The cursor hardware uses two display planes called the A and B planes. The A plane is connected to VDAC overlay plane 3, while the B plane is connected to VDAC overlay plane 2. VDAC overlay planes 1 and 0 are always zero in the video logic. Table 6-4 lists the effect of various cursor values. As shown by the table, the cursor in a color system may have up to three colors, while the cursor in a monochrome system may have up to three greyscale values.

**Table 6-4: Cursor Overlays**

| A Plane | B Plane | Overlay Entry | Display Result |
|---|---|---|---|
| 0 | 0 | 0 | Frame buffer contents |
| 0 | 1 | 4 | Overlay map value |
| 1 | 0 | 8 | Overlay map value |
| 1 | 1 | 12 | Overlay map value |

### 6.9.2. Cursor Coordinates

The PCC calculates pixel coordinates relative to the video sync and blank pulses. Consequently, display coordinates must be translated by (212, 34) before loading into the PCC coordinate registers. That is, the upper-left-corner pixel of the display is at PCC coordinate (212, 34), while the lower-right-corner pixel of the display is at PCC coordinate (1235, 897).

The X offset of 212 applies to the PCC_XPOS, PCC_XMIN1, PCC_XMAX1, PCC_XMIN2, and PCC_XMAX2 registers. The Y offset of 34 applies to the PCC_YPOS, PCC_YMIN1, PCC_YMAX1, PCC_YMIN2, and PCC_YMAX2 registers.

### 6.9.3. PCC Registers

The PCC contains the following programmable elements:

- Two 16-entry arrays to store a 16-by-16 bit sprite pattern for each cursor plane.
- X and Y position registers to control where the cursor pattern is displayed in the raster.
- Two region detectors, one of which defines a rectangle in the raster which can be used to clip the display of of a crosshair cursor. The other region detector may be used to generate a video interrupt in order to synchronize updates of the cursor planes or color map with vertical retrace.
- A control register which determines how the cursor is generated.

To a program the cursor chip appears as 12 write-only register, each 16 bits wide. These registers should always be written with half-word access instructions; they cannot be read. Their contents after power-up are indeterminate.

The PCC registers are buffered such that the changes do not affect the display until the next vertical retrace period. To avoid transient display images from cursor updates, the PCC should not be written during the first three scan lines of vertical blank. The video interrupt may be used to coordinate cursor updates so they do not occur during this period of time.

### 6.9.4. Cursor Command Register (PCC_CMDR)

The cursor command register is a 16-bit write-only register.

```
 +-------+-------+-------+-------+-------+-------+-------+-------+-
 | TEST  | HSHI  | VBHI  | LODSA | FORG2 | ENRG2 | FORG1 | ENRG1 |
 +-------+-------+-------+-------+-------+-------+-------+-------+-

-+-------+-------+-------+-------+-------+-------+-------+-------+
 | XHWID | XHCL1 | XHCLP | XHAIR | FOPB  | ENPB  | FOPA  | ENPA  |
-+-------+-------+-------+-------+-------+-------+-------+-------+
```

TEST Diagnostic test. This bit is used to manipulate the cursor test flipflop. When this bit is set to one, the test flipflop is forced to zero. The test flipflop state is visible in the system control and status register CRSRTST bit. Writing a 1 to TEST clears the flipflop. When any PCC video output pin asserts, the flipflop is set. By configuring the PCC to have only one active output from a bit of the cursor planes, or either of the PARD circuits, it is possible to verify the function of these outputs.

HSHI Horizontal sync polarity. This bit must be zero to indicate that the sync input to the PCC is active low.

VBHI Vertical blank polarity. This bit must be zero to indicate that the blank input to the PCC is active low.

LOADSALoad/display sprite array. When this bit is zero, the cursor sprite is displayed normally from the contents of the sprite arrays. When this bit is one, display of the sprite is inhibited and the sprite arrays can be loaded by successive writes to the PCC_MEMORY register. Upon the transition of LODSA from one to zero, the internal array address counter is reset so that the next write to PCC_MEMORY will load the top row of sprite plane A.

FORG2 Force region detector 2 output to one. When this bit is one, the output of region detector 2 is forced to one. When this bit is zero, the detector operates normally.

ENRG2 Enable region detector 2. When this bit is zero, the output of region detector 2 is inhibited; it will be zero unless the FORG2 bit is also set, which takes precedence and forces the output to one. When ENRG2 is one, the detector operates normally.

FORG1 Force region detector 1 output to one. When this bit is one, the output of region detector 1 is forced to one. When this bit is zero, the detector operates normally.

ENRG1 Enable region detector 1. When this bit is zero, the output of region detector 1 is inhibited; it will be zero unless the FORG1 bit is also set, which takes precedence and forces the output to one. When ENRG1 is one, the detector operates normally.

XHWID Crosshair cursor line width. When this bit is zero, the crosshair cursor lines are one pixel wide. When this bit is one, the lines are two pixels wide. The extra pixels are added to the right of and below the pixels which lie on the lines corresponding to the cursor X and Y positions.

XHCL1 Select crosshair clipping region. If this bit is one, region detector 1 is used to clip the crosshair cursor; if it is zero, region detector 2 is used. This bit is effective only if the crosshair cursor is selected and crosshair clipping is selected.

XHCLP Clip crosshair inside region. If this bit is one, the crosshair cursor is clipped so that it is displayed only within the region selected by the XHCL1 bit. If this bit is zero, the crosshairs extend to the edges of the displayed raster. This bit is effective only if the crosshair cursor is selected.

FOPB Force cursor plane B output to one. When this bit one, the output from cursor plane B is force to one throughout the display, regardless of the settings of bits ENPB, XHAIR, XHCLP, XHCL1, XHWID, and of the contents of the sprite plane B array. When this bit is zero, the cursor is displayed normally.

ENPB Enable cursor plane B. When this bit is zero, the output from cursor plane B is inhibited; it will be zero throughout the display. When this bit is one, the output from the cursor plane B is displayed normally.

FOPA Force cursor plane A output to one. When this bit one, the output from cursor plane B is force to one throughout the display, regardless of the settings of bits ENPA, XHAIR, XHCLP, XHCL1, XHWID, and of the contents of the sprite plane B array. When this bit is zero, the cursor is displayed normally.

ENPA Enable cursor plane A. When this bit is zero, the output from cursor plane A is inhibited; it will be zero throughout the display. When this bit is one, the output from the cursor plane A is displayed normally.

### 6.9.5. Loading the Cursor Sprite Pattern

The cursor sprite pattern is stored in two arrays, each of sixteen 16-bit words. Each word of an array is displayed as 16 pixels on a scan line with bit 0 (least significant) in the leftmost display position. All 32 words are loaded by writing to the PCC_MEMORY register. An internal address counter in the chip is incremented after each write to point to the next word in the array to be loaded.

Cursor command register bit LODSA controls access to the sprite array. When this bit is zero, the arrays are read during normal raster scanning to display the sprite pattern. When LODSA is one, normal display of the sprite is inhibited and data can be written into the arrays. The act of

changing LODSA from one to zero resets the internal array address counter. The next write to PCC_MEMORY loads the top line of the A plane array; the next 15 writes load its remaining lines. The 16th through 32nd writes load the B plane array from top to bottom. When loading is completed, cursor command register bit LODSA must be reset to zero to resume normal sprite display.

Note that software must wait for the processor write buffers to flush after each write to the PCC_MEMORY register, or undesired byte gathering will occur.

Loading the sprite arrays should be synchronized by waiting for the video interrupt so that is it done during the vertical blanking interval.

### 6.9.6. Cursor Region Detectors

There are two region detectors, 1 and 2, each of which defines a rectangular area of the raster which can be used to clip the display of a crosshair cursor. Region detector 1 may also be used to sample three comparators on the analog VDAC outputs. Region detector 2 may also be used to generate a video interrupt to synchronize with vertical retrace. Each region detector is programmed by setting four registers: PCC_XMIN, PCC_XMAX, PCC_YMIN, PCC_YMAX. The horizontal boundaries of a region are controlled by the X registers and can be specified only to a four-pixel boundary: the least significant two bits of their contents are ignored and the system behaves as if those two bits were always zero. The vertical boundaries are controlled by the Y register and can be specified to any scan line boundary. The offsets described above must be applied to the values loaded into these registers.

The contents of the MIN registers determine the leftmost pixel or topmost line in a region. The contents of the MAX registers determine the first subsequent pixel or line which is no longer in the region. That is, the MAX registers should be loaded with the sum of the MIN value and the width or height of the region. The contents of a MAX register must always be greater than those of its corresponding MIN register; otherwise the behavior of the detector is unspecified.

### 6.9.7. Displaying a Sprite Cursor

A 16-by-16 pixel sprite cursor is displayed when cursor command register bit XHAIR is cleared to zero. The displayed position of the upper left corner of the sprite is controlled by the contents of the PCC_XPOS and PCC_YPOS registers. The values loaded into these registers must include an offset as described above. The cursor may be positioned to any pixel in both axes and may be positioned so that part of it falls outside the visible raster.

### 6.9.8. Displaying a Crosshair Cursor

A crosshair cursor is displayed when cursor command register bit XHAIR is set to one. This cursor consists of a vertical line and a horizontal line which cross at the point determined by the contents of the PCC_XPOS and PCC_YPOS register. The values loaded into these registers must include an offset as describe above. The cursor may be positioned at any pixel in both axes.

Cursor command register bit XHWID controls the width of the lines: if it is zero the lines are one pixel wide; if it is one the lines are doubled in width by adding another line one pixel to the right of the vertical line and below the horizontal line.

The length of the lines is controlled by cursor command register bit XHCLP. If it is zero, the lines extend the full width and height of the raster. If XHCLP is one, the lines are clipped by the region detector selected by cursor command register bit XHCL1; one selects region 1; zero selects region 2.

### 6.9.9. Controlling Cursor Plane Outputs

For each cursor plane (A and B) there are two bits in the cursor command register which control its output: the enable bit and the force bit.

The enable bits are ENPA for plane A and ENPB for plane B. If one of these is one, normal cursor data (sprite or crosshair) is generated for the corresponding plane. If one of these is zero, the corresponding plane output is always zero. Setting both of these bits to zero suppresses the cursor display so that the screen shows only the contents of the frame buffer. These bits are buffered so that they take effect only at the start of a vertical blanking interval.

The force bits are FOPA for plane A and FOPB for plane B. If one of these is one, the output of the corresponding plane is always one throughout the entire display raster and regardless of the state of the plane's enable bit. The force bits are not buffered; they take effect immediately upon loading. These bits must be zero for normal display operation.

### 6.9.10. Blanking the Display

The screen may be blanked without disturbing the frame buffer or cursor by using the cursor plane control bits to force the output of both planes and changing VDAC overlay map entry 12 to 0x000000.

### 6.9.11. Cursor Chip Test

The cursor chip has a test flipflop which can be used to verify that the chip is functioning correctly. The state of this flipflop appears in bit CRSRTST of the system control and status register SYS_CSR.

The test flipflop is forced to zero whenever the TEST bit in the cursor command register is one. Whenever the TEST bit of the command register is zero, the test flipflop will be set to one by the logical OR of the outputs from any of cursor plane A, cursor plane B, region detector 1, and region detector 2, as qualified by the enable or force bits for each plane and detector in the command register.

### 6.9.12. Generating Video Interrupts

To generate a video interrupt, enable PARD2 and position it on the first scan line below the visible display. That is, PCC_XMIN2 at 212+0, PCC_XMAX2 at 212+1023, PCC_YMIN1 at 34+864, and PCC_YMAX1 at 34+864.

The video interrupt handler should disable PARD2 and then write a 1 to SYS_CSR<VINT> to dismiss the video interrupt.

From the time the video interrupt initially asserts until the first pixel of the next frame is displayed is 37 scan lines, or approximately 684 microseconds.

### 6.9.13. Sampling the VDAC Outputs

To sample the analog comparators on the VDAC outputs, enable PARD1 and position it at the end of a scan line. For example, PCC_XMIN1 at 212+1020, PCC_XMAX1 at 212+1024, PCC_YMIN1 at 34+0, and PCC_YMAX1 at 34+1 to sample the first scan line.

All pixels of the first scan line must have the same RGB values to allow time for the analog comparators to settle. If the cursor A and B planes are forced on, VDAC overlay 12 may be used to set the RGB values for the entire display. The comparators cannot resolve a cross-channel intensity span of less than 32. For example, ABS(RED - GREEN) >= 32 must be satisfied before examining the SYS_CSR<VRGTRG> bit.

## 6.10. Color Map and Video DAC

The graphics color map and video DAC, based upon the BT478 RAMDAC, supports three 256 by 8-bit color maps for the red, green, and blue video DACs. The RAMDAC also supports four overlay planes, of which two are used by the PCC.

The color map is used for both the color and monochrome frame buffer options. For a monochrome frame buffer, color map entries [0..127] must be set to 0x000000, and color map entries [128..255] must be set to 0xFFFFFF. Overlay planes [3..2] are driven by the PCC outputs and should be programmed to the desired cursor foreground and background color/intensity. Overlay planes [1..0] are not used and must be programmed to off values. The RAMDAC plane mask must be set to 0xFF.

The RAMDAC only support byte reads and writes. The RAMDAC registers are word-aligned in the processor address space. Reads of the RAMDAC stall the CPU for 5 cycles. Writes to the RAMDAC complete in 8 cycles. Table 6-5 lists the addresses of the RAMDAC registers.

**Table 6-5: RAMDAC Register Addresses**

| Address | Name | Register |
|---|---|---|
| 0xB2000000 | VDAC_MAPWA | Address register (color map write) |
| 0xB2000004 | VDAC_MAP | Color map |
| 0xB2000008 | VDAC_MASK | Pixel read mask |
| 0xB200000C | VDAC_MAPRA | Address register (color map read) |
| 0xB2000010 | VDAC_OVERWA | Address register (overlay map write) |
| 0xB2000014 | VDAC_OVER | Overlay map |
| 0xB200001C | VDAC_OVERRA | Address register (overlay map read) |

### 6.10.1. Updating the Color Map

To write the color map, write the VDAC_MAPWA register with the desired pixel value, and then write the red, green, and blue values in three successive writes to the VDAC_MAP register. Software must wait for the processor write buffers to flush after each VDAC_MAP register write (or undesired byte-gathering will occur).

During the write of the blue value (third data write), the VDAC concatenates the data into a 24-bit RGB value and updates the color map. The VDAC_MAPWA register inside the VDAC automatically increments to the next map location after this write completes. Software may update this next location by writing another sequence of red, green, and blue data. After writing location 0xFF, the VDAC_MAPWA register resets to location 0x00.

The color map may be read in a similar fashion, by writing the VDAC_MAPRA register and then reading a sequence of red, green, blue data values from the VDAC_MAP register. It is recommended that color map read-back only be used by diagnostic software; operating system software should maintain the master color map in memory.

### 6.10.2. Updating the Overlay Map

To write the overlay map, write the VDAC_OVERWA register with the desired pixel value, and then write the red, green, and blue values in three successive writes to the VDAC_OVER register. Software must wait for the processor write buffers to flush after each VDAC_OVER register write (or undesired byte-gathering will occur).

During the write of the blue value (third data write), the VDAC concatenates the data into a 24-bit RGB value and updates the overlay map. The VDAC_OVERWA register inside the VDAC automatically increments to the next map location after this write completes. Software may update this next location by writing another sequence of red, green, and blue data. After writing location 15, the VDAC_OVERWA register resets to location 0x00.

The overlay map may be read in a similar fashion, by writing the VDAC_OVERRA register and then reading a sequence of red, green, blue data values from the VDAC_OVER register. It is recommended that overlay map read-back only be used by diagnostic software; operating system software should maintain the master overlay map in memory.

### 6.10.3. Grey-scale Displays

The color map and overlay map may be used to set gray-scale values for a monochrome system. That is, the grey-scale intensity of the entire display may be set by loading color map entries [128..255] with less than 0xFFFFFF, and the cursor foreground and background may be set by loading the overlay map with less than 0xFFFFFF. It is not possible to control the grey-scale intensity of individual frame-buffer pixels.

## 6.11. Color Plane Mask

The color plane mask allows processor writes to the color frame buffer to affect only specific bits of a pixel. This allows modification of a given plane of the color frame buffer using only write cycles. The color plane mask applies to all four pixels in a given word of the frame buffer. For example, setting the plane mask to 01 and then writing a word of the frame buffer will update only bits [24,16,8,0] of that word.

The color plane mask only supports byte writes. The color plane mask is at location 0xB0000000. Writes to the color plane mask complete in 9 cycles.

Note that the color plane mask also affects the monochrome frame buffer. However, no useful function is accomplished. The color plane mask should be set to 0xFF for monochrome systems.

## 6.12. Monochrome Frame Buffer

The VFB01 monochrome frame buffer option contains a 1024-by-864-pixel, 1-plane, frame buffer. The frame buffer appears as a 64-K-by-32-bit region of memory in the processor address space. The frame buffer is organized as 2048 by 1024 pixels, of which the upper-left-hand 1024-by-864 pixels are displayed.

Each word of the frame buffer contains 32 consecutive pixels of a scan line. Each scan line of the

display corresponds to 64 words, of which only the first 32 words are displayed. That is, the stride between displayed columns is 2048 pixels (256 bytes). In a given word, pixels are displayed from bit 0 through bit 31. That is, word 0xAFC00000, bit 0 is the upper-left-hand pixel on the display.

The undisplayed portions of the frame buffer may be used for storage of graphics data structures such as fonts. However, unlike memory, the frame buffer is not parity protected.

### 6.13. Color Frame Buffer

The VFB02 color frame buffer option contains a 1024-by-864-pixel, 8-plane frame buffer. The frame buffer appears as a 256-K-by-32-bit region of memory in the processor address space. The frame buffer is organized as 1024 by 1024 pixels, of which the upper 1024-by-864 pixels are displayed.

Each word of the frame buffer contains 4 consecutive pixels of a scan line. Each scan line of the display corresponds to 256 words. That is, the stride between displayed columns is 1024 pixels (1024 bytes). In a given word, pixels are displayed from byte 0 through byte 3. That is, word 0xAFC00000, byte 0 is the upper-left-hand pixel on the display.

The unused portions of the frame buffer may be used for storage of graphics data structures such as fonts. However, unlike memory, the frame buffer is not parity protected.

### 6.14. Video Timing

Table 6-6 shows the DS3100 video timing.

**Table 6-6: Video Timing**

| | | | |
|---|---|---|---|
| **Horizontal Timing** | | | |
| | Pixel clock | 69.1968 MHz | 14.45 ns |
| | Active Video | 1024 pixels | 14.8 us |
| | Blanking | 256 pixels | 3.70 us |
| | Total line | 1280 pixels | 18.5 us |
| | Sync front porch | 12 pixels | 173 ns |
| | Sync width | 128 pixels | 1850 ns |
| | Sync back porch | 116 pixels | 1676 ns |
| **Vertical Timing** | | | |
| | Line rate | 54.062 KHz | 18.5 us |
| | Active video | 864 lines | 15.98 ms |
| | Blanking | 37 lines | 684 us |
| | Total lines | 901 lines | 16.67 ms |
| | Sync front porch | 12 pixels | 173 ns |
| | Sync width | 3 lines | 55.5 us |
| | Sync back porch | 34 lines - 12 pixels | 629 us |
| **Video Levels** | | | |
| | Impedence | 75 Ohm double termination | |
| | Sync level | 0.000 V | |
| | Blank/Black level | 0.302 V | |
| | White level | 1.000 V | |

### 6.15. Reset Switch

A momentary push button switch activates the system module reset logic as per a power up condition. The reset condition exists until the push button is released.

Note that all eight status LEDS are illuminated during reset.