ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *Emerson / Motorola / Force Computers SPARC CPU-8VT* at our website: *Click HERE*

A SOLECTRON SUBSIDIARY

SPARC/CPU-8VT

Technical Reference Manual

P/N 204718 Edition 2.1
March 1999

FORCE COMPUTERS Inc./GmbH
All Rights Reserved

This document shall not be duplicated, nor its contents used
for any purpose, unless express permission has been granted.

Copyright by FORCE COMPUTERS

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

A SOLECTRON SUBSIDIARY



World Wide Web: www.forcecomputers.com
24-hour access to on-line manuals, driver updates, and application notes is provided via SMART, our SolutionsPLUS customer support program that provides current technical and services information.

Headquarters

The Americas

FORCE COMPUTERS Inc.
2001 Logic Drive
San Jose, CA 95124-3468
U.S.A.

Tel.: +1 (408) 369-6000
Fax: +1 (408) 371-3382
Email support@fci.com

Europe

FORCE COMPUTERS GmbH
Prof.-Messerschmitt-Str. 1
D-85579 Neubiberg/München
Germany

Tel.: +49 (89) 608 14-0
Fax: +49 (89) 609 77 93
Email support@force.de

Asia

FORCE COMPUTERS Japan KK
Miyakeya Building 4F
1-9-12 Hamamatsucho
Minato-ku, Tokyo 105
Japan
Tel.: +81 (03) 3437 3948
Fax: +81 (03) 3437 3968
Email smiyagawa@fci.com

NOTE

The information in this document has been carefully checked and is believed to be entirely reliable. FORCE COMPUTERS makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors which may appear in this document. FORCE COMPUTERS reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance, or design.

FORCE COMPUTERS assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of FORCE COMPUTERS Inc./GmbH. FORCE COMPUTERS does not convey to the purchaser of the product described herein any license under the patent rights of FORCE COMPUTERS Inc./GmbH nor the rights of others. All product names as mentioned herein are the trademarks or registered trademarks of their respective companies.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table of Contents

Using This Manual . . . . . **ix**

1 Introduction **1**

1.1 Specification **2**

1.2 Product Nomenclature **4**

1.3 Ordering Information **4**

2 Installation **7**

2.1 Safety Note **7**

2.2 Installation Prerequisites and Requirements **8**

2.2.1 Requirements 8

2.2.2 Terminal Connection 9

2.2.3 Location Overview 9

2.3 Switch Settings **11**

2.3.1 Memory Module MEM-5 13

2.4 Power Up **14**

2.4.1 VME Slot-1 Device 14

2.4.2 VMEbus SYSRESET 14

2.4.3 Serial Ports 15

2.4.4 RESET and ABORT Key Enable 15

2.4.5 Boot Flash Memory Write Protection 15

2.4.6 User Flash Memory Write Protection 15

2.4.7 Reserved Switches 15

2.4.8 Floppy Interface or SCSI #2 Availability on P2 16

2.4.9 Network Interface Selection (NIS) for Ethernet 17

2.4.10 Parallel Port 17

2.5 SCSI Configuration **17**

2.5.1 SCSI #1 Termination 17

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.5.2 SCSI #2 Termination 19
2.6 Connectors 21
2.6.1 Twisted Pair Ethernet Connector Pinout 21
2.6.2 Serial Port A and B Connector Pinout 22
2.6.3 Keyboard/Mouse Connector Pinout 22
2.6.4 VME P2 Connector Pinout 23
2.7 IOBP-10 24
2.7.1 Jumper Setting for IOBP-10 24
2.7.2 IOBP-10 Connector Pinouts 25
2.8 IOBP-DS 27
2.8.1 Jumper Setting for IOBP-DS 27
2.8.2 IOBP-DS Connector Pinouts 28
2.9 Ethernet Address and Host ID 30
2.10 OpenBoot Firmware 31
2.10.1 Boot the System 31
2.10.2 NVRAM Boot Parameters 33
2.10.3 Diagnostics 34
2.10.4 Display System Information 36
2.10.5 Reset the System 37
2.10.6 OpenBoot Help 38

3 Hardware 41
3.1 Overview 41
3.2 Block Diagram 41
3.3 The TurboSPARC Processor 43
3.3.1 Features of TurboSPARC 43
3.3.2 Address Mapping for TurboSPARC 43
3.4 The L2 Cache 44
3.5 The Shared Memory 44
3.6 The Memory Module MEM-5 46
3.7 SBus Participants 47
3.7.1 Address Mapping for SBus Slots on the SPARC/CPU-8VT 47

204718 2 2000146 420 000
1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.8 NCR89C100 (MACIO #1 and MACIO #2) 48

3.8.1 Features of the NCR89C100 49

3.8.2 SCSI 49

3.8.3 Ethernet 50

3.8.3.1 Network Interface 1 Control And Status Register 51

3.8.3.2 Network Interface 2 Control And Status Register 51

3.9 NCR89C105 (SLAVIO) 52

3.9.1 Features of the NCR89C105 53

3.9.2 Address Map of Local I/O Devices 53

3.9.3 Serial I/O Ports 55

3.9.4 RS-232 Hardware Configuration 56

3.9.5 RS-422 Hardware Configuration 57

3.9.6 Keyboard and Mouse Port 58

3.9.7 Floppy Interface 59

3.9.8 8-Bit Local I/O Devices 59

3.9.9 Boot Flash Memory 60

3.9.10 User Flash Memory 61

3.9.11 Programming the On-board Flash Memories 62

3.9.11.1 Boot EPROM and User Flash Size Control Register 63

3.9.11.2 Flash Memory Programming Voltage Control Register 63

3.9.11.3 Flash Memory Programming Control Register #1 64

3.9.11.4 Flash Memory Programming Control Register #2 65

3.9.12 RTC/NVRAM 66

3.10 VMEbus Interface 66

3.10.1 Features of the FGA-5000 66

3.10.2 Address Mapping for the VMEbus Interface FGA-5000 67

3.10.3 Adaptation of the FGA-5000 68

3.10.4 VMEbus SYSRESET Enable/Disable 68

3.11 On-Board Configuration Registers 69

3.12 Front Panel 70

3.12.1 RESET and ABORT Keys 70

3.12.2 Front Panel Status LEDs 71

3.12.3 Seven-Segment LED Display 72

3.12.4 Rotary Switch 73

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.13 Additional Registers 73
3.13.1 FMB Channel 0 Data Discard Status Register 74
3.13.2 FMB Channel 1 Data Discard Status Register 74

4 Circuit Schematics . . . . . paginated separately

5 FORCE OpenBoot Enhancements 79
5.1 Controlling the VMEbus Master and Slave Interface 80
5.1.1 VMEbus Master Interface 82
5.1.2 VMEbus Slave Interface 84
5.2 VMEbus Interface 86
5.2.1 Generic Information 86
5.2.2 Register Addresses 87
5.2.3 Register Accesses 88
5.2.3.1 SPARC FGA-5000 Registers 88
5.2.3.2 System Configuration Registers 94
5.2.4 VMEbus Interrupt Handler 95
5.2.5 VMEbus Arbiter 97
5.2.6 VMEbus Requester 98
5.2.7 VMEbus Status Signals 99
5.2.8 VMEbus Master Interface 99
5.2.8.1 Initializing and Controlling the VMEbus Master Interface 100
5.2.8.2 Examples: Accessing Address Spaces 102
5.2.8.3 Controlling the SPARC FGA-5000 SBus Interface 105
5.2.9 VMEbus Slave Interface 106
5.2.10 VMEbus Device Node 109
5.2.11 VMEbus NVRAM Configuration Parameters 111
5.2.12 DMA Controller Support 122
5.2.13 Mailboxes and Semaphores 126
5.2.14 FORCE Message Broadcast 127
5.3 Standard Initialisation of the VMEbus Interface 129
5.3.1 SPARC FGA-5000 Registers 129
5.3.2 VMEbus Transaction Timer 130
5.3.3 SBus Rerun Limit 130

204718 2 2000146 420 000
1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.3.4 Interrupts 130
5.3.5 SBus Slot 5 Address Map 130
5.4 System Configuration 131
5.4.1 Watchdog Timer 131
5.4.2 Watchdog Timer NVRAM Configuration Parameters 132
5.4.3 Abort Switch 132
5.4.4 Abort Switch NVRAM Configuration Parameter 133
5.4.5 LEDs, Seven Segment Display and Rotary Switch 133
5.4.6 Reset 134
5.4.7 ID PROM 135
5.5 Flash Memory Support 136
5.5.1 Flash Memory Programming 136
5.5.2 Flash Memory Device 138
5.5.3 Loading and Executing Programs from USER Flash Memory 141
5.5.4 Controlling the Flash Memory Interface 141
5.6 On-board Interrupts 142
5.6.1 VMEbus Interrupts 142
5.6.2 SYSFAIL Interrupt 144
5.6.3 ACFAIL Interrupt 144
5.6.4 ABORT Interrupt 145
5.6.5 Watchdog Timer Interrupt 145
5.7 Second SCSI and Ethernet Interface 146
5.7.1 Commands 146
5.8 BusNet Support 148
5.8.1 Limitations 148
5.8.2 Loading Programs 149
5.8.3 The BusNet Device 149
5.8.3.1 Device Properties 149
5.8.3.2 Device Methods 151
5.8.3.3 NVRAM Configuration Parameters 152
5.8.4 Device Operation 156
5.8.5 How to Use BusNet 157
5.8.6 Using bn-dload to Load from the Backplane 158
5.8.7 Booting from a Solaris/SunOS BusNet Server 159

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.8.8 Booting from a VxWorks BusNet Server 160
5.8.9 Setting NVRAM Configuration Parameters 161

6 Sun OpenBoot (= OPEN BOOT PROM 2.0 MANUAL SET) . . . . . paginated separately

204718 2 2000146 420 000
1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

List of Tables and Figures

	Page	Tab./Fig.
History of manual publication	xi	Tab. a
Fonts, notations and conventions	xi	Tab. b
Specification of the SPARC/CPU-8VT	2	Tab. 1
Product nomenclature of the SPARC/CPU-8VT	4	Tab. 2
Excerpt from the data sheet's ordering information	5	Tab. 3
Location diagram of the SPARC/CPU-8VT (schematic)	10	Fig. 1
Default switch settings	11	Tab. 4
Floppy or SCSI #2 availability on P2	16	Fig. 2
SPARC/CPU-8VT connectors	21	Tab. 5
Twisted pair Ethernet connector pinout	21	Fig. 3
Serial port A and B connector pinout	22	Fig. 4
Keyboard/mouse connector pinout	22	Fig. 5
VME P2 connector pinout	23	Fig. 6
The IOBP-10 back panel	24	Fig. 7
IOBP-10 P1 pinout	25	Fig. 8
IOBP-10 P2 pinout (SCSI #1)	25	Fig. 9
IOBP-10 P3 pinout (floppy)	26	Fig. 10
IOBP-10 P5 pinout (serial A and B)	26	Fig. 11
IOBP-10 P6 pinout (Ethernet #1 – AUI)	26	Fig. 12
The IOBP-DS back panel	27	Fig. 13
IOBP-DS P2 pinout	28	Fig. 14
IOBP-DS J1 pinout (SCSI #1)	28	Fig. 15
IOBP-DS J2 pinout (SCSI #2)	29	Fig. 16
IOBP-DS J3 pinout (Ethernet #1 – AUI)	29	Fig. 17
IOBP-DS J4 pinout (serial A and B)	29	Fig. 18
IOBP-DS J5 pinout (keyboard/mouse)	29	Fig. 19
The 48-bit (6-byte) Ethernet address	30	Fig. 20
The 32-bit (4-byte) host ID	30	Fig. 21
Device alias definitions	32	Tab. 6
Setting configuration parameters	33	Tab. 7
Diagnostic routines	34	Tab. 8
Commands to display system information	37	Tab. 9
Block diagram of the SPARC/CPU-8VT	42	Fig. 22
Physical memory map of TurboSPARC	43	Tab. 10
Bank selection	45	Tab. 11
Physical memory map of main memory	45	Tab. 12
MEM-5 memory banks	46	Tab. 13
Physical memory map of SBus on SPARC/CPU-8VT	47	Tab. 14
I/O Space of CPU-8VT in SBus slot 5	48	Tab. 15

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

	Page	Tab./Fig.
NCR89C100 MACIO #1 address map	48	Tab. 16
Network interface 1 control and status register	51	Tab. 17
Layout of network interface 1 control and status register	51	Tab. 18
Bit definition for network interface 1 control and status register	51	Tab. 19
Network interface 2 control and status register	51	Tab. 20
Layout of network interface 2 control and status register	52	Tab. 21
Bit definition for network interface 2 control and status register	52	Tab. 22
NCR89C105 address map	53	Tab. 23
RS-232, RS-422 configuration	56	Tab. 24
Serial ports A and B pinout list (RS-232)	56	Tab. 25
Switch settings for ports A and B (RS-232)	57	Tab. 26
Serial ports A and B pinout list (RS-422)	57	Tab. 27
Switch settings for ports A and B (RS-422)	58	Tab. 28
8-bit local I/O devices	59	Tab. 29
Boot flash memory capacity	60	Tab. 30
User flash memory capacity	61	Tab. 31
Boot EPROM and user flash size control register	63	Tab. 32
Flash memory programming voltage control register	64	Tab. 33
Flash memory programming control register #1	64	Tab. 34
Flash memory control register pages	64	Tab. 35
Flash memory programming control register #2	65	Tab. 36
Flash memory programming: SEL_ROM and SEL_BOOT	65	Tab. 37
Memory map of the VMEbus interface	67	Tab. 38
On-board configuration registers	69	Tab. 39
LCA identification register	70	Tab. 40
SYS and user LED control registers	72	Tab. 41
Seven-segment LED display control register (SEV_SEG_CTRL)	72	Tab. 42
Naming the parts of the hexadecimal display	73	Fig. 23
Rotary switch status register (ROTARY_SWITCH_STAT)	73	Tab. 43
FMB channel 0 data discard status register	74	Tab. 44
FMB channel 1 data discard status register	74	Tab. 45
Address translation (master): microSPARC – SBus – VMEbus	82	Fig. 24
Mapping a VMEbus area to the processor's virtual address space	83	Fig. 25
Address translation (slave): VMEbus – SBus – microSPARC	84	Fig. 26
SBUS slots utilization	86	Tab. 46
Interrupt mapping	96	Tab. 47
SBus slot 5	130	Fig. 27
Watchdog timer timeout values	131	Tab. 48
Calling the OpenBoot `boot` command using `busnet-tftp`	156	Fig. 28
Transferring data using the BusNet protocol	158	Fig. 29
NVRAM configuration parameters	162	Tab. 49

2047I8.2.20000146.420.000
1.March.1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Using This Manual

This section does not provide information on the product, but on standard features of the manual itself:

- its structure,
- special layout conventions,
- and related documents.

Audience of the Manual

This *Technical Reference Manual* is intended for hard- and software developers installing and integrating the SPARC/CPU-8VT into their systems.

Overview of the Manual

This *Technical Reference Manual* provides a comprehensive hardware and software guide to your board.

IMPORTANT



Please take a moment to examine the “Table of Contents” to see how this documentation is structured. This will be of value to you when looking for information in the future.

It includes

- a brief overview of the product, the specification, the ordering information: see section 1 “Introduction” on page 1.
- the installation instructions for powering up the board: see section 2 “Installation” on page 7. It includes the default configuration (switches and the like), initialization, and connector pinouts.

  The installation instructions also appear as the product’s installation guide – a separate manual delivered together with each product shipped.
- a detailed hardware description: see section 3 “Hardware” on page 41
- the circuit schematics of the board for reference purposes.

  The circuit schematics are packaged separately to enable easy updating. They will always be shipped together with this manual. Therefore:

☞ **Insert the circuit schematics now: see section 4 “Circuit Schematics”.**

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

- a detailed description of OpenBoot which controls the CPU board operations: see section 5 “FORCE OpenBoot Enhancements” on page 79 and section 6 “Sun OpenBoot (= OPEN BOOT PROM 2.0 MANUAL SET)”.

  The Sun OpenBoot section is packaged separately to enable easy updating. It is always shipped together with this manual. Therefore:

  ☞ **Insert the Sun OpenBoot section now: see section 6 “Sun OpenBoot (= OPEN BOOT PROM 2.0 MANUAL SET)”.**

  The section 6 “Sun OpenBoot (= OPEN BOOT PROM 2.0 MANUAL SET)” includes the OPEN BOOT PROM 2.0 MANUAL SET, which is a manual in its own with the following sections:

  – Open Boot 2.0 Quick Reference

  – Open Boot 2.0 Command Reference

  – FCODE Programs

Data sheets

The *SPARC/CPU-8VT Data Sheets* (P/N –) are an integral part of the *SPARC/CPU-8VT Technical Reference Manual* (P/N 204718). They will always be shipped together with the *Technical Reference Manual*. Yet, they are packaged separately to ease handling. The following data sheets are delivered:

- NCR SBus I/O Chipset Data Manual
- SGS-THOMSON M48T18
- TurboSPARC
- AMD Flash EPROM – AM29F040
- AMD Flash EPROM – AM29F016
- Intel Flash Memory – 28F008SA-L

204718 2 20000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Publication History of the Manual

Table a **History of manual publication**

Edition	Date	Description
1.0	March 1997	First print
2.0	December 1997	Figure 22, “Block diagram of the SPARC/CPU-8VT,” on page 42 has been corrected. Table 23, “NCR89C105 address map,” on page 53 has been completed.
2.1	Mrach 1998	P2 factory option changed from 5-row to 3-row and 5-row became the standard. Table 19, “Bit definition for network interface 1 control and status register,” on page 51 and Table 22, “Bit definition for network interface 2 control and status register,” on page 52 have been corrected. Data Sheet “T7213” removed.

Fonts, Notations and Conventions

Table b **Fonts, notations and conventions**

Notation	Description
	All numbers are decimal numbers except when used with the following notations:
0000.0000_{16}	Typical notation for hexadecimal numbers (digits are 0 through F), e.g. used for addresses and offsets. Note the dot marking the 4th (to its right) and 5th (to its left) digit.
0000_{8}	Same for octal numbers (digits are 0 through 7)
0000_{2}	Same for binary numbers (digits are 0 and 1)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table b **Fonts, notations and conventions**

Notation	Description
`Program`	Typical character format used for names, values, and the like. It is used to indicate when to type literally the same word. Also used for on-screen output.
Variable	Typical character format for words that represent a part of a command, a programming statement, or the like, and that will be replaced by an applicable value when actually applied.

Icons for Ease of Use: Safety Notes and Tips & Tricks

There are 3 levels of safety notes used in this manual which are described below in brief by displaying a typical layout example.

Be sure to always read and follow the safety notes of a section first – before acting as documented in the other parts of the section.

CAUTION



Dangerous situation: injuries to people or severe damage to objects possible.

NOTICE



Possibly dangerous situation: no injuries to people but damage to objects possible.

IMPORTANT



No danger encountered. Only application hints and time-saving tips & tricks or information on typical errors when using the information mentioned below this safety hint.

204718 2 20000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

1 Introduction

The SPARC/CPU-8VT implements the capabilities of Sun Microsystems' SPARCstation 5 workstation on a single-slot VMEbus board. It is a single-board computer combining workstation performance and functionality with the ruggedness and expandability of an industry standard 6U VMEbus card. Using SBus modules, the board becomes a VMEbus two-slot solution.
Through a combination of processing power with a full set of I/O interfaces including 2 fast SCSI devices, 2 Ethernet devices, floppy disk, serial I/O and keyboard/mouse ports, the SPARC/CPU-8VT becomes a high performance cost effective solution for embedded applications.

The SPARC/CPU-8VT is a VMEbus board based on:

- the TurboSPARC CPU
- and the FGA-5000 VMEbus-to-SBus interface gate array.

TurboSPARC

The TurboSPARC CPU is a processor based on microSPARC-II technology. It is a highly integrated implementation of the SPARC RISC microprocessor and runs at up to 170 MHz. The TurboSPARC processor with 512-KByte L2 cache provides an estimated performance of 143 SPECint92 and 119 SPECfp92 at 170 MHz. The TurboSPARC interfaces directly to the L2 cache and via processor-controlled buffers to the shared DRAM. 2 memory module slots are available on each SPARC/CPU-8VT enabling memory expansion with 16- or 64-MByte DRAM modules (MEM-5).

FGA-5000

The FGA-5000 provides the SPARC/CPU-8VT with high speed VMEbus transfer capabilities for standard transfers and extended 64-bit MBLT transfers.

This complete 64-bit VMEbus interface and 2 industry standard SBus sockets enable the expansion of memory, I/O and processing performance via a broad range of off-the-shelf solutions.

Every SPARC/CPU-8VT includes an EPROM based monitor/debugger called OpenBoot™, which provides the functionality of the boot device as well as the setup for the VMEbus interface. The software support for the SPARC/CPU-8VT ranges from Solaris™, the most popular implementation of the UNIX operating system, to sophisticated hard real-time operating systems such as VxWorks.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

1.1 Specification

Table 1 Specification of the SPARC/CPU-8VT

Processor	TurboSPARC with 512-KByte L2 cache estimated 143 SPECint92 estimated 119 SPECfp92
Memory management unit	SPARC Reference MMU
Data/instruction cache	16 KByte/16 KByte
L2 cache	512 KByte 256 KByte (factory option) 1 MByte (factory option)
Shared main memory	16- or 64-MByte DRAM modules Upgradable to 128 MByte on 2 slots
SBus slots	2, mechanically compatible to CPU-2CE, CPU-3CE, CPU-5CE, CPU-5TE, CPU-5V, CPU-5VT, CPU-7V
SCSI #1 with DMA to SBus	NCR89C100 (MACIO #1) 10 MByte/sec fast SCSI-2 53C90A superset I/O on front panel and P2
Ethernet #1 with DMA to SBus	NCR89C100 (MACIO #1) 10 MBits/sec AM7990 compatible I/O on front panel as twisted pair and on P2 as AUI
Parallel port with DMA to SBus	NCR89C100 (MACIO #1) 3.4 MByte/sec Centronics compatible Uni- or bidirectional I/O on 5-row P2 connector
SCSI #2 with DMA to SBus	NCR89C100 (MACIO #2) 10 MByte/sec fast SCSI-2 53C90A superset I/O on 3-row connector and 5-row P2 connector I/O on 3-row P2 connector via switch matrix (instead of floppy interface)

20-718-2-2000146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1 Specification of the SPARC/CPU-8VT (cont.)

Ethernet #2 with DMA to SBus	NCR89C100 (MACIO #2) 10 Mbits/sec AM7990 compatible I/O on front panel as Twisted Pair and on 5-row P2 connector (factory option) as AUI
Floppy disk interface	NCR89C105 250, 300, 500 KByte/sec and 1 MByte/sec 82077AA-1 compatible I/O on P2 via switch matrix (instead of SCSI #2)
Serial I/O	NCR89C105 2 ports with RS-232 configuration (RS-422 option via hybrid modules), 8530 compatible I/O on front panel or P2
Keyboard/mouse port	Sun compatible, on front panel and P2
Counters/timers	Two 22-bit, programmable, 500 ns resolution
Boot flash memory	512 KByte (1-MByte option) On-board programmable Hardware write protection
User flash memory	2 or 4 MByte (optional) On-board programmable Hardware write protection
RTC/NVRAM/battery Usable memory	M48T18 8 KByte
VMEbus interface	FGA-5000
Additional features	Reset and abort switches Status LEDs, HEX display, rotary switch, power-on reset circuitry, voltage sensor
Firmware	OpenBoot with diagnostics
Power consumption	+5V 5.5 A (no SBus module installed) +12V 0.7A -12V 0.2A
Environmental cond. Temperature (operating) Temperature (storage) Humidity	 0^{o} C to $+55^{o}$ C -40^{o} C to $+85^{o}$ C 5% to 95% noncondensing

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1 Specification of the SPARC/CPU-8VT (cont.)

Board size	Single-slot 6U VMEbus 160.00 x 233.35 mm; 6.29 x 9.18 inches
Standards compliance	ANSI/VITA 1-1994

The SPARC/CPU-8VT is available in several memory options. Consult your local sales representative to confirm availability of specific combinations.

1.2 Product Nomenclature

Table 2 Product nomenclature of the SPARC/CPU-8VT

CPU-8VT/16-170-2	170-MHz TurboSPARC CPU board with 512-KByte L2 cache, one 16-MByte DRAM memory module, 1 free memory expansion slot, 2-MByte user flash, dual SCSI-2, dual Ethernet, floppy disk, keyboard/mouse port, 2 serial I/O ports, 64-bit VMEbus interface, 2 SBus slots, OpenBoot firmware.
CPU-8VT/64-170-2	Same as above, except 64-MByte DRAM.
MEM-5/16	16-MByte mezzanine memory module.
MEM-5/64	64-MByte mezzanine memory module.

1.3 Ordering Information

The following table is an excerpt from the SPARC/CPU-8VT data sheet. Please ask your local FORCE COMPUTERS representative for the current data sheet.

20478 2 2000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 3 Excerpt from the data sheet's ordering information

Product name	Description
SBus modules	
SBus/GX	Color 2-D and 3-D wire frame graphics accelerator, 1152x900, 8 bits per pixel, single SBus slot.
SBus/TGX	Color 2-D and 3-D wire frame high-performance graphics accelerator, up to 1152x900, 1 MByte VRAM, 8 bits per pixel, single SBus slot.
SBus/TGX+	Color 2-D and 3-D wire frame high performance graphics accelerator, up to 1600 x 1280, 4 MByte VRAM, 8 bits per pixel, double buffering, single SBus slot.
SBus/FP	6U front panel for up to 2 SBus cards.
Accessories	
CPU-8VT/TM	*SPARC/CPU-8VT Technical Reference Manual* including *OpenBoot User's Manual*, a *Set of Data Sheets*, a *Set of Circuit Schematics*, and a *FGA-5000 Technical Reference Manual*.
IOBP-DS	I/O back panel on VMEbus P2 with micro D-Sub connector for one AUI Ethernet, 8-pin mini-circular DIN connector for keyboard/mouse, flat cable connectors for dual SCSI and 2 serial I/O interfaces. For use with the SPARC/CPU-8VT.
Serial-2CE	Adapter cable for one serial port, 26-pin microHD to 25-pin D-Sub. For use with SPARC/CPU-8VT.
CPU-8VT/FP-AccKit	Cable set for front-panel connectors containing Ethernet, SCSI, Centronics and serial split cable.
CPU-8VT/P2-AccKit	Cable set for one P2 connector containing the IOBP-DS itself, Ethernet, SCSI and serial split cable.
FH-003/SET	Hybrid modules for RS-422 serial I/O configuration.
FH-005/SET	Hybrid modules for RS-485 serial I/O configuration.
FH-422T/SET	Hybrid modules with termination for RS-422 serial I/O configuration,
Software	
Solaris 1.x/CPU-8VT	Current version of Solaris 1.x. Please contact your local sales representative for current version information, VMEbus driver support, and documentation.
Solaris 2.x/CPU-8VT	Current version of Solaris 2.x. Please contact your local sales representative for current version information, VMEbus driver support, and documentation.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

20-F18-2.2000.0146 420.000.1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2 Installation

2.1 Safety Note

To ensure proper functioning of the product during its usual lifetime take the following precautions before handling the board.

CAUTION



Malfunction or damage to the board or connected components

Electrostatic discharge and incorrect board installation and uninstallation can damage circuits or shorten their lifetime.

- Before installing or uninstalling the board read this *Installation* section.
- Before installing or uninstalling a memory module, read the *MEM-5 Installation Guide* packaged together with the module.
- Before installing or uninstalling the board in a VME rack:
  - Check all installed boards for steps that you have to take before turning off the power.
  - Take those steps.
  - Finally turn off the power.
- Before touching integrated circuits ensure that you are working in an electrostatic free environment.
- Ensure that the board is connected to the VMEbus via both the P1 and the P2 connectors and that the power is available on both VMEbus connectors.
- When operating the board in areas of strong electromagnetic radiation ensure that the board
  - is bolted on the VME rack
  - and shielded by closed housing.

NOTICE



Damage to components caused by inappropriate IDE drive installation

There are IDE drives providing a means to connect the IDE drive frame electrically with DC ground, e.g., by inserting a jumper on the IDE drive.

- Before installing an IDE drive always make sure that the IDE drive's frame is not electrically connected with DC ground.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Damaging SPARC/CPU-8VT components

To ensure proper functioning of the SPARC/CPU-8VT board:

- Remove the jumper for IACKIN-IACKOUT- and BGIN-BGOUT-bypass on the backplane. This is not necessary on active backplanes.

CAUTION



Maintenance

The Lithium battery integrated in the NVRAM/RTC provides a data retention of at least 10 years when the board is not powered. After this period the Ethernet address, the host ID address, and all other NVRAM/RTC contents may be lost. Therefore:

- Contact FORCE COMPUTERS before expiry of the 10 years period, that is if the board remained unused and without power for 10 years. This period of time begins with the date of purchase.
- Never exchange the NVRAM/RTC.

2.2 Installation Prerequisites and Requirements

IMPORTANT



Before powering up

- check this section for installation prerequisites and requirements
- and check the consistency of the current switch setting (see section 2.3 “Switch Settings” on page 11).

2.2.1 Requirements

The installation requires only

- a power supply
- and a VMEbus backplane with P1 and P2 connectors.

Power supply

The power supply must meet the following specifications:

- required for the processor board: +5V
  - 5.2 A typical
- required for the RS-232 serial interface:
  - +12 V (0.1 A typical)
  - –12 V (0.1 A typical)

20 47 28 2 2000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.2.2 Terminal Connection

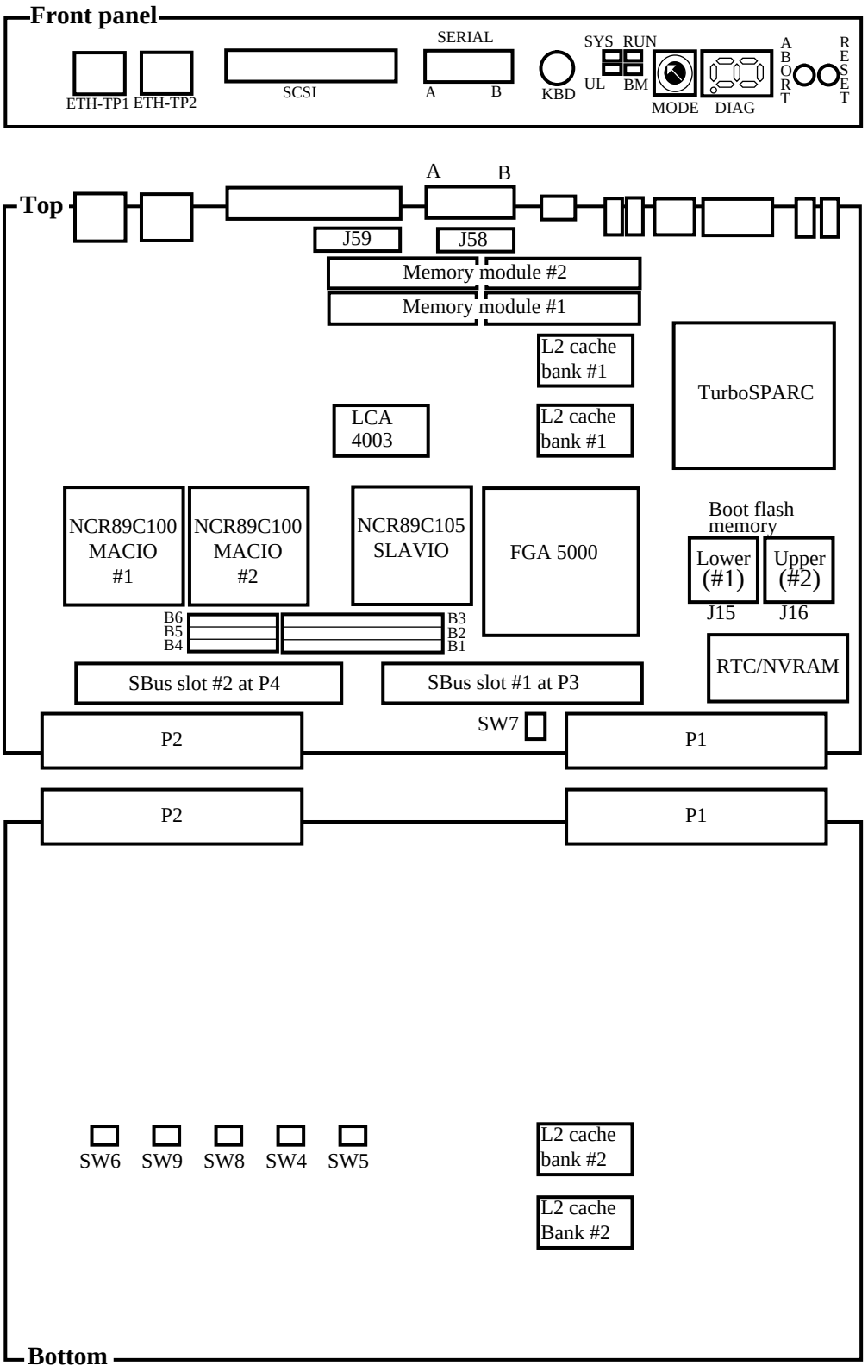
For the initial power-up, a terminal can be connected to the 26-pin MicroD-Sub connector of the serial port, which is located at the front panel (see section 2.4.3 “Serial Ports” on page 15 and section 2.6.2 “Serial Port A and B Connector Pinout” on page 22).

2.2.3 Location Overview

The figure 1 “Location diagram of the SPARC/CPU-8VT (schematic)” on page 10 highlights the position of the important SPARC/CPU-8VT components. Depending on the board type it might be that your board does not include all components named in the location diagram.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Figure 1 Location diagram of the SPARC/CPU-8VT (schematic)



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

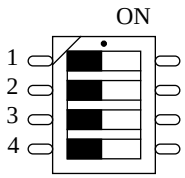
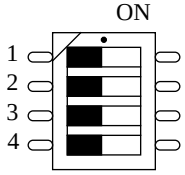
2.3 Switch Settings

The following table lists the functions and the default settings of all switches shown in figure 1 “Location diagram of the SPARC/CPU-8VT (schematic)” on page 10.

IMPORTANT

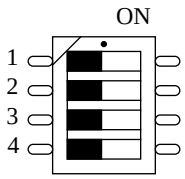
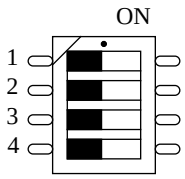
- Before powering up the board check the current switch settings for consistency.
- Do not switch during operation.

Table 4 Default switch settings

Name and default setting		Function
Serial A config. 	SW4-1 OFF	OFF = TRXC on front-panel connector for RS-232 ON = reserved for RS-485
	SW4-2 OFF	OFF = CTS (CTS+/-) on front-panel connector for RS-232 (RS-422) ON = RTXC +/- on front-panel connector for RS-422
	SW4-3 OFF	OFF = RTS (RTS+/-) on front-panel connector for RS-232 (RS-422) ON = TRXC +/- on front-panel connector for RS-422
	SW4-4 OFF	reserved: must be OFF.
Serial B config. 	SW5-1 OFF	OFF = TRXC on front-panel connector for RS-232 ON = reserved for RS-485
	SW5-2 OFF	OFF = CTS (CTS+/-) on front-panel connector for RS-232 (RS-422) ON = RTXC +/- on front-panel connector for RS-422
	SW5-3 OFF	OFF = RTS (RTS+/-) on front-panel connector for RS-232 (RS-422) ON = TRXC +/- on front-panel connector for RS-422
	SW5-4 OFF	reserved; must be OFF.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

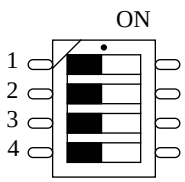
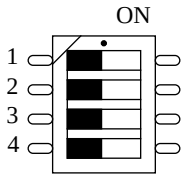
Table 4 Default switch settings (cont.)

Name and default setting		Function
	SW6-1 OFF	SCSI Termination for SCSI #1 on front panel OFF = SCSI-Term front panel automatic ON = SCSI-Term front panel disabled
	SW6-2 OFF	SCSI Termination for SCSI #1 on P2 OFF = disabled ON = enabled
	SW6-3 OFF	SCSI Termination for SCSI #2 on P2 OFF = enabled ON = disabled
	SW6-4 OFF	Reset key on front-panel control OFF = RESET key enabled ON = RESET key disabled
	SW7-1 OFF	VMEbus SYSRESET on power-up OFF = enabled ON = disabled
	SW7-2 OFF	External VMEbus SYSRESET OFF = VMEbus SYSRESET generates on-board RESET ON = VMEbus SYSRESET does not generate on-board RESET
	SW7-3 OFF	VMEbus SYSRESET generation OFF = SYSRESET is driven to VMEbus ON = SYSRESET is not driven to VMEbus
	SW7-4 OFF	Abort key control OFF = ABORT key enabled ON = ABORT key disabled

20-718-2-2000146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 4 Default switch settings (cont.)

Name and default setting		Function
	SW8-1 OFF	Automatic VMEbus slot-1 detection OFF = Automatic detection of VME Slot 1 function ON = Automatic detection of VME slot 1 function disabled. Use SW8-2 instead.
	SW8-2 OFF	Manual VMEbus slot-1 selection OFF = VME slot 1 function enabled ON = VME slot 1 function disabled SW8-2 is active only when SW8-1 = ON!
	SW8-3 OFF	Test switch, must be OFF
	SW8-4 OFF	Voltage sensor sensibility select OFF = Power sense 4.75V ON = Power sense 4.5V
	SW9-1 OFF	Boot flash EPROM write protection OFF = Write boot flash disabled ON = Write boot flash enabled
	SW9-2 OFF	User flash EPROM write protection OFF = Write user flash disabled ON = Write user flash enabled
	SW9-3 OFF	Local LCA configuration mode OFF = LCA configuration mode serial PROM ON = LCA configuration mode download
	SW9-4 OFF	Test switch, must be OFF

2.3.1 Memory Module MEM-5

Per default, memory module #1 is already installed on the delivered CPU board. The memory module #1 is required for powering up because it holds configuration information for booting the board. Memory module #2 is optional for increasing memory capacity. For instructions on installing the MEM-5, see the *How to Install MEM-5 Installation Guide.*
For the location of the memory module connectors on the board, see figure 1 “Location diagram of the SPARC/CPU-8VT (schematic)” on page 10.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.4 Power Up

The initial power up can easily be done by connecting a terminal to TTYA (serial port A). The advantage of using a terminal is that no frame buffer, monitor, or keyboard is used for initial power up, which facilitates a simple start-up.

IMPORTANT



For the initial power up, do not connect a keyboard to the board when using TTYA (serial port A). For more detailed information on booting the system, see section 2.10.1 “Boot the System” on page 31.

2.4.1 VME Slot-1 Device

Automatic VME slot-1 detection

The SPARC/CPU-8VT is configured by default for an automatic detection of VMEbus slot-1 position (SW8-1 is OFF).

IMPORTANT



Automatic VMEbus slot-1 detection will function properly only if all VMEbus boards installed in the system support this feature.

It is necessary that all boards installed in the system drive the VMEbus BG3OUT* signal at power-up to support the automatic VME slot-1 detection.
To disable this automatic VMEbus slot-1 detection feature, turn SW8-1 to ON.
If automatic detection of VMEbus slot-1 position is turned off (SW8-1 is ON), then SW8-2 is used to enable the VMEbus slot-1 controller functions of the SPARC/CPU-8VT.

IMPORTANT



Before installing the SPARC/CPU-8VT in a miniforce chassis, first disable the VMEbus system controller function by setting switch SW8-2 to ON. Ensure that SW8-1 is turned to ON to disable the automatic detection of VMEbus slot-1.

2.4.2 VMEbus SYSRESET

SYSRESET input

A SYSRESET received from VMEbus generates an on-board RESET if switch SW7-2 is OFF (default setting). When SW7-2 is ON, the SYSRESET received from the VMEbus does not generate an on-board RESET.

20-718-2.2000.146 420.000.1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SYSRESET output

A SYSRESET signal is generated to the VMEbus when an on-board local SBus reset occurs on the SPARC/CPU-8VT (e.g. the front panel reset key is toggled or power failure is detected). This SYSRESET signal can be disabled by setting switch SW7-3 to ON.
As written in the VME specification, each board must assert SYSRESET output at power-up when the power supply reaches 3 V until power is stable. This feature is enabled by default (SW7-1 is OFF). It can be disabled by setting SW7-1 to ON.

2.4.3 Serial Ports

By default, both serial ports are configured as RS-232 interfaces. It is also possible to configure these ports as RS-422 interfaces. This optional configuration is achieved with the special FORCE Hybrid FH-003 or FH-422T.

The table 4 “Default switch settings” on page 11 shows the necessary switch settings for RS-232 operation, where SW4 controls serial port A and SW5 controls serial port B. Ensure that the switches are set accordingly.

2.4.4 RESET and ABORT Key Enable

By default, the RESET and ABORT key functions on the front panel are enabled. To disable the RESET or the ABORT key functions on the front panel, set switches SW6-4 (RESET) and SW7-4 (ABORT) to ON.

2.4.5 Boot Flash Memory Write Protection

Both boot flash memories are write-protected via the switch SW9-1. When SW9-1 is OFF, the devices are write-protected (default setting).

2.4.6 User Flash Memory Write Protection

The optional user flash memories are write-protected via SW9-2. When SW9-2 is OFF, the user flash EPROMs are write-protected (default setting).

2.4.7 Reserved Switches

SW4-4, SW5-4, SW8-3, and SW9-4 are reserved for test purposes. They must be OFF.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.4.8 Floppy Interface or SCSI #2 Availability on P2

The availability of both the floppy and SCSI #2 devices at the same time depends on the availability of a 5-row P2 connector. When using a 3-row P2 connector (factory option), you have the choice of either the floppy or the SCSI #2 on P2. In the following it is described how to configure the board for floppy or SCSI #2:

Via a 3-piece configuration switch matrix, it is possible for either the floppy interface or the SCSI #2 to be available on the VME P2 connector on row C:

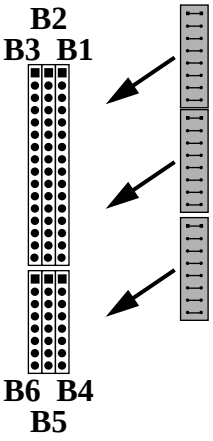
- For the floppy interface on row C plug the switch matrix into sockets B3/B2 and B6/B5. This is the default setting.
- For the SCSI #2 interface on row C plug the switch matrix into sockets B2/B1 and B5/B4.

NOTICE

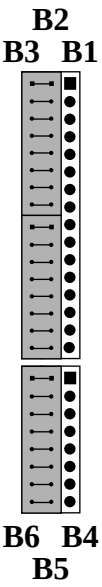


- If you use an IOBP-DS, the switch matrix must be located on B2/B1 and B5/B4 in order to route SCSI #2 to P2 row C.
- If you use an IOBP-10, the switch matrix must be located on B3/B2 and B6/B5 in order to route the floppy interface to P2 row C.

Figure 2 **Floppy or SCSI #2 availability on P2**

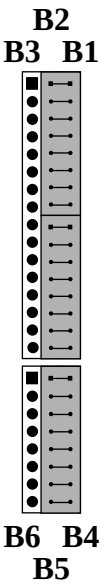


- Floppy interface on row C if the switch matrix is plugged in B3/B2 and B6/B5:



This configuration must be used when using an IOBP-10.

- SCSI #2 interface on row C if the switch matrix is plugged in B2/B1 and B5/B4:



This configuration must be used when using an IOBP-DS.

20-718-2.2000.146 420.000.1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.4.9 Network Interface Selection (NIS) for Ethernet

The Ethernet is selected either via the twisted pair connector or the AUI (Attachment Unit Interface).

- When you boot your system and a connection exists with an AUI network, then the AUI is automatically selected, i.e. when you have a successful connection with a network, the AUI is used.
- When you have no connection with the network, then the twisted pair is selected. This is valid for both Ethernet #1 and Ethernet #2. The Ethernet #1 channel and the Ethernet #2 channel function independently of each other. For both Ethernet interfaces there is one Ethernet address. This means that you are not allowed to connect both interfaces to one physical cable.

IMPORTANT



The Ethernet #2 AUI interface on P2 depends on the availability of a 5-row P2 connector. On the 3-row P2 connector (factory option), only Ethernet #1 AUI-port is available.

2.4.10 Parallel Port

The parallel port is only available on a 5-row P2 connector. When using a 3-row P2 connector (factory option), it is not available.

2.5 SCSI Configuration

In the following 2 sections the SCSI #1 termination or the SCSI #2 termination respectively are described.

2.5.1 SCSI #1 Termination

The SCSI #1 bus is accessible via the CPU board's front-panel SCSI #1 connector and via row A of the VMEbus P2 connector. Therefore, the CPU board holds 2 distinct SCSI bus terminations to enable correct termination of the SCSI #1 bus. Associated to the 2 terminations there are 2 switches – SW6-1 and SW6-2 – which allow easy selection of a valid SCSI #1 bus configuration.

There are 3 valid CPU board switch settings corresponding to valid SCSI #1 bus configurations. The respective SCSI #1 bus configuration is determined by the cable connectors. There are the following 3 possibilities:

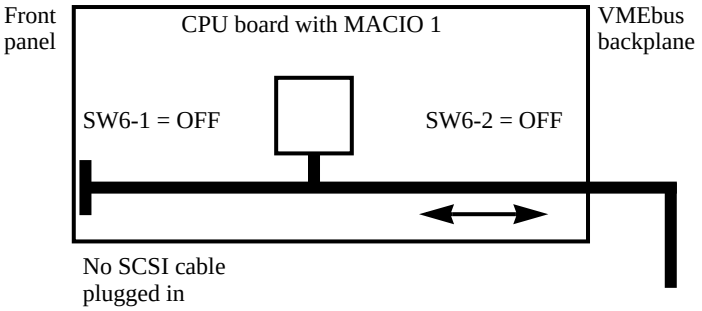
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

- cable connector to the VMEbus P2 connector,
- cable connector to the front panel,
- or a cable connector to the VMEbus P2 connector and the front panel.

Each of the following configuration descriptions starts with identifying the SCSI #1 bus configuration being covered and ends with defining the correct switch setting corresponding to the considered configuration.

Configuration 1

- The configuration 1 is covered by the default switch setting. The CPU board is located at an endpoint of the SCSI #1 bus, the SCSI #1 bus is extended via the VMEbus P2 connector, but no SCSI cable is plugged into the front-panel SCSI connector:



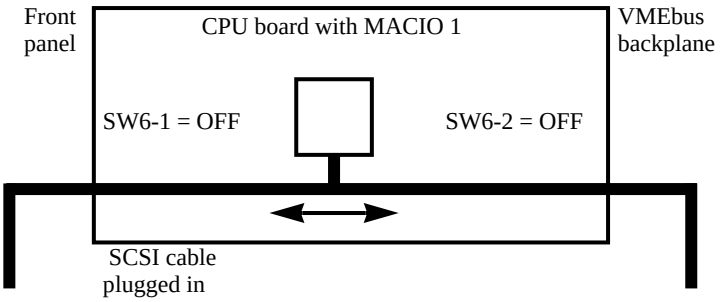
In this configuration:

– SW6-1 must be set to OFF = SCSI-Term front panel automatic (default “OFF”, see page 12)

– and SW6-2 must be set to OFF = disabled (default “OFF”, see page 12).

Configuration 2

- The configuration 2 is also covered by the default switch setting. The CPU board is not located at an endpoint of the SCSI #1 bus, the SCSI 1 bus is extended via the VMEbus P2 connector and via the front-panel SCSI connector:



20-718-2 2.0000/46 420.000/1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

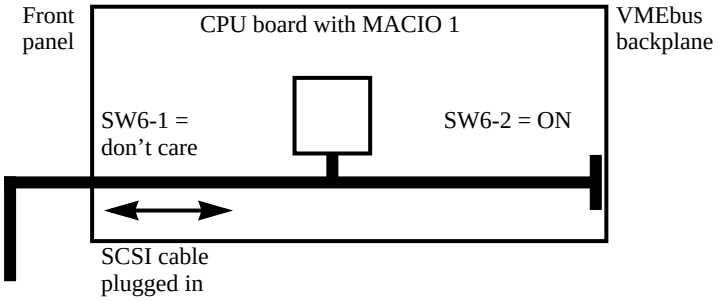
In this configuration:

– SW6-1 must be set to OFF = SCSI-Term front panel automatic (default “OFF”, see page 12)

– and SW6-2 must be set to OFF (default “OFF”, see page 12).

Configuration 3

• In configuration 3 the CPU board is located at an endpoint of the SCSI #1 bus and the VMEbus P2 connector is not used for SCSI #1 bus signalling, but the SCSI #1 bus is extended via the front-panel connector:



In this configuration

– both settings of SW6-1 are valid

– and SW6-2 must be set to ON = enabled (default “OFF”, see page 12).

Since in this configuration the SCSI #1 bus is extended via the front-panel connector setting SW6-1 to ON = SCSI-Term front panel disabled, termination reflects this configuration explicitly.

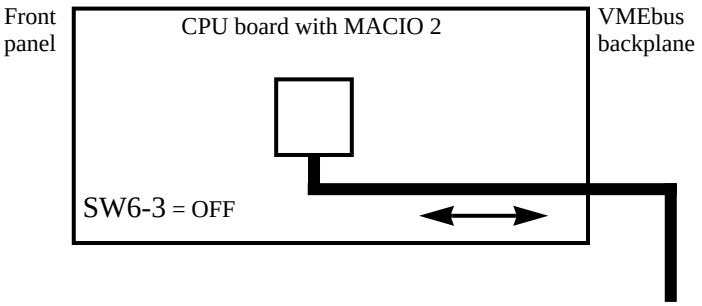
2.5.2 SCSI #2 Termination

The SCSI #2 bus is only available on row C of the VMEbus P2 connector if it is enabled via switch matrix instead of floppy interface. It is terminated – i.e. it is at one endpoint of the SCSI #2 bus – if SW6-3 is set appropriately: OFF = enabled termination (default “OFF”, see page 12).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Configuration 1

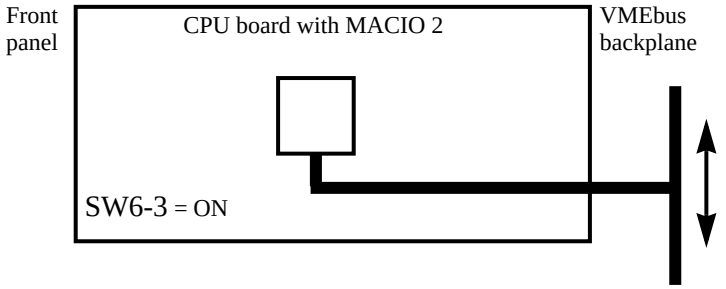
- The configuration 1 is covered by the default switch setting: The board is located at an endpoint of the SCSI #2 bus, i.e., the SCSI #2 bus is extended via the VMEbus P2 connector:



In this configuration SW6-3 must be set to OFF = enabled termination (default “OFF”, see page 12).

Configuration 2

- In configuration 2 the CPU board is not located at an endpoint of the SCSI #2 bus, i.e., the SCSI #2 bus is accessed via the VMEbus P2 connector:



In this configuration SW6-3 must be set to ON = disabled termination (default “OFF”, see page 12).

20-718-2 2000146 420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.6 Connectors

The SPARC/CPU-8VT connectors are listed in the following table.

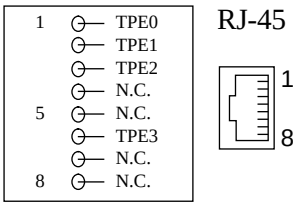
Table 5 SPARC/CPU-8VT connectors

Function	Location	Type	Manufacturer part number
Ethernet #1 (twisted pair)	Front panel	RJ-45	AMP 555131-1
Ethernet #2 (twisted pair)	Front panel	RJ-45	AMP 555131-1
Serial ports A + B	Front panel	26-pin micro D-Sub	AMP 749831-2
SCSI #1	Front panel	50-pin micro D-Sub	AMP 749831-5
Keyboard/mouse	Front panel	8-pin mini DIN	AMP 749232-1
SBus slot2 (SBus slave select 1)	P3	96-pin SMD	FUJITSU FCN-234J096-G/V
SBus slot3 (SBus slave select 2)	P4	96-pin SMD	FUJITSU FCN-234J096-G/V
VMEbus P1	P1	96-pin VGA	Various
VMEbus P2	P2	96-pin VGA	Various

2.6.1 Twisted Pair Ethernet Connector Pinout

The following figure shows the pinout of the twisted pair Ethernet connector. The pinout for both of the connectors is identical.

Figure 3 Twisted pair Ethernet connector pinout

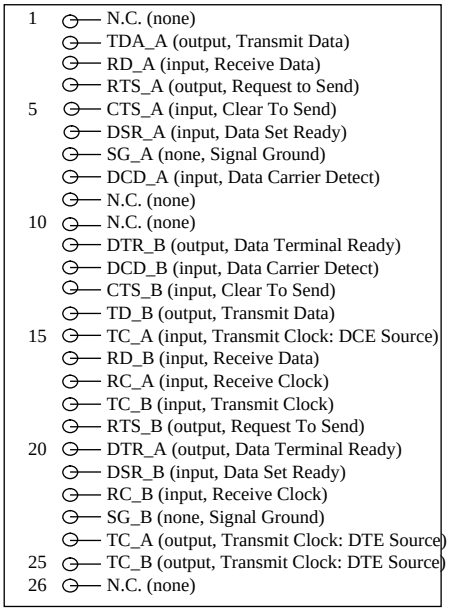


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.6.2 Serial Port A and B Connector Pinout

The following figure shows the pinout of the serial port connector.

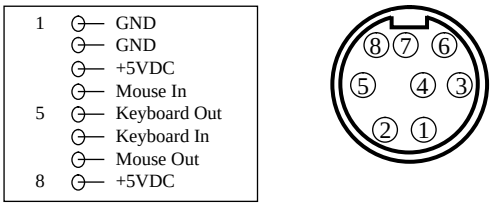
Figure 4 Serial port A and B connector pinout



2.6.3 Keyboard/Mouse Connector Pinout

The keyboard and mouse port is available on the front panel via a mini DIN connector.

Figure 5 Keyboard/mouse connector pinout



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.6.4 VME P2 Connector Pinout

The SCSI #2 interface is an alternative to the FDC interface on row C. The signals for rows Z and D are not available on the 3-row P2 connector (factory option).

Figure 6 **VME P2 connector pinout**

Z	A			C	D
CENTR DS	SCSI#1-D0		1	FDC HD IN/OUT (SCSI#2-D0*)	NC
GND	SCSI#1-D1			FDC HEAD LOAD (SCSI#2-D1*)	NC
CENTR D0	SCSI#1-D2			FDC NC (SCSI#2-D2*)	SCSI#2-D0
GND	SCSI#1-D3			FDC INDEX (SCSI#2-D3*)	SCSI#2-D1
CENTR D1	SCSI#1-D4		5	FDC DS0 (SCSI#2-D4*)	SCSI#2-D2
GND	SCSI#1-D5			NC (SCSI#2-D5*)	SCSI#2-D3
CENTR D2	SCSI#1-D6			NC (SCSI#2-D6*)	SCSI#2-D4
GND	SCSI#1-D7			FDC MOTORON (SCSI#2-D7*)	SCSI#2-D5
CENTR D3	SCSI#1-DP			FDC DIR (SCSI#2-DP*)	SCSI#2-D6
GND	GND		10	FDC STEP (SCSI#2-ATTN*)	SCSI#2-D7
CENTR D4	GND			FDC WDATA (SCSI#2-BSY*)	SCSI#2-DP
GND	GND			FDC WGATE (SCSI#2-ACK*)	TERMPWR#2
CENTR D5	TERMPWR#1			FDC TRACK00 (SCSI#2-RST*)	SCSI#2-ATTN
GND	GND			FDC WPROT (SCSI#2-MSG*)	SCSI#2-BSY
CENTR D6	GND		15	FDC RDATA (SCSI#2-SEL*)	SCSI#2-ACK
GND	SCSI#1-ATTN			FDC SIDESEL (SCSI#2-CD*)	SCSI#2-RST
CENTR D7	GND			FDC DISKCH/RDY (SCSI#2-REQ*)	SCSI#2-MSG
GND	SCSI#1-BSY			FDC EJECT (SCSI#1-IO*)	SCSI#2-SEL
CENTR ACK	SCSI#1-ACK			ETH#1_POW	SCSI#2-CD
GND	SCSI#1-RST		20	GND (TERMPWR#2*)	SCSI#2-REQ
CENTR BSY	SCSI#1-MSG			GND	SCSI#1-IO
GND	SCSI#1-SEL			ETH#1_REC+	CENTR_SLCTIN
CENTR PE	SCSI#1-CD			ETH#1_REC-	MOUSEOUT
GND	SCSI#1-REQ			ETH#1_TRA+	ETH#2_POW
CENTR AF	SCSI#1-IO		25	ETH#1_TRA-	ETH#2_REC+
GND	MOUSEIN			ETH#1_COL+	ETH#2_REC-
CENTR INIT	TXD_KBD			ETH#1_COL-	ETH#1_TRA+
GND	RXD_KBD			GND	ETH#1_TRA-
CENTR ERR	TXD_A			TXD_B	ETH#1_COL+
GND	RXD_A		30	RXD_B	ETH#1_COL-
CENTR SLCT	RTS_A			RTS_B	NC
GND	CTS_A		32	CTS_B	NC

* The SCSI #2 interface is an alternative to the FDC interface (see section 2.4.8 “Floppy Interface or SCSI #2 Availability on P2” on page 16).

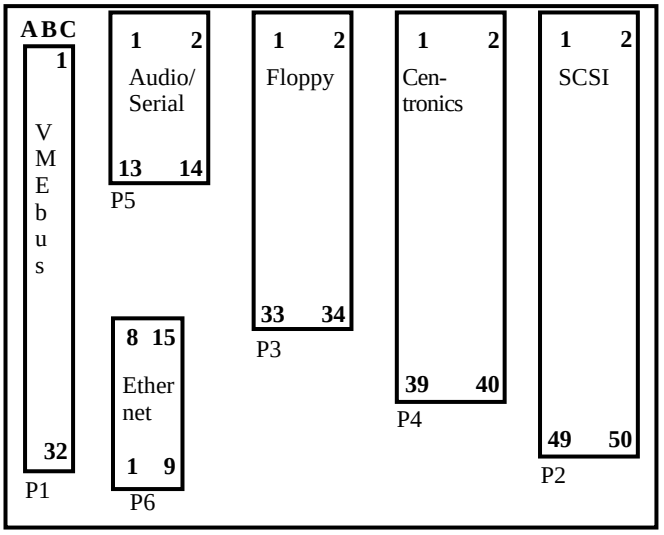
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.7 IOBP-10

IOBP-10 and IOBP-DS can be plugged to the back side of a VMEbus backplane. They only fit into 3-row backplanes. Any attempt to attach them to a 5-row backplane connector might damage the backplane.

The IOBP-10 is an I/O back panel on VMEbus P2 with flat cable connectors for SCSI, serial I/O, Centronics/floppy interface, and a micro D-Sub connector for the Ethernet #1 interface. The Centronics interface on the IOBP-10 is not supported by the SPARC/CPU-8VT. This back panel can be plugged into the VMEbus P2 connector. The diagram below shows all connectors. The IOBP-10 back panel and the IOBP-DS are especially designed for the SPARC/CPU-8VT. Do not use any other I/O back panels on the SPARC/CPU-8VT.

Figure 7 The IOBP-10 back panel



2.7.1 Jumper Setting for IOBP-10

NOTICE

- Ensure that the configuration switch matrix is plugged into sockets B3/B2 and B6/B5, that is the configuration for floppy interface on P2 (see section 2.4.8 “Floppy Interface or SCSI #2 Availability on P2” on page 16).
- The IOBP-10 back panel and the IOBP-DS are especially designed for the SPARC/CPU-8VT. Do not use any other I/O back panels on the SPARC/CPU-8VT.

20-718-2.2000.00 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.7.2 IOBP-10 Connector Pinouts

Figure 8 IOBP-10 P1 pinout

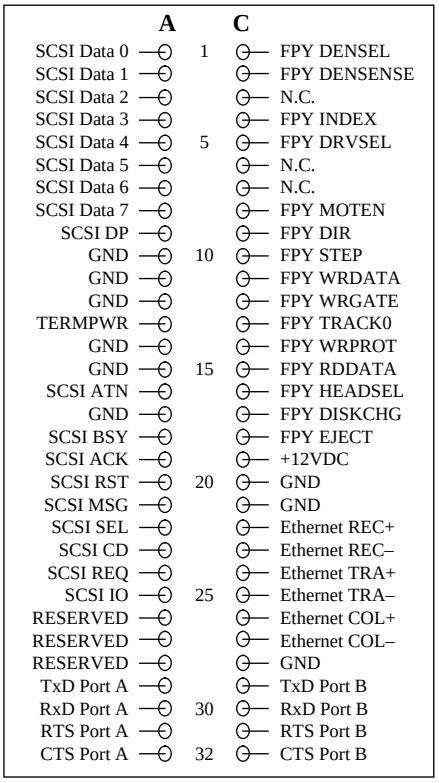
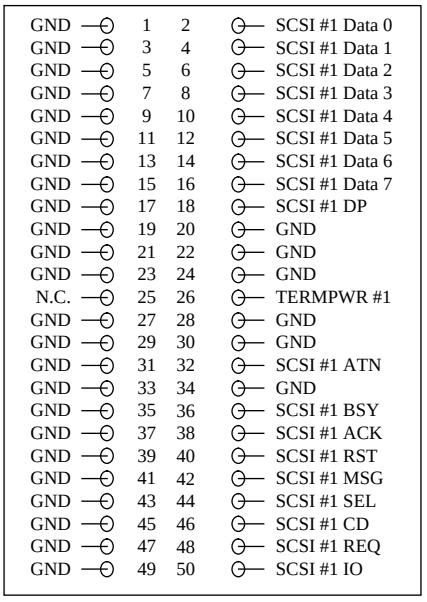


Figure 9 IOBP-10 P2 pinout (SCSI #1)



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Figure 10 **IOBP-10 P3 pinout (floppy)**

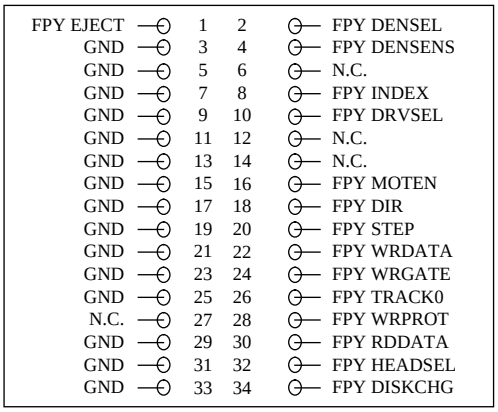


Figure 11 **IOBP-10 P5 pinout (serial A and B)**

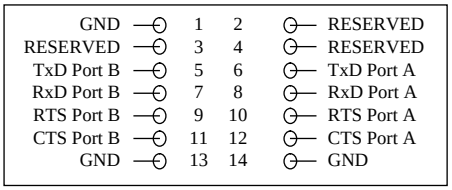
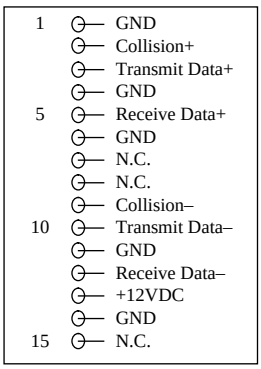


Figure 12 **IOBP-10 P6 pinout (Ethernet #1 – AUI)**



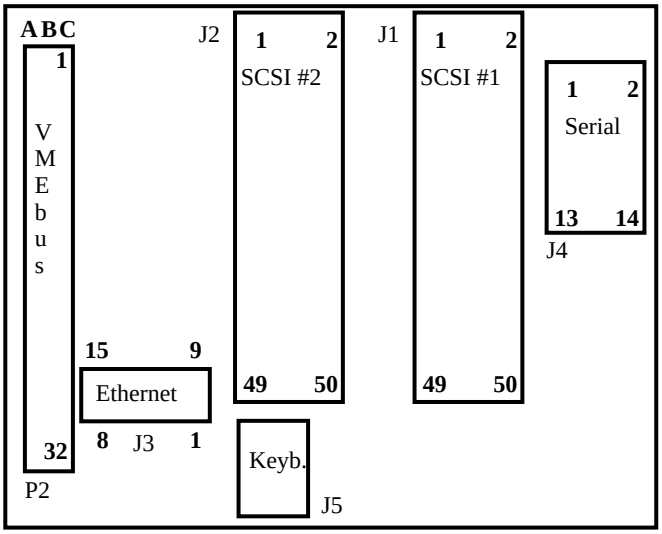
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.8 IOBP-DS

IOBP-10 and IOBP-DS can be plugged to the back side of a VMEbus backplane. They only fit into 3-row backplanes. Any attempt to attach them to a 5-row backplane connector might damage the backplane.

The IOBP-DS is an I/O back panel on VMEbus P2 with flat cable connectors for SCSI #1, SCSI #2, serial I/O, keyboard/mouse, and a micro D-Sub connector for the Ethernet #1 interface (AUI). This back panel can be plugged into the VMEbus P2 connector. The diagram below shows all connectors. The IOBP-I/O back panel and the IOBP-DS are especially designed for the SPARC/CPU-8VT. Do not use any other I/O back panels on the SPARC/CPU-8VT.

Figure 13 The IOBP-DS back panel



2.8.1 Jumper Setting for IOBP-DS

NOTICE



- Please ensure that the configuration switch matrix is plugged into sockets B2/B1 and B5/B4, that is the configuration for dual SCSI interface on P2 (5-row connector) (see section 2.4.8 “Floppy Interface or SCSI #2 Availability on P2” on page 16).
- The IOBP-DS back panel and the IOBP-10 are especially designed for the SPARC/CPU-8VT. Do not use any other I/O back panels on the SPARC/CPU-8VT.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.8.2 IOBP-DS Connector Pinouts

Figure 14 IOBP-DS P2 pinout

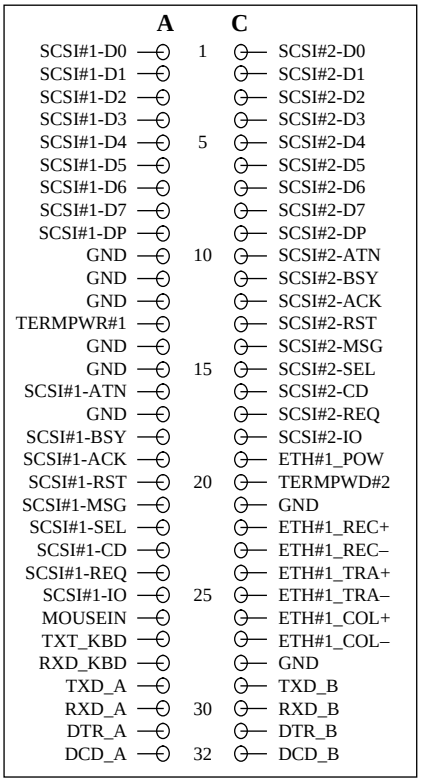
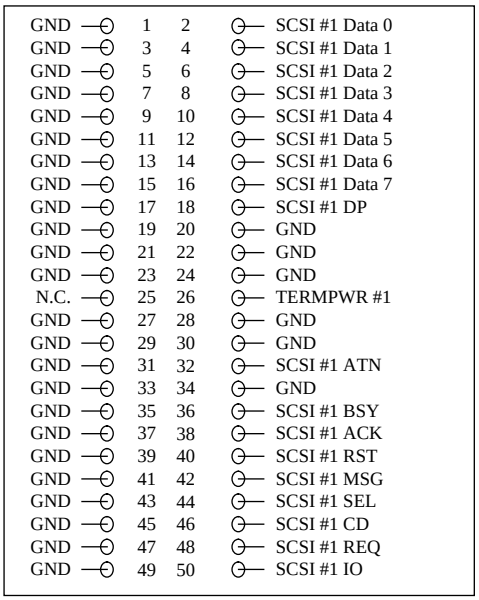


Figure 15 IOBP-DS J1 pinout (SCSI #1)



20-718-2-2000146-420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Figure 16 IOBP-DS J2 pinout (SCSI #2)

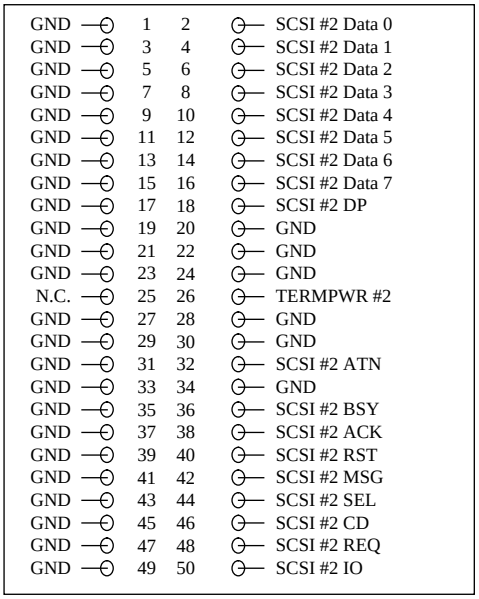


Figure 17 IOBP-DS J3 pinout (Ethernet #1 – AUI)

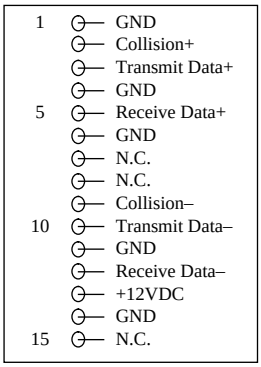


Figure 18 IOBP-DS J4 pinout (serial A and B)

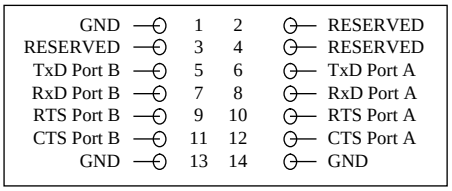
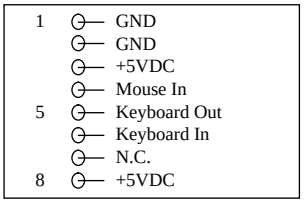


Figure 19 IOBP-DS J5 pinout (keyboard/mouse)



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.9 Ethernet Address and Host ID

In order to see the Ethernet address and host ID, type the following command at the prompt:

```
ok banner
```

The information below explains how the SPARC/CPU-8VT Ethernet address and the host ID are determined.

Figure 20 The 48-bit (6-byte) Ethernet address

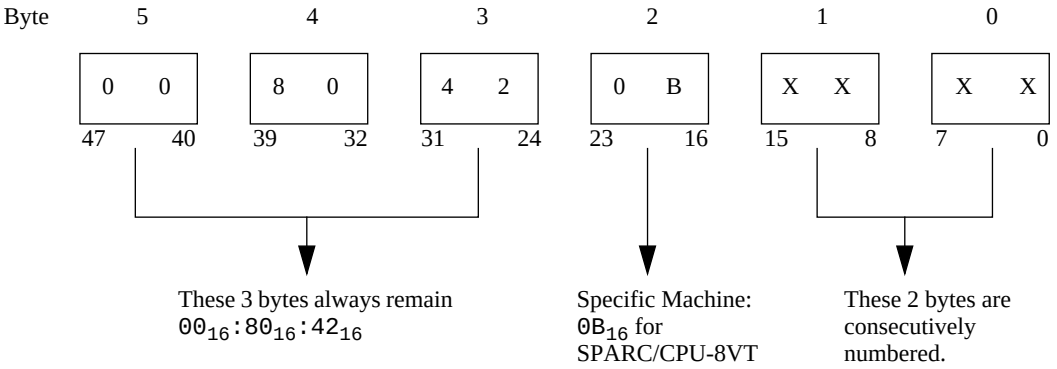
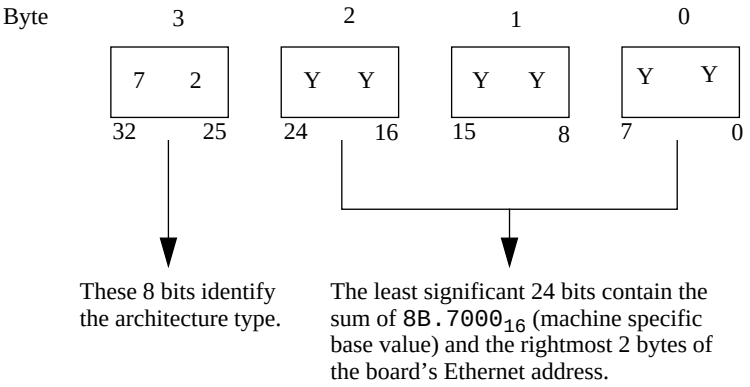


Figure 21 The 32-bit (4-byte) host ID



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.10 OpenBoot Firmware

The following tasks are described in this section:

- Boot the system
- Run diagnostics
- Display system information
- Reset the system
- OpenBoot help

For detailed information concerning OpenBoot, see the *OPEN BOOT PROM 2.0 MANUAL SET*.

2.10.1 Boot the System

The most important function of OpenBoot firmware is the booting of the system. Booting is the process of loading and executing a stand-alone program such as the operating system. After it is powered on, the system usually boots automatically after it has passed the power-on self test (POST). This occurs without user intervention.
If necessary, you can explicitly initiate the boot process from the Open-Boot command interpreter. Automatic booting uses the default boot device specified in nonvolatile RAM (NVRAM); user initiated booting uses either the default boot device or one specified by the user.

To boot the system from the default boot device, enter the following command at the Forth monitor prompt `ok`:

```
ok boot
```

If you are at the restricted monitor prompt `>`, enter:

```
> b
```

The boot command has the following format:
`boot [device-specifier] [filename] [-ah]`

Optional boot parameters

IMPORTANT



These options are specific to the operating system and may differ from system to system.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`[device-specifier]`
The name (full path or alias) of the boot device. Typical values are `cdrom`, `disk`, `floppy`, `net`, or `tape`.

[*filename*]
The name of the program to be booted. *`filename`* is relative to the root of the selected device. If no filename is specified, the boot command uses the value of *`boot-file`* NVRAM parameter. The NVRAM parameters used for booting are described in the following section.

`[-a]`
`-a` prompt interactively for the device and name of the boot file.

`[-h]`
`-h` halt after loading the program.

Devices to boot from

To explicitly boot from the internal disk using the Forth monitor enter:

```
ok boot disk
```

At the restricted monitor prompt enter:

```
> b disk
```

To retrieve a list of all device alias definitions, type `devalias` at the Forth monitor command prompt. The following table lists some typical device aliases:

Table 6 **Device alias definitions**

Alias	Description
`disk`	Default disk (1st internal) SCSI-ID 3
`disk3`	First internal disk SCSI-ID 3
`disk2`	Additional internal disk SCSI-ID 2
`disk1`	External disk SCSI-ID 1
`disk0`	External disk SCSI-ID 0
`tape`	First tape drive SCSI-ID 4
`tape0`	First tape drive SCSI-ID 4
`tape1`	Second tape drive SCSI-ID 5

20-47 18-2-2000.146 420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 6 Device alias definitions (cont.)

Alias	Description
`cdrom`	CD-ROM partition d, SCSI-ID 6
`net`	Ethernet
`floppy`	Floppy drive

2.10.2 NVRAM Boot Parameters

The OpenBoot firmware holds configuration parameters in NVRAM. At the Forth monitor prompt, type `printenv` to see a list of all available configuration parameters. The OpenBoot command `setenv` may be used to set these parameters:
`setenv` *`[configuration parameter] [value]`*

This information refers only to those configuration parameters which are involved in the boot process. The following table lists these parameters.

Table 7 Setting configuration parameters

Parameter	Default value	Description
`auto-boot?`	`true`	If `true`, automatic booting after power-on or reset
`boot-device`	`disk`	Device from which to boot
`boot-file`	`empty string`	File to boot
`diag-switch?`	`false`	If `true`, run in diagnostic mode
`diag-device`	`net`	Device from which to boot in diagnostic mode
`diag-file`	`empty string`	File to boot in diagnostic mode

When booting an operating system or another stand-alone program, and neither a boot device nor a filename is supplied, the `boot` command of the Forth monitor takes the omitted values from the NVRAM configuration parameters. If the parameter `diag-switch?` is `false`, `boot-device`, and `boot-file` are used. Otherwise, the OpenBoot firmware uses `diag-device` and `diag-file` for booting.
For detailed information on all NVRAM configuration parameters, see the *OPEN BOOT PROM 2.0 MANUAL SET.*

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.10.3 Diagnostics

At power-on or after reset the OpenBoot firmware executes POST. If the NVRAM configuration parameter `diag-switch?` is `true` for each test, a message is displayed on a terminal connected to the serial I/O port A. If the system does not work correctly, error messages are displayed indicating the problem. After POST the OpenBoot firmware boots an operating system or enters the Forth monitor, if the NVRAM configuration parameter `auto-boot?` is `false`.
The Forth monitor includes several diagnostic routines. These on-board tests let you check devices such as network controller, SCSI devices, floppy disk system, memory, clock, and installed SBus cards. User installed devices can be tested if their firmware includes a self-test routine. The table below lists several diagnostic routines.

Table 8 Diagnostic routines

Command	Description
`probe-scsi`	Identify devices connected to the on-board SCSI bus
`probe-scsi-all` *`[device-path]`*	Perform probe-scsi on all SCSI buses installed in the system below the specified device tree node. (If *device-path* is omitted, the root node is used).
`test` *`device-specifier`*	Execute the specified device's self-test method. *`device-specifier`* may be a device path name or a device alias. For example: `test` *`net`* - test network connection `test` *`/memory`* - test number of MByte specified in the self-test-#megs NVRAM parameter or test all of memory if `diag-switch?` is `true`
`test-all` *`[device-specifier]`*	Test all devices (that have a built-in self-test method) below the specified device tree node. (If *device-path* is omitted, the root node is used.)
`watch-`*`clock`*	Monitor the clock function
`watch-`*`net`*	Monitor network connection

Examples:

SCSI bus — To check the on-board SCSI bus for connected devices, enter:

```
ok probe-scsi
Target 0
Unit 0 Disk SEAGATE ST31230W 0456
ok
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

All SCSI busses

To test all the SCSI busses installed in the system, enter the following (The actual response depends on the devices on the SCSI busses):

```
ok probe-scsi-all
/iommu@0,10000000/sbus@0,10001000/esp@2,100000

Target 0
Unit 0 Disk SEAGATE ST31230W 0456

/iommu@0,10000000/sbus@0,10001000/espdma@4,8400000/esp@4,8800000

Target 5
Unit 0 Removable Read Only device TOSHIBA CD-ROM XM-4101TA1084
ok
```

Single device

To test a single installed device enter:

```
ok test device-specifier
```

This executes the `self-test` device method of the specified device node.
`device-specifier` may be a device path name or a device alias as described in Table 6, “Device alias definitions,” on page 32. The response depends on the self-test of the device node.

Group of devices

To test a group of installed devices enter:

```
ok test-all
```

All devices below the root node of the device tree are tested. The response depends on the devices having a self-test routine. If a device specifier option is supplied at the command line, all devices below the specified device tree node are tested.

Memory

When using the memory testing routine, the system tests the number of MByte of memory specified in the NVRAM configuration parameter `self-test-#megs`. If the NVRAM configuration parameter `diag-switch?` is `true`, the whole memory is tested.

```
ok test /memory
testing 32 megs of memory at addr 0 27
ok
```

The command `test-memory` is equivalent to `test /memory`. In the above-mentioned example, the first number (`0`) is the base address of the memory bank to be tested, the second number (`27`) is the number of the remaining MByte.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

- If the CPU board works correctly, the memory is erased and tested and the **ok** prompt will appear.
- If the PROM or the on-board memory does not work, one of several potential error messages indicating the problem will appear.

Clock

To test the clock function enter:

```
ok watch-clock
Watching the ‘seconds’ register of the real time clock
chip.
It should be ‘ticking’ once a second.
Type any key to stop.
22
ok
```

The system responds by incrementing a number once a second. Press any key to stop the test.

Network

To monitor the network connection enter:

```
ok watch-net
Using AUI Ethernet Interface
Lance register test -- succeeded.
Internal loopback test -- succeeded.
External loopback test -- succeeded.
Looking for Ethernet packets.
‘.’ is a good packet. ‘X’ is a bad packet.
Type any key to stop.
...........X.........................X.............
ok
```

The system monitors the network traffic displaying

- a dot (`.`) when receiving a valid packet
- and an `X` when receiving a packet with an error which can be detected by the network hardware interface.

2.10.4 Display System Information

The Forth monitor provides several commands to display system information. These commands let you display the system banner, the Ethernet address for the Ethernet controller, the contents of the ID PROM, and the version number of the OpenBoot firmware.
The ID PROM contains specific information to the individual machine, including the serial number, date of manufacture, and assigned Ethernet address.

20-718-2-2000/146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The following table lists these commands:

Table 9 Commands to display system information

Command	Description
`banner`	Displays system banner
`show-sbus`	Displays list of installed and probed SBus devices
`.enet-addr`	Displays current Ethernet address
`.idprom`	Displays ID PROM contents, formatted
`.traps`	Displays a list of SPARC trap types
`.version`	Displays version and date of the boot PROM
`show-devs`	Displays a list of all device tree nodes
`devalias`	Displays a list of all device aliases

2.10.5 Reset the System

If your system needs to be reset

- press the reset button on the front panel
- or, if you are in the Forth monitor, type **reset** on the command line.

```
ok reset
```

The system immediately begins executing the power-on self test (POST) and the initialization procedures. When the POST is completed, the system either boots automatically or enters the Forth monitor, just as it would have done after a power-on cycle.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.10.6 OpenBoot Help

The Forth monitor contains an on-line help which can be activated by entering `help`:

```
ok help
Enter ‘help command-name’ or ‘help category-name’ for more help
(Use ONLY the first word of a category description)
Examples: help select -or- help line
Main categories are:
File download and boot
Resume execution
Diag (diagnostic routines)
Power on reset
>-prompt
Floppy eject
Select I/O devices
Ethernet
System and boot configuration parameters
Line editor
Tools (memory,numbers,new commands,loops)
Assembly debugging (breakpoints,registers,disassembly,symbolic)
Sync (synchronize disk data)
Nvramrc (making new commands permanent)
ok
```

A list of all available help categories is displayed. These categories may also contain subcategories. To get help for special Forth words or subcategories just type `help [name]`.
The on-line help shows you the Forth word, the parameter stack before and after execution of the Forth word (before -- after), and a short description.
The on-line help of the Forth monitor is located in the boot PROM, that means that the on-line help does not exist for all Forth words.

Example:

How to get help for special Forth words or subcategories:

```
ok help tools
 Category: Tools (memory,numbers,new commands,loops)
 Subcategories are:
Memory access
Arithmetic
Radix (number base conversions)
Numeric output
Defining new commands
Repeated loops
ok
```

20-d738-2-2000/146-420.000/1-March.1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ok help memory
 Category: Memory access
dump ( addr length -- ) display memory at addr for length bytes
fill ( addr length byte -- ) fill memory starting at addr with byte
move ( src dest length -- ) copy length bytes from src to dest address
map? ( vaddr -- ) show memory map information for the virtual address
l? ( addr -- ) display the 32-bit number from location addr
w? ( addr -- ) display the 16-bit number from location addr
c? ( addr -- ) display the 8-bit number from location addr
l@ ( addr -- n ) place on the stack the 32-bit data at location addr
w@ ( addr -- n ) place on the stack the 16-bit data at location addr
c@ ( addr -- n ) place on the stack the 8-bit data at location addr
l! ( n addr -- ) store the 32-bit value n at location addr
w! ( n addr -- ) store the 16-bit value n at location addr
c! ( n addr -- ) store the 8-bit value n at location addr
ok
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

20-0718-2-2/2000/146 420.000/1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3 Hardware

3.1 Overview

The SPARC/CPU-8VT is powered by the TurboSPARC processor. The TurboSPARC processor with 512-KByte L2 cache delivers an estimated processing performance of 143 SPECint92 and 119 SPECfp92 at 170 MHz. The TurboSPARC interfaces directly to the 64-bit wide L2 cache and via processor-controlled buffers to the shared DRAM. Cache and DRAM also include 2 bits for parity. The SPARC/CPU-8VT is available with 16- or 64-MByte DRAM modules (both sizes are provided by the memory module MEM-5).

The TurboSPARC processor also includes a direct SBus interface for I/O-purposes. The complete suite of I/O functions includes 2 fast SCSI-2, 2 Ethernet, a floppy disk, 2 serial I/O, a Centronics parallel I/O, and keyboard/mouse ports. The SPARC/CPU-8VT utilizes the FORCE FGA-5000 VME64 device to provide a complete 64-bit VMEbus interface. This makes the SPARC/CPU-8VT the ideal solution for computing and VME transfer intensive embedded applications.

Using SBus modules the board becomes a VMEbus two-slot solution. The SCSI #1 and the Ethernet #1 are realized via one NCR89C100 (MACIO #1). The SCSI #2 and the Ethernet #2 are realized via another NCR89C100 (MACIO #2).

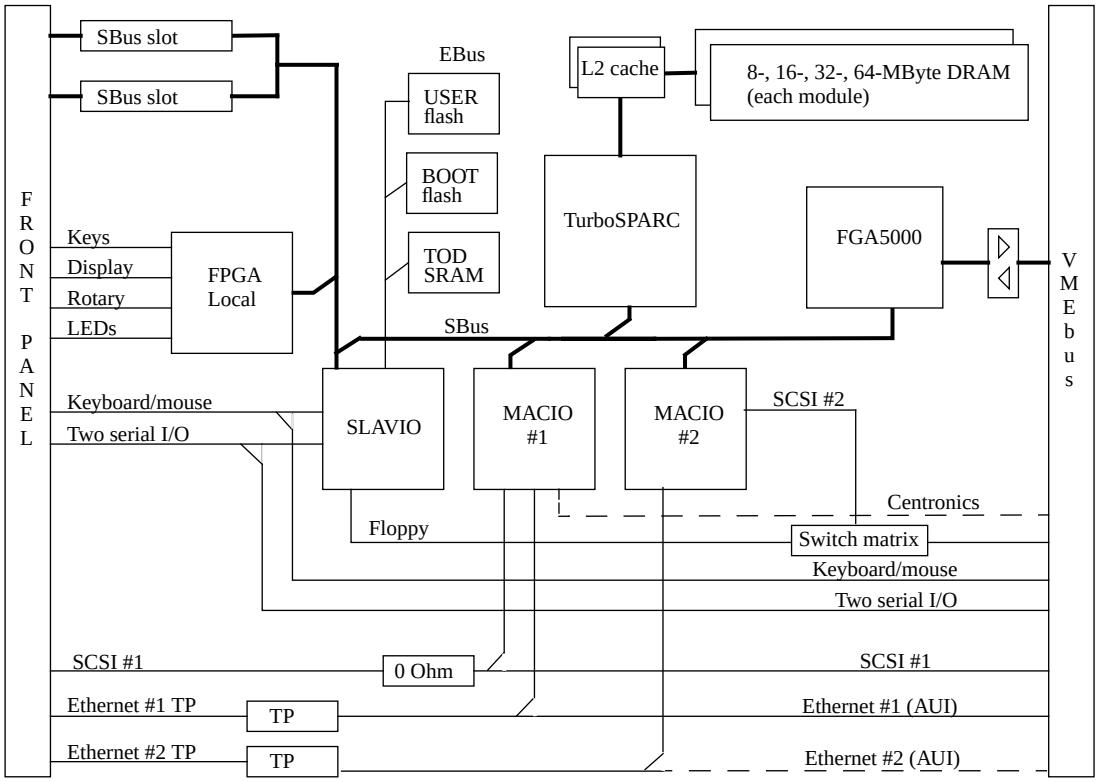
The floppy disk interface, 2 serial I/O ports, and the keyboard/mouse interface are provided by the NCR89C105 device (SLAVIO), which additionally provides the interface to the boot flash memory, the RTC/NVRAM, and the user flash memory via its 8-bit expansion port.

3.2 Block Diagram

A block diagram showing a functional overview of the SPARC/CPU-8VT is shown on the next page.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Figure 22 Block diagram of the SPARC/CPU-8VT



Ethernet #2 (AUI) and Centronics on P2 are not available on a 3-row P2 connector (factory option).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.3 The TurboSPARC Processor

The TurboSPARC CPU is the core of the SPARC/CPU-8VT. This device is realized in a 416-pin BGA package. A floating point unit, an integer unit, an MMU, an instruction cache, and a data cache are integrated in the TurboSPARC processor. For further information see the TurboSPARC User's Manual, which is available from Fujitsu.

3.3.1 Features of TurboSPARC

- TurboSPARC device running with up to 170 MHz
- Integer unit with 9-stage pipeline
- Floating point unit
- SPARC reference memory management unit
- A 16-KByte instruction cache and a 16-KByte write-back data cache, directly mapped
- Memory interface which supports up to 256-MByte DRAM
- SBus controller supports up to five SBus slots plus one "master-only" slot

3.3.2 Address Mapping for TurboSPARC

The table below lists the physical addresses of the TurboSPARC processor.

Table 10 Physical memory map of TurboSPARC

Address	Function	SBus slot #	SBus slave select #
0000.0000_{16} …$0FFF.FFFF_{16}$	User memory		
1000.0000_{16} …$1FFF.FFFF_{16}$	Control space		
2000.0000_{16} …$2FFF.FFFF_{16}$	AFX frame buffer	0	
3000.0000_{16} …$3FFF.FFFF_{16}$		1	0
4000.0000_{16} …$4FFF.FFFF_{16}$		2	1
5000.0000_{16} …$5FFF.FFFF_{16}$		3	2

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 10 Physical memory map of TurboSPARC (cont.)

Address	Function	SBus slot #	SBus slave select #
6000.0000_{16} …$6FFF.FFFF_{16}$		4	3
7000.0000_{16} …$7FFF.FFFF_{16}$		5	4

3.4 The L2 Cache

The TurboSPARC interfaces directly to a 64-bit wide level 2 cache. Furthermore, 2 bits for parity are provided by the processor. Via different configurations up to 1 MByte of cache is supported by the processor. On the SPARC/CPU-8VT the L2 cache consists of 2 banks, providing a total size of 512 KByte. Other cache sizes can be supported by using different SRAM-chips.

3.5 The Shared Memory

The shared memory is accessible through processor-controlled buffers. These buffers provide the interface between processor/level 2 cache and the memory banks.

The TurboSPARC supports up to 8 memory banks (bank 0 to 7). Each bank can have a max. of 32 MByte for a total of 256-MByte memory supported by the TurboSPARC. The signals for all memory banks are routed to the memory module connectors for module #1 and module #2.

IMPORTANT



Bank A of the memory module on connector #1 must be assembled.

The memory connector for memory module #1 supports banks 0, 1, 2, and 3. The memory connector for memory module #2 supports banks 4, 5, 6, and 7. Memory modules with up to 4 memory banks can be used (resulting in up to 128-MByte DRAM per module). As shown in the table below, the memory bank structure is organized so that memory modules with a bank count from 1 to 4 (if available) can be used in any combination. Each module has up to 4 banks, only up to 8 banks in total are allowed. A memory module can contain:

20-718-2-2000146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

- bank A,
- or banks A and B,
- or banks A, B, and C,
- or banks A, B, C, and D.

.

Table 11 Bank selection

Bank select from processor	Module on connector #1 Bank …				Module on connector #2 Bank …			
	A	B	C	D	A	B	C	D
0	x							
1		x						
2			x					
3				x				
4					x			
5						x		
6							x	
7								x

The shaded area above shows an example of how the banks are selected by the processor. In this example, the processor can select bank D of the module on connector #1 by its own bank select 3.

Table 12 Physical memory map of main memory

Address	Function	Size	Select #
0000.0000_{16} … $01FF.FFFF_{16}$	Bank #0	32 MByte	CAS0
0200.0000_{16} … $03FF.FFFF_{16}$	Bank #1	32 MByte	CAS1
0400.0000_{16} … $05FF.FFFF_{16}$	Bank #2	32 MByte	CAS2
0600.0000_{16} … $07FF.FFFF_{16}$	Bank #3	32 MByte	CAS3
0800.0000_{16} … $09FF.FFFF_{16}$	Bank #4	32 MByte	CAS4

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 12 Physical memory map of main memory (cont.)

Address	Function	Size	Select #
$0A00.0000_{16}$ … $0BFF.FFFF_{16}$	Bank #5	32 MByte	CAS5
$0C00.0000_{16}$ … $0DFF.FFFF_{16}$	Bank #6	32 MByte	CAS6
$0E00.0000_{16}$ … $0FFF.FFFF_{16}$	Bank #7	32 MByte	CAS7

3.6 The Memory Module MEM-5

Memory module MEM-5

The MEM-5 provides 16- or 64-MByte DRAM. There are 4-Mbit devices used to realize 16 MByte and 16-Mbit devices to realize 64 MByte. The following table shows the board memory capacity and the memory banks used on the TurboSPARC.

Memory module structure

Each memory module implements its total memory capacity on 2 banks of equal size:

- bank #0 and bank #1 for a module installed at connector #1
- and bank #6 and bank #7 for a module installed at connector #2.

For example, a MEM-5/16 installed at connector #2 is accessible via the physical addresses $0C00.0000_{16}$ … $0C7F.FFFF_{16}$ (8 MByte on bank #6) and $0E00.0000_{16}$ … $0E7F.FFFF_{16}$ (8 MByte on bank #7).

Table 13 MEM-5 memory banks

MEM-5 memory capacity	Memory banks A and B are used
16 MByte	X
64 MByte	X

The installation of a memory module on the SPARC/CPU-8VT is described in the *Installation Guide How to Install MEM-5*, which is available from FORCE COMPUTERS.

20 4718 2 2000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.7 SBus Participants

There are 2 SBus slots located on the component side of the board. SBus slot #1 is located at connector P3 and SBus slot #2 is located at connector P4 (see figure 1 “Location diagram of the SPARC/CPU-8VT (schematic)” on page 10).

The TurboSPARC supports up to 5 SBus slots plus an additional "master-only" slot. The SBus controller is included in the TurboSPARC.

3.7.1 Address Mapping for SBus Slots on the SPARC/CPU-8VT

The following table shows the TurboSPARC physical address map including all of its SBus slots and their functions on the SPARC/CPU-8VT.

Table 14 Physical memory map of SBus on SPARC/CPU-8VT

Address	Function	SBus slot #	SBus slave select #
2000.0000_{16} …$2FFF.FFFF_{16}$	AFX frame buffer	0	
3000.0000_{16} …$3FFF.FFFF_{16}$	VMEbus interface	1	0
4000.0000_{16} …$4FFF.FFFF_{16}$	SBus module P3 (or VMEbus interface)	2	1
5000.0000_{16} …$5FFF.FFFF_{16}$	SBus module P4 (or VMEbus interface)	3	2
6000.0000_{16} …$6FFF.FFFF_{16}$	VMEbus interface	4	3
7000.0000_{16} …$77FF.FFFF_{16}$	NCR89C105 (SLAVIO)	5	4
7800.0000_{16} …$7DFF.FFFF_{16}$	NCR89C100 (MACIO #1 and MACIO #2)	5	4
$7E00.0000_{16}$ …$7FFF.FFFF_{16}$	VMEbus interface	5	4

If no SBus cards are installed, the address space 4000.0000_{16}...$5FFF.FFFF_{16}$ is available for master accesses.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.8 NCR89C100 (MACIO #1 and MACIO #2)

There are 2 MACIO NCR89C100 devices on the SPARC/CPU-8VT.

- MACIO #1 is located in SBus slot 5 (SBus slave select 4) at physical address 7800.0000_{16}. This device drives the SCSI #1 and the Ethernet #1 interfaces.
- MACIO #2 is located in SBus slot 5 (SBus slave select 4). MACIO #2 drives the SCSI #2 and the Ethernet #2 interfaces.

Table 15 I/O Space of CPU-8VT in SBus slot 5

Physical address	Select #	Function
7000.0000_{16} …$77FF.FFFF_{16}$	SBus slave select 4	NCR89C105 SLAVIO, see table 29 “8-bit local I/O devices” on page 59 and table 23 “NCR89C105 address map” on page 53
7800.0000_{16} …$7DFF.FFFF_{16}$	SBus slave select 4	NCR89C100 MACIO #1 (see table 16 “NCR89C100 MACIO #1 address map” on page 48) For MACIO #2 registers add offset 40_{16} to MACIO #1 registers.
$7E00.0000_{16}$ …$7FFF.FFFF_{16}$	SBus slave select 4 (FGA-5000 slave select 5)	FGA-5000 registers and VMEbus interface (A24/A16)

Table 16 NCR89C100 MACIO #1 address map

Physical address	Device	Type	Access
7800.0000_{16}	DMA2 internal ID register	R	W
7840.0000_{16} …$7840.000F_{16}$	DMA2 ESP registers	R/W	W
7840.0010_{16} …$7840.001F_{16}$	DMA2 Ethernet registers	R/W	W
$7C80.0000_{16}$ …$7C80.001F_{16}$	DMA2 parallel port registers	R/W	W
7880.0000_{16} …$7880.003F_{16}$	SCSI controller registers	R/W	B
$78C0.0000_{16}$ …$78C0.000F_{16}$	Ethernet controller registers	R/W	B

20-718-2.2/000.146 420.000/1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The MACIO #2 registers are accessible with an offset of 40_{16} to the MACIO #1 registers. For example, the DMA2 Ethernet registers for MACIO #1 are located at physical address 7840.0010_{16} … $7840.001F_{16}$. DMA2 Ethernet registers for MACIO #2 are located at physical address 7840.0050_{16} … $7840.006F_{16}$.

Overview

The NCR89C100 SBus master integrates high-performance I/O macrocells and logic including a fast 53C9X SCSI core, an Ethernet controller core, a DMA2 controller, a high-speed parallel port, and an SBus interface.
The DMA2 block comprises the logic used to interface each of these functions to the SBus. It provides buffering for each of the functions. Buffering takes the form of a 64-byte data cache and 16-bit wide buffer for the Ethernet channel, and a 64-byte FIFO for the SCSI channel. The DMA2 incorporates an improved cache and FIFO draining algorithm which allows better SBus utilization than previous DMA implementations.
The availability of the parallel port depends on the availability of a 5-row P2 connector. When using a 3-row P2 connector (factory option), the parallel port is not available.

3.8.1 Features of the NCR89C100

- Fast 8-bit SCSI: supports fast SCSI mode, backward compatible to 53C90A
- 7990-compatible Ethernet
- LS64854-compatible DMA2 controller
- Glueless SBus interface clocked with 21.25 MHz at 170-MHz processor frequency
- Concurrently supports 10 MByte/s SCSI transfers and 1.25 MByte/s Ethernet transfers
- 64-byte FIFO for SCSI
- Supports SBus burst modes: 4-word, 8-word, and no burst

For further information, see *NCR SBus I/O Chipset Data Manual.*

3.8.2 SCSI

The SCSI interface provides a standard interface to a wide variety of mass storage devices, such as hard disks, tapes, and CD-ROMs. The SCSI transfers up to 10 MByte per second.

The SPARC/CPU-8VT board has 2 independent SCSI interfaces (SCSI #1 and SCSI #2). They are realized via two MACIO NCR89C100 devices (MACIO #1 and MACIO #2). The NCR89C100 has 48-mA drivers

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

and therefore provides direct drive of a single-ended SCSI bus. The SCSI core is a superset of the industry standard NCR53C90A which has been modified to support fast SCSI.

The SCSI interface is single-ended and supports “TERMPWR”. The NCR89C100 DMA2 core is able to transfer the data to and from the shared main memory.

All signals of the SCSI #1 interface are routed to the VME P2 connector and the front panel. The connection of SCSI #1 on P2 is compatible to the CPU-3CE, CPU-5V, CPU-5VT, and CPU-7V.

For information on the availability of the SCSI #2 interface on the VMEbus P2 connector, see section 2.4.8 “Floppy Interface or SCSI #2 Availability on P2” on page 16.
The SCSI signals on the VME P2 connector are shown in figure 6 “VME P2 connector pinout” on page 23.
For information on configuring the SCSI termination, see section 2.5 “SCSI Configuration” on page 17.

3.8.3 Ethernet

Ethernet #1 is realized via MACIO #1, Ethernet #2 via MACIO #2. For both Ethernet interfaces there is one Ethernet address, i.e. that you do not have to connect both interfaces to one physical cable.
Both Ethernet interfaces are available as TP-Ethernet on the front panel. Additionally, Ethernet #1 is routed as AUI port to the VME P2 connector.

The NCR89C100 DMA controller enables the Ethernet interface to transfer data to and from the shared main memory. The Ethernet core is register level compatible with the AMD Am7990, Revision F, standard Ethernet controller, which is capable of transferring Ethernet data up to 10 Mbit/s.

Network interface selection (NIS) for Ethernet

The Ethernet is selected either via

- the twisted pair connector
- or the AUI (Attachment Unit Interface).

When you boot your system and a connection exists with an AUI network, then the AUI is automatically selected. I.e. when you have a successful connection to a network, the AUI is used.
When you have no connection with the AUI network, the twisted pair is selected. This is valid for both Ethernet #1 and Ethernet #2. The Ethernet #1 and the Ethernet #2 channels function independently of each other.

20-F78 2.2000.146 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.8.3.1 Network Interface 1 Control And Status Register

The network interface 1 control and status register is used for the twisted pair network of Ethernet #1.

Table 17 Network interface 1 control and status register

Physical address	Register name	Read/write	Access
7138.0004_{16}	Network interface 1	R/W	8 bit

Table 18 Layout of network interface 1 control and status register

7138.0004_{16}								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	1	1	TP1 STAT	TP1 TENA

Table 19 Bit definition for network interface 1 control and status register

Setting	Read/ write	Function
TP1_TENA = 0	R/W	Link test disabled for Ethernet #1 TP
TP1_TENA = 1	R/W	Link test enabled for Ethernet #1 TP
TP1_STAT = 0	R	AUI #1 selected or linktest #1 disabled or link for TP #1 succeeded
TP1_STAT = 1	R	TP #1 selected, linktest #1 enabled, and link for TP #1 failed

3.8.3.2 Network Interface 2 Control And Status Register

The network interface 2 control and status register is used for twisted pair network of Ethernet #2. The availability of AUI #2 depends on the availability of a 5-row P2 connector. When using a 3-row P2 connector (factory option), AUI #2 is not available!

Table 20 Network interface 2 control and status register

Physical address	Register name	Read/write	Access
7138.0005_{16}	Network interface 2	R/W	8 bit

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 21 Layout of network interface 2 control and status register

7138.0005_{16}								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	1	1	TP2 STAT	TP2 TENA

Table 22 Bit definition for network interface 2 control and status register

Setting	Read/ write	Function
TP2_TENA = 0	R/W	Link test disabled for Ethernet #2 TP
TP2_TENA = 1	R/W	Link test enabled for Ethernet #2 TP
TP2_STAT = 0	R	AUI #2 selected or linktest #2 disabled or link for TP #2 succeeded
TP2_STAT = 1	R	TP #2 selected, linktest enabled, and link for TP #2 failed

3.9 NCR89C105 (SLAVIO)

The NCR89C105 SBus slave integrates most of the 8-bit system I/O functions including 2 dual-channel 8530-compatible serial controllers, a high speed 8277AA-1-compatible floppy disk controller, counter/timers, interrupt controllers, and system reset logic. It also provides an SBus interface for several other byte-wide peripherals through an external expansion bus.

The primary serial controller is 8530-compatible and can be used as 2 general purpose serial ports.

The second serial controller is subset of the 8530 standard and is dedicated for the keyboard/mouse connection.

20-718-2-2000146-420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.9.1 Features of the NCR89C105

- Dual-channel serial ports (8530-compatible)
- Keyboard/mouse port
- 82077AA-1 floppy disk controller, supporting up to 1 Mbit/s data transfer rate
- 8-bit expansion bus with control to support RTC/NVRAM, EPROM, and generic 8-bit devices externally
- Glueless SBus interface clocked with 21.25 MHz at 170-MHz processor frequency
- Interrupt controller
- System reset control
- Programmable 22-bit counters and timers
- Auxiliary I/O registers

For further information on the NCR89105, refer to the *NCR SBus I/O Chipset Data Manual*.

3.9.2 Address Map of Local I/O Devices

The following table lists the physical addresses for all local I/O devices and the accesses permitted [(B)yte, (H)alf Word, (W)ord, and (D)ouble Word].

Table 23 NCR89C105 address map

Physical Addr.	Device	Access
7000.0000_{16} -> $70FF.FFFF_{16}$	Boot flash memory and user flash memory	B, H,W
7100.0000_{16} -> $711F.FFFF_{16}$	Keyboard, mouse, and serial ports	B
7100.0000_{16}	Mouse control port	
7100.0002_{16}	Mouse data port	
7100.0004_{16}	Keyboard control port	
7100.0006_{16}	Keyboard data port	
7110.0000_{16}	TTYB control port	
7110.0002_{16}	TTYB data port	
7110.0004_{16}	TTYA control port	
7110.0006_{16}	TTYA data port	

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 23 NCR89C105 address map (cont.)

Physical Addr.	Device		Access
7120.0000_{16} -> $712F.FFFF_{16}$	RTC/NVRAM		B, H,W
7130.0000_{16} -> $7137.FFFF_{16}$	Boot flash and user flash memory programming window (512KB)	Generic ports (1 MB)	B
7138.0000_{16} -> $713F.FFFF_{16}$	On-board configuration registers of the SPARC/CPU-8VT		B
7140.0000_{16} -> $714F.FFFF_{16}$	Floppy controller		B
7140.0002_{16}	Digital output register (DOR)		
7140.0004_{16}	Main status register (MSR, read only)		
7140.0004_{16}	Datarate select register (DSR, write only)		
7140.0005_{16}	FIFO		
7140.0006_{16}	Reserved (test mode select)		
7140.0007_{16}	Digital input register (DIR, read only)		
7140.0007_{16}	Configuration control register (CCR, write only)		
7150.0000_{16} -> $717F.FFFF_{16}$	reserved		
7180.0000_{16}	89C105 configuration register		B
7190.0000_{16} -> $719F.FFFF_{16}$	Auxiliary I/O registers		B
7190.0000_{16}	Aux 1 register (miscellaneous system functions)		
7191.0000_{16}	Aux 2 register (software power-down control)		
$71A0.0000_{16}$	Diagnostic message register		B
$71B0.0000_{16}$	Modem register		B
$71C0.0000_{16}$ -> $71CF.FFFF_{16}$	reserved		

20-718.2.2000146-420.000.1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 23 NCR89C105 address map (cont.)

Physical Addr.	Device	Access
$71D0.0000_{16}$ -> $71DF.FFFF_{16}$	Counter/timers	W, D
$71D0.0000_{16}$	Processor counter limit register or user timer MSW	
$71D0.0004_{16}$	Processor counter register or user timer LSW	
$71D0.0008_{16}$	Processor counter limit register (non-resetting port)	
$71D0.000C_{16}$	Processor counter user timer start/stop register	
$71D1.0000_{16}$	System limit register (level 10 interrupt)	
$71D1.0004_{16}$	System counter register	
$71D1.0008_{16}$	System limit register (non-resetting port)	
$71D1.000C_{16}$	reserved	
$71D1.0010_{16}$	Timer configuration register	
$71E0.0000_{16}$ -> $71EF.FFFF_{16}$	Interrupt controller	W
$71E0.0000_{16}$	Processor interrupt pending register	
$71E0.0004_{16}$	Processor clear-pending pseudo-register	
$71E0.0008_{16}$	Processor set-soft-interrupt pseudo-register	
$71E1.0000_{16}$	System interrupt pending register	
$71E1.0004_{16}$	Interrupt target mask register	
$71E1.0008_{16}$	Interrupt target mask clear pseudo-register	
$71E0.000C_{16}$	Interrupt target mask set pseudo-register	
$71E0.0010_{16}$	Interrupt target register (reads as 0, write has no effect)	
$71F0.0000_{16}$	System control/status register	W

3.9.3 Serial I/O Ports

The 8530 SCC block implementing the 2 serial I/O ports is functionally compatible with the standard NMOS 8530 and therefore provides 2 fully independent full-duplex ports. The physical address map for the serial ports is shown in table 23 “NCR89C105 address map” on page 53.
The 2 serial I/O ports are available

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

- via the VMEbus P2 connector, each with 4 signals (RXD, TXD, RTS, CTS)
- and the front-panel 26-pin shielded connector.

RS-232 or RS-422 configuration

To simplify the configuration of the serial interfaces, FORCE COMPUTERS has developed RS-232 and RS-422 hybrid modules: FH-002, FH-003, and FH-422T. These 21-pin SIL modules are installed in sockets so that they may easily be changed to meet specific application needs. Each hybrid provides the configuration for one serial interface. Thereby, the SPARC/CPU-8VT allows RS-232 or RS-422 support via choosing the respective hybrid assembly option. By default, the FH-002 hybrid module is installed for RS-232 operation.

To change the configuration of

- serial port A: Insert the respective hybrid in socket J59 (see figure 1 “Location diagram of the SPARC/CPU-8VT (schematic)” on page 10).
- serial port B: Insert the respective hybrid in socket J58.

Table 24 RS-232, RS-422 configuration

Configuration	Installed hybrid	Socket for serial port A	Socket for serial port B	Default
RS-232	FH-002	J59	J58	*
RS-422	FH-003 or FH-422T	J59	J58	

3.9.4 RS-232 Hardware Configuration

The serial ports A and B are configured by default for RS-232 operation. The following individual I/O signals are available for the serial ports A and B on the front-panel connector.

Table 25 Serial ports A and B pinout list (RS-232)

Pin		Transmitted signals	Pin		Received signals
2	14	TXD-Transmit Data	3	16	RXD-Receive Data
4	19	RTS-Request to Send	5	13	CTS-Clear to Send
7	23	Ground	6	21	SYNC
20	11	DTR- Data Terminal Ready	8	12	DCD- Data Carrier Detect

20 47 28 2 2000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 25 Serial ports A and B pinout list (RS-232)

24	25	TRXC- DTE Transmit Clock	15	18	TRXD- DCE Transmit Clock
			17	22	RTXC- DCE Receive Clock

The pinout for serial port A is shown in the white area and the pinout for serial port B in the grey area.

The table below shows the switch settings for each port.

Table 26 Switch settings for ports A and B (RS-232)

Port A	Port B	Default	Functions of RS-232
SW4-1	SW5-1	OFF	TRXC is available on front-panel connectors, pin 24
SW4-2	SW5-2	OFF	CTS is available on front-panel connectors, pin 5
SW4-3	SW5-3	OFF	RTS is available on front-panel connectors, pin 4

3.9.5 RS-422 Hardware Configuration

The serial ports A and B can be reconfigured for RS-422 operation. In order to configure the serial ports to RS-422, the hybrid module FH-003 or FH-422T must be used. The difference between these hybrids is, that the FH422T already includes termination resistors for the input- and output-signals.

In the following table the pinout for serial port A is shown in the white area and the pinout for serial port B is shown in the grey area.

Table 27 Serial ports A and B pinout list (RS-422)

Pin		Transmitted signals	Pin		Received signals
24	25	TXD+ Transmit Data	20	11	RXD+ Receive Data
8	12	TXD- Transmit Data	7	23	RXD- Receive Data
4*	19*	RTS+ Request to Send	2*	14*	CTS+ Clear to Send
3*	16*	RTS- Request to Send	5*	13*	CTS- Clear to Send

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 27 Serial ports A and B pinout list (RS-422)

4*	19*	TRXC+ Transmit Clock	2*	14*	RTXC+ Receive Clock
3*	16*	TRXC- Transmit Clock	5*	13*	RTXC- Receive Clock

* Signals RTS and TRXC can be switched so that they are available on the connector pins 3 and 4 (16, 19). Signals CTS and RTXC can also be switched so that they are available on the connector pins 2 and 5 (14, 93). This is done by SW4 for port A and by SW5 for port B.

The following table shows the corresponding switch settings.

Table 28 Switch settings for ports A and B (RS-422)

Port A	Port B	Default	Functions of RS-422
SW4-1	SW5-1	OFF	OFF for RS-422
SW4-2	SW5-2	OFF	CTS+/- on front-panel connectors, pins 2/14 and 5/13 available
SW4-2	SW5-2	ON	RTXC+/- on front-panel connectors, pins 2/14 and 5/13 avail.
SW4-3	SW5-3	OFF	RTS+/- on front-panel connectors, pins 3/16 and 4/19 available
SW4-3	SW5-3	ON	TRXC+/- on front-panel connectors, pins 3/16 and 4/19 avail.

3.9.6 Keyboard and Mouse Port

The keyboard and mouse port is available on the front panel via an 8-pin mini DIN connector and on the VME P2 connector.

The serial port controller used for the keyboard and mouse port is compatible with the NMOS 8530 controller.

The pinout of the keyboard and mouse port is described in section 2.6.3 “Keyboard/Mouse Connector Pinout” on page 22.

The physical address for the keyboard and mouse port is shown in table 23 “NCR89C105 address map” on page 53.

20 47 18 2 2000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.9.7 Floppy Interface

The floppy disk interface is 82077AA-1-compatible. It is able to transfer data rates of 250, 300, 500 KByte/s, and 1 MByte/s. The floppy disk controller block is functionally compatible with the Intel 82077AA-1. It integrates drivers, receivers, data separator, and a 16-byte bidirectional FIFO. The floppy disk controller supports all standard disk formats (typically 720-KByte and 1.44-MByte floppies). It is also compatible with the 2.88-MB floppy format.

For information on the availability of the floppy interface on the VMEbus P2 connector, see section 2.4.8 “Floppy Interface or SCSI #2 Availability on P2” on page 16.

3.9.8 8-Bit Local I/O Devices

The following local I/O devices are interfaced via the NCR89C105.

Table 29 8-bit local I/O devices

Physical base address	Function	IRQ
$(7000.0000_{16}$ …$7003.FFFF_{16})$ 7000.0000_{16} …$7007.FFFF_{16}$	Boot flash memory device # 1 (256-KByte device #1) 512-KByte device #1 (default)	No
$(7004.0000_{16}$ …$7007.FFFF_{16})$ $(7008.0000_{16}$ …$700F.FFFF_{16})$	Boot flash memory device # 2 (256-KByte device #1) (512-KByte device #1)	No
7010.0000_{16} …$701F.FFFF_{16}$ $(7010.0000_{16}$ …$702F.FFFF_{16})$	User flash memory device # 1 1-MByte device #1 (2-MByte device #1)	No
7020.0000_{16} …$702F.FFFF_{16}$ $(7030.0000_{16}$ …$704F.FFFF_{16})$	User flash memory* 1-MByte device # 2 (2-MByte device #2)	No

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 29 8-bit local I/O devices (cont.)

Physical base address	Function	IRQ
7120.0000_{16} …$712F.FFFF_{16}$	RTC/NVRAM	No
7130.0000_{16} …$7137.FFFF_{16}$	Flash memory programming area	No
7138.0000_{16} …$713F.FFFF_{16}$	On-board configuration register	No

* The user flash memory is accessible for read access in the linear access area beginning at 7010.0000_{16}. To reprogram the user flash memory, the 512-KByte programming window beginning at 7130.0000_{16} is used.

3.9.9 Boot Flash Memory

The boot flash memory consists of up to two 2-Mbit or 4-Mbit flash memory devices. In the default configuration, there is one 4-Mbit device installed. Other variants are additional assembly options.

The boot flash memory devices can be reprogrammed on-board and can also be write-protected via hardware switch SW9-1. When SW9-1 is OFF, the devices are write-protected (default setting).

The boot flash memory devices are installed in sockets at location J15 and J16. This permits programming them in a standard programmer which may be necessary if the power fails during reprogramming. In this case, the contents of the boot flash memory would be lost and the board would not be able to boot.

Table 30 Boot flash memory capacity

Devices	Count	Capacity	Default
256 K * 8	2	512 KByte	
512 K* 8	1	512 KByte	X
512 K* 8	2	1 MByte	

The on-board programming of the boot flash memory devices requires setting some bits in the flash memory programming control register #1, #2, and the flash memory program voltage control register (see section 3.9.11 “Programming the On-board Flash Memories” on page 62).

20-4718-2-2000/146 420.000/1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

IMPORTANT



The boot flash memory is accessible for read access in the linear access area beginning at address 7000.0000_{16}. To reprogram the boot flash memory the 512-KByte programming window beginning at 7130.0000_{16} is used.

3.9.10 User Flash Memory

The user flash memory area consists of a maximum of two 8-Mbit or two 16-Mbit flash memory devices, providing a capacity of 2 or 4 MByte respectively. The capacity of user flash memory devices is outlined in the product nomenclature (see table 2 “Product nomenclature of the SPARC/CPU-8VT” on page 4).

The user flash memory area can be used to store ROMable operating systems as well as application specific code.

Table 31 User flash memory capacity

Devices	Count	Capacity	Default
1M * 8 (8 Mbit)	2	2 MByte	X
2M * 8 (16 Mbit)	1	2 MByte	
2M * 8 (16 Mbit)	2	4 MByte	

The user flash memory devices can be reprogrammed on-board. They can also be write-protected via hardware switch SW9-2.

- When SW9-2 is ON, write accesses are possible.
- When SW9-2 is OFF, the devices are write-protected.

The on-board programming of the user flash memory devices requires setting some bits in the flash memory programming control register #1, #2, and the flash memory programming voltage control register (see section 3.9.11 “Programming the On-board Flash Memories” on page 62).

IMPORTANT



The user flash memory is accessible for read access in the linear access area beginning at address 7010.0000_{16}. To reprogram the user flash memory, the 512-KByte programming window beginning at 7130.0000_{16} is used.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.9.11 Programming the On-board Flash Memories

Both areas of flash memories, the boot area and the user area, can be re-programmed on-board. For further details about programming the on-board memory devices, see section 5.5 “Flash Memory Support” on page 136 f.

The address range in which the flash memory devices can be programmed is located in a 512-KByte page (programming window) of the Generic Port area of the NCR89C105 (SLAVIO). The physical address range is 7130.0000_{16} … $7137.FFFF_{16}$.

To program the on-board flash memory devices, the following steps are required:

- Disable hardware write protection in order to program the flash memory devices:
  - SW9-1 must be ON in order to program the boot flash memory
  - SW9-2 must be ON in order to program the user flash memory.

  For the location of the switches on the board, see figure 1 “Location diagram of the SPARC/CPU-8VT (schematic)” on page 10.
- Switch the programming voltage ON by setting the appropriate bit in the flash memory programming voltage control register (not necessary for 5Vdevices 29F016).
- Set address lines A[21:19] of flash memory programming control register #1 to the requested address range. Set the device number of the device to be selected.
- Select either the user flash memory or the boot flash memory for programming (flash memory programming control register #2).
- After the flash memory devices have been programmed, return to the default settings of SW9-1 and SW9-2. This protects the flash memory devices from being programmed by accident.

To enable programming and to decide which area is to be mapped to the programming window, the following six bits are used: VPP_ON, A[21:19], SEL_ROM, and SEL_BOOT (see the following description of the flash memory programming voltage control register and flash programming control register #1 and #2).

20-4718-2 2.0000.146 420.000.1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.9.11.1 Boot EPROM and User Flash Size Control Register

Table 32 Boot EPROM and user flash size control register

7138.0008_{16}								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	USERSEL [1…0]		1	BOOT SEL0

BOOTSEL0 (R/W) BOOTSEL0 selects the size of the boot EPROM devices installed in the boot EPROM sockets.

= 0 512K * 8 (4 Mbit) EPROM devices

= 1 256K * 8 (2 Mbit) EPROM devices

USERSEL[1…0] (R/W) USERSEL[1…0] select the size of the user flash devices installed in the user flash sockets.

= 00_2 reserved

= 01_2 reserved

= 10_2 1MB * 8 (8 Mbit) flash memory devices

= 11_2 2MB * 8 (16 Mbit) flash memory devices

3.9.11.2 Flash Memory Programming Voltage Control Register

To enable the programming of the flash memory devices, the +12V programming voltage must be switched ON. This is done by setting bit VPP_ON in the flash memory programming voltage control register to 1.

IMPORTANT



Do not change unused or reserved bits of the LCA configuration registers to maintain software compatibility to future SPARC/CPU-8VT revisions and avoid malfunctions of the board.

Initialization VPP_ON is cleared on reset. This inhibits the programming of the flash memory devices.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 33 Flash memory programming voltage control register

7138.000A$_{16}$								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	1	1	1	VPP_ON

VPP_ON (R/W) VPP_ON turns the +12V programming voltage for all flash memories ON or OFF.

= 1 The programming voltage is turned ON.

= 0 The programming voltage is turned OFF.

3.9.11.3 Flash Memory Programming Control Register #1

IMPORTANT



Do not change unused or reserved bits of the LCA configuration registers to maintain software compatibility to future SPARC/CPU-8VT revisions and avoid malfunctions of the board.

Table 34 Flash memory programming control register #1

7138.0002$_{16}$								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	A[21:19]			SEL_ROM

A[21…19] (R/W) The outputs of A[21…19] are directly connected with the address pins A21, A20, and A19 of both user flash memories which allows to address flash memories with up to 4 MByte size.
As the flash memories are accessible only in the physical range 7130.0000$_{16}$ … 7137.FFFF$_{16}$, software has to modify these bits to make a specific 512-KByte page available in this address range.

Table 35 Flash memory control register pages

A[21…19]	**Page**	**Accessible area of flash memory (offset)**
000$_2$	0	0000.0000$_{16}$ … 0007.FFFF$_{16}$
001$_2$	1	0008.0000$_{16}$ … 000F.FFFF$_{16}$
010$_2$	2	0010.0000$_{16}$ … 0017.FFFF$_{16}$
011$_2$	3	0018.0000$_{16}$ … 001F.FFFF$_{16}$
100$_2$	4	0020.0000$_{16}$ … 0027.FFFF$_{16}$

20-718 2.20000.146 420.000/1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 35 Flash memory control register pages (cont.)

A[21…19]	Page	Accessible area of flash memory (offset)
101_2	5	0028.0000_{16} … $002F.FFFF_{16}$
110_2	6	0030.0000_{16} … $0037.FFFF_{16}$
111_2	7	0038.0000_{16} … $003F.FFFF_{16}$

SEL_ROM (R/W)

SEL_ROM and SEL_BOOT in the flash memory programming control register #2 are used to select 1 of 4 flash memory devices to be accessible in the physical address range 7130.0000_{16} … $7137.FFFF_{16}$ (see table 37 “Flash memory programming: SEL_ROM and SEL_BOOT” on page 65).

3.9.11.4 Flash Memory Programming Control Register #2

IMPORTANT



Do not change unused or reserved bits of the LCA configuration registers to maintain software compatibility to future SPARC/CPU-8VT revisions and avoid malfunctions of the board.

Table 36 Flash memory programming control register #2

7138.0009_{16}								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	1	1	1	SEL_BOOT

SEL_BOOT (RW)

SEL_BOOT and SEL_ROM in the flash memory programming control register #1 are used to select 1 of 4 flash memories to be accessible in the physical address range 7130.0000_{16} … $7137.FFFF_{16}$. The possible configurations are listed in the following table.

Table 37 Flash memory programming: SEL_ROM and SEL_BOOT

		SEL_BOOT	
		0	1
SEL_ROM	0	1st user flash memory device accessible	1st boot flash memory device accessible
	1	2nd user flash memory device accessible	2nd boot flash memory device accessible

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.9.12 RTC/NVRAM

The RTC/NVRAM M48T18 combines:

- an 8K * 8 full CMOS SRAM,
- a byte-wide accessible real time clock,
- a crystal,
- and a long-life lithium carbon monofluoride battery.

All components are integrated in a single plastic DIP package. The M48T18 is a non-volatile pin- and functional equivalent to any JEDEC standard 8K * 8 SRAM.

For a detailed description of the RTC/NVRAM, see data sheet “SGS-THOMSON M48T18” in SPARC/CPU-8VT Set of Data Sheets.

3.10 VMEbus Interface

The SPARC/CPU-8VT utilizes the FGA-5000 to provide fully SBus and VMEbus compliant interfaces. Supported functions include:

- master and slave data transfer capabilities,
- VMEbus interrupt handling,
- and arbitration functions.

Additional VMEbus utility functions and a special loop-back cycle for stand-alone testing of the interface are provided.

3.10.1 Features of the FGA-5000

The FGA-5000 provides the following features:

- VMEbus master and slave interface
- DMA controller
- Interrupts
- VMEbus arbiter
- FORCE message broadcast, mailboxes, and semaphores
- Reset functions
- System controller functions
- Timers

20-4718-2-2000146-420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

For a complete description of the FGA-5000, see the *FGA-5000 Technical Reference Manual*, available from FORCE COMPUTERS.

3.10.2 Address Mapping for the VMEbus Interface FGA-5000

The table below lists the physical addresses of the VMEbus interface FGA-5000.

Table 38 Memory map of the VMEbus interface

		TurboSPARC		FGA-5000
Address	**Function**	**SBus Slot #**	**SBus slave select #**	**Select #**
3000.0000_{16} …$3FFF.FFFF_{16}$	VMEbus interface	1	0	SSEL[0]
4000.0000_{16} …$4FFF.FFFF_{16}$	VMEbus interface (SBus module)	2	1	SSEL [1]
5000.0000_{16} …$5FFF.FFFF_{16}$	VMEbus interface (SBus module)	3	2	SSEL [2]
6000.0000_{16} …$6FFF.FFFF_{16}$	VMEbus interface	4	3	SSEL [3]
$7E00.0000_{16}$ …$7FFF.FFFF_{16}$	VMEbus interface	5	4	SSEL [5]

The FGA-5000 can be selected with up to 6 SBus select input signals SSEL [5…0]. The TurboSPARC CPU supports only 5 select signals SLVSEL[4…0]. The remaining select signal of the FGA-5000 can be used to expand the VMEbus address area.

The I/O devices NCR89C100 (MACIO #1, #2) and NCR89C105 (SLAVIO) are selected in SBus slot 5 by SBus slave select 4. As the upper part in this range is not used by these devices, the SBus slave select 4 has been splitted into 2 signals: SB_SEL [4] and SB_SEL [5]. Now the I/O devices are selected by SB_SEL[4] and the VMEbus interface FGA-5000 is selected by SB_SEL[5]. With this expansion, the VMEbus interface FGA-5000 shares SBus slot 5 with the I/O devices and can use an additional address range of up to 32 MByte in this SBus slot.

On the base board, SBus slot 2 (SBus slave select 1) and SBus slot 3 (SBus slave select 2) are provided for SBus modules. When no SBus modules are installed, the SBus slots can be used to expand the VMEbus address range again. In this case, 256 MByte are gained with every additional SBus slot.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

When using all address range resources for the VMEbus interface, the TurboSPARC CPU can access the VMEbus interface FGA-5000, internal registers, and VMEbus slaves, in an address area of 1056 MByte (1GByte + 32 MByte).

3.10.3 Adaptation of the FGA-5000

Some aspects of the FGA-5000 require a small amount of glue logic to be built around this device in order to use it on the SPARC/CPU-8VT.

When the FGA-5000 is not VMEbus master, the VMEbus input signal BERR has not been directly routed to the FGA-5000. This masking is required for normal VMEbus transfers. However, during FMB transfers the VMEbus slave should see the input signal BERR.

When the FGA-5000 is one of several selected slaves during an FMB cycle, and one or more of the other slaves acknowledges the transfer with bus error, then the FGA-5000 does not recognize that the message is invalid. In order to enable the software to discard this invalid message, additional registers have been implemented in a separate programmable device on the base board.

For information on the access and the interpretation of the contents of the added registers, see section 3.13 “Additional Registers” on page 73.

For information on the FMB implementation in the FGA-5000 device, refer to the *FGA-5000 Technical Reference Manual*.

3.10.4 VMEbus SYSRESET Enable/Disable

SYSRESET input

An external SYSRESET generates an on-board RESET in the default switch setting, i.e., SW7-2 is OFF. When SW7-2 is ON, the external SYSRESET does not generate an on-board RESET.

SYSRESET output

A SYSRESET signal is generated to the VMEbus

- when an on-board local SBus reset occurs on the SPARC/CPU-8VT (e.g. the front-panel reset key is toggled or power failure is detected). This SYSRESET signal can be disabled by setting SW7-3 to ON.
- by power-up reset. Power-up reset occurs by switching on the power and determines when the board supply voltage reached a stable value. This SYSRESET signal can be disabled by setting SW7-1 to ON.

As written in the VME specification each board must assert SYSRESET output at power-up when power supply reaches 3 V until power is stable.

2047.18.2.2000.00.146.420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This feature is enabled by default (SW7-1 is OFF). It can be disabled by setting SW7-1 to ON.

3.11 On-Board Configuration Registers

The following table lists the physical address of the registers used for system configuration.

Table 39 On-board configuration registers

Address	Reset value	Size	Description
7138.0000_{16}	$F0_{16}$	8 bit	SYS LED control register (see page 72)
7138.0001_{16}	$F0_{16}$	8 bit	USER LED control register (see page 72)
7138.0002_{16}	$F0_{16}$	8 bit	Flash memory programming control register #1 (see page 64)
7138.0003_{16}	FX_{16}	8 bit	Rotary switch status register (see page 73)
7138.0004_{16}	FC_{16}	8 bit	Network interface 1 control and status register (see page 51)
7138.0005_{16}	FC_{16}	8 bit	Network interface 2 control and status register (see page 51)
7138.0006_{16}	FD_{16}	8 bit	reserved
7138.0007_{16}	XX_{16}	8 bit	reserved
7138.0008_{16}	FE_{16}	8 bit	Boot EPROM and user flash size control register (see page 63)
7138.0009_{16}	FE_{16}	8 bit	Flash memory programming control register 2 (see page 65)
$7138.000A_{16}$	FE_{16}	8 bit	Flash memory programming voltage control register (see page 64)
$7138.000B_{16}$	XX_{16}	8 bit	Seven-segment LED display control register (see page 72)
$7138.000C_{16}$	FE_{16}	8 bit	FMB channel 0 data discard status register (see page 74)
$7138.000D_{16}$	FE_{16}	8 bit	FMB channel 1 data discard status register (see page 74)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 39 On-board configuration registers (cont.)

Address	Reset value	Size	Description
$7138.000E_{16}$	00_{16}	8 bit	reserved
$7138.000F_{16}$	FX_{16}	8 bit	LCA identification register (see below)

Table 40 LCA identification register

$7138.000F_{16}$								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	REV R			

REV (R) REV indicates the revision of the LCA.

= F_{16} Not valid

= E_{16} Revision 0

= D_{16} Revision 1

3.12 Front Panel

For an overview and a diagram of the front panel, see figure 1 “Location diagram of the SPARC/CPU-8VT (schematic)” on page 10.

3.12.1 RESET and ABORT Keys

The front panel on the SPARC/CPU-8VT has 2 mechanical keys which directly influence the system.

RESET key

The RESET key enables the user to reset the whole board. If the board is VMEbus system controller (Slot-1 device) and the SYSRESET output is enabled by SW7-3, the SYSRESET signal of the VMEbus also becomes active with the RESET key. This resets the complete VMEbus system. Via on-board switch SW6-4, it is possible to deactivate the RESET key:

- When SW6-4 is OFF, the RESET key works.
- When SW6-4 is ON, toggling the RESET key has no effect.

20-718-2.2000.00-146 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

ABORT key

The ABORT key on the front panel can be used to generate a non-maskable interrupt (level 15). The ABORT key function is controlled by switch SW7-4:

- When SW7-4 is OFF, the ABORT key works.
- When SW7-4 is ON, toggling the ABORT key has no effect.

If the ABORT key produces a non-maskable interrupt, the pending signal can be read in the miscellaneous control and status register 0 (MCSR0 register). The ABORT interrupt request mapping register (ABORT_IRQ_MAP) is used to map and enable an interrupt, generated by assertion of the ABKEY signal.

For further information see the FGA-5000 Technical Reference Manual.

3.12.2 Front Panel Status LEDs

There are 4 single LEDs on the front panel.

- RUN/RESET LED
- VME BM LED (bus master)
- SYS status LED
- UL (user) status LED

RUN/RESET LED

The RUN/RESET LED

- is red when any reset signal on the board is active.
- begins blinking when SB_SEL<4> is inactive for more than 0.5 s in order to signal a hang-up.
- is green in all other cases.

BM LED

The BM LED reflects all VMEbus master activities on the SPARC/CPU-8VT. The BM LED

- lights up green when the board accesses the VMEbus.
- turns red when SPARC/CPU-8VT is asserting SYSFAIL to the VME-bus.

SYS LED and user LED

A system (SYS) and a user (UL) programmable LED on the front panel allow for system- and user-defined diagnostic features. They can be accessed via the 2 LED control registers.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 41 SYS and user LED control registers

7138.0000_{16} for SYS LED; 7138.0001_{16} for user LED								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	BLINK_FREQ		COLOR	

BLINK_FREQ (R/W) BLINK_FREQ controls the frequency at which the respective LED is blinking.

- = 00_2 disabled
- = 01_2 0.5 Hz blinking frequency
- = 10_2 1 Hz blinking frequency
- = 11_2 2 Hz blinking frequency

COLOR (R/W) COLOR controls the status and the color of the respective LED.

- = 00_2 turned off
- = 01_2 green LED
- = 10_2 red LED
- = 11_2 yellow LED

3.12.3 Seven-Segment LED Display

A user programmable seven-segment LED display on the front panel allows for user-defined diagnostic features. It can be accessed via the seven-segment LED display control register.

Table 42 Seven-segment LED display control register (SEV_SEG_CTRL)

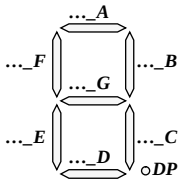
$7138.000B_{16}$								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	DP	SEG_[G…A]						

DP and SEG_[G…A] (W) DP controls the status of the decimal point (DP) and SEG_[G… A] controls the segments (SEG_G…SEG_A) of the hexadecimal display (see figure 23 “Naming the parts of the hexadecimal display” on page 73).

- = 0 The respective part of the display is turned off.
- = 1 The respective part of the display is turned on.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Figure 23 Naming the parts of the hexadecimal display



3.12.4 Rotary Switch

A rotary switch on the front panel allows for user-defined parameterizing of user applications. Its setting can be read via the rotary switch status register.

Table 43 Rotary switch status register (ROTARY_SWITCH_STAT)

F.F124.0003$_{16}$								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	ROT_SWI[3…0]			

ROT_SWI[3…0] (R) ROT_SWI[3…0] indicates the setting of the rotary switch.

= 0000_2 The setting of the rotary switch is 0_{16}.

= 0001_2 The setting of the rotary switch is 1_{16}.

…

= 1110_2 The setting of the rotary switch is E_{16}.

= 1111_2 The setting of the rotary switch is F_{16}.

3.13 Additional Registers

The following additional registers are provided to increase functionality. They are related to the FMB channels of the FGA-5000. A complete description of the FGA-5000 device is found in the *FGA-5000 Technical Reference Manual*, available from FORCE COMPUTERS.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.13.1 FMB Channel 0 Data Discard Status Register

The FMB channel 0 consists of an 8-stage FIFO and so does the FMB channel 0 data discard status register. Read accesses to this register switch the internal read pointer one step ahead in the FIFO. Whenever the software needs to perform elementary functions as such, we recommend coordinating accesses to this register and the related FGA-5000 registers so that synchronisation of both FIFOs is not broken.

Table 44 FMB channel 0 data discard status register

7138.000C$_{16}$								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	1	1	1	MSG_ VALID

MSG_VALID (R) MSG_VALID indicates whether the data in the FMB channel 1 of the FGA-5000 has to be discarded.

= 0 FMB channel's data are invalid.

= 1 FMB channel's data are valid.

3.13.2 FMB Channel 1 Data Discard Status Register

Table 45 FMB channel 1 data discard status register

7138.000D$_{16}$								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	1	1	1	1	1	1	1	MSG_ VALID

MSG_VALID (R) MSG_VALID indicates whether the data in the FMB channel 1 of the FGA-5000 has to be discarded.

= 0 FMB channel's data are invalid.

= 1 FMB channel's data are valid.

20-718-2.2000.146 420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Please Note…

The circuit schematics section is an integral part of the *SPARC/CPU-8VT Technical Reference Manual (P/N 204718)*. Yet, it is packaged separately to enable easy updating.

The circuit schematics section will always be shipped with the *Technical Reference Manual*.

Please:

- ☞ **Insert the circuit schematics section (P/N –) now into the *SPARC/CPU-8VT Technical Reference Manual* (P/N 204718*)*.**
- ☞ **Remove this sheet.**

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

4 Circuit Schematics

Copies of the SPARC/CPU-8VT are found on the following pages.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5 FORCE OpenBoot Enhancements

This section describes the enhancements to the standard OpenBoot firmware that have been done for the SPARC/CPU-8VT. For a description of standard OpenBoot firmware features, see the *OPEN BOOT PROM 2.0 MANUAL SET*.

Besides the commands already provided by the standard OpenBoot firmware, the OpenBoot firmware available on the SPARC/CPU-8VT includes further words for

- accessing and controlling the VMEbus interface,
- accessing and programming available flash memories,
- controlling the operating mode of the watchdog timer,
- and making use of the diagnostics.

The following subsections describe these words in detail, and examples are given when it seems necessary to convey the usage of a particular or a group of words. In general, each word is described using the notation stated below:

`name` *(stack-comment)* description

The `name` field identifies the name of the word being described.

The stack parameters passed to and returned from a word are described by the *stack-comment* notation – enclosed in parentheses –, and shows the effect of the word on the evaluation stack. The notation used is:

parameters before execution — parameters after execution

The parameters passed and returned to the word are separated by the "—".

The *description* body describes the semantics of the word and conveys the purpose and effect of the particular word.

The OpenBoot ported to the SPARC/CPU-8VT is based upon the OpenBoot 2.15 obtained from Sun Microsystems.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.1 Controlling the VMEbus Master and Slave Interface

VMEbus addressing

The VMEbus has a number of distinct address spaces represented by a subset of the 64 possible values encoded by the 6 address modifier bits. The size of the address space depends on the particular address space, for example the *standard* (A24) address space is limited to 16 MByte, whereas the *extended* (A32) address space allows to address 4 GByte. An additional bit – which corresponds with the VMEbus LWORD* signal – is used to select between 16-bit and 32-bit data.

A *physical* VMEbus address is represented numerically by the pair *phys.high* (also called *space*) and *phys.low* (also called *offset*). The *phys.high* consists of the 6 address modifier bits `AM0` through `AM5` corresponding with bit `0` through `5`; and the data width bit `LWORD*` (`0` = 16-bit data, `1` = 32-bit data) in bit `6`.
OpenBoot provides a number of constants combining the information mentioned above. These constants are called `AML` constants. `AML` is the combination of the first letters of the words address modifier and `LWORD*`. Each `AML` constant specifies a unique address space:

`vmea16d16` (— *h# 2d*) returns the `AML` constant $2D_{16}$ identifying the privileged *short* (A16) address space with 16-bit data transfers.

`vmea16d32` (— *h# 6d*) returns the `AML` constant $6D_{16}$ identifying the privileged *short* (A16) address space with 32-bit data transfers.

`vmea24d16` (— *h# 3d*) returns the `AML` constant $3D_{16}$ identifying the privileged *standard* (A24) address space with 16-bit data transfers.

`vmea24d32` (— *h# 7d*) returns the `AML` constant $7D_{16}$ identifying the privileged *standard* (A24) address space with 32-bit data transfers.

`vmea32d16` (— *h# 0d*) returns the `AML` constant $0D_{16}$ identifying the privileged *extended* (A32) address space with 16-bit data transfers.

`vmea32d32` (— *h# 4d*) returns the `AML` constant $4D_{16}$ identifying the privileged *extended* (A32) address space with 32-bit data transfers.

The `AML` modifiers described below are available to modify the `AML` in such a way that additional VMEbus address spaces may be identified:

`burst` (*phys.high-single — phys.high-burst*) converts the numeric representation of any VMEbus `AML` constant in single-transaction form to its burst-transaction (BLT) form.

```
ok vmea24d32 burst .
3f
ok
```

20-718-2.2000.00-146 420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vme-user` (*phys.high-privileged — phys.high-non-privileged*) converts the numeric representation of any VMEbus `AML` constant in privileged form to its non-privileged (user-mode) form.

```
ok vmea16d32 vme-user .
69
ok
```

`vme-program` (*phys.high-data — phys.high-program*) converts the numeric representation of any VMEbus `AML` constant in data-transaction form to its program form.

```
ok vmea32d16 vme-program .
e
ok
```

The *offset* specifies the VMEbus address of an area within the selected address *space*. The value of the offset depends on the address space. For example the *standard* (A24) address space is limited to 16 MByte (24-bit addresses ranging from 00.0000_{16} to $FF.FFFF_{16}$), whereas the *extended* (A32) address space allows to address 4 GByte (32-bit addresses ranging from 0000.0000_{16} to $FFFF.FFFF_{16}$), and the *short* (A16) address space is limited to 64 KByte (16-bit addresses ranging from 0000_{16} to $FFFF_{16}$).

Example:

The example below shows how to specify the address of a VMEbus board that is accessible within the *extended* (A32) address space (`vmea32d32`) beginning at offset 4080.0000_{16}:

```
ok h# 4080.0000 vmea32d32
```

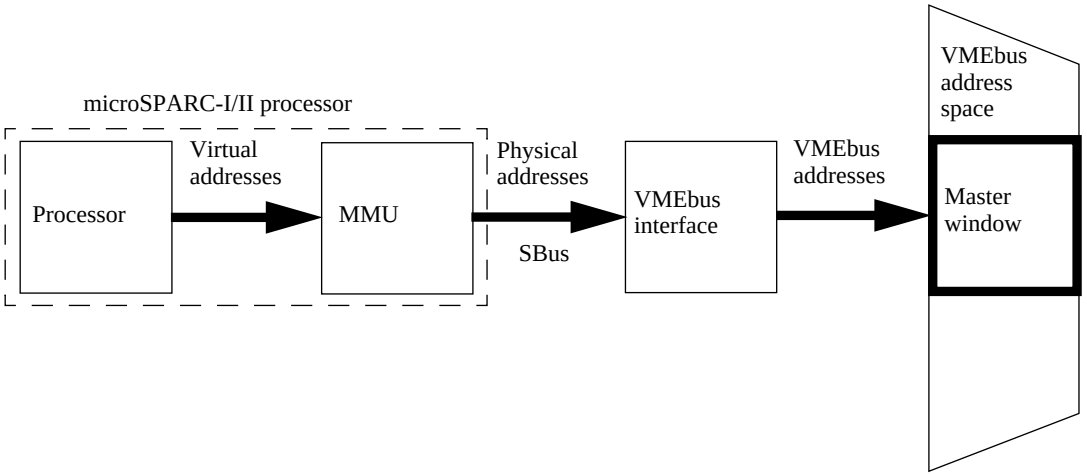
The first part represents the *offset* (*phys.low*) and the second part represents the *space* (*phys.high*).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.1.1 VMEbus Master Interface

As shown in the figure below the processor emits virtual addresses during a data transfer cycle which are translated to physical addresses by the MMU. Within a microSPARC-I/II environment, the VMEbus is connected with the SBus and the VMEbus interface responds to unique physical SBus addresses and executes the appropriate VMEbus transfer. Depending on the physical addresses and the state of specific registers within the VMEbus interface, the interface addresses a specific VMEbus address space.

Figure 24 Address translation (master): microSPARC – SBus – VMEbus



Before the processor may access a specific area within one of the VMEbus address spaces, the steps described below must be taken:

- The VMEbus interface has to be set up to respond to specific physical SBus addresses to forward the access to a certain VMEbus address space.
- The contents of the MMU table are modified to make the SBus address range available to the processor's address range and thus allowing accesses to the specific VMEbus area using virtual addresses. In general, this means that the VMEbus area is made available to the processor's virtual address space.
- The VMEbus interface has to be enabled, in order to allow accesses to the VMEbus address space.

OpenBoot provides commands to make VMEbus areas available to the processor's virtual address space and to remove these VMEbus areas from the processor's virtual address space. The command `vme-memmap` performs all steps to make specified VMEbus areas available to the processor's virtual address space. The command `vme-free-virtual` removes the VMEbus area which has been made available previously from the processor's virtual address space.

20-47.18-2.2000.00.146 420.000.1 March 1993

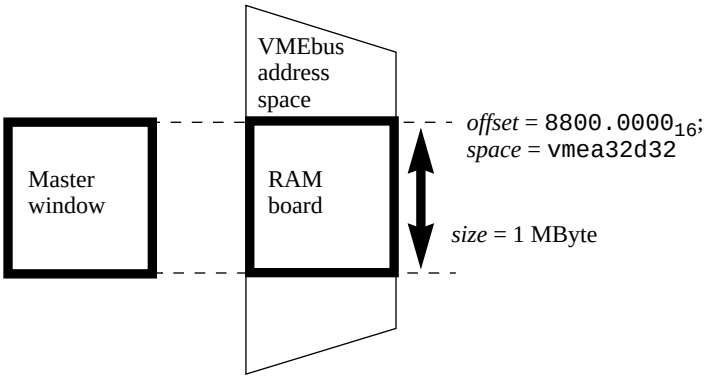
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vme-memmap` (*offset space size — vaddr*) initializes the VMEbus master interface according to the parameters *offset* and *space* and returns the virtual address *vaddr* to be used to access the specified VMEbus area.
The parameters *space* and *offset* describe the VMEbus address area in detail: *offset* specifies the physical VMEbus address of the area to be accessed and *space* specifies the address space where the VMEbus area is located in. The size of the VMEbus area is given by *size*.

Example:

Assumed a memory board is accessible within the *extended* (A32) VMEbus address space beginning at address 8800.0000_{16} and ranging to $880F.FFFF_{16}$ (1 MByte) as shown in the figure below:

Figure 25 Mapping a VMEbus area to the processor's virtual address space



In order to make this VMEbus area available to the processor's virtual address space, the commands listed below have to be used:

```
ok 0 value vme-ram
ok h# 8800.0000 vmea32d32 1Meg vme-memmap is vme-ram
ok
```

The first command defines a variable `vme-ram` which is later used to store the virtual address of the VMEbus area. The second command listed above makes 1 MByte beginning at physical address 8800.0000_{16} within the *extended* (A32) VMEbus address space available to the processor's virtual address space. The virtual address returned by the command is stored in the variable `vme-ram` which has been defined by the first command `value`. The variable `vme-ram` may be used later to access this VMEbus area.

`vme-free-virtual` (*vaddr size* —) removes the VMEbus area associated with the virtual address *vaddr* from the processor's virtual address space.
The VMEbus area previously made available to the processor's virtual address space is removed from the virtual address space using the `vme-free-virtual` command as shown below:

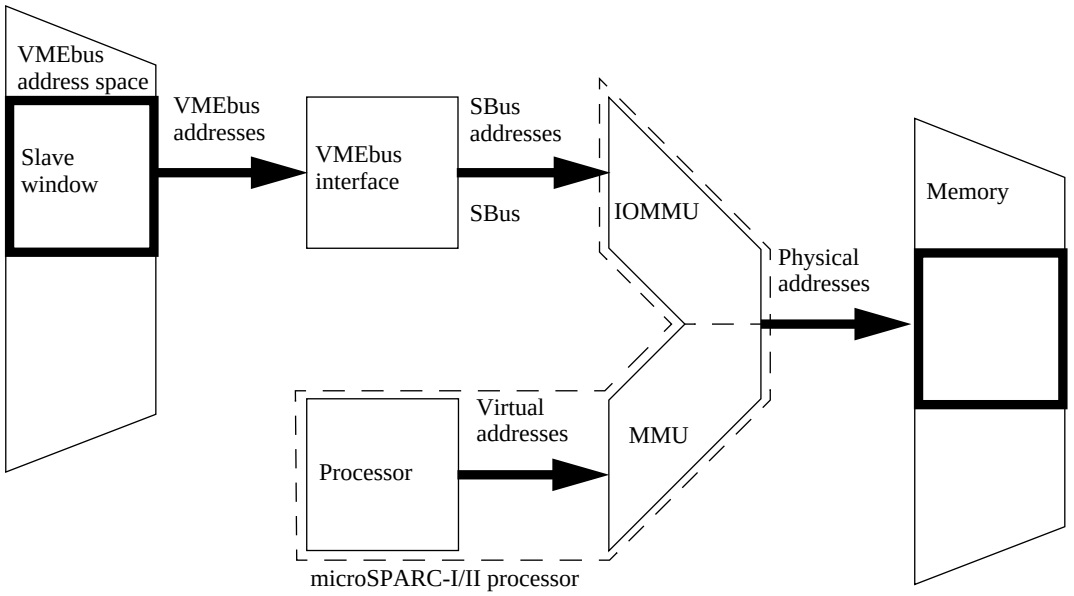
```
ok vme-ram 1Meg vme-free-virtual
ok
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.1.2 VMEbus Slave Interface

As shown in the figure below the VMEbus interface responds to unique VMEbus addresses and translates these addresses to virtual SBus addresses. The IOMMU translates these virtual SBus addresses to physical addresses, which address a certain area within the on-board memory.

Figure 26 Address translation (slave): VMEbus – SBus – microSPARC



The processor accesses the same on-board memory by applying virtual addresses to the MMU which are translated to the appropriate physical addresses.
Before another VMEbus master may access the on-board memory, the following steps have to be taken to make a certain amount of on-board memory available to one of the VMEbus address spaces, e.g. s*tandard* (A24), or *extended* (A32) address space:

- A certain amount of the available on-board memory has to be allocated to make it available to one of the VMEbus address spaces.
- The VMEbus interface has to be set up to respond to specific addresses within the selected VMEbus address spaces. In general, registers within the VMEbus interface are modified to accomplish this.
- The contents of the IOMMU table are modified to associate the virtual SBus addresses, which are emitted by the VMEbus interface during a slave access, with the physical addresses of the allocated memory. Furthermore, the contents of the MMU table are modified to associate the virtual addresses, which are emitted by the processor during accesses to the on-board memory, with the physical addresses of the allocated memory.

20-4718-2-2000148-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

- The VMEbus interface has to be enabled, in order to allow accesses from the VMEbus to the on-board memory.

OpenBoot provides commands to make the on-board memory available to one of the VMEbus address spaces, and to remove the on-board memory from these VMEbus address spaces. The command `set-vme-slave` performs all steps to make a specified amount of memory available at a specific VMEbus address space. The command `reset-vme-slave` removes the on-board memory from the VMEbus address space.

`set-vme-slave` (*offset space size — vaddr*) initializes the VMEbus slave interface according to the parameters passed to the command and returns the virtual address *vaddr* of the memory which has been made available to the VMEbus. OpenBoot provides all necessary mappings (MMU and IOMMU) to access the memory from the processor and the VMEbus.
The parameters *space* and *offset* specify where the slave interface is accessible within the VMEbus address range. The parameter *offset* specifies the physical base address of the slave interface within the particular address space. The size of the memory that should be made available to the VMEbus is given by *size*.

Example:

Assumed that 1 MByte of on-board memory should be made available to the *extended* (A32) address space of the VMEbus beginning at the VMEbus address 4080.0000_{16}, the commands listed below have to be used.

```
ok 0 value my-mem
ok 4080.0000 vmea32d32 1meg set-vme-slave is my-mem
ok
```

The first command defines a variable `my-mem` which is later used to store the virtual address of the on-board memory which has been made available to the VMEbus. The second command listed above makes 1 MByte beginning at physical address 4080.0000_{16} available within the *extended* (A32) VMEbus address space. The virtual address returned by the command is stored in the variable `my-mem` which has been defined by the first command `value`. The variable `my-mem` may be used later to access the on-board memory.

`reset-vme-slave` (*vaddr size* —) resets the VMEbus slave interface associated with the virtual address *vaddr* and destroys all mappings which were necessary to make the memory available to VMEbus.

```
ok my-mem 1Meg reset-vme-slave
ok
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.2 VMEbus Interface

The VMEbus Interface on the SPARC/CPU-8VT consists of FORCE COMPUTERS' SPARC FGA-5000 (VSI) chip.

5.2.1 Generic Information

The variable – which is declared as `value` – described below is used to retrieve generic information about the VMEbus interface:

`vsi-va` (— *vaddr*) returns the virtual base address *vaddr* of the registers included in the SPARC FGA-5000.

The base address of the SPARC FGA-5000 may be modified by the command described below:

`vsi-base-addr!` (*offset sbus-slot#* —) sets the base address of the SPARC FGA-5000 according to the given Sbus slot number *sbus-slot#* and the offset *offset* within the specified SBus slot. The values *sbus-slot#* and *offset* are stored in the appropriate variables `vsi-sbus-slot#` and `vsi-offset`. Furthermore, the command sets the variables `vsi-pa` and `vsi-va` according to the given parameters.

On the SPARC/CPU-8VT the SBus slots are utilized as stated in the following table:

Table 46 SBUS slots utilization

sbus slot#	Select	Address range		Description
0	–	2000.0000_{16}	...2FFF.FFFF16	SBus slot #0 (AFX)
1	SSEL*0*	3000.0000_{16}	$...3FFF.FFFF_{16}$	SBus slot #1 (reserved for VMEbus accesses through the SPARC FGA-5000)
2	SSEL*1*	4000.0000_{16}	$...4FFF.FFFF_{16}$	SBus slot #2 (Sbus Card 1)
3	SSEL*2*	5000.0000_{16}	$...5FFF.FFFF_{16}$	SBus slot #3 (Sbus Card 2)
4	SSEL*3*	6000.0000_{16}	$...6FFF.FFFF_{16}$	SBus slot #4 (reserved for VMEbus accesses through the SPARC FGA-5000)
5	SSEL*4*	7000.0000_{16}	$...7FFF.FFFF_{16}$	SBus slot #5 (MACIO#1 + #2, SLAVIO, SPARC FGA-5000 registers)

20 47 28 2 2000146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vsi-base-addr@` (— *offset sbus-slot#*) returns the base address of the SPARC FGA-5000 represented by the SBus slot number *sbus-slot#* and the offset *offset* within the specific SBus slot.

5.2.2 Register Addresses

The following commands are available to get the virtual addresses of the System Configuration Registers.

`sysconfig-va` *(— vaddr)* returns the virtual base address *vaddr* of the System Configuration Registers.

`led1-ctrl` *(— vaddr)* returns the virtual address *vaddr* of the SYS LED Control Register.

`led2-ctrl` *(— vaddr)* returns the virtual address *vaddr* of the User LED Control Register.

`flash-ctrl1` *(— vaddr)* returns the virtual address *vaddr* of the Flash Memory Control Register 1.

`ni1-csr` *(— vaddr)* returns the virtual address *vaddr* of the Network Interface 1 Control and Status Register.

`ni2-csr` *(— vaddr)* returns the virtual address *vaddr* of the Network Interface 2 Control and Status Register.

`xchg-macio-ctrl` *(— vaddr)* returns the virtual address *vaddr* of the Exchange MACIO Control Register.

`rotary-switch-stat` *(— vaddr)* returns the virtual address *vaddr* of the Rotary Switch Status Register.

`boot-rom-size-ctrl` *(— vaddr)* returns the virtual address *vaddr* of the Boot ROM Size Control Register.

`flash-ctrl2` *(— vaddr)* returns the virtual address *vaddr* of the Flash Memory Control Register 2.

`flash-vpp-ctrl` *(— vaddr)* returns the virtual address *vaddr* of the Flash Memory Programming Voltage Control Register.

`led-display-ctrl` (— *vaddr*) returns the virtual address *vaddr* of the LED Display Control Register.

`fmb-0-data-discard` (— *vaddr*) returns the virtual address *vaddr* of the FMB Channel 0 Data Discard Status Register.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`fmb-1-data-discard` (— *vaddr*) returns the virtual address *vaddr* of the FMB Channel 1 Data Discard Status Register.

`switch-override` (— *vaddr*) returns the virtual address *vaddr* of the Switch Override Register.

`lca-id` (— *vaddr*) returns the virtual address *vaddr* of the LCA ID Register.

5.2.3 Register Accesses

5.2.3.1 SPARC FGA-5000 Registers

The commands described below are used to read data from and to store data in specific registers of the SPARC FGA-5000:

`vsi-sbus-base@` (— *long*) returns the contents – a 32-bit data – of the SBus Base Address Register.

`vsi-sbus-base!` (*long* —) stores the 32-bit data *long* in the SBus Base Address Register.

`vsi-id@` (— *id-code*) returns the contents – the 32-bit data *id-code* – of the Identification Register.

`vsi-vme-range@` (*range#* — *long*) returns the contents – a 32-bit data – of the SBus Address Decoding And Translation Register identified by its register number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 SBus Address Decoding and Translation Registers.

`vsi-vme-range!` (*long range#* —) stores the 32-bit value *long* in the SBus Address Decoding And Translation Register identified by its register number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 SBus Address Decoding and Translation Registers.

`vsi-vme-master-cap@` (*range#* — *byte*) returns the contents – an 8-bit data – of the VMEbus Master Capability Register identified by its register number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 VMEbus Master Capability Registers.

`vsi-vme-master-cap!` (byte *range#* —) stores the 8-bit data *byte* in the VMEbus Master Capability Register identified by its register number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 VMEbus Master Capability Registers.

20-d718-2-2000.bk 420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vsi-vme-cap@` (— *byte*) returns the contents – an 8-bit data – of the VMEbus Capability Register.

`vsi-vme-cap!` (*byte* —) stores the 8-bit data *byte* in the VMEbus Master Capability Register.

`vsi-vme-handshake@` (— *byte*) returns the contents – an 8-bit data – of the VMEbus Handshake Register.

`vsi-vme-handshake!` (*byte* —) stores the 8-bit data *byte* in the VMEbus Master Handshake Register.

`vsi-sbus-ssel@` (*range#* — *byte*) returns the contents – an 8-bit data – of the SBus Slave Slot Select Register identified by its register number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 SBus Slave Slot Select Registers.

`vsi-sbus-ssel!` (*byte range#* —) stores the 8-bit data *byte* in the SBus Slave Slot Select Register identified by its register number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 SBus Slave Slot Select Registers.

`vsi-sbus-cap@` (— *byte*) returns the contents – an 8-bit data – of the SPARC FGA-5000's SBus Capability Register.

`vsi-sbus-cap!` (*byte* —) stores the 8-bit data *byte* in the SPARC FGA-5000's SBus Capability Register.

`vsi-sbus-retry-time-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the SPARC FGA-5000's SBus Retry Time Control Register.

`vsi-sbus-retry-time-ctrl!` (*byte* —) stores the 8-bit data *byte* in the SPARC FGA-5000's SBus Retry Time Control Register.

`vsi-sbus-rerun-limit-ctrl@` (— *word*) returns the contents – a 16-bit data – of the SPARC FGA-5000's SBus Rerun Limit Control Register.

`vsi-sbus-rerun-limit-ctrl!` (*word* —) store the 16-bit data *word* in the SPARC FGA-5000's SBus Rerun Limit Control Register.

`vsi-swpar@` (— *long*) returns the contents – a 32-bit data – of the SBus Write Posting Error Address Register.

`vsi-vwpar@` (— *long*) returns the contents – a 32-bit data – of the VMEbus Write Posting Error Address Register.

`vsi-slerr@` (— *long*) returns the contents – a 32-bit data – of the SBus Late Error Address Register.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vsi-iack-emu@` (*level* — *byte*) returns the contents – an 8-bit data – of the IACK Cycle Emulation Register associated with the given *level*. The value of *level* may be one of the values in the range one through seven. Each value specifies one of the seven VMEbus interrupt request levels.

Only the least significant three bits of *level* are considered and when *level* is zero then the command treats it as if the value “one” has been passed to the command.

`vsi-vme-base@` (— *byte*) returns the contents – an 8-bit data – of the VMEbus Base Address Register.

`vsi-vme-base!` (*byte* —) stores the 8-bit data *byte* in the VMEbus Base Address Register.

`vsi-sbus-range@` (*range#* — *long*) returns the contents – a 32-bit data – of the VMEbus Address Decoding and Translation Register identified by its range number *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three VMEbus Address Decoding and Translation Registers.

`vsi-sbus-range!` (*long range#* —) stores the 32-bit data *long* in the VMEbus Address Decoding and Translation Register identified by its range number *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three VMEbus Address Decoding and Translation Registers.

`vsi-vme-ext@` (*range#* — *byte*) returns the contents – an 8-bit data – of the VMEbus Address Extension Register identified by its range number *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three VMEbus Address Extension Registers.

`vsi-vme-ext!` (*byte range#* —) stores the 8-bit data *byte* in the VMEbus Address Extension Register identified by its range number *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three VMEbus Address Extension Registers.

`vsi-reset-stat@` (— *byte*) returns the contents – an 8-bit data – of the Reset Source Register.

`vsi-intr-stat@` (— *long*) returns the contents – a 32-bit data – of the Interrupt Status Register.

`vsi-intr-stat!` (*long* —) stores the 32-bit data *long* in the Interrupt Status Register.

`.vsi-intr-stat` (—) displays the actual contents of the Interrupt Status Register. The contents of the register is displayed as shown below:

20-4718-2-2000148-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ok .vsi-intr-stat
VME-IRQ1:0 VME-IRQ2:0 VME-IRQ3:0 VME-IRQ4:0  VME-IRQ5:0
VME-IRQ6:0 VME-IRQ7:0 VME-IACK:0 FMB1    :0 FMB0     :0
IBOX    :0 LERR    :0 WDOG    :0 DMATERM :0 VWPERR   :0
SWPERR  :0 MAILBOX :0 ARBTOUT :0 ABORT   :0 SYSFAIL+ :0
SYSFAIL-:0 ACFAIL  :0
ok
```

When an interrupt is pending the command displays the one (1); otherwise it displays the zero (0) to indicate that the interrupt is not pending.

IMPORTANT



The state of the entry `SYSFAIL-` reports the occurrence of a negative edge of the VMEbus `SYSFAIL*` signal which indicates that the SYSFAIL* signal has been asserted. The state of the entry `SYSFAIL+` reports the occurrence of a positive edge of the VMEbus `SYSFAIL*` signal which indicates that the `SYSFAIL*` signal has been negated.

`vsi-arb-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the Arbiter Control Register.

`vsi-arb-ctrl!` (*byte* —) stores the 8-bit data *long* in the Arbiter Control Register.

`vsi-req-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the Requester Control Register.

`vsi-req-ctrl!` (*byte* —) stores the 8-bit data *byte* in the Requester Control Register.

`vsi-bus-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the Bus Capture Control Register.

`vsi-bus-ctrl!` (*byte* —) stores the 8-bit data *byte* in the Bus Capture Control Register.

`vsi-irq-stat@` (— *byte*) returns the contents – an 8-bit data – of the VMEbus Interrupt Status Register.

`vsi-mbox@` (*mailbox# — byte*) returns the contents – an 8-bit data – of the Mailbox Register identified by its mailbox number *mailbox#.* The value of *mailbox#* may be one of the values in the range zero through 15. Each value specifies one of the 16 Mailbox Registers.

`vsi-mbox!` (*byte mailbox# —*) stores the 8-bit data *byte* in the Mailbox Register identified by its mailbox number *mailbox#.* The value of *mailbox#* may be one of the values in the range zero through 15. Each value specifies one of the 16 Mailbox Registers.

`vsi-mbox-stat@` (— *word*) returns the contents – a 16-bit data – of the Mailbox Status Register.

`vsi-mbox-stat!` (*word* —) stores the 16-bit data *long* in the Mailbox Status Register.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vsi-sem@` (*semaphore# — byte*) returns the contents – an 8-bit data – of the Semaphore Register identified by its semaphore number *semaphore#*. The value of *semaphore#* may be one of the values in the range zero through 47. Each value specifies one of the 48 Semaphore Registers.

`vsi-sem!` (*byte semaphore# —*) stores the 8-bit data *byte* in the Semaphore Register identified by its semaphore number *semaphore#*. The value of *semaphore#* may be one of the values in the range zero through 47. Each value specifies one of the 48 Semaphore Registers.

`vsi-fmb-ctrl@` (*— byte*) returns the contents – an 8-bit data – of the Message Broadcast Control Register.

`vsi-fmb-ctrl!` (*byte —*) stores the 8-bit data *byte* in the Message Broadcast Control Register.

`vsi-fmb-addr@` (*— byte*) returns the contents – an 8-bit data – of the Message Broadcast Address Register.

`vsi-fmb-addr!` (*byte —*) stores the 8-bit data *byte* in the Message Broadcast Address Register.

`vsi-fmb-stat@` (*channel# — byte*) returns the contents – an 8-bit data – of the Message Broadcast Status Register identified by its channel number *channel#*. The value of *channel#* may be one of the values in the range zero to one. Each value specifies one of the two Message Broadcast Status Registers.

`vsi-fmb-stat!` (*byte channel# —*) stores the 8-bit data *byte* in the Message Broadcast Status Register identified by its channel number *channel#*. The value of *channel#* may be one of the values in the range zero to one. Each value specifies one of the two Message Broadcast Status Registers.

`vsi-fmb-msg@` (*channel# — long* `true`| `false`) returns the contents – a 32-bit data – of the Message Broadcast Register identified by its channel number *channel#*. The value of *channel#* may be one of the values in the range zero to one. Each value specifies one of the two Message Broadcast Registers.

`vsi-gcsr@` (*— byte*) returns the contents – an 8-bit data – of the Global Control and Status Register.

`vsi-gcsr!` (*byte —*) stores the 8-bit data *byte* in the Global Control and Status Register.

`vsi-mcsr0@` (*— byte*) returns the contents – an 8-bit data – of the Miscellaneous Control and Status Register 0.

`vsi-mcsr0!` (*byte —*) stores the 8-bit data *byte* in the Miscellaneous Control and Status Register 0.

20-47 18-2 2000 146 420 000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vsi-mcsr1@` (— *byte*) returns the contents – an 8-bit data – of the Miscellaneous Control and Status Register 1.

`vsi-mcsr1!` (*byte* —) stores the 8-bit data *byte* in the Miscellaneous Control and Status Register 1.

`vsi-wdt-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the Watchdog Timer Control Register.

`vsi-wdt-ctrl!` (*byte* —) stores the 8-bit data *byte* in the Watchdog Timer Control Register.

`vsi-wdt-restart@` (— *byte*) returns the contents – an 8-bit data – of the Watchdog Restart Register.

`vsi-wdt-restart!` (*byte* —) stores the 8-bit data *byte* in the Watchdog Restart Register.

`vsi-dma-ctrl@` (— *word*) returns the contents – a 16-bit data – of the DMA Control Register.

`vsi-dma-ctrl!` (*word* —) stores the 16-bit data *word* in the DMA Control Register.

`vsi-dma-mode@` (— *byte*) returns the contents – an 8-bit data – of the DMA Mode Register.

`vsi-dma-mode!` (*byte* —) stores the 8-bit data *byte* in the DMA Mode Register.

`vsi-dma-stat@` (— *byte*) returns the contents – an 8-bit data – of the DMA Status Register.

`vsi-dma-stat!` (*byte* —) stores the 8-bit data *byte* in the DMA Status Register.

`vsi-dma-src@` (— *long*) returns the contents – a 32-bit data – of the DMA Source Address Register.

`vsi-dma-src!` (*long* —) stores the 32-bit data *long* in the DMA Source Address Register.

`vsi-dma-dest@` (— *long*) returns the contents – a 32-bit data – of the DMA Destination Address Register.

`vsi-dma-dest!` (*long* —) stores the 32-bit data *long* in the DMA Destination Address Register.

`vsi-dma-cap@` (— *long*) returns the contents – a 32-bit data – of the DMA Capability and Transfer Count Register.

`vsi-dma-cap!` (*long* —) stores the 32-bit data *long* in the DMA Capability and Transfer Count Register.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vsi-ibox-ctrl@` (— *word*) returns the contents – a 16-bit data – of the Interrupt Box Control Register.

`vsi-ibox-ctrl!` (*word* —) stores the 16-bit data *word* in the Interrupt Box Control Register.

`vsi-ibox-addr@` (— *word*) returns the contents – a 16-bit data – of the Interrupt Box Address Register.

`vsi-ibox-addr!` (*word* —) stores the 16-bit data *word* in the Interrupt Box Address Register.

5.2.3.2 System Configuration Registers

The following commands are available to read data from and store data in the System Configuration Registers.

`led1-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the SYS LED Control Register.

`led1-ctrl!` (*byte* —) stores the 8-bit data *byte* in the SYS LED Control Register.

`led2-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the User LED Control Register.

`led2-ctrl!` (*byte* —) stores the 8-bit data in the User LED Control Register.

`flash-ctrl1@` (— *byte*) returns the contents – an 8-bit data – of the Flash Memory Control Register 1.

`flash-ctrl1!` (*byte* —) stores the 8-bit data in the Flash Memory Control Register 1.

`ni1-csr!` (*byte* —) stores the 8-bit data *byte* in the Network Interface1 Control and Status Register.

`ni2-csr@` (— *byte*) returns the contents – an 8-bit data – of the Network Interface 2 Control and Status Register.

`xchg-macio-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the Exchange MACIO Control Register.

`xchg-macio-ctrl!` (*byte* —) stores the 8-bit data *byte* in the Exchange MACIO Control Register.

`rotary-switch-stat@` (— *byte*) returns the contents – an 8-bit data – of the Rotary Switch Status Register.

20-47.18-2.2/2000.146 420.000/1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`boot-rom-size-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the Boot ROM Size Control Register.

`boot-rom-size-ctrl!` (*byte* —) stores the 8-bit data *byte* in the Boot ROM Size Control Register.

`flash-ctrl2@` (— *byte*) returns the contents – an 8-bit data – of the Flash Memory Control Register 2.

`flash-ctrl2!` (*byte* —) stores the 8-bit data in the Flash Memory Control Register 2.

`flash-vpp-ctrl@` (— *byte*) returns the contents – an 8-bit data – of the Flash Memory Programming Voltage Control Register.

`flash-vpp-ctrl!` (*byte* —) stores the 8-bit data *byte* in the Flash Memory Programming Voltage Control Register.

`led-display@` (— *byte*) returns the contents – an 8-bit data – of the LED Display Control/Status Register. Because the LED Display Control Register is only writable, the command returns the contents of the LED Display Control Shadow Register.

`led-display!` (*byte* —) stores the 8-bit data *byte* in the LED Display Control/Status Register. Because the LED Display Control Register is only writable, the command stores the given data in the LED Display Control Shadow Register, too.

`fmb-0-data-discard@` (— *byte*) returns the contents – an 8-bit data – of the FMB Channel 0 Data Discard Status Register.

`fmb-1-data-discard@` (— *byte*) returns the contents – an 8-bit data – of the FMB Channel 1 Data Discard Status Register.

`lca-id@` (— *byte*) returns the contents – an 8-bit data – of the LCA ID Register.

5.2.4 VMEbus Interrupt Handler

`vme-intr-pending?` (*level* — `true`| `false`) checks whether an interrupt is pending on a given interrupt request *level* and returns a flag. When an interrupt is pending the flag is `true`; otherwise it is `false`. The value of *level* may be one of the values in the range one through seven. Each value specifies one of the seven VMEbus interrupt request levels.

Only the least significant three bits of *level* are considered and when *level* is zero then the command treats it as if the value “one” has been passed to the command.

The command verifies the state of the interrupt pending bit

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

in the Interrupt Status register associated with the given *level*. When the corresponding status bit is set then no VMEbus interrupt is pending and the command returns `false`. Otherwise — the status bit is cleared — the value `true` is returned.

`vme-iack@` (*level* — *vector*) initiates an interrupt acknowledge cycle at the given VMEbus interrupt request *level* and returns the obtained 8-bit *vector*. The value of *level* may be one of the values in the range one through seven. Each value specifies one of the seven VMEbus interrupt request levels.
Typically, the *vector* returned is within the range 0 through 255, but when no interrupt is pending, and therefore no interrupt has to be acknowledged, the value -1 is returned.

Only the least significant three bits of *level* are considered and when *level* is zero then the command treats it as if the value “one” has been passed to the command.

Table 47 Interrupt mapping

Mapping	Constant	Interrupt generated by SPARC FGA-5000
0	vsi-nmi	INT (connected with *non-maskable* interrupt)
1	vsi-sbus-irq-1	SINT1 (connected with SBus IRQ1)
2	vsi-sbus-irq-2	SINT2 (connected with SBus IRQ2)
3	vsi-sbus-irq-3	SINT3 (connected with SBus IRQ3)
4	vsi-sbus-irq-4	SINT4 (connected with SBus IRQ4)
5	vsi-sbus-irq-5	SINT5 (connected with SBus IRQ5)
6	vsi-sbus-irq-6	SINT6 (connected with SBus IRQ6)
7	vsi-sbus-irq-7	SINT7 (connected with SBus IRQ7)

IMPORTANT



The words listed in the second column of the table may be used to specify a valid interrupt mapping.

`install-vme-intr-handler` (*mapping level* —) installs the interrupt service routine dealing with the given VMEbus interrupt *level*. The parameter *mapping* defines the interrupt asserted by the SPARC FGA-5000 when the certain VMEbus interrupt request level is asserted. The value of *mapping* may be one of the values in the range zero through seven. Each value specifies one of the eight SPARC FGA-5000 interrupt request lines. Table “Interrupt mapping” on page 96 lists all allowed mappings.

20 47 18 2 20000146 420 000 1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The value of *level* may be one of the values in the range one through seven. Each value specifies one of the seven VMEbus interrupt request levels. The address of the interrupt service routine currently in effect is preserved.

Only the least significant three bits of *mapping* and *level* are considered. When *level* is zero then the command treats it as if the value “one” has been passed to the command.

`uninstall-vme-intr-handler` (*level* —) removes the interrupt service routine dealing with the given VMEbus interrupt *level* and installs the *old* interrupt service routine. The value of *level* may be one of the values in the range one through seven. Each value specifies one of the seven VMEbus interrupt request levels.

Only the least significant three bits of *level* are considered and when *level* is zero then the command treats it as if the value “one” has been passed to the command.

`.vme-vectors` (–) displays the VMEbus interrupt vectors received during the last interrupt acknowledge cycle.

OpenBoot maintains seven variables called `vme-intr{1|2|3|4|5|6|7}-vector` which are modified by the VMEbus interrupt handlers. In general, the interrupt handlers store the vector obtained during an interrupt acknowledge cycle in the appropriate variable.

5.2.5 VMEbus Arbiter

The commands listed below are available to control the arbiter:

`vme-arb-mode@` (— *mode*) returns the *mode* the arbiter is currently operating in. The value of *mode* may range from zero to three. Each value specifies a particular mode: the values zero and three indicate that the arbiter is operating in the *priority* mode; the value one specifies the *round-robin* mode; and the value two specifies the *prioritized-round-robin* mode.

Three constants are available to specify one of the three bus arbitration modes: **`pri`** — prioritized — (3_{10}), **`rrs`** — round robin select— (1_{10}), **`prr`** — prioritized round robin — (6_{10}).

`vme-arb-mode!` (*mode* —) selects the arbiter mode specified by *mode*. The value of *mode* may range from zero to three. Each value specifies a particular mode: the values zero and three indicate that the arbiter operates in the *priority* mode; the value one specifies the *round-robin* mode; and the value two specifies the *prioritized-round-robin* mode.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`set-arb-mode` (*addr length* —) sets the arbiter mode according to the contents of the string representation of the arbiter mode. The string is specified by its address *addr* and its length *length*. The string may contain the following:.

String representation	Mode
“rrs”	Round Robin Select
“pri”	Prioritized
“prr”	Prioritized Round Robin Select

`.vsi-arb-ctrl` (—) displays the current contents of the VMEbus Arbiter Control Register.

5.2.6 VMEbus Requester

The commands listed below are available to control the requester and to obtain some information about the requesters’s operational state:

`set-rel-mode` (*addr length* —) sets the release mode according to the contents of the string representation of the release mode. The string is specified by its address *addr* and its length *length*. The string may contain the following:

String representation	Mode
“ ror”	Release On Request
“ roc”	Release On Bus Clear
“ rat”	Release After Timeout
“ rwd”	Release When Done

Any combination of the three release modes ROR, ROC, and RAT may be specified. In this case the release modes in the string must be separated by a comma, as described below:

```
ok "roc,rat" set-rel-mode
```

`.vsi-req-ctrl` (—) displays the current contents of the VMEbus Requester Control Register.

`vme-bus-capture!` (`true`|`false` —) enables or disables the bus-capture-and-hold capability of the SPARC FGA-5000. If the value `true` is passed to the command the VMEbus Interface starts to capture the bus and when it gains the ownership of the bus it holds the as long as the bus is released. The bus is released when the command is called and the value `false` is passed to it.

20-718-2 2.2000.146 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vme-bus-captured?` (— `true`|`false`) determines whether the VMEbus interface gains the ownership of the bus. The value `true` is returned when the VMEbus interface gains the ownership of the VMEbus. Otherwise the value `false` is returned to indicated that the VMEbus interface has not gained the ownership of the bus.
In general this command is called immediately after a capture-and-hold cycle has been initiated as shown in the example below:

```
ok true vme-bus-capture!
ok begin vme-bus-captured? until
ok …
ok …
ok false vme-bus-capture!
ok
```

5.2.7 VMEbus Status Signals

The commands listed below are available to access and control the VMEbus status signals.

`vme-sysfail-set` (—) asserts (sets) the VMEbus `SYSFAIL*` signal.

`vme-sysfail-clear` (—) negates (clears) the VMEbus `SYSFAIL*` signal.

`vme-sysfail?` (— `true`|`false`) determines the state of the VMEbus `SYSFAIL*` signal and returns a flag set according to the signal's state. When the `SYSFAIL*` signal is asserted the flag returned is `true`; otherwise its value is `false`.

`vme-acfail?` (— `true`|`false`) determines the state of the VMEbus `ACFAIL*` signal and returns a flag set according to the signal's state. When the `ACFAIL*` signal is asserted the flag returned is `true`; otherwise it is `false`.

5.2.8 VMEbus Master Interface

The SPARC FGA-5000 provides 16 sets of registers to control any VMEbus master operation. Each set may be used to address a certain address range within the VMEbus' address space. A register set is identified by a unique number, the *range number* (*range#*), in the range zero through 15. When the VMEbus is being accessed the part of the SPARC FGA-5000 connected with the SBus is considered as the SBus slave device, whereas the part of the SPARC FGA-5000 that is connected with the VMEbus is operating as VMEbus master. This fact is reflected in the names of the commands available to control the VMEbus master interface.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.2.8.1 Initializing and Controlling the VMEbus Master Interface

The commands listed and described in the following are available to initialize and control the VMEbus master interface:

`#vme-ranges` (— *#vme-ranges*) returns the number *#vme-ranges* of available register sets which are used to control accesses to the VMEbus.

`vme-master-ena` (*range#* —) enables the address decoding associated with the range number *range#* to access the VMEbus. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation.

`vme-master-dis` (*range#* —) enables the address decoding associated with the range number *range#* to access the VMEbus. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation.

`vme-master-wp-ena` (*range#* —) enables *write posting* within the VMEbus address range associated with the range number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation.

`vme-master-wp-dis` (*range#* —) disables *write posting* within the VMEbus address range associated with the range number *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation.

`vme-supervisor!` (`true`|`false` —) selects the mode in which the VMEbus is being accessed. When the value `true` is passed to the command, the VMEbus is accessed in the *privileged* mode. Otherwise – the value `false` is passed to the command – the VMEbus is accessed in the *non-privileged* mode.
The mode selected with this command applies to **all** ranges used to access the VMEbus.

`sbus-slave-range@` (*range# — offset sbus-slot# size*) returns the SBus slave parameters associated with the range identified by *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation. The parameters returned by the command specify the SBus address range to be accessed to reach the VMEbus. The address range is represented by the triple *offset*, *sbus-slot#*, and *size*.

`sbus-slave-range!` (*offset sbus-slot# size range#* —) sets the SBus slave parameters associated with the range identified by *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation.

20-47 SE 2-2000.0146 420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The parameters passed to the command specify the SBus address range to be accessed to reach the VMEbus. The address range is represented by the triple *offset*, *sbus-slot#*, and *size*.

`vme-master-range@` (*range# — addr data-capability address-capability size*) returns the VMEbus master capabilities associated with the range number identified by *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation.
The VMEbus address range being accessed is represented by the *addr-size* pair, where *addr* specifies the physical VMEbus address and *size* identifies the address range covered. The value of *data-capability* and *address-capability* may be one of the values listed in the table below.

`vme-master-range!` (*addr data-capability address-capability size range# —*) sets the VMEbus master capabilities associated with the range number identified by *range#*. The value of *range#* may be one of the values in the range zero through 15. Each value specifies one of the 16 register sets controlling any VMEbus master operation.
The VMEbus address range being accessed is represented by the *addr-size* pair, where *addr* specifies the physical VMEbus address and *size* identifies the address range covered. The value of *data-capability* and *address-capability* may be one of the values listed in the table below:

Value	Data-capability	Address-capability
000_2	cap-d8	cap-a16
001_2	cap-d16	cap-a24
010_2	cap-d32	cap-a32
011_2	cap-blt	reserved
100_2	cap-mblt	cap-a64
101_2	reserved	reserved
110_2	reserved	reserved
111_2	reserved	reserved

`.vme-master-ranges` (—) displays the current settings of all register sets of the VMEbus master interface.

`.vme-cap` (—) displays the contents of the VMEbus Capability Register.

`vme-master-map` (*range# — vaddr*) makes the physical address range, as defined by the contents of the range register set specified by the range number *range#*,

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

available to the processor's virtual address space and returns the virtual address *vaddr*.
Because the command obtains all information from the specific range register set, the particular range register must be initialized before.

`vme-master-unmap` (*vaddr range#* —) removes the physical address range, as defined by the contents of the range register set specified by the range number *range#*, from the processor's virtual address space.

5.2.8.2 Examples: Accessing Address Spaces

The examples given in this section describe how to initialize the VMEbus interface for subsequent VMEbus master accesses.

Assumed the first three register sets have been used to access the VMEbus address spaces as described in the examples below, the following command may be used to display the settings of the registers sets:

```
ok .vme-master-ranges
```

Extended address space

Example 1: Access to Extended Address Space

The first example shows how to access a 1 MByte area within the extended address space (A32) of the VMEbus beginning at address 4080.0000_{16}. The register set associated with the range number zero (*range#* is `0`) is used to access the VMEbus area mentioned above.

```
ok h# 4080.0000 cap-d32 cap-a32 1Meg 2 * 0 vme-master-range!
ok 1Meg d# 10 * 1 1Meg 2 * 0 sbus-slave-range!
ok 0 vme-master-ena
ok 0 vme-master-map value vmebus
ok
```

1st command — The first command initializes the VMEbus master interface. It sets the data- and address capabilities, as well as the VMEbus address and the size of the area being accessed. The *data capability* is defined using the predefined constant `cap-d32` which enables the VMEbus master interface to access bytes (8bit data), half-words (16bit data), and words (32-bit data) within the VMEbus area. The *address capability* is defined using the predefined constant `cap-a32` that enables the VMEbus interface to access the extended address space (A32) of the VMEbus.

2nd command — The SBus slave interface is initialized by the second command which specifies that the VMEbus is accessed when the SBus slot one (1) is being accessed at offsets $A0.0000_{16}$ to $BF.FFFF_{16}$ which corresponds to the VMEbus addresses in the range 4080.0000_{16} to $409F.FFFF_{16}$ of the extended address space (A32).

3rd command — The third command enables any access to the VMEbus.

20-718-2 2.2000.146 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

4th command

The fourth command maps the physical address area to be accessed in order to address the VMEbus to the virtual address space of the processor and stores the virtual address in the variable `vmebus`. This variable may be used to access the VMEbus area using the commands to read and write data provided by OpenBoot.

Translation no longer used

When the translation (SBus to VMEbus) defined by the contents of the register set associated with the range number zero is no longer used, then the memory mapped to the processor's virtual address space to access the VMEbus must be released. This has to be done before the contents of this register set are modified using the command `vme-master-unmap`:

```
ok vmebus 0 vme-master-unmap
ok
```

Standard address space

Example 2: Access to Standard Address Space

In the second example the VMEbus interface is initialized to allow accesses to the standard address space (A24) of the VMEbus beginning at address 98.0000_{16}. The size of this area is 512 Kbyte and the register set associated with the range number one (*range#* is `1`) is used to access this VMEbus area.

```
ok h# 98.0000 cap-d16 cap-a24 1Meg 2 / 1 vme-master-range!
ok 1Meg d# 18 * 2 1Meg 2 / 1 sbus-slave-range!
ok 1 vme-master-ena
ok 1 vme-master-map value vmebus
ok
```

1st command

The first command initializes the VMEbus master interface. It sets the data- and address capabilities, as well as the VMEbus address and the size of the area being accessed. The *data capability* is defined using the predefined constant `cap-d16` which enables the VMEbus master interface to access bytes (8bit data) and half-words (16bit data) within the VMEbus area. The *address capability* is defined using the predefined constant `cap-a24` that enables the VMEbus interface to access the standard address space (A24) of the VMEbus.

2nd command

The SBus slave interface is initialized by the second command which specifies that the VMEbus is accessed when the SBus slot two (2) is being accessed at offsets 120.0000_{16} to $127.FFFF_{16}$ which corresponds to the VMEbus addresses in the range 98.0000_{16} to $9F.FFFF_{16}$ of the standard address space (A24).

3rd command

The third command enables any access to the VMEbus.

4th command

The fourth command maps the physical address area to be accessed in order to address the VMEbus to the virtual address space of the processor and stores the virtual address in the variable `vmebus`. This variable may be used to access the VMEbus area using the commands to read and write data provided by OpenBoot.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Translation no longer used

When the translation (SBus to VMEbus) defined by the contents of the register set associated with the range number one is no longer used, then the memory mapped to the processor's virtual address space to access the VMEbus must be released. This has to be done before the contents of this register set are modified using the command `vme-master-unmap`:

```
ok 1 vme-master-unmap
ok
```

Short address space

Example 3: Access to Short Address Space

The third example describes how to initialize the VMEbus interface to allow accesses to the short address space (A16) of the VMEbus beginning at address 0000_{16}. The size of this area is 64 Kbyte and therefore covers the entire short address space. The register set associated with the range number two (*range#* is `2`) is used to access this VMEbus area.

```
ok h# 0000 cap-d8 cap-a16 h# 1.0000 2 vme-master-range!
ok 1Meg d# 64 * 3 h# 1.0000 2 sbus-slave-range!
ok 2 vme-master-ena
ok 2 vme-master-map value vmebus
ok
```

1st command

The first command initializes the VMEbus master interface. It sets the data- and address capabilities, as well as the VMEbus address and the size of the area being accessed. The *data capability* is defined using the predefined constant `cap-d8` which limits the VMEbus master interface to access only bytes (8bit data) within the VMEbus area. The *address capability* is defined using the predefined constant `cap-a16` that enables the VMEbus interface to access the short address space (A16) of the VMEbus.

2nd command

The SBus slave interface is initialized by the second command which specifies that the VMEbus is accessed when the SBus slot three (3) is being accessed at offsets 400.0000_{16} to $400.FFFF_{16}$ which corresponds to the VMEbus addresses in the range 0000_{16} to $FFFF_{16}$ of the short address space (A16).

3rd command

The third command enables any access to the VMEbus.

4th command

The fourth command maps the physical address area to be accessed in order to address the VMEbus to the virtual address space of the processor and stores the virtual address in the variable `vmebus`. This variable may be used to access the VMEbus area using the commands to read and write data provided by OpenBoot.

Translation no longer used

When the translation (SBus to VMEbus) defined by the contents of the register set associated with the range number two is no longer used, then the memory mapped to the processor's virtual address space to access the VMEbus must be released. This has to be done before the contents of this register set are modified using the command `vme-master-unmap`:

20-718-2 2.2000.146 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ok 2 vme-master-unmap
ok
```

5.2.8.3 Controlling the SPARC FGA-5000 SBus Interface

The following commands are available to control the various operating modes of the SPARC FGA-5000 SBus interface.

`sbus-rerun!` (`true`|`false`) enables or disables the SPARC FGA-5000's capability to generate reruns on the SBus. When the value `true` is passed to the command the SPARC FGA-5000 will initiate SBus rerun if necessary. Otherwise –the value `false` is passed to the command – the SPARC FGA-5000's capability to initiate SBus reruns is disabled.

`sbus-rerun-limit@` (— *#rerun-limit*) returns the number of reruns before the SPARC FGA-5000 terminates an SBus cycle with an error. The value of *#rerun-limit* is in the range zero through 255 and specifies the number of reruns.

`sbus-rerun-limit!` (*#rerun-limit* —) sets the number of reruns before the SPARC FGA-5000 terminates an SBus cycle with an error. The value of *#rerun-limit* may be in the range zero through 255 and specifies the number of reruns. The command treats the value of *#rerun-limit* as a modulo 256 number.
When the command is called it verifies whether the given number of reruns falls below the limit specified by `min-rerun-limit`. If this value falls below the given limit, then the commands uses the value of `min-rerun-limit` instead. This ensures that the SBus interface is operating properly.

`sbus-split-ena` (—) enables the SPARC FGA-5000 capability to split SBus cycles. `sbus-split-dis` disables the SPARC FGA-5000 capability to split SBus cycles.

`.sbus-cap` (—) displays the current contents of the SBus Master Capability Register as shown below:

```
ok .sbus-cap
Split: 1   Split Flow: 1   Arbiter: 1   Burst: 1
Master Read Stop Point: 32 bytes  Max. Burst Length: 32
bytes
ok
```

`.sbus-retry-time-ctrl` (—) displays the current contents of the SPARC FGA-5000's SBus Retry Time Control Register as stated below:

```
ok .sbus-retry-time-ctrl
Retry time: 10
ok
```

`.sbus-rerun-limit-ctrl` (—) displays the current contents of the SBus Rerun Limit Control Register as depicted below:

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ok .sbus-rerun-limit-ctrl
Enable Reruns: 0   Rerun limit: 255
ok
```

5.2.9 VMEbus Slave Interface

The SPARC FGA-5000 provides three sets of registers to control any VMEbus slave access. Each set may be used to make a certain slave address range – *standard*- (A24) or *extended* (A32) slave address range – available to the VMEbus' address space. A register set is identified by a unique number, the *range number* (*range#*), in the range zero through two.
Only the A24 and A32 slave mode allows a VMEbus master to access the memory of the SPARC/CPU-8VT. In the A16 slave mode all VMEbus master accesses are limited to the registers of the SPARC FGA-5000 which are accessible from the VMEbus.

When the VMEbus interface is being accessed from the VMEbus, then the part of the SPARC FGA-5000 connected with the VMEbus is considered as **VMEbus slave** device. Whereas the part of the SPARC FGA-5000 that is connected with the SBus is operating as the **SBus master**. This fact is reflected in the names of the commands available to control the VMEbus master interface.

The commands listed and described in the following are available to initialize and control the A16 VMEbus slave interface:

`vme-a16-slave-ena` (—) enables the capability to access the SPARC FGA-5000 registers from the VMEbus in the *short* address space (A16).

`vme-a16-slave-dis` (—) disables the capability to access the SPARC FGA-5000 registers from the VMEbus in the *short* address space (A16).

`vme-a16-slave-addr@` (— *addr*) returns the 16-bit address *addr* at which the registers of the SPARC FGA-5000 are accessible within the *short* address space (A16).

`vme-a16-slave-addr!` (*addr* —) defines the 16-bit address *addr* at which the registers of the SPARC FGA-5000 are accessible within the *short* address space (A16).
The least significant nine bits of the address *addr* are ignored by the command – the command treats them as if they are cleared –, because the SPARC FGA-5000 is only accessible from the VMEbus beginning at 512-Byte boundaries.
The commands listed and described in the following are available to initialize and control the A24 and A32 VMEbus slave interface:

20-4718-2-2000146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vme-slave-ena` (*range#* —) enables the address decoding associated with the range number *range#* to allow accesses from the VMEbus. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

`vme-slave-dis` (*range#* —) disables the address decoding associated with the range number *range#* to allow accesses from the VMEbus. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

`vme-slave-wp-ena` (*range#* —) enables *write posting* within the VMEbus slave address range associated with the range number *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

`vme-slave-wp-dis` (*range#* —) disables *write posting* within the VMEbus slave address range associated with the range number *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

`sbus-a24-master-range@` (*range# — vaddr size*) returns the SBus master parameters associated with the A24 slave interface identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.
The parameters *vaddr* and *size* identify the virtual address range within the SBus, into which all A24 slave accesses are translated.

`sbus-a24-master-range!` (*vaddr size range#* —) defines the SBus master parameters associated with the A24 slave interface identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.
The parameters *vaddr* and *size* identify the virtual address range within the SBus, into which all A24 slave accesses are translated.

`vme-a24-slave-range@` (*range# — paddr size*) returns the VMEbus base address *paddr* and the size *size* of the A24 slave window associated with the range identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

`vme-a24-slave-range!` (*paddr size range#* —) sets the VMEbus base address *paddr* and the size *size* of the A24 slave window associated with the range identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`sbus-a32-master-range@` (*range# — vaddr size*) returns the SBus master parameters associated with the A32 slave interface identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.
The parameters *vaddr* and *size* identify the virtual address range within the SBus, into which all A32 slave accesses are translated.

`sbus-a32-master-range!` (*vaddr size range# —*) defines the SBus master parameters associated with the A32 slave interface identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.
The parameters *vaddr* and *size* identify the virtual address range within the SBus, into which all A32 slave accesses are translated.

`vme-a32-slave-range@` (*range# — paddr size*) returns the VMEbus base address *paddr* and the size *size* of the A32 slave window associated with the range identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

`vme-a32-slave-range!` (*paddr size range# —*) sets the VMEbus base address *paddr* and the size *size* of the A32 slave window associated with the range identified by *range#*. The value of *range#* may be one of the values in the range zero through two. Each value specifies one of the three register sets controlling any VMEbus slave access.

`.vme-slave-ranges` (—) displays the current settings of **all** register sets controlling any VMEbus slave access.

The following example lists all steps to be taken to initialize the VMEbus interface for A32 accesses from the VMEbus beginning at address 2340.0000_{16} and ranging to $235F.FFFF_{16}$. The register set associated with the *range number* zero (0) is used to control this particular VMEbus slave interface.

```
ok h# 2340.0000 1Meg 2 * 0 vme-a32-slave-range!
ok h# ffe0.0000 1Meg 2 * 0 sbus-a32-master-range!
ok h# 10.0000 obmem h# ffe0.0000 1Meg 2 * iomap-pages
ok 0 vme-slave-ena
ok
```

As shown above the first command defines the VMEbus slave interface's base address and size of the slave window. The second command defines that any A32 access is translated to an access of the SBus beginning at SBus address $FFE0.0000_{16}$. And the third command creates all necessary entries within the IOMMU to translate the SBus access to an access of the on-board memory beginning at physical address 10.0000_{16} .

20-4718-2-2000146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Finally, the VMEbus slave interface is enabled using the fourth command.

The next example list all steps to be taken, to initialize the VMEbus interface for A24 accesses from the VMEbus beginning at address $C0.0000_{16}$ and ranging to $CF.FFFF_{16}$. The register set associated with the *range number* one (1) is used to control this particular VMEbus slave interface.

```
ok h# c0.0000 1Meg 1 vme-a24-slave-range!
ok h# fff0.0000 1Meg 1 sbus-a24-master-range!
ok h# 20.0000 obmem h# fff0.0000 1Meg iomap-pages
ok 1 vme-slave-ena
ok
```

As shown above the first command defines the VMEbus slave interface's base address and size of the slave window. The second command defines that any A24 access is translated to an access of the SBus beginning at SBus address $FFF0.0000_{16}$. And the third command creates all necessary entries within the IOMMU to translate the SBus access to an access of the on-board memory beginning at physical address 20.0000_{16}.
Finally, the VMEbus slave interface is enabled using the fourth command.

5.2.10 VMEbus Device Node

The OpenBoot device tree contains the device node for the VMEbus interface and is called "**VME**". It is a *child device* of the device node "**/iommu/sbus**" (The full pathname of the VMEbus interface device node is displayed by the command **show-devs**). The device *alias* **vme** is available as a shorthand representation of the VMEbus interface device-path.

The vocabulary of the VMEbus device includes the standard commands recommended for a *hierarchical* device. The words of this vocabulary are only available when the VMEbus device has been selected as shown below:

```
ok cd vme
ok words
selftest    reset    close    open ...
... list of further methods of the device node
ok selftest .
0
ok device-end
ok
```

The example listed above, selects the VMEbus device and makes it the current node. The word **words** displays the names of the *methods* of the VMEbus device. And the third command calls the method **selftest** and the value return by this method is displayed. The last command *unselects* the current device node, leaving no node selected.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The following methods are defined in the vocabulary of the VMEbus device:

`open` (— `true`) prepares the package for subsequent use. The value `true` is always returned.

`close` (—) frees all resources allocated by open.

`reset` (—) puts the VMEbus Interface into *quiet* state.

`selftest` (— *error-number*) performs a test of the VMEbus interface, and returns an *error-number* to report the course of the test. In the case that the device has been tested successfully the value zero is returned; otherwise it returns a specific error number to indicate a certain fail state.

`decode-unit` (*addr len — low high*) converts the *addr* and *len*, a text string representation, to *low* and *high* which is a numerical representation of a physical address within the address space defined by the package.

`map-in` (*low high size — vaddr*) creates a mapping associating the range of physical address beginning at *low*, extending for *size* bytes, within the package's physical address space, with a processor virtual address *vaddr*.

`map-out` (*vaddr size* —) destroys the mapping set by map-in at the given virtual address *vaddr* of length *size*.

`dma-alloc` (*size — vaddr*) allocates a virtual address range of length *size* bytes that is suitable for direct memory access by a bus master device. The memory is allocated according to the most stringent alignment requirements for the bus. The address of the acquired virtual memory *vaddr* is returned via the stack.

`dma-free` (*vaddr size* —) releases a given virtual memory, identified by its address *vaddr* and *size*, previously acquired by `dma-alloc`.

`dma-map-in` (*vaddr size cachable? — devaddr*) converts a given virtual address range, specified by *vaddr* and *size*, into an address *devaddr* suitable for direct memory access on the bus. The virtual memory must be allocated already by `dma-alloc`. The SPARC/CPU-8VT does not support caching. Thus the *cachable?* flag is ignored.

`dma-map-out` (*vaddr devaddr size* —) removes the direct memory access mapping previously created by `dma-map-in`.

`dma-sync` (*vaddr devaddr size* —) synchronizes memory caches associated with a given direct memory access mapping, specified by its virtual address *vaddr*, the *devaddr* and its *size* that has been established by dma-map-in.

20-4718-2-2000146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.2.11 VMEbus NVRAM Configuration Parameters

The NVRAM configuration parameters listed below are available to control the initialisation and operation of the VMEbus Interface. The current state of these configuration parameters are displayed using the `printenv` command, and are modified using either the `setenv`, or the `setdefault` command provided by OpenBoot.

`vme-sysfail-clear?` when the value of the configuration parameter is `true` the SYSFAIL* signal will be cleared by OpenBoot. In the case that the configuration parameter is `false` OpenBoot will not clear the SYSFAIL* signal, but the operating system which is loaded has to clear it. (default: `true`)

The state of this NVRAM configuration parameter is considered independent from the state of the `vme-init?` configuration parameter.

`vme-bus-timer?` controls whether the VMEbus transaction timer in the SPARC FGA-5000 is used to watch each VMEbus access. When the flag is `true` the transaction timer is enabled. If the flag is `false` the transaction timer is disabled. (default: `true`)
The state of this NVRAM configuration parameter is considered independent from the state of the `vme-init?` configuration parameter.

`vme-bus-timeout` contains the timeout value of the SPARC FGA-5000 VMEbus transaction timer and is a value in the range one to three. Each value selects a particular timeout period. Independent of the state of the configuration parameter `vme-bus-timer?` the timeout value is stored in the appropriate register. When the value of this configuration parameter is not in the range one through three, then the value three is used instead. (default: 3_{10})

The state of this NVRAM configuration parameter is considered independent from the state of the `vme-init?` configuration parameter.

`vme-slot#` specifies the *logical* VMEbus slot number assigned to the SPARC/CPU-8VT board. The values may be in the range one through 255, but preferably should be set in such a way that it corresponds with the number of an available VMEbus slot.
The state of this configuration parameter does not control whether the VMEbus interface is operating as system controller when the configuration parameter's value is one. (default: 1_{10})

`vme-fair-req?` specifies whether the VMEbus requester operates in the *fair* mode when requesting the VMEbus. When the value of the configuration parameter is `true`, the VMEbus requester operates in the *fair* mode. Otherwise – the value of the configuration parameter is `false` – the requester operates **not** in the fair mode. (default: `false`)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`vme-req-level` specifies the level on which the SPARC FGA-5000 requests the VMEbus. The value of this configuration parameter may be in the range zero through three. Each value corresponds directly with one of the four available bus request levels. (default: 3_{10}, BR3* request level)

`#sbus-burst-len` selects the maximum length of SBus burst transactions generated by the SPARC FGA-5000. The value of this configuration parameter may be 8, 16, 32, or 64 – corresponding with the SBus bursts of 8 bytes, 16 bytes, 32 bytes, and 64 bytes. When the value differs from the values listed above the SBus burst length is set to 32 bytes. (default: 32 bytes)

`#sbus-read-stop` selects the SBus "Read Stop Boundary". The value of this configuration parameter may be 8, 16, 32, or 64 – corresponding with the SBus read stop points of 8 bytes, 16 bytes, 32 bytes, and 64 bytes. When the value differs from the values listed above the SBus read stop point is set to 64 bytes. (default: 64 bytes)

`sbus-burst?` controls whether the SPARC FGA-5000 generates SBus burst transactions. When the configuration flag is `true`, then the SPARC FGA-5000 capability to generate SBus bursts is enabled. In the case that the configuration flag is `false` the SPARC FGA-5000 will carry out only single SBus transactions. (default: `true`)

`sbus-hidden-arb?` controls whether the SPARC FGA-5000 operates in the "Hidden SBus Arbitration" mode. When the configuration flag is `true`, then the SPARC FGA-5000 operates in the hidden arbitration mode. In the case that the configuration flag is `false` the SPARC FGA-5000 does not operate in the hidden arbitration mode. (default: `true`)

`sbus-split-flow?` controls whether the SPARC FGA-5000 operates in the "Flow Through" mode. When the configuration flag is `true`, then the SPARC FGA-5000 operates in the flow through mode. In the case that the configuration flag is `false` the SPARC FGA-5000 does not operate in the flow through mode. (default: `true`)

`#sbus-retry` selects the maximum number of SBus clocks before a retry occurs. The value of this configuration parameter may be in the range zero through 255. In the case that the value is below a certain limit – typically three SBus clocks – the SBus retry counter is prevented from being set below this limit. (default: 20 SBus clocks)

`vme-sgl-filter?` controls whether the strobe glitch filter for the VMEbus handshake signals are enabled. When the configuration flag is `true`, then the strobe glitch filters for these signals are enabled. In the case that the configuration flag is `false` the strobe glitch filters are disabled. (default: `false`)

`vme-as-slow?` controls whether the VMEbus AS handshake signal operates in the slow mode. When the configuration flag is `true`, then the AS handshaking

20-47 18-2-2000-146-420-000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

operates in the slow mode. In the case that the configuration flag is `false` AS handshaking operates in the fast mode. (default: `false`)

`vme-ds-slow?` controls whether the VMEbus DS handshake signal operates in the slow mode. When the configuration flag is `true`, then the DS handshaking operates in the slow mode. In the case that the configuration flag is `false` DS handshaking operates in the fast mode. (default: `false`)

`vme-arb-mode` select the arbitration mode of the VMEbus arbiter. This configuration string shall identify one of the following arbitration modes: "`pri`" (priority), "`rrs`" (round robin select), "`sgl`" (single level – not supported), or "`prr`" (priority round robin select). (default: `pri`)

`vme-rel-mode` select the release mode of the VMEbus requestor. This configuration string shall identify one of the following release modes: "`ror`" (release on request), "`roc`" (release on bus clear), "`rat`" (release after timeout), or "`rwd`" (release when done). (default: `ror`)
When the VMEbus interface requestor is capable of supporting more than one type of release mode, then the value may be as many of the release modes mentioned as are applicable, but separated by a comma. (For example "`ror,rat`")

`vme-early-rel?` controls whether the SPARC FGA-5000 releases the VMEbus at the beginning or end of the current transaction. When the configuration flag is `true`, then the SPARC FGA-5000 releases the bus at the beginning of the current transaction. In the case that the configuration flag is `false` the VMEbus is released at the end of the transaction. (default: `false`)

`vme-bbsy-filter?` controls whether the `BBSY*` glitch filter within the SPARC FGA-5000 is enabled or disabled. When the configuration flag is `true`, then the BBSY* glitch filter is enabled. In the case that the configuration flag is `false` the `BBSY*` glitch filter is disabled. (default: `false`)

`vme-init?` controls whether the VMEbus interface is initialized by OpenBoot. When this flag is `true` the VMEbus interface is initialized according to the state of the NVRAM parameter listed below. In the case that the flag is `false` the VMEbus interface is not initialized. The VMEbus interface is initialized after OpenBoot set up the main memory. (default: `true`)
The state of the NVRAM configuration parameters listed in the following are only considered by OpenBoot when the configuration parameter `vme-init? is true!`

`vme-intr1` controls whether the VMEbus interrupt request level 1 has to be enabled. When the value is 255 then the VMEbus interrupt request level 1 is not enabled. In the case that the value is within the range one to seven, the corresponding interrupt handler is activated and the VMEbus interrupt request level 1 is enabled. The values one to seven specify the SPARC

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FGA-5000 interrupt request line to be asserted when a VMEbus interrupt request level 1 occurs. (default: 255_{10})

`vme-intr2` controls whether the VMEbus interrupt request level 2 has to be enabled. When the value is 255 then the VMEbus interrupt request level 2 is not enabled. In the case that the value is within the range one to seven, the corresponding interrupt handler is activated and the VMEbus interrupt request level 2 is enabled. The values one to seven specify the SPARC FGA-5000 interrupt request line to be asserted when a VMEbus interrupt request level 2 occurs. (default: 255_{10})

`vme-intr3` controls whether the VMEbus interrupt request level 3 has to be enabled. When the value is 255 then the VMEbus interrupt request level 3 is not enabled. In the case that the value is within the range one to seven, the corresponding interrupt handler is activated and the VMEbus interrupt request level 3 is enabled. The values one to seven specify the SPARC FGA-5000 interrupt request line to be asserted when a VMEbus interrupt request level 3 occurs. (default: 255_{10})

`vme-intr4` controls whether the VMEbus interrupt request level 4 has to be enabled. When the value is 255 then the VMEbus interrupt request level 4 is not enabled. In the case that the value is within the range one to seven, the corresponding interrupt handler is activated and the VMEbus interrupt request level 4 is enabled. The values one to seven specify the SPARC FGA-5000 interrupt request line to be asserted when a VMEbus interrupt request level 4 occurs. (default: 255_{10})

`vme-intr5` controls whether the VMEbus interrupt request level 5 has to be enabled. When the value is 255 then the VMEbus interrupt request level 5 is not enabled. In the case that the value is within the range one to seven, the corresponding interrupt handler is activated and the VMEbus interrupt request level 5 is enabled. The values one to seven specify the SPARC FGA-5000 interrupt request line to be asserted when a VMEbus interrupt request level 5 occurs. (default: 255_{10})

`vme-intr6` controls whether the VMEbus interrupt request level 6 has to be enabled. When the value is 255 then the VMEbus interrupt request level 6 is not enabled. In the case that the value is within the range one to seven, the corresponding interrupt handler is activated and the VMEbus interrupt request level 6 is enabled. The values one to seven specify the SPARC FGA-5000 interrupt request line to be asserted when a VMEbus interrupt request level 6 occurs. (default: 255_{10})

`vme-intr7` controls whether the VMEbus interrupt request level 7 has to be enabled. When the value is 255 then the VMEbus interrupt request level 7 is not enabled. In the case that the value is within the range one to seven, the corresponding interrupt handler is activated and the VMEbus interrupt request level 7 is enabled. The values one to seven specify the SPARC

20-4718-2 2.000.0146 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FGA-5000 interrupt request line to be asserted when a VMEbus interrupt request level 7 occurs. (default: 255_{10})

`vme-sysfail-assert?` controls whether a non-maskable interrupt is generated upon the assertion of the VMEbus signal SYSFAIL*. When the flag is `true` an interrupt handler, dealing with this interrupt, is installed and the ability to generate a non-maskable interrupt upon the assertion of the SYSFAIL* signal is enabled. In the case that the flag is `false` the ability to generate a non-maskable interrupt upon the assertion of the SYSFAIL* signal is enabled. (default: `false`)

`vme-sysfail-negate?` controls whether a non-maskable interrupt is generated upon the negation of the VMEbus signal SYSFAIL*. When the flag is `true` an interrupt handler, dealing with this interrupt, is installed and the ability to generate a non-maskable interrupt upon the negation of the SYSFAIL* signal is enabled. In the case that the flag is `false` the ability to generate a non-maskable interrupt upon the negation of the SYSFAIL* signal is enabled. (default: `false`)

`vme-acfail-assert?` controls whether a non-maskable interrupt is generated upon the assertion of the VMEbus signal ACFAIL*. When the flag is `true` an interrupt handler, dealing with this interrupt, is installed and the ability to generate a non-maskable interrupt upon the assertion of the ACFAIL* signal is enabled. In the case that the flag is `false` the ability to generate a non-maskable interrupt upon the assertion of the ACFAIL* signal is enabled. (default: `false`)

`vme-ibox-addr` the least significant 16 bits of this 32-bit configuration parameter defines the address at which the interrupt box (`IBOX`) of the SPARC FGA-5000 is accessible within the *short* address space (A16). Only the least significant 16 bits of this configuration parameter are considered, and the state of the remaining bits are ignored. Independent of the configuration parameter `vme-ibox-ena?` OpenBoot will set the address of the IBOX. (default: 0_{16})

`vme-ibox-ena?` indicates whether the interrupt box (IBOX), accessible in the *short* (A16) address range of the VMEbus, should be enabled. When this NVRAM configuration parameter is `true` then the IBOX is enabled. In the case that the NVRAM configuration parameter is `false` the IBOX is not enabled.
The default value of this NVRAM configuration parameter is `false`.

`fmb-init?` controls whether the FMB system is initialized by OpenBoot. When this flag is `true` the FMB system is initialized according to the state of the NVRAM parameter listed below. In the case that the flag is `false` the FMB system is not initialized. The FMB system is initialized only during the initialisation of the VMEbus interface, which means that the `vme-`

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`init?` configuration parameter must be `true`, in order to set up the FMB system. (default: `true`)

`fmb-slot#` specifies the *logical* slot number assigned to the FMB channels of the SPARC/CPU-8VT board. The values may be in the range one through 21, and preferably should be set in such a way that it corresponds with the number of an available VMEbus slot. (default: 1_{10})

`fmb-addr` specifies the address – the most significant eight bits of a 32-bit address – where the FMB system resides in the *extended* address space (A32) of the VMEbus. (default: fa_{16})
The NVRAM configuration parameters listed below are associated with the slave interface accessible in the ***short*** (**A16**) address range.

`vme-a16-slave-addr` specifies the base address of the slave interface accessible in the *short* (A16) address range of the VMEbus.
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a16-slave-size` specifies the size of the memory which is made available to the *short* (A16) address range of the VMEbus. When the value of this configuration parameter is zero OpenBoot will not initialize the slave interface, even if the `vme-a16-slave-ena?` configuration parameter is `true`!
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a16-slave-ena?` indicates whether the slave interface, accessible in the *short* (A16) address range of the VMEbus, should be enabled. When this NVRAM configuration parameter is `true` then the VMEbus slave interface is enabled. In the case that the NVRAM configuration parameter is `false` the VMEbus slave interface is not enabled, and any attempt to access the VMEbus slave interface from the VMEbus will lead to an error termination on the VMEbus.
The default value of this NVRAM configuration parameter is `false`.

In the case that the NVRAM configuration parameter `vme-init?` is `true` the OpenBoot will initialize the slave interface according to the configuration parameters described above.
When the `vme-a16-slave-ena?` configuration parameter is `true`, then OpenBoot will initialize the VMEbus slave interface according to the NVRAM configuration parameters `vme-a16-slave-addr` and `vme-a16-slave-size`. It will provide the required amount of physical on-board memory and builds up the necessary MMU and IOMMU settings to make the memory available to the VMEbus. The *virtual* base address of the physical on-board memory provided for VMEbus slave accesses is stored in the variable `vme-a16-slave-mem`.
Thus, applications executed within the OpenBoot environment may benefit from this mechanism, because OpenBoot will initialize the slave in-

20-47 28 2-20000 146 420 000 1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

terface completely according to the NVRAM configuration parameters associated with the slave interface.

In addition, this mechanism allows to report the parameters of the slave interface to an operating system loaded, which in turn provides its own memory and the corresponding MMU and IOMMU settings. In this case the VMEbus device driver is responsible for the access to the slave interface from the VMEbus. In general, the configuration parameter `vme-a16-slave-ena?` must be set to `false` to prevent OpenBoot from initialising and enabling the slave interface when an operating system will be loaded. Supposed that the slave interface is initialized and enabled by OpenBoot prior to loading the operating system, any access from the VMEbus to the slave interface while loading the operating system may alter memory and cause severe damage.

IMPORTANT



The SPARC/CPU-8VT does not provide the ability to access its on-board memory from the VMEbus within the short (A16) address range. Therefore, the NVRAM configuration parameters associated with the A16 slave interface, control the access to the registers of the SPARC FGA-5000, which are accessible within the short address range. The configuration parameter `vme-a16-slave-size` is not of any importance and will be ignored.

The NVRAM configuration parameters listed below are associated with the slave interface accessible in the standard (A24) address range.

`vme-a24-slave-addr` specifies the base address of the slave interface accessible in the standard (A24) address range of the VMEbus.
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a24-slave-size` specifies the size of the memory which is made available to the *standard* (A24) address range of the VMEbus. When the value of this configuration parameter is zero OpenBoot will not initialize the slave interface, even if the `vme-a24-slave-ena?` configuration parameter is `true`!
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a24-slave-ena?` indicates whether the slave interface, accessible in the *standard* (A24) address range of the VMEbus, should be enabled. When this NVRAM configuration parameter is `true` then the VMEbus slave interface is enabled. In the case that the NVRAM configuration parameter is `false` the VMEbus slave interface is not enabled, and any attempt to access the VMEbus slave interface from the VMEbus will lead to an error termination on the VMEbus.
The default value of this NVRAM configuration parameter is `false`.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

In the case that the NVRAM configuration parameter `vme-init?` is `true` the OpenBoot will initialize the slave interface according to the configuration parameters described above.

When the `vme-a24-slave-ena?` configuration parameter is `true`, OpenBoot will initialize the VMEbus slave interface according to the NVRAM configuration parameters `vme-a24-slave-addr` and `vme-a24-slave-size`. It will provide the required amount of physical on-board memory and builds up the necessary MMU and IOMMU settings to make the memory available to the VMEbus. The *virtual* base address of the physical on-board memory provided for VMEbus slave accesses is stored in the variable `vme-a24-slave-mem`.
Thus, applications executed within the OpenBoot environment may benefit from this mechanism, because OpenBoot will initialize the slave interface completely according to the NVRAM configuration parameters associated with the slave interface.

In addition, this mechanism allows to report the parameters of the slave interface to an operating system loaded, which in turn provides its own memory and the corresponding IOMMU settings. In this case the VMEbus device driver is responsible for the access to the slave interface from the VMEbus. In general, the configuration parameter `vme-a24-slave-ena?` must be set to `false` to prevent OpenBoot from initialising and enabling the slave interface when an operating system will be loaded. Supposed that the slave interface is initialized and enabled by OpenBoot prior to loading the operating system, any access from the VMEbus to the slave interface while loading the operating system may alter memory and cause severe damage.

The NVRAM configuration parameters listed below are associated with the slave interface accessible in the extended (A32) address range.

`vme-a32-slave-addr` specifies the base address of the slave interface accessible in the extended (A32) address range of the VMEbus.
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a32-slave-size` specifies the size of the memory which is made available to the *extended* (A32) address range of the VMEbus. When the value of this configuration parameter is zero OpenBoot will not initialize the slave interface, even if the `vme-a24-slave-ena?` configuration parameter is `true`!
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a32-slave-ena?` indicates whether the slave interface, accessible in the extended (A32) address range of the VMEbus, should be enabled. When this NVRAM configuration parameter is `true` the VMEbus slave interface is

20-47 28 2 2000.146 4/20.000/1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

enabled. In the case that the NVRAM configuration parameter is `false` the VMEbus slave interface is not enabled, and any attempt to access the VMEbus slave interface from the VMEbus will lead to an error termination on the VMEbus.
The default value of this NVRAM configuration parameter is `false`.

In the case that the NVRAM configuration parameter `vme-init?` is `true` the OpenBoot will initialize the slave interface according to the configuration parameters described above.
When the `vme-a32-slave-ena?` configuration parameter is `true`, then OpenBoot will initialize the VMEbus slave interface according to the NVRAM configuration parameters `vme-a16-slave-addr` and `vme-a32-slave-size`. It will provide the required amount of physical on-board memory and builds up the necessary MMU and IOMMU settings to make the memory available to the VMEbus. The *virtual* base address of the physical on-board memory provided for VMEbus slave accesses is stored in the variable `vme-a32-slave-mem`.

Thus, applications executed within the OpenBoot environment may benefit from this mechanism, because OpenBoot will initialize the slave interface completely according to the NVRAM configuration parameters associated with the slave interface.

In addition, this mechanism allows to report the parameters of the slave interface to an operating system loaded, which in turn provides its own memory and the corresponding IOMMU settings. In this case the VMEbus device driver is responsible for the access to the slave interface from the VMEbus. In general, the configuration parameter `vme-a32-slave-ena?` must be set to `false` to prevent OpenBoot from initialising and enabling the slave interface when an operating system will be loaded. Supposed that the slave interface is initialized and enabled by OpenBoot prior to loading the operating system, any access from the VMEbus to the slave interface while loading the operating system may alter memory and cause severe damage.

The NVRAM configuration parameters listed below are associated with the master interface to access the ***short*** (**A16**) address range.

`vme-a16-master-addr` specifies the base address of the *short* (A16) address range to be accessed on the VMEbus.
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a16-master-size` specifies the size of the area in the *short* (A16) address range of the VMEbus which will be accessed. When the value of this configuration parameter is zero OpenBoot will not initialize the master interface, even if the `vme-a16-master-ena?` configuration parameter is `true`! If the specified size exceeds the size of the *short* (A16) address

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

range, then it limits the specified size to 64 Kbyte. Due to the capabilities of the SPARC FGA-5000 OpenBoot will always adjust the specified size to 64 Kbyte.
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a16-master-ena?` indicates whether the master interface, to access the *short* (A16) address range of the VMEbus, should be enabled. When this NVRAM configuration parameter is `true` then the VMEbus master interface is enabled. In the case that the NVRAM configuration parameter is `false` the VMEbus master interface is not enabled.
The default value of this NVRAM configuration parameter is `false`.

In the case that the NVRAM configuration parameter `vme-init?` is `true` OpenBoot will initialize the master interface according to the configuration parameters described above. When the `vme-a16-master-ena?` configuration parameter is `true`, then OpenBoot will initialize the necessary registers in the master interface and provides the virtual memory to access the VMEbus. The *virtual* base address necessary to access the VMEbus is stored in the variable `vme-a16-master-mem`.
Thus, applications executed within the OpenBoot environment may benefit from this mechanism, because OpenBoot will initialize the master interface completely according to the NVRAM configuration parameters associated with the master interface.

In addition, this mechanism allows to report the parameters of the master interface to an operating system loaded, which in turn provides its own virtual memory to access the VMEbus. In this case the VMEbus device driver is responsible for providing the necessary virtual address range to access the VMEbus. In general, the configuration parameter `vme-a16-master-ena?` must be set to `false` to prevent OpenBoot from initialising and enabling the master interface when an operating system will be loaded.

The NVRAM configuration parameters listed below are associated with the master interface to access the standard (A24) address range.

`vme-a24-master-addr` specifies the base address of the *standard* (A24) address range to be accessed on the VMEbus.
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a24-master-size` specifies the size of the area in the *standard* (A24) address range of the VMEbus which will be accessed. When the value of this configuration parameter is zero OpenBoot will not initialize the master interface, even if the `vme-a24-master-ena?` configuration parameter is `true`! If the specified size exceeds the size of the *standard* (A24) address range, then it limits the specified size to 16 Mbyte.

20-47.16-2.2/2000.146 420.000/1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a24-master-ena?` indicates whether the master interface, to access the *standard* (A24) address range of the VMEbus, should be enabled. When this NVRAM configuration parameter is `true` then the VMEbus master interface is enabled. In the case that the NVRAM configuration parameter is `false` the VMEbus master interface is not enabled.
The default value of this NVRAM configuration parameter is `false`.

In the case that the NVRAM configuration parameter `vme-init?` is `true` OpenBoot will initialize the master interface according to the configuration parameters described above. When the `vme-a24-master-ena?` configuration parameter is `true`, then OpenBoot will initialize the necessary registers in the master interface and provides the virtual memory to access the VMEbus. The *virtual* base address necessary to access the VMEbus is stored in the variable `vme-a24-master-mem`.
Thus, applications executed within the OpenBoot environment may benefit from this mechanism, because OpenBoot will initialize the master interface completely according to the NVRAM configuration parameters associated with the master interface.
In addition, this mechanism allows to report the parameters of the master interface to an operating system loaded, which in turn provides its own virtual memory to access the VMEbus. In this case the VMEbus device driver is responsible for providing the necessary virtual address range to access the VMEbus. In general, the configuration parameter `vme-a24-master-ena?` must be set to `false` to prevent OpenBoot from initialising and enabling the master interface when an operating system will be loaded.

The NVRAM configuration parameters listed below are associated with the master interface to access the extended (A32) address range.

`vme-a32-master-addr` specifies the base address of the extended (A32) address range to be accessed on the VMEbus.
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a32-master-size` specifies the size of the area in the standard (A24) address range of the VMEbus which will be accessed. When the value of this configuration parameter is zero OpenBoot will not initialize the master interface, even if the `vme-a32-master-ena?` configuration parameter is `true`!
The default value of this 32-bit NVRAM configuration parameter is zero (0).

`vme-a32-master-ena?` indicates whether the master interface, to access the extended (A32) address range of the VMEbus, should be enabled. When this

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

NVRAM configuration parameter is `true` then the VMEbus master interface is enabled. In the case that the NVRAM configuration parameter is `false` the VMEbus master interface is not enabled.
The default value of this NVRAM configuration parameter is `false`.

In the case that the NVRAM configuration parameter `vme-init?` is `true` OpenBoot will initialize the master interface according to the configuration parameters described above. When the `vme-a24-master-ena?` configuration parameter is `true`, then OpenBoot will initialize the necessary registers in the master interface and provides the virtual memory to access the VMEbus. The virtual base address necessary to access the VMEbus is stored in the variable `vme-a24-master-mem`.

Thus, applications executed within the OpenBoot environment may benefit from this mechanism, because OpenBoot will initialize the master interface completely according to the NVRAM configuration parameters associated with the master interface.

In addition, this mechanism allows to report the parameters of the master interface to an operating system loaded, which in turn provides its own virtual memory to access the VMEbus. In this case the VMEbus device driver is responsible for providing the necessary virtual address range to access the VMEbus. In general, the configuration parameter `vme-a32-master-ena?` must be set to `false` to prevent OpenBoot from initialising and enabling the master interface when an operating system will be loaded.

5.2.12 DMA Controller Support

The commands listed below are available to control the DMA controller of the SPARC FGA-5000, as well as to get information about the actual state of the DMA controller.

`dma-ip?` (— `true`|`false`) checks whether an interrupt is pending because a DMA process has been terminated. The value `true` is returned when an interrupt is pending due to the termination of a DMA process. Otherwise the value `false` is returned to indicate that no interrupt is pending.

`dma-ena` (—) enables the DMA controller and starts a DMA process.

`dma-dis` (—) disables the DMA controller and stops the DMA process currently running.

`dma-halt` (—) halts the DMA process currently running.

`dma-resume` (—) resumes the DMA process that has been halted before.

20-4718-2-2000146-420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`dma-src-cap@` (— *data-capability address-capability*) returns the *data-capability* and *address-capability* currently defined for the source of the DMA process.

`dma-src-cap!` (*data-capability address-capability* —) sets the *data-capability* and *address-capability* for the source of the DMA process.
The constants listed below are available to specify the *data-capability* and the *address-capability*:

Value	**Data-capability**	**Address-capability**
000_2	cap-d8	cap-a16
001_2	cap-d16	cap-a24
010_2	cap-d32	cap-a32
011_2	cap-blt	reserved
100_2	cap-mblt	reserved
101_2	reserved	reserved
110_2	reserved	reserved
111_2	reserved	reserved

`dma-dest-cap@` (— *data-capability address-capability*) returns the *data-capability* and *address-capability* currently defined for the destination of the DMA process.

`dma-dest-cap!` (*data-capability address-capability* —) sets the *data-capability* and *address-capability* for the destination of the DMA process.
The constants listed below are available to specify the *data-capability* and the *address-capability*

Value	**Data-capability**	**Address-capability**
000_2	cap-d8	cap-a16
001_2	cap-d16	cap-a24
010_2	cap-d32	cap-a32
011_2	cap-blt	reserved
100_2	cap-mblt	reserved
101_2	reserved	reserved
110_2	reserved	reserved
111_2	reserved	reserved

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`dma-count@` (— *transfer-count*) returns the current state of the transfer count. The value *transfer-count* indicates the number of bytes to be transferred by the DMA controller.

Because the DMA controller only transfers a multiple of 32-bit data (*longword*, which is a *word* in the SPARC terminology), the command returns the appropriate number of *words* to be transferred.

`dma-count!` (*transfer-count* —) sets the number of bytes – *transfer-count* – to be transferred by the DMA controller.

Because the DMA controller only transfers a multiple of 32-bit data (*longword*, which is a *word* in the SPARC terminology), the command calculates the appropriate number of *words* to be transferred. The *transfer-count* is considered to be a module 4 Mbyte less four bytes number.

`dma-running?` (— `true`|`false`) checks whether the DMA controller is in the *running* state. The value `true` is returned when the DMA controller is currently running. Otherwise the value `false` is returned to indicate that the DMA controller is disabled.

`dma-waiting?` (— `true`|`false`) checks whether the DMA controller is in the *waiting* state. The value `true` is returned when the DMA controller is currently waiting, which means that it has been halted. Otherwise the value `false` is returned to indicate that the DMA controller is not waiting.

`dma-normal-terminated?` (— `true`|`false`) checks whether the DMA process has been terminated successfully. It returns the value `true` when the DMA process has been terminated successfully. Otherwise the value `false` is returned to indicate that the DMA process has been terminated due to a fail state, or because the DMA process is still in progress.

`dma-error-terminated?` (— `true`|`false`) checks whether the DMA process has been terminated unsuccessfully. It returns the value `true` when the DMA process has been terminated due to a fail state. Otherwise the value `false` is returned to indicate that the DMA process has been terminated due to normal termination, or because the DMA process is still in progress.

`.dma-stat` (—) displays the current state of the DMA Status Register.

```
ok .dma-stat
ERR:3 NT:0 HALT:0 RUN:0
ok
```

The fields `NT`, `HALT`, and `RUN` reflect the current state of the DMA controller. When the `NT` field is set to one (1) then the DMA controller terminated successfully (*normal termination*). In the case that the `HALT` field

20-47 B 2 2000.06 420.000 1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

is set to one (1) then the DMA controller is halted – in general, this field is set along with the RUN field. The DMA controller is running when the RUN field is set to one (1). When the one of the fields described previous are cleared (0) the DMA controller is not in the particular state.
Typically, the ERR field indicates the course of the DMA controller operation and may indicate the fail states listed in the table below:

Error code	Description
0	Error occurred on source bus
1	Error occurred on destination bus
2	No error termination
3	No error termination

The following two commands to initiate a DMA transfer do not set the data- and address capabilities of the source area and destination area. The capabilities must be set with the `dma-src-cap!` and `dma-dest-cap!` commands appropriately before the DMA transfer is started.

`dma-mem>vme` (*src-addr dest-addr count* — `true`|`false`) initiates a DMA transfer from the SBus to the VMEbus and awaits the termination of the DMA process. The amount of bytes given by *count* are transferred from *src-addr* – an address area on the SBus (*virtual* address) – to *dest-addr* – an address area on the VMEbus (*physical* address). The command returns the value `false` when all data have been transferred successfully. Otherwise the value `true` is returned to indicate that an error occurred during the DMA process.

Because the DMA controller only transfers a multiple of 32-bit data (*longword*, which is a *word* in the SPARC terminology), the command calculates the appropriate number of *words* to be transferred. Furthermore, the *count* is considered to be a module 4 Mbyte less four bytes number.

`dma-vme>mem` (*src-addr dest-addr count* — `true`|`false`) initiates a DMA transfer from the VMEbus to the SBus and awaits the termination of the DMA process. The amount of bytes given by *count* are transferred from *src-addr* – an address area on the VMEbus (*physical* address) – to *dest-addr* – an address area on the SBus (*virtual* address). The command returns the value `false` when all data have been transferred successfully. Otherwise the value `true` is returned to indicate that an error occurred during the DMA process.

Because the DMA controller only transfers a multiple of 32-bit data (*longword*, which is a *word* in the SPARC terminology), the command calculates the appropriate number of *words* to be transferred. Furthermore, the *count* is considered to be a module 4 Mbyte less four bytes number.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.2.13 Mailboxes and Semaphores

The commands described in this section control the mailboxes, the semaphores, and the interrupt box (IBOX).

`vme-mbox-take` (*mailbox#* — `true`|`false`) takes the mailbox semaphore specified by *mailbox#* and returns the value `true` when the mailbox semaphore has been taken successfully. The value `false` is returned when the mailbox semaphore has already been taken.
The value of *mailbox#* may be one of the values in the range zero through 15. Each value specifies one of the 16 Mailbox Registers

`vme-mbox-give` (*mailbox#* —) gives – releases – the mailbox semaphore specified by *mailbox#*.
The value of *mailbox#* may be one of the values in the range zero through 15. Each value specifies one of the 16 Mailbox Registers.

`vme-sem-take` (*semaphore#* — `true`|`false`) takes a semaphore specified by *mailbox#* and returns the value `true` when the semaphore has been taken successfully. The value `false` is returned when the semaphore has already been taken.
The value of *semaphore#* may be one of the values in the range zero through 47. Each value specifies one of the 48 Semaphore Registers.

`vme-sem-give` (*semaphore#* —) gives – releases – the semaphore specified by *semaphore#*.
The value of *semaphore#* may be one of the values in the range zero through 47. Each value specifies one of the 48 Semaphore Registers.

The Interrupt Box is only accessible from the VMEbus within the *short* address space (A16). Any byte access – reading or writing – may lead the SPARC FGA-5000 to generate an interrupt. The address of the interrupt box within the *short* address space may be any byte location in the range 0000_{16} through $FFFF_{16}$.
The commands listed below are available to control and initialize the Interrupt Box.

`vme-ibox-ena` (—) enables the interrupt box.

`vme-ibox-dis` (—) disables the interrupt box.
`vme-ibox-addr@` (— *addr*) returns the physical address *addr* of the interrupt box.

`vme-ibox-addr!` (*addr* —) sets the physical address *addr* of the interrupt box.

As shown in the example below the first command sets the address of the interrupt box. The interrupt box is accessible at the address 4002_{16} within the VMEbus *short* address space. An Sbus IRQ 5 is generated by the

20-47.18-2.2000.00.146-420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SPARC FGA-5000 whenever the interrupt box is accessed from the VMEbus. The fourth command enables the interrupt box.

```
ok h# 4002 vme-ibox-addr!
ok 5 ibox vsi-irq-mapping!
ok true ibox vsi-irq!
ok vme-ibox-ena
ok
```

5.2.14 FORCE Message Broadcast

The commands listed below are available to control the FORCE Message Broadcast (FMB) system and to obtain status information about the state of the FMB system.

`fmb-super-only` (`true`|`false` —) allows or prevents the FMB message registers from being accessed in the *non-privileged* mode. When the value `true` is passed to the command the FMB message register is accessible in the *privileged* mode, as well as in the *non-privileged* mode. Otherwise – the value `false` is passed to the command – the FMB message registers are only accessible in the *privileged* mode.

`fmb-ena` (*channel#* —) enables the FMB channel specified by *channel#*. The value of *channel#* may be one of the values in the range zero to one. Each value specifies one of the two FMB channels.

`fmb-dis` (*channel#* —) disables the FMB channel specified by *channel#*. The value of *channel#* may be one of the values in the range zero to one. Each value specifies one of the two FMB channels.

`fmb!` ([`true`|`false`] *channel#* —) enables or disables the FMB channel specified by *channel#*. When the value `true` is passed to the command the FMB channel is enabled. Otherwise – the value `false` is passed to the command – the FMB channel is disabled.

`fmb-slot@` (— *slot#*) returns the slot number *slot#* assigned to the FMB channels.

`fmb-slot!` (*slot#* —) assigns the slot number *slot#* to the FMB channels. The value of *slot#* may be one of the values in the range zero to 21. Each value specifies a specific slot.

`fmb-addr@` (— *fmb-space*) returns the most significant eight bits – the *fmb-space* – of the 32-bit VMEbus address the FMB will respond to when an FMB transaction on the VMEbus is detected.

`fmb-addr!` (*fmb-space* —) sets the most significant eight bits – the *fmb-space* – of the 32-bit VMEbus address the FMB will respond to when an FMB transaction on the VMEbus is detected.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`fmb-ip?` (*channel#* — `true`|`false`) checks whether an interrupt is pending because an FMB message has been accepted or rejected by the channel specified by *channel#*. The value of *channel#* may be one of the values in the range zero through one. Each value specifies one of the two FMB channels.

The value `true` is returned when an interrupt is pending. Otherwise the value `false` is returned to indicate that no interrupt is pending.

`fmb-msg@` (*channel# — message* `true`|`false`) fetches a message – a 32-bit data – from the FMB channel specified by *channel#*. The *message* and the value `true` are returned when an FMB is available. Otherwise the value `false` is returned to indicate that no FMB message is available.

`fmb-msg!` (*message slot-list channel# —* `true`|`false`) sends the *message* – a 32-bit data – to all FMB channels identified by the *slot-list* and *channel#*. The value `true` is returned when the messages has been sent out successfully. Otherwise the value `false` is returned to indicate that one or more FMB channels have rejected the message.
The value of *channel#* may be one of the values in the range zero through one. Each value specifies one of the two FMB channels.
The value of *slot-list* identifies the hosts participating in the FMB transaction. Each bit of the slot list is associated with a host identified by a *unique* FMB slot number. The first bit – bit 0 – relates to the host with the FMB slot number one (1); the second bit – bit 1 – relates to the host with the FMB slot number two (2); and so forth.
Because the FMB system allows only up to 21 hosts, the command considers only the least significant bits of the parameter *slot-list* (bit 0 through 20).

`fmb-init` (*slot# fmb-space* —) performs all rudimentary steps to initialize the SPARC FGA-5000 in such a way that the subsequent FMB cycles are carried out using the `fmb-msg!` command.
The slot number *slot#* specifies the slot number the FMB channels are associated with. The value of *slot#* may be one of the values in the range zero to 21. Each value specifies a specific slot.
The last available register set in the SPARC FGA-5000 is initialized to carry out an FMB cycle on the VMEbus within the appropriate VMEbus address area that has been specified by *fmb-space*. The parameter *fmb-space* defines the most significant eight bits (one of 256 16-Mbyte pages) of the VMEbus address where the FMB area is located. The capabilities of this VMEbus master range are A32/D32 and write posting is disabled.
The variable `fmb-va` contains the virtual address to be accessed to execute an FMB cycle on the VMEbus.

The example below assigns the slot number 15_{10} to the FMB channels available (all other hosts must have a different FMB slot number). The FMB address space is set to FA_{16} which means that the FMB system is accessed when the address $FAXX.XXXX_{16}$ appears on the VMEbus ad-

20-4718-2 2000146 420 000 1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

dress lines (the least significant 24 bits are used to select a specific FMB channel and specific hosts). And the second command enables the second FMB channel.

```
ok d# 15 h# fa fmb-init
ok true 1 fmb!
ok h# 1234AA55 h# 0010.800f 1 fmb-msg!
ok 1 fmb-msg@
ok .s 2drop
1234AA55 ffffffff
ok 1 fmb-msg@
ok .s drop
0
ok
```

Finally the message $1234.AA55_{16}$ is sent to the second FMB channel available on the hosts with the FMB slot number one, two, three, four, 15, and 20. Because the message is sent to the host with the FMB slot number 15 – the host that sent the message –, the message is read from the second FMB channel on the host, as shown by the fourth command. When the FMB channel is read again, and supposed the host did not receive another FMB message, the command `fmb-msg@` will return the value `false` to indicate that no more messages are available.

5.3 Standard Initialisation of the VMEbus Interface

Besides the initialisation performed according to the state of the NVRAM configuration parameters, the VMEbus interface – mainly the SPARC FGA-5000 – is initialized as described in the subsections below.

5.3.1 SPARC FGA-5000 Registers

The register of the SPARC FGA-5000 are accessible beginning at offset $0FFF.FE00_{16}$ within the SBus slot 5 and occupy the last 512 bytes in this slot ($0FFF.FE00_{16}$ … $0FFF.FFFF_{16}$). This corresponds with the physical address range $7FFF.FE00_{16}$ through $7FFF.FFFF_{16}$.
The area in the range $0E00.0000_{16}$ through $0FFF.FDFF_{16}$ is available for any application. Preferably, this area may be used to access the *standard* (A24, max 16 MB) and *short* (A16, max 64 KB) address space of the VMEbus.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.3.2 VMEbus Transaction Timer

The SPARC FGA-5000 contains a VMEbus transaction timer which is disabled after a RESET. This timer is enabled during the initialisation phase of OpenBoot and the transaction timeout period is set to the longest possible value (512 us).

5.3.3 SBus Rerun Limit

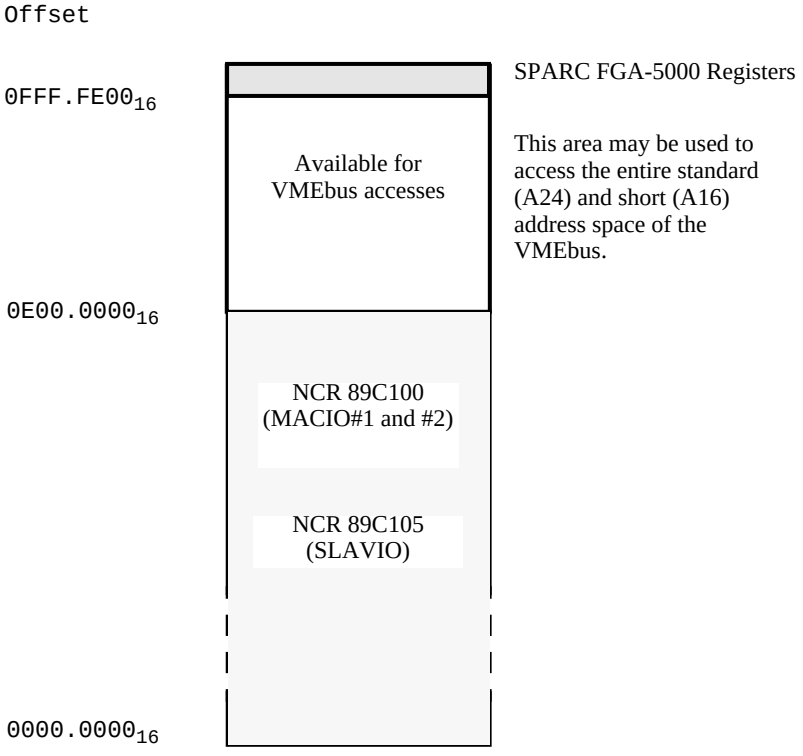
The SBus Rerun Limit counter, within the SPARC FGA-5000, is disabled to avoid any improper behavior of the system.

5.3.4 Interrupts

The SPARC FGA-5000 is initialized in such a way that in the case of the occurrence of pressing the ABORT switch a non-maskable interrupt (level 15 interrupt) is generated.

5.3.5 SBus Slot 5 Address Map

Figure 27 SBus slot 5



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.4 System Configuration

5.4.1 Watchdog Timer

`wd-ena` (—) enables and starts the watchdog timer.

`wd-dis` (—) stops and disables the watchdog timer.

`wd-timeout@` (— *timeout*) returns the watchdog timer's reference value in use. The value of *timeout* may be one of the values in the range zero through seven. Each value identifies a particular timeout period as shown in the table below.

`wd-timeout!` (*timeout* —) sets the watchdog timer's reference value for timeout according to the given *timeout*. The value of *timeout* may be one of the values in the range zero through seven. Only the least significant three bits of the value *timeout* are considered. The values select a particular timeout period. The table below lists all possible values:

Table 48 Watchdog timer timeout values

Timeout	$t_{wd\text{-}timeout\text{-}min}$
0	408 ms
1	1.68 s
2	6.7 s
3	26.8 s
4	1 min 48 s
5	7 min 9 s
6	28 min 38 s
7	1 h 54 min

`wd-restart` (—) resets the watchdog timer and starts a new time count. In particular the command invokes one of the commands **`vsi-wdt-restart@`** or **`vsi-wdt-restart!`** to restart the watchdog timer.

The watchdog timer is started by the commands listed below:

```
ok 3 wd-timeout!
ok vsi-nmi wdt vsi-irq-mapping!
ok true wdt vsi-irq!
ok wd-ena
ok
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

In this example the *watchdog timer timeout* is set to 26.8 seconds, and a ***non-maskable*** interrupt is generated whenever half of the watchdog time has expired. The OpenBoot already contains an interrupt handler dealing with the interrupt generated by the watchdog timer, and this interrupt handler increments an internal variable by one, whenever the watchdog timer emits an interrupt. The state of this variable is determined by:

```
ok wdnmi-occurred? ?
6
ok
```

This variable is cleared – set to zero – by

```
ok wdnmi-occurred? off
ok
```

`wd-reset?` (— `true`|`false`) determines whether a reset has been generated because the watchdog timer has expired. If a reset has been generated because the watchdog timer reached the timeout value, then the value `true` is returned; otherwise the value `false` is returned

5.4.2 Watchdog Timer NVRAM Configuration Parameters

The NVRAM configuration parameters listed below are available to control the initialisation and operation of the watchdog timer. The current state of these configuration parameters are displayed using the `printenv` command, and are modified using either the `setenv`, or the `setdefault` command provided by OpenBoot.

`wd-ena?` controls whether the watchdog timer has to be started. When the flag is `true`, the watchdog timer is started after it has been initialized according to the configuration parameter `wd-timeout`. If the flag is `false` the watchdog timer is not started, but the watchdog timer registers are initialized according to the configuration parameter `wd-timeout`. (default: `false`)

`wd-timeout` contains the timeout value of the watchdog timer and is a value in the range zero to seven. Each value selects a particular timeout period. Independent of the state of the configuration parameter `wd-ena?` the timeout value is stored in the appropriate watchdog timer register. (default: 7_{10})

5.4.3 Abort Switch

`abort-switch?` (— `true`|`false`) determines the current state of the abort switch. The value `true` is returned when the abort switch is pressed. And the value `false` is returned when the abort switch is released.

20-4718-2-2/2000/146 420.000/1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.4.4 Abort Switch NVRAM Configuration Parameter

The NVRAM configuration parameter listed below is available to control the initialisation and operation of the abort switch. The current state of these configuration parameters are displayed using the `printenv` command, and are modified using either the `setenv`, or the `set-default` command provided by OpenBoot.

`abort-ena?` controls whether the abort switch has to be enabled. When this flag is `true` the abort switch is enabled and has the same effect as pressing the `STOP-A` key on an available keyboard. If the flag is `false` then the abort switch is disabled. (default: `false`)

5.4.5 LEDs, Seven Segment Display and Rotary Switch

The commands described below are available to control the seven segment LED display, the system and the user LED, as well as to get information about the state of the rotary switch.

`diag-led!` (*byte* —) stores the data *byte* passed to the command in the register used to control the seven segment display.

`>7-seg-code` (*u* — *7-seg-code*) converts the value *u* to its corresponding seven segment code *7-seg-code*. Only the least significant four bits of the value *u* are considered.

`led!` (*colour freq led#* —) controls the LED identified by *led#*. The value of *led#* may be either zero or one. The value zero specifies the SYS LED, and the value one specifies the user LED. The command only considers the state of the bit 0 of the value *led#*.
The parameters *colour* and *freq* define the color of the LED and the frequency at which the LED is blinking. The following constants are defined to specify the *colour*: **`black`**, **`green`**, **`red`**, and **`yellow`**. When the color **`black`** is specified the LED is turned off.
And the constants **`no-blinking`**, **`slow`**, **`moderate`**, and **`fast`** are available to specify a frequency. The constant **`no-blinking`** causes the LED to be turned on permanently.

The following example shows how to let the user LED blinking at round about 2 Hz (moderate) in red

```
ok red moderate 1 led!
ok
```

`led-on` (*led#* —) turns the LED identified by *led#* on. The value of *led#* may be either zero or one. The value zero specifies the SYS LED, and the value one specifies the user LED. The command only considers the state of the bit 0 of the value *led#*.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`led-off` (*led#* —) turns the LED identified by *led#* off. The value of *led#* may be either zero or one. The value zero specifies the SYS LED, and the value one specifies the user LED. The command only considers the state of the bit 0 of the value *led#*.

`led?` (*led#* — `true`|`false`) determines the state of the LED identified by *led#*, and returns either `true` or `false` to indicate if the LED is turned on or off. The value of *led#* may be either zero or one. The value zero specifies the SYS LED, and the value one specifies the user LED. The command only considers the state of the bit 0 of the value *led#*.
When the LED is turned on, the value `true` is returned; otherwise the value `false` is returned.

`toggle-led` (*led#* —) determines the state of the LED identified by *led#*, and turns the LED on or off. The LED is turned on when it was turned off before, and vice versa.
The value of *led#* may be either zero or one. The value zero specifies the SYS LED, and the value one specifies the user LED. The command only considers the state of the bit 0 of the value *led#*.

`rotary-switch@` (— *byte*) returns the current state of the rotary switch. The value of *byte* may be one of the values in the range zero through 15. The value zero corresponds to the position 0 of the rotary switch, the value one corresponds to position 1, and so forth.

5.4.6 Reset

The command listed below are available to initiate various RESETs, and to obtain information about a previous RESET.

`vme-sysreset` (—) asserts the VMEbus SYSRESET* signal and thus causes a *system* reset.

`reset-call` (—) forces a *local* reset. This command provides the same function as the OpenBoot command `reset`.

`vme-sysreset-in!` (`true`|`false`) allows or prevents the board from being reset by the assertion of the VMEbus SYSRESET* signal. When the value `true` is passed to the command the board will be reset whenever the VMEbus SYSRESET* signal is asserted. Otherwise – the value `false` is passed to the command – the board will not be reset by the assertion of the SYSRESET* signal.

`sbus-reset?` (— `true`|`false`) determines whether the last reset occurred was due to an SBus reset. The value `true` is returned when the last reset was because of an SBus reset. Otherwise it returns the value `false` to indicate that the last reset was not because of an SBus reset.

20-47.18-2.2000.0146 420.000.1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`wdt-reset?` (— `true`|`false`) determines whether a reset has been generated because the watchdog timer has expired. If a reset has been generated because the watchdog timer reached the timeout value, the value `true` is returned; otherwise the value `false` is returned.

`vme-sysreset?` (— `true`|`false`) determines whether the last reset occurred was due to the assertion of the VMEbus SYSRESET* signal. The value `true` is returned when the last reset was a VMEbus SYSRESET* reset. Otherwise it returns the value `false` to indicate that the last reset was **not** a VMEbus SYSRESET* reset.

`vme-sysreset-call?` (— `true`|`false`) determines whether the last reset occurred was due to a VMEbus SYSRESET* call. The value `true` is returned when the last reset was because of a VMEbus SYSRESET* call. Otherwise it returns the value `false` to indicate that the last reset was not a VMEbus SYSRESET* call.
A VMEbus SYSRESET* call is done by clearing the SYSRESET bit in the SPARC FGA-5000's Miscellaneous Control and Status Register.

`reset-call?` (— `true`|`false`) determines whether the last reset occurred was due to a local reset call. The value `true` is returned when the last reset was because of a local reset call. Otherwise it returns the value `false` to indicate that the last reset was not a local reset call.
A local reset call is done by clearing the RESET bit in the SPARC FGA-5000's Miscellaneous Control and Status Register.

`vme-reset-call?` (— `true`|`false`) determines whether the last reset occurred was due to a reset call initiated by an access via the VMEbus. The value `true` is returned when the last reset was because of a reset call. Otherwise it returns the value `false` to indicate that the last reset was **not** because of a reset call initiated by an access via the VMEbus.
A reset call is done by clearing the LOCRESET bit in the SPARC FGA-5000's Global Control and Status Register.

5.4.7 ID PROM

On the CPU-8VT an ID PROM (a serial E^2PROM X24C04) is connected via an I^2C bus. The OpenBoot provides the following commands to access the ID PROM:

`mem>idprom` (*src-addr dest-addr size* —) copies a number of bytes from the on-board memory to the ID PROM. The number of bytes to be copied are specified by *size*. The data are stored beginning at the virtual address *src-addr* and are copied to the ID PROM beginning at address *dest-addr*. The destination address may range from `0` to `511` and specifies the location within the ID PROM. The value of *size* may be one of the values in the range `1` through `512`.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`idprom>mem` (*src-addr dest-addr size* —) copies a number of bytes from the ID PROM to the on-board memory. The number of bytes to be copied are specified by *size*. The data are stored beginning at the virtual address *dest-addr* and are copied from the ID PROM beginning at address *src-addr*. The source address may range from `0` to `511` and specifies the location within the ID PROM. The value of *size* may be one of the values in the range `1` through `512`.

5.5 Flash Memory Support

5.5.1 Flash Memory Programming

The commands listed below are available to access and program the flash memories available on the SPARC/CPU-8VT.

`flash-messages` (— *vaddr*) returns the virtual address of the *variable* `flash-messages`. The state of this variable controls whether the words to erase and program the flash memories will display messages while erasing or programming the flash memories. Messages will not be displayed after *turning off* this variable by `flash-messages off`, and are displayed after *turning on* this variable by `flash-messages on`.

`flash-va` (— *vaddr*) returns the virtual base address *vaddr* of the flash memory programming window. The virtual address returned is only valid when the flash memories have been previously prepared for accessing using the `select-flash` word.

`boot-flash-va` (— *vaddr*) returns the virtual base address *vaddr* of the BOOT flash memory.

`user-flash-va` (— *vaddr*) returns the virtual base address *vaddr* of the USER flash memory. When the USER flash memory is not accessible directly, but only through the flash memory programming window, the address returned is zero. On the SPARC/CPU-8VT the USER flash memory is accessible only through the flash memory programming window. Thus, the commands described above have to be used to access the USER flash memory.

`select-flash` (“USER<eol>” | “BOOT<eol>” —) prepares either the BOOT flash memories, or the USER flash memories for programming. In detail, the number and size of the available flash memories are determined, as well as the size of the flash memory programming window. The flash memory programming window is mapped and the virtual base address of the window is stored internally, and may be obtained by using the word `flash-va`.

20-4718-2-2000146-420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`user-flash?` (— `true`|`false`) checks whether the BOOT flash memory or the USER flash memory is accessible through the flash memory programming window. It returns `true` in the case that the USER flash memory is accessible through the programming window; otherwise it returns `false`.

`move>flash` (*source-addr dest-addr count* —) programs the selected flash memory beginning at *dest-addr* with a number of bytes, specified by *count* and stored at *source-addr*.

`flash>move` (*source-addr dest-addr count* —) copies a number of bytes, specified by *count*, from the selected Flash Memory beginning at *source-addr* to *dest-addr*. The Flash Memory is accessed through the Flash Memory programming window for reading data from the memory. Thus, the Flash Memory has to be prepared for accessing using the command `select-flash`.

`fill-flash` (*dest-addr count pattern* —) fills the selected flash memory beginning at *dest-addr* with a particular *pattern*. The number of bytes to be programmed in the flash memory is given by *count*.

`erase-flash` (*device-number* —) erases a flash memory device identified by its *device-number*. The devices are numbered beginning from zero (0).

`c!-flash` (*byte addr* —) stores the *byte* at the location within the selected flash memory identified by *addr*.

`w!-flash` (*half-word addr* —) stores the *half-word* (16 bits) at the location within the selected flash memory identified by *addr*.

`l!-flash` (*word addr* —) stores the *word* (32 bits) at the location within the selected flash memory identified by *addr*.

The USER flash memory is prepared for programming by:

```
ok select-flash USER
USER flash memory is selected for programming
Flash memory programming window at $ffe98000 size 512 Kbyte
512 Kbyte BOOT flash memory is available at $ffe58000.
2048 Kbyte USER flash memory is available.
ok
```

As shown above, the word `select-flash` informs the user that the USER flash memory has been made accessible through the flash memory programming window. It displays the base address (*virtual* address) of the window and its size.

The total amount of the available BOOT flash memory and USER flash memory is displayed, too. After the USER flash memory has been prepared for programming, all commands described above operate on the USER flash memory. And the BOOT flash memory is only read and programmed by these commands when the BOOT flash memory has been prepared for these operations by:

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ok select-flash BOOT
BOOT flash memory is selected for programming
Flash memory programming window at $ffe98000 size 512 Kbyte
512 Kbyte BOOT flash memory is available at $ffe58000.
2048 Kbyte USER flash memory is available.
ok
```

To read data from the selected flash memory – in the current context from the USER flash memory – the command `flash>move` is used as follows:

```
ok flash-va h# 10.0000 h# 20.0000 flash>move
ok
```

The contents of the entire USER flash memory is copied to main memory beginning at address 10.0000_{16}. A specific area within the selected flash memory is read by:

```
ok flash-va h# 6.8000 + h# 10.0000 h# 5.8c00 flash>move
ok
```

and copies 363520 bytes beginning from address `flash-va` + 6.8000_{16} to main memory beginning at address 10.0000_{16}.

5.5.2 Flash Memory Device

The device tree of OpenBoot for the SPARC/CPU-8VT contains a device node associated with the USER flash memories. Thus, it is possible to load an executable image stored in the available USER flash into memory and start such an executable.
The device is called "**`flash-memory@0,300000`**" and is attached to the device node "**`/obio`**". The device alias **`flash`** is available as an abbreviated representation of the flash memory device path.
The vocabulary of the flash memory device includes the standard commands recommended for a *byte* device. The words of this vocabulary are only available when the flash memory device has been selected as shown below:

```
ok cd flash
ok words
close         open         selftest     reset     load
write-blocks  read-blocks  seek         write     read
max-transfer  block-size
ok selftest .
0
ok device-end
ok
```

The example listed above, selects the flash memory device and makes it the current node. The word `words` displays the names of the *methods* of the VMEbus device. And the third command calls the method `selftest` and the value returned by this method is displayed. The last command *unselects* the current device node, leaving no node selected.

20-47.18-2-2000.146-420.000-1-March.1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

When the command `select-dev` is used to select the flash memory device, the NVRAM configuration parameters `bootflash-#megs` and `bootflash-#devices` have to be set properly, before the device can be selected.

The NVRAM configuration parameters listed below are available to control the loading of an image from the USER flash memory. The current state of these configuration parameters is displayed using the `printenv` command, and is modified using either the `setenv`, or the `set-default` command provided by OpenBoot.

`bootflash-#megs` specifies the amount of available USER flash memory in megabyte. (default: 0 megabyte)

`bootflash-#devices` specifies the number of available USER flash memory devices. (default: no devices)

`bootflash-load-base` specifies the address where the data loaded from the available flash memory are stored when the **`load`** or **`boot`** command, provided by OpenBoot, is used to load an image from the flash memory.
When this parameter is set to -1 – which is the parameter's default value – then the image loaded from the flash memory is stored beginning at the address *addr*. But when the value of the configuration parameter differs from -1, the image loaded from the flash memory is stored beginning at the address specified by the configuration parameter `bootflash-load-base`. The same address is stored in the variable `load-base` maintained by OpenBoot.
The methods listed below are available in the vocabulary of the flash memory device:

`open` (— `true`) prepares the package for subsequent use. The value `true` is returned when the device has been opened successfully; otherwise the value `false` is returned. Usually, the fail state is indicated when the NVRAM configuration parameters `bootflash-#megs` and `bootflash-#devices` are not consistent.

`close` (—) frees all resources allocated by open.

`reset` (—) puts the flash memory device into *quiet* state.

`selftest` (— *error-number*) always returns the value zero.

`read` (*addr length* — *actual*) reads at most *length* bytes from the flash memory device into memory beginning at address *addr*. If *actual* is zero or negative, the read failed. The value of *length* may not always be a multiple of the device's normal block size.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`write` (*addr length — actual*) discards the information passed to the command and always returns zero to indicate that the device does not support this function.

`seek` (*offset file# — error?*) seek to byte *offset* within the file identified by *file#*. The flash memory device package maintains an internal position counter that is updated whenever a method to read data from or to store data in the flash memories is called. If *offset* and *file#* are both zero, the internal position counter is reset to offset zero, otherwise the value of *offset* is assigned to the internal position counter, and a subsequent access to the flash memories starts at the offset selected.
As the flash memory device does not support any file system, the parameter *file#* is ignored, except in the case mentioned above.
When the seek succeeded the value of *error?* is zero, otherwise the value `-1` is returned to indicate the fail state.

`read-blocks` (*addr block# #blocks — #read*) reads the number of blocks identified by *#blocks* of length *block-size* bytes, each from the device beginning at the device block *block#*, into memory at address *addr*. It returns the number of blocks actually read (*#read*).

`write-blocks` (*addr block# #blocks — #written*) discards the information passed to the command and always returns zero to indicate that the device does not support this function.

`block-size` (*— bytes*) returns the size in bytes *bytes* of a block which is always the size of the flash memory programming window.

`max-transfer` (*— bytes*) returns the size in bytes *bytes* of the largest single transfer the device can perform. The command returns a multiple of `block-size`.

`load` (*addr — length*) reads a stand-alone program from the flash memory beginning at offset 0_{16} and stores it beginning at address *addr*. It returns the number of bytes *length* read from the flash memory.
This method considers the state of the NVRAM configuration parameter `bootflash-load-base`: when this parameter is set to -1 – which is the parameter's default value – then the image loaded from the flash memory is stored beginning at the address *addr*. But when the value of the configuration parameter differs from `-1`, then the image loaded from the flash memory is stored beginning at the address specified by the configuration parameter `bootflash-load-base`. And the same address is stored in the variable `load-base` maintained by OpenBoot.

20-FB-2.2000.06-420.000.1-March.1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.5.3 Loading and Executing Programs from USER Flash Memory

Besides the ability to load and execute an executable image from disk, or via a network, or other components, the OpenBoot for the SPARC/CPU-8VT provides a convenient way to load and execute an executable image from the available USER flash memory. The executable image to be loaded has to be either a **binary** image (a.out format), a ***FORTH*** program, or a ***FCode*** program.
As mentioned at the beginning of this section the device alias `flash` is available as an abbreviated representation of the flash memory device. The command listed below is used to explicitly load and execute an image from the flash memory:

```
ok boot flash
```

The following NVRAM configuration parameters can be modified to determine whether or not the system will load an executable image automatically after a power-up cycle or system reset:

```
auto-boot?
boot-device
```

Assuming that the SPARC/CPU-8VT is equipped with one USER flash memory device which size is 1Mbyte, the commands listed in the following have to be used to load and execute an image from the flash memory automatically after a power-up cycle or system reset:

```
ok setenv bootflash-#devices 1
bootflash-#devices =  1
ok setenv bootflash-#megs 1
bootflash-#megs =     1
ok setenv boot-device flash
boot-device =         flash
ok setenv auto-boot? true
auto-boot? =          true
ok reset
```

5.5.4 Controlling the Flash Memory Interface

The commands listed below are available to control the flash memory interface. These commands are used to make a specific flash memory device available in the flash memory programming window, and to control the flash memory programming voltage.

`flash-vpp-on` (—) turns the programming voltage on.

`flash-vpp-off` (—) turns the programming voltage off.

`userprom-select-page` (*page* —) makes a *page* (one of eight possible 512 KB pages) of a USER flash memory available in the *flash memory programming window*.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`bootprom-select-page` (*page* —) makes a *page* (one of eight possible 512 KB pages) of a BOOT flash memory available in the *flash memory programming window*.

`select-bootprom-1` (—) makes the first BOOT flash memory device available in the *flash memory programming window*.

`select-bootprom-2` (—) makes the second BOOT flash memory device available in the *flash memory programming window*.

`select-bootprom` (*device-number* —) makes a BOOT flash memory device, identified by its *device-number*, available in the *flash memory programming window*. The devices are numbered beginning from zero (0).

`select-userprom-1` (—) makes the first USER flash memory device available in the *flash memory programming window*.

`select-userprom-2` (—) makes the second USER flash memory device available in the *flash memory programming window*.

`select-userprom` (*device* —) makes a USER flash memory device, identified by its *device-number*, available in the *flash memory programming window*. The devices are numbered beginning from zero (0).

5.6 On-board Interrupts

Besides the interrupt handlers already available in the standard OpenBoot, the OpenBoot of the SPARC/CPU-8VT provides further handlers that deal with the interrupts generated by following:

- one of the VMEbus interrupt levels one to seven;
- the assertion and negation of the `SYSFAIL*` signal;
- the assertion of the `ACFAIL*` signal;
- pressing the `ABORT` switch;
- the watchdog timer, when half the time has expired.

5.6.1 VMEbus Interrupts

The interrupt handlers for any VMEbus interrupt are not installed automatically by OpenBoot; however, appropriate words are available to *activate* and *deactivate* an interrupt handler serving a specific VMEbus interrupt. Such an interrupt handler is activated by:

20-4718-2-2000148-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ok 0 pil!
ok 3 5 install-vme-intr-handler
ok
```

The **`pil!`** command decreases the processor interrupt level to allow the processor to respond to all interrupts. By default, OpenBoot sets the mask to `13` and allows the processor to respond to interrupts above interrupt level `13`. The second command installs the interrupt handler that deals with the VMEbus interrupt level `5`. Furthermore, this command specifies that an SBus interrupt level `3` will be generated upon the occurrence of a VMEbus interrupt `5`. Any of the seven SBus interrupt levels may be specified to be generated upon a VMEbus interrupt.
OpenBoot maintains seven variables called `vme-intr{1|2|3|4|5|6|7}-vector` which are modified by the VMEbus interrupt handlers. In general, the interrupt handlers store the vector obtained during an interrupt acknowledge cycle in the appropriate variable. The state of these variables is displayed by

```
ok .vme-vectors
1: -- 2: --   3: --   4: --   5: 33   6: --   7: --
ok
```

By default, the value -1 (`true`) is assigned to these variables to indicate that no VMEbus interrupt occurred. So, the word `.vme-vectors`, as shown above, will display "**`--`**" indicating that no interrupt occurred; otherwise it shows the vector obtained (a value in the range 0 to FF_{16}).
Another way to display the state of a variable used to store the interrupt vector is

```
ok vme-intr5-vector ?
33
ok
```

and the variable is set to -1 (`true`) by

```
ok vme-intr5-vector on
ok
```

An interrupt handler is removed and the corresponding interrupt is disabled by

```
ok 5 uninstall-vme-intr-handler
ok
```

All interrupt handlers serving all VMEbus interrupts are installed by

```
ok 0 pil!
ok 8 1 do i i install-vme-intr-handler loop
ok
```

In this case, all interrupt handlers are installed and the VMEbus interrupt to SBus interrupt mapping is as follows: SBus interrupt level 1 is generated upon the occurrence of a VMEbus interrupt 1; SBus interrupt level 2 is generated upon the occurrence of a VMEbus interrupt 2; and so forth.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.6.2 SYSFAIL Interrupt

OpenBoot for the SPARC/CPU-8VT already includes an interrupt handler to serve the non- maskable interrupt generated upon the assertion and negation of the SYSFAIL* signal. This handler need not to be installed because it is already installed by OpenBoot.
By default, the interrupts that will be emitted by a status change of the SYSFAIL* signal are disabled and have to be enabled by

```
ok true sysfail- vsi-irq!
ok true sysfail+ vsi-irq!
ok
```

which enable the generation of a non-maskable interrupt whenever the SYSFAIL* signal is asserted and negated.

When a non-maskable interrupt occurred due to the assertion of the SYSFAIL* signal, the appropriate interrupt handler increments the variable sysfail-asserted? by one to report the occurrence of such an interrupt. The variable sysfail-negated? is incremented by the interrupt handler when the SYSFAIL* signal has been negated and caused a non- maskable interrupt. The state of both variables are obtained by

```
ok sysfail-asserted? ?
0
ok
```

and

```
ok sysfail-negated? ?
1
ok
```

And these variables are cleared – set to zero – by

```
ok sysfail-asserted? off
ok sysfail-negated? off
ok
```

5.6.3 ACFAIL Interrupt

OpenBoot for the SPARC/CPU-8VT already includes an interrupt handler to serve the non- maskable interrupt generated upon the assertion of the ACFAIL* signal. This handler need not to be installed because it is already installed by OpenBoot.
By default, the interrupt that will be emitted by asserting the ACFAIL* signal is disabled and has to be enabled by

```
ok true acfail vsi-irq!
ok
```

which enables the generation of a non-maskable interrupt whenever the ACFAIL* signal is asserted.

When a non-maskable interrupt occurred due to the assertion of the ACFAIL* signal, the appropriate interrupt handler increments the variable

20-118-2.2000.146 420.000.1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`acfail-asserted?` by one to report the occurrence of such an interrupt. The state of this variable is obtained by

```
ok acfail-asserted? ?
2
ok
```

And the variable is cleared – set to zero – by

```
ok acfail-asserted? off
ok
```

5.6.4 ABORT Interrupt

OpenBoot for the SPARC/CPU-8VT already includes an interrupt handler to serve the non- maskable interrupt generated by pressing the front panel abort switch. This handler need not be installed because it is already installed by OpenBoot.
By default, the interrupt that will be emitted when the abort switch has been pressed is disabled and has to be enabled by

```
ok true abort-key vsi-irq!
ok
```

which enables the generation of a non-maskable interrupt whenever the abort switch is pressed.

When a non-maskable interrupt occurred due to pressing the abort switch, the appropriate interrupt handler increments the variable `abort-occurred?` by one to report the occurrence of such an interrupt. The state of both variables are obtained by

```
ok abort-occurred? ?
7
ok
```

And these variables are cleared – set to zero – by

```
ok abort-occurred? off
ok
```

Besides the effects described above, the pressing of the abort switch has the same effect as giving the **`Stop-A`** keyboard command. The program currently running is aborted and the FORTH interpreter appears immediately.

5.6.5 Watchdog Timer Interrupt

OpenBoot for the SPARC/CPU-8VT already includes an interrupt handler to serve the non- maskable interrupt generated by the watchdog timer when half of the time has expired. This handler need not to be installed because it is already installed by OpenBoot.
By default, the interrupt that will be emitted by the watchdog timer is disabled – the watchdog timer is disabled – and has to be enabled by

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ok true wdt vsi-irq!
ok wd-ena
ok
```

In this example a non-maskable interrupt is generated whenever half of the watchdog time has expired. The interrupt handler included in OpenBoot restarts the watchdog timer to ensure that the watchdog time will not expire and cause a reset. Additionally, the interrupt handler increments the variable `wdnmi-occurred?` by one whenever the watchdog timer emits an interrupt. The state of this variable is determined by

```
ok wdnmi-occurred? ?
6
ok
```

This variable is cleared – set to zero – by

```
ok wdnmi-occurred? off
ok
```

5.7 Second SCSI and Ethernet Interface

5.7.1 Commands

`ni-test!` (`true`|`false` *ni#* —) enables or disables the twisted pair network interface's link test capability. The network number *ni#*, which specifies the proper network interface, may be one or two. Each value specifies one of the two available network interfaces. The value one specifies the **first** network interface and the value two specifies the **second** network interface. In the case that the value of *ni#* is neither one nor two, the command assumes that the first network interface is specified.
When the value `true` is passed to the command then the network interface's test capability is enabled. Otherwise – the value `false` is passed to the command – the network interface's test capability is disabled.

`ni-test-ena` (*ni#* —) enables the twisted pair network interface's link test capability. The network number *ni#*, which specifies the proper network interface, may be one or two. Each value specifies one of the two available network interfaces. The value one specifies the **first** network interface and the value two specifies the **second** network interface. In the case that the value of *ni#* is neither one nor two, the command assumes that the first network interface is specified.

`ni-test-dis` (*ni#* —) disables the twisted pair network interface's link test capability. The network number *ni#*, which specifies the proper network interface, may be one or two. Each value specifies one of the two available network interfaces. The value one specifies the **first** network interface and the val-

20-47.18-2.2/000.146-420.000/1-March.1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

ue two specifies the **second** network interface. In the case that the value of *ni#* is neither one nor two, the command assumes that the first network interface is specified.

`ni-stat?` (*ni#* — `true`|`false`) determines the state of the twisted pair network interface specified by the network number *ni#*. The network number *ni#* may be one or two, and specifies one of the two available network interfaces. Each value specifies one of the two available network interfaces. The value one specifies the *first* network interface and the value two specifies the *second* network interface. In the case that the value of *ni#* is neither one nor two, the command assumes that the first network interface is specified.
When the network link is up the value `true` is returned; otherwise the value `false` is returned to indicate that the network link is down.

`select-macio` (*macio#* —) selects the NCR 89105 (MACIO) device that will be accessible at the predefined addresses within the SBus slot 5. The MACIO's device number *macio#* may be one or two. Each value specifies one of the two available MACIO devices. When the value one is passed to the command the *first* MACIO device (MACIO #1) is selected, and if the value two is passed to the command the second MACIO device (MACIO #2) is selected.
In the case that the value of *macio#* is neither one nor two as mentioned above, the command assumes that the first MACIO device is selected.

`macio-selected?` (— *macio#*) returns the number of the NCR89105 (MACIO) device that is currently accessible at the predefined addresses within the SBus slot 5.

The NVRAM configuration parameter is available to control which of the two available MACIO devices is available at the predefined addresses within the SBus slot 5 after reset.

`use-second-macio?` controls whether the *second* NCR 89105 (MACIO) device is accessible at the predefined addresses within the SBus slot 5. When the value of this configuration parameter is `true`, the second MACIO device is accessible at the predefined addresses within the SBus slot 5. Otherwise – the value of the configuration parameter is `false` – the first MACIO device is accessible at the predefined addresses. (default: `false`)

`tpe-link-test?` controls whether to enable or disable the link test capability of the first on-board 10baseT Ethernet Interface (TPE). When the value of this configuration parameter is `true`, the link test capability is enabled. Otherwise – the value of the configuration parameter is `false` – the link test capability is disabled. (default: `true`)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`tpe-link-2-test?` controls whether to enable or disable the link test capability of the *second* on-board 10baseT Ethernet Interface (TPE). When the value of this configuration parameter is `true`, the link test capability is enabled. Otherwise – the value of the configuration parameter is `false` – the link test capability is disabled. (default: `true`)

Device Aliases

The following device aliases are provided by the OpenBoot for the SPARC/CPU-8VT to identify a certain device associated with the second MACIO:

```
disk20   /iommu/sbus/espdma@5,8400040/esp@5,8800040/sd@0,0
disk21   /iommu/sbus/espdma@5,8400040/esp@5,8800040/sd@1,0
disk22   /iommu/sbus/espdma@5,8400040/esp@5,8800040/sd@2,0
disk23   /iommu/sbus/espdma@5,8400040/esp@5,8800040/sd@3,0
tape21   /iommu/sbus/espdma@5,8400040/esp@5,8800040/st@5,0
tape20   /iommu/sbus/espdma@5,8400040/esp@5,8800040/st@4,0
tape2    /iommu/sbus/espdma@5,8400040/esp@5,8800040/st@4,0
cdrom2   /iommu/sbus/espdma@5,8400040/esp@5,8800040/sd@6,0:d
disk-2   /iommu/sbus/espdma@5,8400040/esp@5,8800040/sd@3,0
net2     /iommu/sbus/ledma@5,8400050/le@5,8c00040
net2-tpe /iommu/sbus/ledma@5,8400050:tpe/le@5,8c00040
net2-aui /iommu/sbus/ledma@5,8400050:aui/le@5,8c00040
scsi2    /iommu/sbus/espdma@5,8400040/esp@5,8800040
```

5.8 BusNet Support

In general, the OpenBoot should provide the capability to load and execute (*boot*) an executable image via the VMEbus backplane using the BusNet protocol.

5.8.1 Limitations

Due to the fact that OpenBoot is a simple *booter*, rather than an operating system, the limitations listed below apply to the BusNet protocol implementation:

- The OpenBoot support for the BusNet protocol only allows a participant to operate as a **slave**.
- The network management services are currently **not** supported. Every received packet containing such a request is refused by the BusNet driver.
- The OpenBoot provides only *single-buffering* mode which means that only one buffer is provided for every participant.
- In general, OpenBoot does not use any interrupt mechanism while loading an image from the *boot* device. Therefore, OpenBoot will not

20-47.28-2 2.2000.146 420.000.1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

enable a mailbox—available on the machine—even if the NVRAM configuration parameters allow the use of a mailbox.

5.8.2 Loading Programs

The OpenBoot provides several methods for loading and executing a program on the machine. These methods load a file from a remote machine across the communication channel into memory, and support execution of FORTH-, FCode- and binary executable programs.

An executable program is loaded across the VMEbus using the BusNet protocol with the following two command provided by OpenBoot:
`$ boot device-specifier argument`
or
`$ load device-specifier argument`

The parameter *device-specifier* represents the name – full path name or alias – of the BusNet boot device. The OpenBoot provides the following device alias definitions associated with this device:

Alias	Boot path	Description
busnet	`/iommu/VME/BusNet:tftp`	TFTP is used to load program
busnet-tftp	`/iommu/VME/BusNet:tftp`	TFTP is used to load program
busnet-raw	`/iommu/VME/BusNet:raw`	pure binary data is loaded (*raw* device)

IMPORTANT



Many commands – like `boot` and `test` – that require a device name, accept either a full device path name or a device alias. In this documentation, the term *device-specifier* is used to indicate that either a device path or a device alias is acceptable for such commands

5.8.3 The BusNet Device

The BusNet device is a *packet* oriented device capable of sending and receiving packets. The BusNet device available in OpenBoot is called BusNet and is attached to the device path /iommu/VME.

5.8.3.1 Device Properties

Device properties identify the characteristics of the package and its associated physical device. The BusNet device is characterized by the properties described below – these properties are static:

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`name` property identifies the package. The BusNet package is identified by the string `busnet`.

`device_type` declares the type of the device. As the BusNet device is intended for booting across a *network* (VMEbus), its device type is declared as `network`.

`address-bits` specifies the number of address bits necessary to address this device on its network. Typically, the BusNet address consists of `32` bits, but only the least significant five bits are important. All remaining bits must be cleared (`0`). Therefore, the property `address-bits` is set to `32`.
The property's size is `32` bits (integer).

`reg` property describes the VMEbus address ranges which are accessible by the BusNet device driver. The information given by this property is crucial for the operating of the operating system's own BusNet device driver. The register property is declared as follows:

VMEbus A16 space	`h# 0000.0000 vmea16d32 h# 0001.0000`
VMEbus A24 space	`h# 0000.0000 vmea24d32 h# 00ff.0000`
VMEbus A32 space	`h# 0000.0000 vmea32d32 h# ff00.0000`

The properties listed below are created dynamically whenever the device is opened for subsequent accesses:

`bn-packet-size` specifies the size of a BusNet packet – including the BusNet packet header. The value of this property depends on the value of the NVRAM configuration parameter `bn-packet-size`. When the value of the configuration parameter is below the minimum of 2048 bytes, the property's value is set to `2048`. In the case that the value of the configuration parameter is not a multiple of 64 bytes, the value of the property is *downsized* to the next 64 byte boundary.
The property's size is 32 bits (integer).

`max-frame-size` indicates the maximum allowable size of a packet (in bytes). This property is created dynamically when the BusNet device is opened and depends on the property `bn-packet-size`.
The property's size is 32 bits (integer).

`bn-master-offset` specifies the physical address of the participant designated as master.
The property's size is 32 bits (integer).

`bn-master-space` specifies the space in which the master's BusNet region is accessible.
The property's size is 32 bits (integer).

20-47.18.2-2000146-420.000-1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`bn-master-access` specifies the access mode of the master's BusNet region.
The property's size is 32 bits (integer).

`bn-p-offset` specifies the physical address of the participant's own BusNet region.
The property's size is 32 bits (integer).

`bn-p-space` `specifies` the space in which the participant's own BusNet region is accessible.
The property's size is 32 bits (integer).

`bn-p-access` specifies the access mode of the participant's own BusNet region.
The property's size is 32 bits (integer).

`bn-logical-addr` specifies the logical address assigned to the participant.
The property's size is 32 bits (integer).

`bn-p-mbox-offset` specifies the physical address of the participant's mailbox.
The property's size is 32 bits (integer).

`bn-p-mbox-space` specifies the space in which the mailbox of the participant is accessible.
The property's size is 32 bits (integer).

`bn-p-mbox-access` specifies the access mode of the participant's mailbox.
The property's size is 32 bits (integer).

`bn-p-mbox-intr` specifies the interrupt generated when the participant's mailbox is being accessed from the bus.
The property's size is 32 bits (integer).

`bn-p-mbox?` specifies whether the participant provides a mailbox. When the value of this property is `true` then the participant provides a mailbox. Otherwise, the participant does not provide a mailbox.

5.8.3.2 Device Methods

The BusNet device intended for use by OpenBoot implements the methods described below.

`open` (— *ok?*) prepares the device for subsequent use. The value `true` is returned upon successful completion; otherwise, the value `false` is returned to indicate a failure. When `open` is called, the parent instance chain has already been opened, and this method may call its parent's methods.
Typically, the device builds up its BusNet region, makes this region available to the VMEbus address space, and tries to connect with the BusNet master for registering.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`close` (—) restores the device to its *not-in-use* state. Typically, it informs all known BusNet participants about its intention to withdraw from the protocol, and disables its VMEbus slave interface to prevent it from being accessed by other BusNet participants.

`reset` (—) puts the device into its *quiescent* state, and afterwards starts to register with the master again. In particular, the `reset` method executes the `close` and immediately afterwards the `open` method.

`selftest` (— *error#*) normally tests the package and returns an error number *error#* which identifies a specific failure. But the BusNet device provides this method only for completeness, and returns the value zero when the method is called. The value zero is returned to indicate that no failure has been detected.

`load` (*addr* — *length*) reads the default stand-alone program into memory starting at *addr* using the network booting protocol. The *length* parameter returned specifies the size in bytes of the image loaded.

`read` (*addr length* — *actual*) receives a network packet and stores at most the first *length* bytes in memory beginning at address *addr*. It returns the *actual* number of bytes received (not the number copied), or it returns zero if no packet is currently available.
The BusNet device driver copies only the data contained in the BusNet packet into memory and discards all information related to the BusNet protocol.

`write` (*addr length* — *actual*) transmits the network packet of size *length* stored in memory beginning at address *addr*, and returns the number of bytes actually transmitted, or zero if the packet has not been transmitted due to a failure.
The BusNet device driver copies the data into the data field of a BusNet packet and transmits the packet to the specified recipient.

`seek` (*poslow poshigh* — *-1*) operation is invalid and the method therefore always returns -1 to indicate the failure.

5.8.3.3 NVRAM Configuration Parameters

The OpenBoot provides the NVRAM configuration parameters as defined by the BusNet Protocol Specification 1.4.2. The NVRAM configuration parameters may be modified using the `set-default` or `setenv` commands provided by OpenBoot. The actual state of the NVRAM configuration parameters are displayed by the `printenv` command.

`bn-master-offset` specifies the physical address of the participant designated as master. The default value of this 32-bit configuration parameter is zero.

20-4718-2-2/2000/146 420.000/1 March 1999

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`bn-master-space` specifies the space in which the master's BusNet region is accessible. Typically, this configuration parameter identifies one of the address spaces available in the address range of the bus. The default value of this 32-bit configuration parameter is $3D_{16}$ (privileged *standard* address space).

`bn-master-access` specifies the access mode of the master's BusNet region. The default value of this 32-bit configuration parameter is 32_{16} (D32, read/write, no LOCKed cycles are supported).

`bn-p-offset` specifies the physical address of the participant's own BusNet region. The default value of this 32-bit configuration parameter is zero.

`bn-p-space` specifies the space in which the participant's own BusNet region is accessible. Typically, this configuration parameter identifies one of the address spaces available in the address range of the bus. The default value of this 32-bit configuration parameter is $3D_{16}$ (privileged *standard* address space).

`bn-p-access` specifies the access mode of the participant's own BusNet region. The default value of this 32-bit configuration parameter is 32_{16} (D32, read/write, no LOCKed cycles are supported).

`bn-logical-addr` specifies the logical address assigned to the participant. The value of this configuration parameter may be in the range zero through 31. The default value of this 32-bit configuration parameter is zero.

`bn-p-mbox-offset` specifies the physical address of the participant's mailbox. The default value of this 32-bit configuration parameter depends on the hardware capabilities of the specific machine.

`bn-p-mbox-space` specifies the space in which the participant's mailbox is accessible. Typically, this configuration parameter identifies one of the address spaces available in the address range of the bus.
The default value of this 32-bit configuration parameter depends on the hardware capabilities of the specific machine.

`bn-p-mbox-access` specifies the access mode of the participant's mailbox. The default value of this 32-bit configuration parameter depends on the hardware capabilities of the specific machine.

`bn-p-mbox-intr` specifies the interrupt generated when the participant's mailbox is being accessed from the bus. The default value of this 32-bit configuration parameter depends on the hardware capabilities of the specific machine.

`bn-p-mbox?` specifies whether the participant provides a mailbox. When this configuration parameter is `true` then the participant provides a mailbox. Otherwise, the participant does not provide a mailbox. The default value of this

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

configuration parameter depends on the hardware capabilities of the specific machine.

`bn-packet-size` specifies the size of a BusNet packet. The minimum packet size allowed by the BusNet protocol is 2 KByte. The default value of this configuration parameter is 2 KByte. If set to another value it must be a multiple of 64 bytes.
The BusNet protocol does not permit participants to use different packet buffer sizes during initialisation.
The default value of this 32-bit configuration parameter is 2048_{10}.

A participant is designated as *master* when the following pairs of configuration parameters `bn-master-space`, `bn-p-space` and `bn-master-offset`, `bn-p-offset` are identical. When these configuration parameters are different, the participant is designated as *slave*. However, OpenBoot does *not* support the master operation of a participant.

IMPORTANT



The default values of some described NVRAM configuration parameters may vary depending on the VMEbus interface of the particular machine (S4, MVIC, FGA-5000), especially the parameters describing the mailbox of the participant.

The state of the NVRAM configuration parameters listed below are only considered when the Trivial File Transfer Protocol (TFTP) is used to load and execute an image across the network using the BusNet protocol:

`bn-arp?` specifies whether the BusNet driver should scrutinize all outgoing packets and verifies whether an Ethernet frame carries an ARP request.
When the flag is `true`, the BusNet driver checks whether an Ethernet frame contains an ARP request and, if so, it resolves the request and passes the response to the receiving part of the BusNet driver automatically. The Ethernet frame is *not* sent across the network.
The BusNet driver uses the contents of the NVRAM configuration parameters `bn-master-ip-addr`, `bn-p-ip-addr`, `bn-master-en-addr`, and `bn-p-en-addr` to build up the appropriate response.
In the case that the flag is `false`, it sends all Ethernet frames without any further verification across the network. (default: `false`)

`bn-rarp?` specifies whether the BusNet driver should scrutinize all outgoing packets and verifies whether an Ethernet frame carries an RARP request.
When the flag is `true`, the BusNet driver checks whether an Ethernet frame contains a RARP request and, if so, it resolves the request and passes the response to the receiving part of the BusNet driver automatically. The Ethernet frame is *not* send across the network.
The BusNet driver uses the contents of the NVRAM configuration parameters `bn-master-ip-addr`, `bn-p-ip-addr`, `bn-master-en-addr`, and `bn-p-en-addr` to build up the appropriate response.

20-47 18 2-2000.146 420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

In the case that the flag is `false`, it sends all Ethernet frames without any further verification across the network. (default: `false`)

`bn-master-ip-addr` specifies the Internet Protocol (IP) Address of the master. The default value of this 32-bit configuration parameter is zero (`0`). The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-master-ip-addr 0x83030001
```

In the example, the Internet address `131.3.0.1` (83030001_{16}) is assigned to the NVRAM configuration parameter.
This configuration parameter *must* be set when one of the two configuration parameters `bn-arp?` or `bn-rarp?` are set to `true`.

`bn-p-ip-addr` specifies the Internet Protocol (IP) Address of the participant. The default value of this 32-bit configuration parameter is zero (`0`). The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-p-ip-addr 0x83030002
```

In the example, the Internet address `131.3.0.2` (83030002_{16}) is assigned to the NVRAM configuration parameter.
This configuration parameter *must* be set when one of the two configuration parameters `bn-arp?` or `bn-rarp?` are set to `true`.

`bn-master-en-addr` specifies the Ethernet address of the master. The Ethernet address is represented by an ASCII string in the following format: `XX:XX:XX:XX:XX:XX` – where `XX` is a hexadecimal number. The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-master-en-addr 0:80:42:b:10:ac
```

This configuration parameter *must* be set when one of the configuration parameters `bn-arp?` and `bn-rarp?` are set to `true`.

`bn-p-en-addr` specifies the Ethernet address of the participant. The Ethernet address is represented by an ASCII string in the following format: `XX:XX:XX:XX:XX:XX` – where `XX` is a hexadecimal number. The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-p-en-addr 0:80:42:b:10:ad
```

This configuration parameter *must* be set when one of the configuration parameters `bn-arp?` and `bn-rarp?` are set to `true`.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.8.4 Device Operation

In general, OpenBoot provides the `boot` command to load a program through a communication channel into memory. (For detailed information about the `boot` command and the associated NVRAM configuration parameters refer to the OpenBoot Command Reference.) The *device-specifier* specifies the physical device that is attached to the communication channel. A program is loaded across the VMEbus – using the BusNet protocol – by

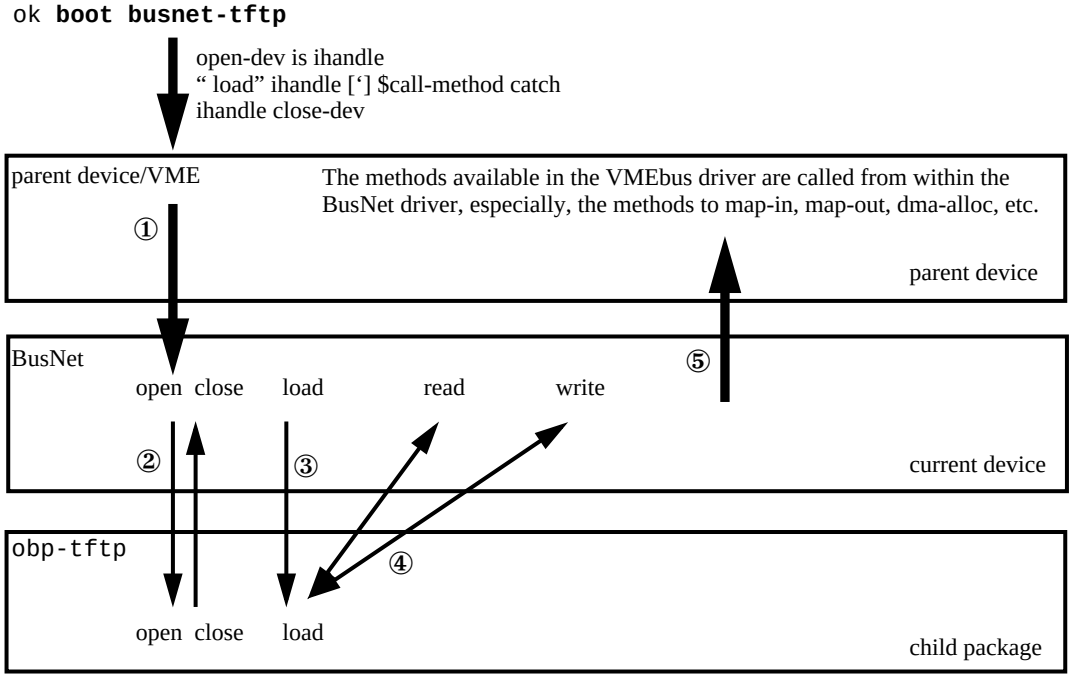
```
ok boot busnet
```

or

```
ok boot busnet-tftp
```

The device aliases `busnet` and `busnet-tftp` specify the BusNet device used to load the program. Both aliases contain the argument string `tftp` which informs the BusNet device to use the Trivial File Transfer Protocol TFTP to load the program, and the BusNet driver replaces the medium access layer MAC, which usually is Ethernet.

Figure 28 Calling the OpenBoot `boot` command using `busnet-tftp`



When the `boot` command is called – as shown in the figure above – OpenBoot tries to locate the specified device in its device tree and, opens each node of the device tree in turn, starting at the top until the BusNet device is reached ①. Assuming the TFTP protocol is used to load the program, the BusNet driver tries to open the package `obp-tftp` provided

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

by OpenBoot and returns control to the `boot` command after the execution of its `open` method is complete ②.
In the next step, the boot command calls the BusNet driver's `load` method, which in turn calls the `load` method of the TFTP package to load the program ③.
During the time the program is loaded, the TFTP package controls operation and calls the methods `read` and `write` of its *parent device* ④– the BusNet device – to receive and transmit packets across the network. Once the program has been loaded, the control is passed back to the BusNet device, and the `boot` command. The latter calls the `close` method of the BusNet device which in turn calls the `close` method of the TFTP package. Finally, control is returned to the `boot` command.
The BusNet device calls the methods of its *parent* device, that is the VMEbus device. Typically, the BusNet driver calls the methods to make its BusNet region available to the VMEbus address space and to map this region to the processor's virtual address space ⑤.

5.8.5 How to Use BusNet

The `/busnet-demo` package is available in OpenBoot to demonstrate how to operate the BusNet driver in the *raw* mode. In this mode pure binary data are sent across the network from one BusNet participant to another participant. The following two definitions are available to initiate the transmission and receipt of data:

`demo-send-data` (*src-addr size dest-p#* —) sends the amount of data specified by *size* and stored beginning at the address *src-addr* to the participant identified by its logical BusNet address *dest-p#*.

`demo-receive-data` (*dest-addr size src-p#* —) receives as much data as specified by *size* from the participant identified by its logical BusNet address *dest-p#* and stores it beginning at the address *dest-addr*.

IMPORTANT

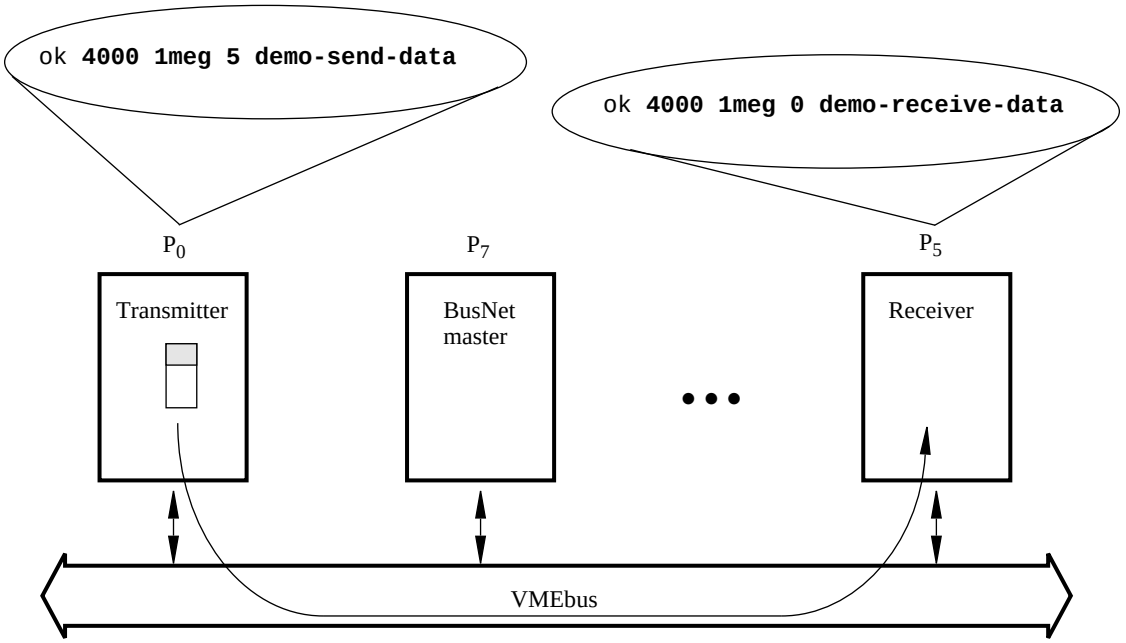


When these commands are used to exchange data between two participants running OpenBoot, then a third participant must be available which provides BusNet master functionality. This is necessary because OpenBoot does not provide BusNet master functionality!

As shown in the figure below, three participants take part in communicating across the network using the BusNet protocol. The logical address of the participants are zero, seven and five. The participants P_0 and P_5 are executing OpenBoot, and the participant P_7 runs an operating system which is capable of providing BusNet master functionality – for example Solaris/SunOS, or VxWorks.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Figure 29 Transferring data using the BusNet protocol



When a certain amount of data located in the on-board memory of the participant zero (P_0) – the transmitter – should be transferred to the participant five (P_5) – the receiver – then the following command must be used on the transmitter:

```
ok 4000 1meg 5 demo-send-data
```

This command initiates a transmission of 1 Mbyte of data located at address 4000_{16} in the transmitter's on-board memory to the receiver. To enable the receiver to receive the data the following command must be used:

```
ok 4000 1meg 0 demo-receive-data
```

This command initiates the receipt of data from the participant zero and stores the data beginning at address 4000_{16} in the receiver's on-board memory.

IMPORTANT

To ensure proper operation of the data exchange, the size applied to the commands on the receiver and transmitter must be the same!

5.8.6 Using `bn-dload` to Load from the Backplane

The command `bn-dload` loads a file across the network and stores it at a specific address, as shown in the example below:

```
ok 4000 bn-dload filename
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The *filename* must be relative to the server's root, and the contents of the file are stored beginning at address 4000_{16} within the on-board memory. The command `bn-dload` uses the Trivial File Transfer Protocol (TFTP) to load the file.

FORTH Programs

FORTH programs to be loaded with `bn-dload` must be ASCII files beginning with the two characters "\ " (backslash immediately followed by a space). To execute the loaded FORTH program, the `eval` command has to be used as follows:

```
ok 4000 file-size @ eval
```

The variable `file-size` contains the size of the loaded file.

FCode Programs

FCode programs to be loaded with `bn-dload` must be in the `a.out` format. To execute the loaded FORTH program, the `byte-load` command has to be used as follows:

```
ok 4000 1 byte-load
```

The command `byte-load` is used by OpenBoot to interpret FCode programs on expansion boards such as SBus cards. The second argument passed to this command – value one (`1`) in the example – specifies the separation between FCode byte in general. Because the `bn-dload` command loads the FCode into on-board memory, the spacing is one (`1`).

Binary Executables

Executable binary programs to be loaded with `bn-dload` must be in the `a.out` format. To execute the binary program, the `go` command has to be used as follows:

```
ok go
```

When the program should be started again, the commands listed below have to be used:

```
ok init-program go
```

5.8.7 Booting from a Solaris/SunOS BusNet Server

When Solaris/SunOS is loaded and executed from a Solaris/SunOS BusNet server, the boot command has to be used as follows:

```
ok boot busnet
```

In this case, OpenBoot will load the appropriate primary booter from the server using the Trivial File Transfer Protocol (TFTP), and start execution of the loaded image.
When the Solaris/SunOS is loaded and executed automatically after each system reset, the NVRAM configuration parameter `auto-boot?` must be set to `true`, and depending on the state of the configuration parameter `diag-switch?`, either `boot-device` or `diag-device` must be

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

set. When the diagnostic mode is disabled, the configuration parameter `boot-device` must be set as follows:

```
ok setenv boot-device busnet
```

And in the case that the diagnostic mode is enabled, the configuration parameter `diag-device` must be set as described in the following:

```
ok setenv diag-device busnet
```

5.8.8 Booting from a VxWorks BusNet Server

Because VxWorks currently is not capable of resolving RARP requests, the NVRAM configuration parameters listed below must be set prior to loading an executable image.

`bn-rarp?` specifies whether the BusNet driver should scrutinize all outgoing packets and verifies whether an Ethernet frame carries an RARP request. The flag must be set to `true`, to enable the BusNet driver to check whether an Ethernet frame contains a RARP request, and if so, it resolves the request and passes the response to the receiving part of the BusNet driver automatically. The Ethernet frame is not sent across the network. The BusNet driver uses the contents of the NVRAM configuration parameters `bn-master-ip-addr`, `bn-p-ip-addr`, `bn-master-en-addr`, and `bn-p-en-addr` to build up the appropriate response.

`bn-master-ip-addr` specifies the Internet Protocol (IP) Address of the master. The default value of this 32-bit configuration parameter is zero (`0`). The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-master-ip-addr 0x83030001
```

In the example, the Internet address `131.3.0.1` (83030001_{16}) is assigned to the NVRAM configuration parameter.

`bn-p-ip-addr` specifies the Internet Protocol (IP) Address of the participant. The default value of this 32-bit configuration parameter is zero (`0`). The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-p-ip-addr 0x83030002
```

In the example, the Internet address `131.3.0.2` (83030002_{16}) is assigned to the NVRAM configuration parameter.

`bn-master-en-addr` specifies the Ethernet address of the master. The Ethernet address is represented by an ASCII string in the following format: `XX:XX:XX:XX:XX:XX` – where `XX` is a hexadecimal number. The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-master-en-addr 0:80:42:b:10:ac
```

20-4718-2-2000146-420.000-1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

`bn-p-en-addr` specifies the Ethernet address of the participant. The Ethernet address is represented by an ASCII string in the following format: `XX:XX:XX:XX:XX:XX` – where `XX` is a hexadecimal number. The `setenv` command is used to set this configuration parameter as shown below:

```
ok setenv bn-p-en-addr 0:80:42:b:10:ad
```

Assuming the participant's Ethernet- and Internet address are `0:80:42:b:10:ad` and `131.3.0.2`, and the VxWorks server's Ethernet- and Internet address are `0:80:42:b:10:ac` and `131.3.0.1`, then the NVRAM configuration parameters listed above must be set as described below:

```
ok setenv bn-master-en-addr 0:80:42:b:10:ac
ok setenv bn-master-ip-addr 0x83030001
ok setenv bn-p-en-addr 0:80:42:b:10:ad
ok setenv bn-p-ip-addr 0x83030002
ok setenv bn-rarp? true
```

After these NVRAM configuration parameters have been set, the OpenBoot BusNet driver scrutinizes every outgoing packet that carries an Ethernet frame and verifies whether the Ethernet frame contains an RARP request. If so, the BusNet driver resolves the RARP request – using the information contained by the configuration parameters mentioned above – and passes the response internally to the receiving part of the BusNet driver. All other packets are sent across the network.
After this, the `boot`, `load` or `bn-dload` command can be used to load an executable image from the VxWorks server. In case of the first two commands, the name of the image being loaded is always the name of the primary booter (e.g. `83030002.SUN4M`).

5.8.9 Setting NVRAM Configuration Parameters

The SPARC/CPU-8VT is equipped with the SPARC FGA-5000 VMEbus Interface Chip which provides a mailbox register located in the *short* address space (A16) of the VMEbus. To enable the mailbox the following NVRAM configuration parameters must be set in addition to the NVRAM configuration parameters listed in the table below:

`vme-a16-slave-addr` must be set to $YY00_{16}$ where `YY` is one of the values 00_{16}, 02_{16}, 04_{16}, …, FC_{16}, or FE_{16}. This means that the base address of the SPARC FGA-5000 registers *must* be aligned to a 512-Byte boundary.

`vme-a16-slave-ena?` must be set to `true`.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 49 NVRAM configuration parameters

Parameter	Default	Description
bn-master-offset	00000000_{16}	
bn-master-space	$3D_{16}$	privileged standard (A24) address range
bn-master-access	32_{16}	read/write/D32
bn-p-offset	00000000_{16}	
bn-p-space	$3D_{16}$	privileged standard (A24) address range
bn-p-access	32_{16}	read/write/D32
bn-p-mbox?	true	mailbox available (FGA-5000 Mailbox #0)
bn-p-mbox-offset	0120_{16}	offset of mailbox #0
bn-p-mbox-space	$2D_{16}$	privileged short (A16) address range
bn-p-mbox-access	10_{16}	read/D8
bn-p-mbox-intr	5	SBus interrupt level 5 is asserted upon a mailbox

20-4718-2 2.000/0.146 420.000/1 March 1993

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Please Note…

The Sun OpenBoot section is an integral part of the *SPARC/CPU-8VT Technical Reference Manual* (P/N 204718). Yet, it is packaged separately to enable easy updating.

The Sun OpenBoot section will always be shipped together with the *Technical Reference Manual*.

Please:

☞ **Insert the Sun OpenBoot section (P/N –) now into the *SPARC/CPU-8VT Technical Reference Manual* (P/N 204718).**

☞ **Remove this sheet.**

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

6 Sun OpenBoot (= OPEN BOOT PROM 2.0 MANUAL SET)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP