

SPARC CPU-2CE
TECHNICAL REFERENCE MANUAL

REVISION NO. A1
JANUARY 1994

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

NOTICE

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 that 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 GmbH/Inc.

FORCE COMPUTERS does not convey to the purchaser of the product described herein any license under the patent rights of FORCE COMPUTERS GmbH/Inc. nor the rights of others.

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

Phone : (408) 371-5900
FAX : (408) 371-3382

FORCE COMPUTERS FRANCE S.A.R.L.
Le Volta
17-19 rue Jeanne Braconnier
F-92366 Meudon La Foret Cedex
France

Phone : (1) 41 07 95 15
FAX : (1) 45 37 06 19

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

Phone : (089) 608 14-0
Telex : 524190 forc-d
FAX : (089) 609 77 93

FORCE COMPUTERS (U.K.) Ltd
No. 1 Holly Court
3 Tring Road
Wendover
Buckinghamshire HP
England

Phone : (0296) 625456
Telex : 838033
FAX : (0296) 624027

FORCE COMPUTERS strives to contribute to the safety and preservation of the environment with the same care it applies to the performance and quality of its products.

To support this effort, this Product Manual has been printed on paper that is completely chlorine free and conservation friendly.

TABLE OF CONTENTS

Section 1

1.0 INTRODUCTION	1-1
1.1 The Processor	1-4
1.2 The Memory Subsystem	1-4
1.3 System EPROM Open Boot™:	1-4
1.4 FLASH EEPROM (Optional Feature)	1-5
1.5 Local I/O	1-5
Serial Interface Ports	1-5
Keyboard/Mouse Port	1-5
Floppy Disk Controller	1-5
Audio Port	1-5
1.6 DMA+	1-6
1.7 Small Computer Systems Interface (SCSI)	1-6
1.8 Ethernet Interface	1-6
1.9 Real-Time Clock/NVRAM	1-6
1.10 SBus Interface	1-7
1.11 VMEbus Interface	1-7
1.11.1 Master Interface	1-7
1.11.2 Slave Interface	1-8
1.11.3 Interrupts	1-8
1.11.4 System Controller	1-8
1.12 Software Description	1-8
1.13 Development	1-9
1.14 Sun Tools	1-9
1.15 Graphical User Interface	1-9
1.16 Real-Time	1-11
1.17 Networking	1-11
1.18 Determining Revision Level	1-13

Section 2

INSTALLATION and DIAGNOSTICS	2-1
2.1 Power Up	2-2
2.2 Post Power Up Procedures	2-4
2.2.1 How to Talk to the CPU-2CE On-Board Memory	2-4
2.2.1.1 Memory Test Command One	2-4
2.2.1.2 Memory Test Command Two	2-4
2.2.1.3 Memory Test Command Three	2-5
2.2.2 Test the Ethernet Port	2-5
2.2.3 How to Talk to the SPARC CPU-2CE Buses	2-6
2.2.3.1 Mapping Memory	2-6
2.2.3.2 Accessing Memory	2-6
2.2.4 How to Talk to the SBus	2-6

TABLE OF CONTENTS "continued"

2.3	Running PROM Diagnostics	2-6
2.4	Running Functional Tests	2-7
2.5	Returning To Monitor Mode	2-7
2.6	Diagnostics	2-7
2.6.1	Main Categories of Diagnostics	2-7
2.6.2	Diagnostic Selections	2-8
2.6.3	Boot PROM Diagnostics	2-11
2.6.3.1	POST	2-11
2.6.3.2	On Board Diagnostics	2-13
2.6.4	Sundiag System Exerciser	2-13
2.6.5	Monitor and Forth Toolkit	2-13
2.7	SPARC CPU-2CE NVRAM Parameters	2-16

Section 3

HARDWARE	3-1
3.1 Card Landmarks	3-1
3.1.1 Card Jumpers	3-5
3.1.2 Device Space	3-11
3.1.3 Control Space	3-12
3.1.4 The Memory Map Page	3-15
3.1.5 Interrupt Map	3-17
3.2 Custom Chips	3-18
3.2.1 Cache+	3-18
3.2.2 RAM+	3-18
3.2.3 DMA+	3-18
3.2.4 MMU+	3-18
3.2.5 SPARC CPU-2CE CACHE Controller	3-18
The Cache Controller features:	3-18
The Cache Controller functions:	3-19
Function of Write Through Cache Control	3-19
3.3 SPARC Expansion Connector	3-22
3.3.1 Overview	3-22
3.3.1.1 Intent of the Expansion Connector	3-22
3.3.2 Reconciling the Differences	3-22
3.3.2.1 Pa26	3-22
3.3.2.2 Pa25	3-22
3.3.2.3 POK	3-23
3.3.2.4 Ramclk N	3-23
3.3.2.5 Ramsel* vs. ramsel1*	3-23
3.3.2.6 Parity N vs parcs1*, ramclk1, pa25, and perr*	3-23

TABLE OF CONTENTS "continued"

3.4	SCSI Interface	3-24
3.4.1	SCSI Definition	3-24
3.4.2	SCSI Performance	3-24
3.4.3	SCSI Addresses	3-25
3.4.4	SCSI Interrupt	3-25
3.4.5	Interface Programming	3-26
3.5	Ethernet Interface	3-27
3.5.1	Ethernet Interface Definition	3-27
3.5.1.1	Ethernet Transmits and Receives	3-27
3.5.1.2	Ethernet Buffer	3-27
3.5.2	Ethernet Performance	3-27
3.5.3	Ethernet Interrupt	3-27
3.5.4	Ethernet Addresses	3-28
3.5.5	Ethernet Interface Programming	3-28
3.6	SBus Interface	3-31
3.6.1	Introduction	3-31
3.6.2	SBus Connections	3-31
3.6.3	SBus Slot Addresses	3-32
3.6.4	SBus Slot 0 Devices	3-33
3.6.4.1	Card ID	3-33
3.6.4.2	DMA Registers	3-34
	DMA Control/Status Register	3-35
	DMA Address Register	3-36
	DMA Byte Count Register	3-36
	DMA Diagnostic Register	3-36
3.6.5	SBus Slot 1 Devices and Slot 2 Devices	3-36
3.6.5.1	SPARC Assembly Language Example	3-36
3.6.5.2	Forth Example	3-39
3.6.5.3	C/SunOS Example	3-39
3.7	RAM+	3-40
3.7.1	FEATURES	3-40
3.7.2	PIN DESCRIPTION	3-40
3.7.2.1	Sbus Interface	3-40
3.7.2.2	DRAM Interface	3-41
3.7.2.3	Power/Ground	3-41
3.7.2.4	LSI Logic Buffer Naming Conventions	3-41
3.7.3	SBus INTERFACE	3-42
3.7.3.1	DRAM Access	3-42
	• Word Read	3-43
	• Word Write - Buffered	3-44
	• Burst Read	3-45
	• Burst Write	3-46
3.7.3.2	Address Mapping	3-47
3.7.4	DETAILED TIMING	3-48
3.7.4.1	DRAM Timing	3-48
	DRAM Errors	3-48
	DRAM	3-48

TABLE OF CONTENTS "continued"

3.8 The MMU+	3-49
3.8.1 Features	3-49
3.8.2 Pin Description	3-50
3.8.2.1 SBus Interface	3-50
3.8.2.2 MMU Interface	3-50
3.8.2.3 Decodes	3-51
3.8.2.4 Miscellaneous	3-52
3.8.3 Clock Generation	3-52
3.8.3.1 General Description	3-52
3.8.3.2 Constant Clock	3-53
3.8.3.3 Serial Clock	3-53
3.8.4 Decodes	3-53
3.8.4.1 Control Space	3-53
• CTL mappings	3-53
• Context Register Access	3-54
• SCC Bypass	3-54
• EPROM Bypass	3-54
• Segment Map Access	3-55
• Page Map Access	3-56
• Page Type Bits	3-56
3.8.4.2 Device Space	3-57
• SBus or DRAM Access	3-58
• TOD, EPROM, Floppy Access	3-59
• Parity Register Access	3-60
• SCC/KB Mouse Access	3-60
3.8.5 Protection Checking	3-61
3.8.6 Statistics Updates	3-61
3.8.7 Interrupts	3-62
3.8.8 Counters	3-62
3.8.9 I/O Register	3-63
3.9 P2 Bus Interface Overview	3-64
3.9.1 P1/P2 Connector Numbering	3-64
3.10 Serial Interface A and B	3-64
3.10.1 Serial Interface A and B Device Address	3-64
3.10.1.1 RS232/RS423 Jumper Selection	3-64
3.10.2 Serial Interface A/B Definition	3-64
3.10.3 Serial Interface A/B Performance	3-64
3.11 Keyboard/Mouse Interface	3-66
3.11.1 Keyboard/Mouse Device Address	3-66
3.11.2 Keyboard/Mouse Interface Definition	3-66
3.11.3 Specifications	3-66

TABLE OF CONTENTS "continued"

Section 4

APPENDICES TO THE HARDWARE USER'S MANUAL	4-1
--	-----

The CPU Card Schematic Diagrams	
DWG Schematics	
Serial Cable Drawing	
Ethernet Cable Drawing	
Reader Comment Card	
Product Error Report	

Section 5

ABSTRACT COPIES OF DATA SHEETS	5-1
--------------------------------------	-----

Section 6

PROGRAMMING & FIRMWARE MANUAL	6-1
6.0 Programming	6-1
6.1 Address Spaces	6-1
6.1.1 Control Space	6-1
6.1.2 System Space (ASI = 2)	6-2
6.1.3 Device Space	6-7
6.2 Memory Management	6-12
6.2.1 Address Translation	6-13
6.2.2 Modifying the MMU	6-13
6.2.3 MMU Protection Criteria	6-13
6.3 Programming LED's	6-14
6.3.1 SPARC CPU-2CE LED Programming Example	6-15
6.3.1.1 Example of the ce-misc.h file	6-16
6.4 Additional Forth Commands for LEDs/Hexswitch	6-16

Section 7

VME	7-1
7.0 VMEbus Interface	7-1
7.1 Features of the SPARC CPU-2CE VMEbus Interface	7-1
7.2 VMEbus Basics - An Introduction	7-3
7.3 VME Performance	7-3
7.4 VME Addresses	7-4
7.5 VME Implementation	7-5

TABLE OF CONTENTS "continued"

7.6	Major VME Register Groups	7-6
7.6.1	Accesses To Byte Registers	7-7
7.7	Master Interface	7-8
7.7.1	A32 Map Register Base Location	7-8
7.7.2	A32 Map Register Initialization	7-8
7.7.3	VME Registers Programming Example	7-10
7.8	Slave Interface	7-12
7.8.1	Slave Transfer Control	7-12
7.8.2	Slave Map Register	7-12
7.8.2.1	Slave Map Register Initialization	7-13
7.9	Mail Box	7-14
7.9.1	Mail Box Register Base Location	7-15
7.9.1.1	Mail Box Register Initialization	7-15
7.9.1.2	Mail Box Register Interrupt Level	7-15
7.10	Mailbox Interrupt Level - Rerun Length Register	7-16
7.11	Bus Locker	7-17
7.11.1	VME Bus Locker Register	7-17
7.11.2	Initialization	7-18
7.12	Interrupt Handler	7-19
7.12.1	Interrupt Enable/Bus Arbiter Mode Register	7-19
7.12.1.1	Interrupt Enable Register Initialization	7-19
7.13	Bus Requester	7-20
7.13.1	Bus Arbiter	7-20
7.14	Bus Time Out Period	7-20
7.14.1	Rerun Time Out	7-20
7.14.2	Abort	7-20
7.15	VMEbus Watchdog Timer	7-21
7.16	System Reset and the Reset Switch	7-21
7.16.1	Sources for a System Reset	7-21
	"The Power-On Reset"	7-21
	"The Reset Switch"	7-21
	"The Watchdog Reset"	7-22
	"The Software Reset"	7-22
7.17	Jumper	7-22
7.18	Programmable Register Settings	7-22
7.19	VME Interrupt Monitor Register	7-23
7.19.1	Fair Mode Requester	7-23
7.20	VME IACK Cycles	7-23
7.20.1	Daisy Chain IACK Driver	7-24
7.20.2	Master Cycles	7-25
7.20.3	Slave Cycles	7-25
7.21	Bus Arbitration	7-25
7.22	Interrupts	7-26
7.23	Example of a VME System	7-x
7.24	VMEbus Device Driver	7-28
7.24.1	VMEbus Device Driver System Calls	7-28

TABLE OF CONTENTS "continued"

7.24.2	VMEbus Address Modifiers	7-30
	Program/Data/Block Transfer Address Modifiers	7-30
	Supervisory/Non-Privileged Address Modifiers	7-31
	Extended/Standard/Short Address Modifiers	7-31
7.24.3	VMEbus Device Driver Limitations	7-31
	Floppy and Audio Interrupt Conflicts	7-31
	Potential for SunOS Patches to Conflict	7-32
	GENERIC.VME versus GENERIC_SMALL.VME	7-32
7.25	VME Programming Examples	7-32
7.25.1	FORTH Programming Examples	7-32
	Map the VMEbus to a Virtual Address	7-32
	Map Local Memory to the VMEbus	7-33
	Map an SBUS Address to a Virtual Address	7-36
7.25.2	C Programming Example	7-36
 Section 8		
ONC/VME	8-1
8.1	Capabilities	8-1
8.2	Using ONC/VME	8-1
8.3	Booting Over the VMEbus Backplane	8-2
8.3.1	Installation Example	8-2
8.3.2	Hardware Configuration	8-3
8.3.3	Software Configuration	8-3
8.3.4	PROM/NVRAM Configuration	8-3
8.3.4.1	vme-slavemap	8-4
8.3.4.2	boot-device	8-4
8.3.4.3	vm-server-slavemap	8-4
8.3.4.4	vm-server-addr	8-4
8.3.4.5	vm-ip-addr	8-4
8.3.5	Initializing the System	8-5
8.3.6	Files To Edit on the Server	8-5
8.3.6.1	/sys/sun4c/conf/GENERIC.VME	8-5
8.3.6.2	/etc/hosts	8-5
8.3.6.3	/etc/ethers	8-5
8.3.6.4	ONC Driver Interrupt Conflict with the Audio and Floppy Drivers	8-6
8.3.7	Adding a Client	8-6
8.3.7.1	/etc/hostname.vm0	8-6
8.3.7.2	/etc/bootparams	8-6
8.3.7.3	/export/root/client-vm/etc/fstab	8-7
8.3.7.4	/export/root/client-vm/etc/hosts	8-7
8.3.7.5	/export/root/client-vm/etc/hostname.vm0	8-7
8.3.8	System Start-Up	8-7
8.4	Activating ONC/VME	8-7
8.4.1	Setting Up a Private Network	8-8
8.5	Theory of Operation	8-9

TABLE OF CONTENTS "continued"

8.6 Memory Usage	8-9
8.6.1 Addressing Scheme	8-9
8.6.2 ONC/VME Region Layout	8-10
8.6.2.1 Magic Number	8-11
8.6.2.2 Packet-Ready Flags	8-11
8.6.2.3 Host States	8-11
8.6.2.4 IP Addresses	8-11
8.6.2.5 Valid-Host Flags	8-11
8.6.2.7 Packet Contents	8-11
8.7 Protocol Operation	8-11
8.7.1 Initialization	8-12
8.7.1.1 Initializing the Hardware	8-12
8.7.1.2 Initializing the Local Region	8-12
8.7.1.3 Probing for Remote Hosts	8-12
8.7.2 Sending a Packet	8-12
8.7.2.1 Broadcast Packets	8-13
8.7.3 Receiving a Packet	8-13
8.7.3.1 Shutting Down	8-14
8.7.3.2 Error Handling	8-14
Section 9	
OPTIONS AND APPLICATIONS	9-1
Section 10	
GLOSSARY	10-1
SPARC CPU-2CE Hardware Documentation Glossary	10-1
SPARC Reference Materials Available from Sun	10-1
Documentation Available Other Places	10-1
Suggested Reference Material for the SCSI Interface	10-2
Ethernet Interface	10-2
Suggested Reference Material for Serial Interface A/B	10-2
Suggested Reference Material for Keyboard/Mouse Interface	10-2
Suggested Reading for FORTH	10-3
Section 11	
CPU-2CE OPEN BOOT SUPPLEMENT	11-1

LIST OF TABLES

Table: 1.1	Board Function & Specification	1-11
Table: 1.2	Compatible Peripherals	1-13
Table: 2.1	Diagnostic Tools	2-8
Table: 2.2	Diagnostic Switches	2-11
Table: 2.3	Non-Volatile System Configuration Parameter Defaults	2-14
Table: 2.4	NVRAM VME Configuration Parameters	2-15
Table: 2.5	Front Panel	2-17
Table: 2.6	CPU-2CE Connectors	2-17
Table: 2.7	SCSI Pinout List	2-21
Table: 2.8	Ethernet Pinout List	2-22
Table: 2.9	Serial Pinout List	2-24
Table: 2.10	Audio Connector (Din 8)	2-26
Table: 2.11	Keyboard/Mouse Connector Pinout List	2-27
Table: 2.12	SBus Connector Pinout	2-28
Table: 2.13	P1 Bus Pinout List	2-29
Table: 2.14	P2 Bus Pinout List	2-30
Table: 2.15	Expansion Connector Electrical Pinout	2-31
Table: 3.1	Virtual Memory Layout	3-9
Table: 3.2	Address Space Indicator Contents	3-11
Table: 3.3	SPARC CPU-2CE Interrupt Priority Levels	3-17
Table: 3.4	SBus Bandwidth Estimation	3-21
Table: 3.5	SCSI Register Addresses	3-25
Table: 3.6	Ethernet Registers	3-28
Table: 3.7	SBus Interrupts	3-31
Table: 3.8	SBus Slot Addresses	3-32
Table: 3.9	SBus Slot 0 Addresses	3-33
Table: 3.10	DMA Register Addresses	3-34
Table: 3.11	Address Ranges	3-42
Table: 3.12	DRAM maps physical addresses to SIMM Locations	3-48
Table: 3.13	CTL mappings	3-53
Table: 3.14	Page Type Bit Definitions	3-56
Table: 3.15	Device Space	3-57
Table: 6.1	Address Space Indicator Contents	6-2
Table: 6.2	System Space (ASI=2)	6-3
Table: 6.3	System Enable Register System Enable Register	6-3
Table: 6.4	Synchronous Error Register	6-4
Table: 6.5	Synchronous Error Register	6-4
Table: 6.6	Synchronous Error Register	6-5
Table: 6.7	Synchronous Error Register	6-5
Table: 6.8	Synchronous Error Register	6-5
Table: 6.9	Asynchronous Error Register	6-6
Table: 6.10	Cache Tags	6-6
Table: 6.11	Type One Space	6-8
Table: 6.12	TOD/NVRAM	6-9
Table: 6.13	Counters timers	6-9

LIST OF TABLES "continued"

Table: 6.14	PError	6-10
Table: 6.15	Type 1 Space for off Board DRAM	6-11
Table: 6.16	Interrupt Control Register	6-11
Table: 6.17	Aux I/O Register	6-12
Table: 6.18	LED/SW3 Register (Longword access only)	6-14
Table: 7.1	VME Concept Definitions	7-3
Table: 7.2	VME Addresses	7-4
Table: 7.3	VME Address Spaces	7-5
Table: 7.4	VME Registers	7-7
Table: 7.5	A32 Map Register	7-9
Table: 7.6	A32 Map Register Address Bits	7-9
Table: 7.7	A32 Map Register Bit Definitions	7-9
Table: 7.8	Slave Map Register Address	7-13
Table: 7.9	Slave Map Register Address Bits	7-13
Table: 7.10	Slave Master Register Bit Definitions	7-14
Table: 7.11	Mail Box Register Address	7-15
Table: 7.12	Mail Box Register Address Bits	7-15
Table: 7.13	Mail Box Register Bit Definitions	7-16
Table: 7.14	Bus Locker Register Bit Definitions	7-18
Table: 7.15	Interrupt Enable Register Address	7-19
Table: 7.16	Interrupt Enable Register Bit Definitions	7-19
Table: 7.17	Slot 1 Jumper & Functions	7-22
Table: 7.18	Programmable Register Settings	7-22
Table: 7.19	VME Interrupt Monitor Register Address	7-23
Table: 7.20	Interrupt Monitor Register Address Definitions	7-23
Table: 7.21	Mail Box Register Address	7-24
Table: 7.22	VME IACK Cycles Register Address Bits	7-24
Table: 7.23	VME IACK Cycles Interrupt Responses	7-24
Table: 7.24	Slave Cycles Duration and Theoretical Bandwidth	7-25
Table: 7.25	Bus Arbitration	7-25
Table: 7.26	VME Interrupt Levels and Sources	7-26
Table: 7.27	A16 - only Address Decode	7-31
Table: 7.28	A24 - Only Address Decode	7-31
Table: 7.29	Address Modifier Codes	7-32
Table: 8.1	Example Server Configuration	8-2
Table: 8.2	Example Client Configuration	8-3
Table: 8.3	ONC/VME Region Layout	8-10
Table: 11.1	SPARC Reference Materials Available from Sun	11-1
Table: 11.2	Documentation Available Other Places	11-1

LIST OF FIGURES

Figure: 1.1	Block Diagram	1-2
Figure: 1.2	Photo Page	1-3
Figure: 2.1	RS-232/423 Jumper Blocks	2-3
Figure: 2.2	POST	2-9
Figure: 2.3	Keyboard LED Diagnostic Codes	2-12
Figure: 2.4	Font Panel	2-18
Figure: 2.5	Component Side	2-19
Figure: 2.6	Solder Side	2-20
Figure: 2.7	SCSI Connector and Pins (Front View)	2-21
Figure: 2.8	Ethernet Cable Connector and Pins (Front View)	2-22
Figure: 2.9	Ethernet Cable	2-23
Figure: 2.10	Serial Cable	2-25
Figure: 2.11	Audio Connector	2-26
Figure: 2.12	Keyboard/Mouse Connector	2-27
Figure: 2.13	Sun Expansion Connector Pin Orientation	2-31
Figure: 3.1	Shows the Front Panel	3-1
Figure: 3.2	Shows the component-side	3-1
Figure: 3.3	Shows the solder	3-1
Figure: 3.4	RS-232/423 Jumper Blocks	3-6
Figure: 3.5	SPARC/CPU-2CE ADDRESS SPACE VIRTUAL	3-7
Figure: 3.6	Physical Memory Map	3-8
Figure: 3.7	Virtual/Physical Memory Map ASI Diagram	3-14
Figure: 3.8	Type 1 Space (On-Board I/O) (obio)	3-15
Figure: 3.9	Type 0 Space (Main Memory) (obmem)	3-16
Figure: 3.10	Safe Decode Circuit	3-23
Figure: 3.11	Word Read	3-43
Figure: 3.12	Word Read 2	3-43
Figure: 3.13	Word Write	3-44
Figure: 3.14	Burst Read	3-45
Figure: 3.15	Burst Write	3-46
Figure: 3.16	Segment Map & Page Map	3-49
Figure: 3.17	Internal Register Cycle	3-54
Figure: 3.18	Clock Cycle	3-55
Figure: 3.19	SBus or DRAM Access	3-58
Figure: 3.20	Scan EPROM Access	3-59
Figure: 3.21	2nd EPROM Access	3-59
Figure: 3.22	Parity Register Access	3-60
Figure: 3.23	SCC/KB Mouse Access	3-60
Figure: 3.24	Statistics Updates	3-61
Figure: 3.25	Counter	3-63
Figure: 3.26	2nd Counter	3-63
Figure: 3.27	RS-232/423 Jumper Blocks	3-65
Figure: 6.1	Illustration Memory Management	6-12
Figure: 7.1	Sample VME System Diagram	7-28

Section 1

1.0 INTRODUCTION

A complete VMEbus-based SPARCstation™ 2 architecture with Sbus expansion. The SPARC CPU-2CE is the latest innovation resulting from the technology partnership of FORCE COMPUTERS and Sun Microsystems. Combining a true SPARCstation 2 architecture with FORCE COMPUTERS expertise and experience in standard 6U Eurocards has resulted in a faithful SPARCstation 2 implementation in a single VMEbus slot.

The SPARC CPU-2CE™ offers the same I/O interfaces as the SPARCstation 2, including DMA supported SCSI and Ethernet ports along with audio, keyboard/mouse, and two serial channels with full modem support. Two Sbus sockets allow the installation of standard, off-the-shelf Sbus modules such as graphic frame buffers or accelerators or any other of over 300 Sbus cards available from third-party vendors.

Through its binary compatibility with the Sun SPARCstation 2 family, the SPARC CPU-2CE runs current versions of SunOS™/Solaris™ as well as the more than 4,000 shrink-wrapped SPARCware™ applications available today. In addition, a variety of real-time operating systems will be available.

Its IEEE 1014 compatible VMEbus interface enables the SPARC CPU-2CE user to build high-performance embedded UNIX® systems, SPARC®- based real-time systems and hybrid UNIX/real-time systems linked via the VMEbus or local- and/or wide-area networks.

The SPARC CPU-2CE is a compact single board computer that brings the power and functionality of the popular Sun SPARCstation 2 to the industry-standard VMEbus for use in UNIX and real-time applications.

Figure: 1.1 Block Diagram

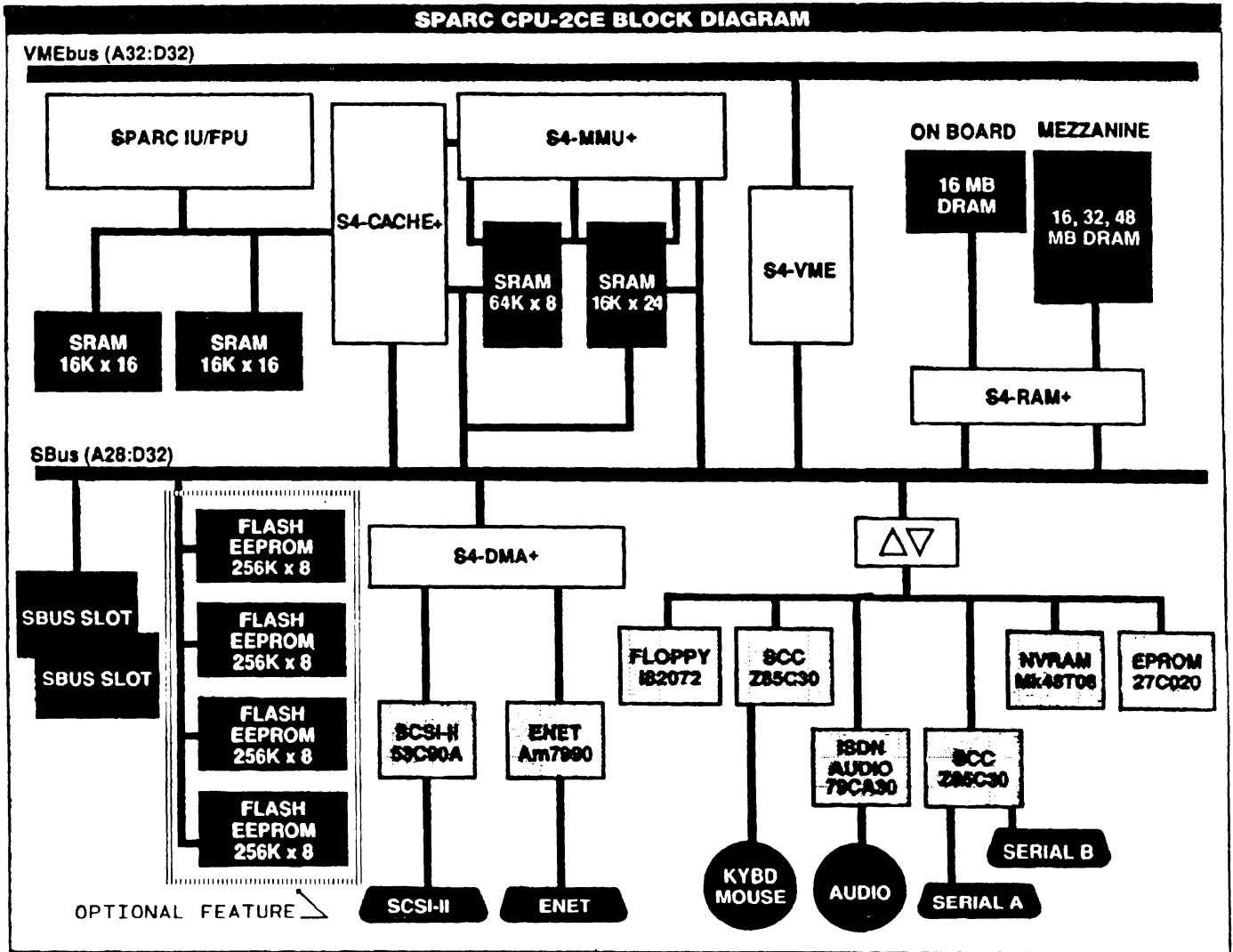
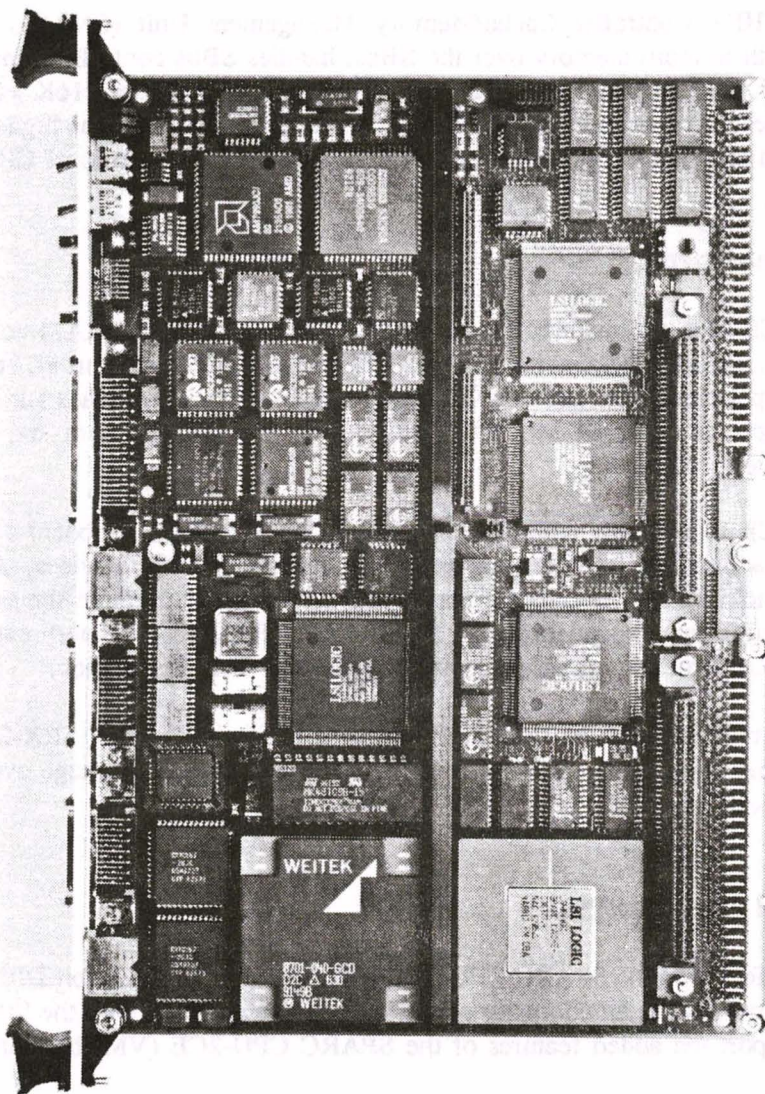


Figure: 1.2 Photo Page



1.1 The Processor

A 40-Mhz SPARC (Scalable Processor ARChitecture) 32-bit RISC processor chip set is at the core of the SPARC CPU-2CE. It is comprised of an integrated Integer Unit/Floating Point Unit (IU/FPU), a Sun standard SRAM-based Memory Management Unit (MMU+), a Cache Controller (Cache+) and two Cache RAM chips. Operating at 40 Mhz, the IU/FPU provides 28.5 MIPS integer performance and 4.2 MFLOPS floating point performance. The Cache Controller provides a 2-Kbyte tag array for a 64-Kbyte, virtual, write-through cache.

The integrated CACHE+ Controller Cache/Memory Management Unit (CMMU) handles the cache memory, provides path to main memory over the SBus, handles SBus controller function, and works at 40 MHz clock speed. The SPARC CPU-2CE has two cache RAMs that are 16K x 16 devices and run at 40 MHz clock speed. The large cache and double word write buffer means higher hit ratios and the ability to sustain high I/O DMA bandwidths with only a small reduction in local CPU performance.

1.2 The Memory Subsystem

The SPARC CPU-2CE chip set includes the RAM+, an Sbus-compatible DRAM controller with single clock burst capability. RAM+ operates the DRAM in fast page mode to support 8-, 16-, 32-, and 64-byte bursts at one transfer per clock after an initial access latency of 4 clocks for reads and 2 clocks for writes. The configuration supports up to 64 Mbytes of 32-bit wide DRAM on board and incorporates parity generation and parity error detection.

The SPARC CPU-2CE is available with 16-, 32-, 48-, or 64-Mbytes of on-board DRAM. 16 Mbytes reside on the base-board and are implemented using high-reliability TSOP devices. Additional capacity, up to 64 Mbytes, is achieved using separate mezzanine modules which retain the single-slot capability without interfering with Sbus expandability on the SPARC CPU-2CE. Memory capacity upgrades are possible since the DRAM mezzanine modules are designed for field installation.

Memory can be expanded beyond the 64-Mbyte on-board limit via an SBus SRX-2 card. This option expands Type0 memory space by up to 64 Mbytes and is cacheable, an advantage over other Sbus-based memory expansion modules.

1.3 System EPROM Open Boot™:

The Open Boot™ EPROM on the SPARC CPU-2CE is located in a single 32 bit DIP socket that enables easy upgrading. It provides the functionality of the boot device supplied with the SPARCstation 2, with enhancements to support the added features of the SPARC CPU-2CE (VMEbus interface and optional Flash EEPROM).

There is an EPROM-based monitor/debugger called Open Boot™. Open Boot is a trademark by Sun.

1.4 FLASH EEPROM (Optional Feature)

In addition to the boot device, the SPARC CPU-2CE employs 4 FLASH EEPROMs in TSOP packages for flexibility and customization. This 1-Mbyte FLASH EEPROM is organized to be 32-bits wide and is mapped in place of the Sun SPARCstation 2 Sbus Slot 3.

These EEPROMs are erasable and writable by the user and allow the system integrator to incorporate FCODE drivers for system specific VME-based products using routines supplied in the boot ROM. With 150 ns EEPROMs, the access time is 4 clocks.

The SPARC CPU-2CE allows integrators to specify their product as the console or boot device without having to modify the on-board firmware. It also allows them to take advantage of SunOS loadable drivers for their VME products. This feature provides a path to firmware-assisted system auto-configuration.

1.5 Local I/O

Ethernet, SCSI interfaces, two serial interface ports, a keyboard specific serial port, a mouse specific serial port, a floppy disk controller, and an audio input / output port are provided.

Serial Interface Ports

The Zilog Z85C30 Serial Communications Controller (SCC) supports two serial interface ports using 26-pin connectors on the front panel for synchronous and asynchronous communications. Data rates up to 64 Kbaud are supported and are user configurable, with a pluggable shunt for either RS-232C or RS-423C, and offer full modem control and interrupt capability. Both ports must be set identical as either RS-232 or RS-423.

Keyboard/Mouse Port

The connector for the keyboard/mouse port uses a SPARCstation Type 2 8-pin Mini-circular DIN connector on the front panel. The Z85C30 SCC for the keyboard/mouse specific port is terminated with TTL buffers and uses all TTL-compatible data signals. The interface is fully compatible with Sun SPARCstation 2 keyboard and mouse product offerings and uses the same cables.

Floppy Disk Controller

An Intel i82072 functions as the floppy disk controller and is fully compatible with the Sun SPARCstation 2. The floppy disk controller signals are available on the User I/O lines of the P2 connector.

Audio Port

The audio interface is implemented with an AMD Am79CA30 ISDN/audio controller device and is fully compatible with the Sun SPARCstation 2. The audio connector on the front panel is an 8-pin Mini-circular DIN connector. As in the SPARCstation 2, the ISDN functionality of the Am79CA30 is not implemented.

1.6 DMA+

The DMA ASIC provides DMA and data assembly-disassembly functions for both the Ethernet and SCSI interfaces. The ASIC contains a 32 byte FIFO for each interface and performs DMA in 16 byte bursts when alignment and transfer length permit.

1.7 Small Computer Systems Interface (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 SPARC CPU-2CE board features a Small Computer System Interface (SCSI) controller device using the NCR 53C90A controller chip augmented by the DMA+(D-channel). This provides SCSI-I functionality and can transfer at up to 5 Mbytes/sec, depending on the speed of the target.

The SCSI controller on the SPARC CPU-2CE acts as a SCSI initiator. Up to seven SCSI target devices can be connected to a SPARC CPU-2CE board. The SCSI bus is properly buffered and terminated on the SPARC CPU-2CE, so that the connection of SCSI devices is simple and straight forward. Just attach a terminated SCSI device with a cable and the CPU-2CE will automatically adjust for proper termination.

The DMA features full SCSI specification, ANSI X3.131/1986, compatibility. It supports both synchronous and asynchronous operation with SCSIbus signals connected to the front panel connector and VMEbus P2 connector. Single ended mode is supported only.

The SCSI bus is routed to both the front panel, via a 50-pin SCSI-II connector, and to the board's P2 User I/O pins.

1.8 Ethernet Interface

The Ethernet interface is comprised of the DMA+ interface chip for DMA, the AMD Am7990 Local Area Network Controller (LANCE) chip and the AMD Am7992B Serial Interface Adapter (SIA). The IEEE 802.3 Ethernet interface is available via a 15-pin Micro-D connector on the front panel. An adapter cable for the Micro-D to the regular DB15 is available. The DMA+ chip contains a 32-byte FIFO and increases performance through its 16-byte DMA burst capability.

The Ethernet interface features compatibility to the IEEE 802.3 Ethernet specification, DMA burst capability, data rate of up to 10 Mbit/sec, interrupt generation (Level 6), and utilization of up to 128 Kbytes of memory.

1.9 Real-Time Clock/NVRAM

The SPARC CPU-2CE board uses the Mostek MK48T08 RTC-NVRAM chip. This device includes an 8-Kbyte non-volatile static RAM, of which 2 Kbyte is user programmable, and a clock/calendar circuit. Both are supported by a 10-year shelf-life lithium battery, in order to retain data when the board is powered down.

1.10 SBus Interface

The SBus is a high performance CMOS bus and the basic interconnection mechanism for the system between the CPU and main memory and I/O devices. Two Sbus connectors are available for I/O expansion into the next VME slot.

The Sbus is a high-performance expansion bus with a bandwidth of over 50 Mbytes/sec. The Sbus forms the local memory and I/O bus for the SPARC CPU-2CE and features 32-bit virtual addressing, 28-bit physical addressing, 32-bit data path, synchronous data transfer and master/slave capability. The SPARC CPU-2CE incorporates two Sbus sockets for I/O expansion. These provide support and compatibility with the hundreds of third-party Sbus expansion products on the market, including products for video and graphics, connectivity and networking, manufacturing and process control, engineering, and scientific instrumentation.

FORCE offers a 6U VMEbus Front Panel with two SBus board cutouts used to mount SBus boards to the CPU-2CE.

1.11 VMEbus Interface

The SPARC CPU-2CE utilizes the Sun VME Chip to provide a complete 32-bit VMEbus interface. Since this is the same device used on the SPARC CPU-1E from FORCE COMPUTERS system performance upgrades with minimal software changes are guaranteed. Supported functions include master and slave data transfer capabilities, and VMEbus interrupt handling and arbitration functions. Additional VMEbus utility functions and a special loop-back cycle for standalone testing of the interface are provided.

The VMEbus interface on the SPARC CPU-2CE is fully supported by a driver for the SunOS/Solaris operating system. The software support also includes an implementation of the ONC/VME protocol which supports standard networking protocols such as TCP/IP, NFS and RPCs across the VMEbus backplane. The VME Driver is needed to implement VME software.

1.11.1 Master Interface

The VMEbus master interface allows 16-, 24-, and 32-bit addressing with 8-, 16-, and 32-bit data transfers. A full 4-GByte address range is available and may be mapped contiguously without the 512-Mbyte limitation typically imposed by the 28-bit Sbus definition. This is done by directing normally unused MMU signals to VMEbus decoding logic, external to the VME chip. A software switch enables the standard VME 512-Mbyte address mode with window mapping registers. This 28-bit mode emulates the SPARC CPU-1E VMEbus addressing scheme, allowing customers a migration path to help preserve a previous investment in custom VMEbus drivers.

Read-modify-write cycles are supported for master accesses, and a 300 μ sec VMEbus timer is included.

NOTE: Unaligned transfers are not supported by the VME chip.

1.11.2 Slave Interface

Access to the on-board DRAM is allowed to a 1-Mbyte page configurable on 1-Mbyte boundaries within the local address space. Addressing is recognized for both 32- and 24-bit standard accesses, with 16-bit accesses reserved for the mail box interrupt. Address modifiers are supported, and any slave access as a Direct Virtual Memory Access (DVMA) device is set to the local supervisor mode in accordance with the Sun-4™ architecture. The 1-Mbyte VME address space selected is always mapped to the highest megabyte in the virtual address space in accordance with the Sun-4 architecture. Unaligned slave accesses are not supported by the VME chip. See the chapter on the VMEbus Interface for more details.

A mail box interrupt function allows other VMEbus devices to interrupt the SPARC CPU-2CE. This mail box detects accesses to the specific A16 address space set in the mail box register. Mail box accesses are acknowledged as standard VMEbus cycles, and trigger an on-board level 13 interrupt.

1.11.3 Interrupts

The on-board interrupt handler selectively supports all seven VMEbus interrupt levels which are routed directly to the MMU interrupt logic. Control of interrupts is handled via software in the interrupt enable register.

1.11.4 System Controller

The SPARC CPU-2CE is capable of providing Slot 1 system controller functions. The VMEbus requestor is a release on request (ROR) requestor. Bus requests are made on BR3.

The VMEbus arbiter function is used when the SPARC CPU-2CE is configured for system controller functions (Slot 1 jumper). Both single-level (SGL) and round robin (RRS) arbitration are provided. Bus timer logic is included to prevent a VMEbus lockup to a non-responding bus requestor or to a non-existent slave device.

Additional system controller functions are automatically activated, when the Slot 1 jumper is enabled. These capabilities include: IACK Daisy Chain Driver, SYSRESET, SYSFAIL and VMEbus system clock.

1.12 Software Description

With full SPARCstation 2 architecture compatibility, the SPARC CPU-2CE runs the Solaris operating system, including enhancements specific to the SPARCstation product line. Solaris provides an advanced development and run-time environment for the SPARC CPU-2CE. As the first "shrink-wrapped" distributed computing solution, Solaris is comprised of SunOS, ONC® networking environments, OpenWindows™, OPEN LOOK® and DeskSet™. SunOS is the highest quality and most widely supported enriched UNIX implementation available today.

By carefully merging the most robust functionality of UNIX System V™, Berkeley BSD™, and Xenix™; SunOS offers the optimum balance of UNIX capability, reliability, and performance. Further functionality includes:

- Standards Conformance (POSIX, X/Open, XPG, SVID and FIPS)
- The OpenWindows "Look and Feel" Graphical User Interface

- Hierarchical and consistent format file system that includes support for MS-DOS®, CD-ROM and RFS, among others
- ONC networking environment for distributed computing across multivendor networks
- Enhanced UNIX kernel with internationalization, system accounting, security and redundancy
- A comprehensive range of third-party software that include databases, design automation and artificial intelligence
- VMEbus driver configured for Sun-4/SPARCserver™ type expansion interface allowing the use of existing Sun-compatible VMEbus device drivers

ONC is the industry standard for heterogeneous networking. OpenWindows is the network-extensible graphical application development platform. OPEN LOOK is an intuitive graphical user interface, and Deskset is a suite of personal and workgroup productivity applications.

1.13 Development

By using the SPARC CPU-2CE as both the development and the target system, equipment costs and time-to-market can be reduced. Solaris on the SPARC CPU-2CE provides all the tools necessary to take a project from conception through delivery. This includes: project planning aids (e.g. mail, diary, print facilities, etc.); design aids (e.g. compilers, assemblers, debuggers, and libraries); test facilities (e.g. compliance checkers, performance monitors, profilers, and system diagnostics); documentation tools (e.g. text formatters, print filters, and PostScript® interfaces); archiving and maintenance (e.g. source-code control systems, archivers, and backup systems); and others.

Optimizing compilers available from Sun include: C, C++, FORTRAN-77, Pascal, Modula-2, and Common Lisp. Independent software developers furnish many other languages, including Ada, APL, BASIC, COBOL, Forth, Mainsail, PL/I, and Prolog. Sun languages have full access to systems, graphics, networking, and user interface packages. Third-party cross compilers allow C, FORTRAN, or Pascal programmers to produce executable binary code for multiple architectures from a single FORCE SPARC CPU-2CE.

1.14 Sun Tools

Standard software provided with Solaris includes a powerful user interface. This allows access to Solaris resource management functions, as well as industry- standard UNIX editors, command shells, and all UNIX commands for file management and manipulation, system administration, I/O system control, and other software development tools.

1.15 Graphical User Interface

OpenWindows is a full-featured windowing environment based on the X-Window Standard and the Network extensible Windowing System (NeWS™). OPEN LOOK is a graphical user interface that uses X-Windows and NeWS to provide an intuitive, "look and feel" desktop. The DeskSet is a group of personal productivity tools and system service applications which includes a File Manager, Calendar, Debugger, Mailer, and others.

1.16 Real-Time

The SPARC CPU-2CE with Solaris, which supports a wide range of VMEbus and SBus-based hardware and software solutions, creates an ideal embedded system. New releases of Solaris will provide the SPARC CPU-2CE with real-time extension for use in "soft" real-time applications. These extensions include: fixed priority processes, priority manipulation, pre-emptive scheduling, process priority inheritance, guaranteed dispatch latency and memory locking.

Several hard real-time operating systems are available for or are being ported to the SPARC CPU-2CE. These typically provide high-speed multi-tasking, pre-emptive scheduling, and fast interrupt response. They include facilities for intertask communications and synchronization, efficient memory management, system clock and timing, optimized floating point support, and high-performance I/O and file systems. Real-time development tools usually include an interactive debug Shell, a linking loader, symbolic debugger, performance monitor, exception and signal handling, libraries, utility routines, extensive system and task information utilities, and source level debuggers.

1.17 Networking

ONC networking environment is a suite of software modules and services that are the de facto standard for distributed computing. FORCE COMPUTERS provides a VMEbus driver utility as a standard item in conjunction with the Solaris operating system. This driver combines the standard Sun-4 SPARCserver treatment of the VMEbus expansion interface with the additional capability to run ONC over the VMEbus backplane. This provides an optimal platform for both tightly coupled networking and distributed processing applications, as well as the ability to utilize existing drivers for add-in VMEbus peripheral controllers.

The Remote Procedure Call (RPC) is the core of ONC and is used to build services such as the Network File System (NFS™), Network Information Service (NIS) and Network System Boot.

Solaris also supports an extensive set of networking tools that allow developers to take advantage of industry standards. Solaris supports internet protocols, including raw (IP), datagram (UDP/IP) and stream (TCP/IP). User-accessible interfaces allow development of custom interfaces and protocols. Sun will also offer a Solaris Serial Line Internet Protocol (SLIP) for serial line_based wide-area networks (WANs).

Optional software provides support for: DECnet; SNA, including IBM's Advanced Program-to-Program Communications (APPC) and Document Interchange Architecture (DIA); OSI support which adheres to MAP 2.1 and TOP 1.0 specifications, Layers 1 through 7 for MAP applications; FTAM (File Transfer Access and Management); and CASE (Common Application Services Elements).

Table: 1.1 Board Function & Specification

Central Processing Unit	SPARC RISC
Clock Frequency	40 MHz
Integer Performance	28.5 MIPS
Floating Point	4.2 MFLOPS @ 40 MHz Double Precision
Block Transfer DMA	50 Mbytes/sec (80 Mbyte/sec peak)
Cache: Data & Instruction	64 Kbytes
Basic DRAM TSOP Capacity	16 Mbytes
Error Detection	Byte Parity
Local DRAM Expansion (Single Slot)	16, 32, 48 via mezzanine
Maximum On-Board DRAM	64 Mbytes
SRAM (Battery Backed Up)	8 Kbytes
MMU	Sun-4 MMU ASIC
SBus Expansion	2 Slots
VMEbus	32-bit IEEE/ANSI-standard
VMEBus Master	A32, A24, A16: D8, D16, D32, RMW
VMEBus Slave	A32, A24: D8, D16, D32, RMW, BLT
VMEBus Interrupt Handler	1 through 7 (Selectable)
VMEBus Arbiter	4 level SGL
Multiprocessor Mailbox	Yes (A16 access)
Ethernet (IEEE 802.3)	7990 Lance, AMD 7992B SIA
Ethernet Transfer Rate	10 Mbits/sec
SCSI Controller	NCR 53C90A and DMA+
SCSI Transfer Rate	ASYC 1.5 Mbyte/sec, SYNC up to 5 Mbyte/sec
Serial Ports	2 Async/Sync; RS-232-C/RS-423
Serial Data Rates	64 Kbaud
NVRAM TOD Clock/Calendar	MK 48T08
EPROM Size	2 Mbyte, 8-bit wide
FLASH EEPROM Size (Optional Feature)	1 Mbyte, 32-bit wide
Reset Switch	Yes
Abort Switch	Yes

Hex Rotary Switch, switch input port	1
LEDs	2 (bi-color Red/Green)
Programmable Timers	2
Watchdog	Yes
Keyboard/Mouse Port	Sun-4 Standard
Floppy Disk Controller	Intel 82072
Audio Controller	AMD 79C30
CD ROM Support	Yes
Front Panel	Yes
Power Requirements + 5V	4 Amps (W/O Optional SBus Cards)
+12 V	0.1 Amp /max .6 Amp
-12 V	0.1 Amp /max 0.2 Amp
2A Minifuse	Schurter P/N 3402.0012.xx
630ma Minifuse	Schurter P/N 3402.0008.xx
Number of VME Slots Used	1
Board	14 Layers
Board Dimensions	6.29" X 9.18" (160 x 233 mm)
SunOS or UNIX™ Operating System	Solaris

See the ordering information table on the next page for adapters available from FORCE. Contact your local sales office for additional options.

Table: 1.2 Compatible Peripherals

Tape Drive	Archive 2150s	1/2" cartridge drive
Tape Drive	Exabyte 8200	8mm cartridge 2.5 GB
Tape Drive	Exabyte 8500	8mm cartridge 5 GB***
Floppy Drive	Sony F17W-FP	3 1/2" 1.44 MB
Hard Drive	Maxtor LXT-213SY	210 MB 3 1/2 inch SUN 207
Hard Drive	Segate ST1480N	424 MB 3 1/2 inch SUN424*
Hard Drive	SEgate ST4766N	669 MB 5 1/4 inch SUN669
Hard Drive	Segate ST41200N	1.GB 5 1/4 inch SUN1.0G**
Hard Drive	Segate ST41600N	1.3 GB SUN1.3G*
CDROM	Sony CDU-8012	

* Not in SUNOS 4.1e

** Not in SUNOS 4.1.1b

*** Only Supported in SUNOS 4.12

1.18 Determining Revision Level

You can determine the revision code and serial number of a SPARC CPU-2CE by inspecting the label attached to the P1 connector of each card.

Section 2**INSTALLATION and DIAGNOSTICS****WARNING**

TO AVOID MALFUNCTIONS AND COMPONENT DAMAGE, PLEASE READ THE COMPLETE INSTALLATION PROCEDURE BEFORE THE BOARD IS INSTALLED IN A SYSTEM ENVIRONMENT.

STATIC KILLS

2.1 Power Up

This chapter will cover the instructions and considerations for powering up to insure proper operation of the CPU-2CE.

The only mechanical jumper blocks for the CPU-2CE are to configure the serial ports and to configure the VME. The serial ports are RS-232-C or RS-423 with the default being RS-232-C. If RS-423 is needed, set the block before power up. Move the jumpers position to set RS-423. Both ports must be set identical as either RS-232 or RS-423. The VME default is VME slot 1 device with the jumper installed in JMP1. Removing Jumper 1 selects the card to be non-slot 1 VME device. VME backplane must be terminated according to spec.

Backplane Slot Configuration Requirements

The SPARC CPU-2CE can be plugged into any VMEbus slot. If a SPARC CPU-2CE is installed in a slot other than 1 with empty slots between it and the slot 1 controller. Each of the empty slots must be configured as described below. See your backplane manual for documentation on the location of these jumpers.

Jumper IACKIN* to IACKOUT* on the backplane:

Pins A21 and A22 on the backplane must be jumpered together.

The signals Bus Grant [0:3] must be jumpered across empty slots, as follows:

BGO: jumper pins 4 and 5 Row B

BG1: jumper pins 6 and 7 Row B

BG2: jumper pins 8 and 9 Row B

BG3: jumpered pins 10 and 11 Row B

The CPU-2CE can power up without a computer terminal and keyboard, but you cannot enter the appropriate commands to test the card without a means to enter commands and view the results. FORCE recommends the initial power up to be performed with a computer terminal connected to serial port A of the CPU-2CE. By using the computer terminal, additional possible error producing factors; a nonfunctioning SBus card, SBus frame buffer ca. connector, or monitor can be detected. Connecting a terminal is simpler than connecting a frame buffer, monitor, and keyboard. The optional serial cable is available by ordering the accessories kit.

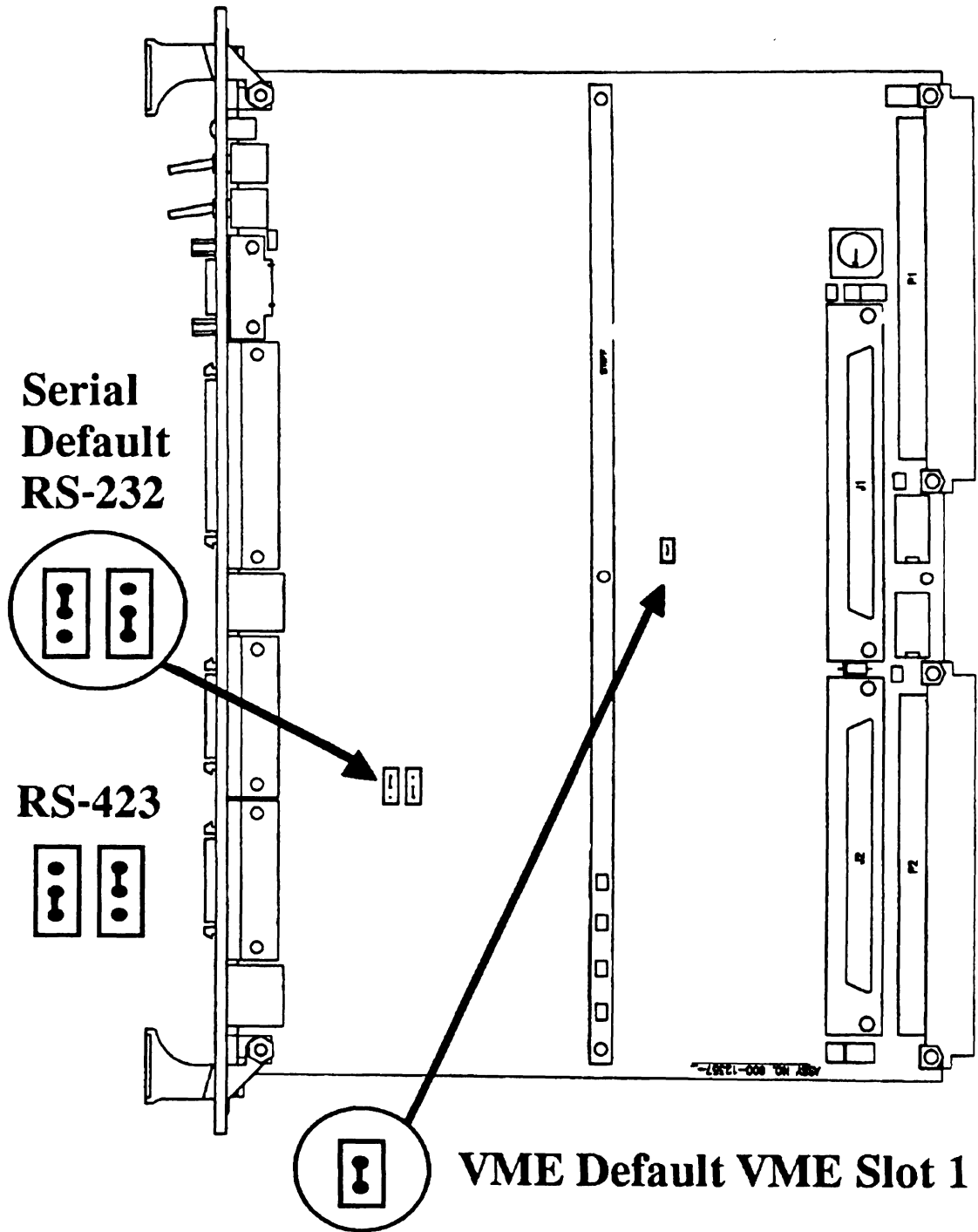
NOTE: Make sure you use a terminal type serial cable. See your terminal manual for ordering or building the proper cable.

The power-up sequence consists of a series of component functional tests and initialization, followed by booting. Turn on the power to the backplane.

In the default autoboot mode, the SPARC CPU-2CE attempts to boot SunOS from an attached SCSI disk. If a disk is not attached, the Open PROM displays an identification banner and enters the command mode of the "PROM Monitor" (a program that monitors the activity of the keyboard). The PROM displays a > prompt.

Figure: 2.1 RS-232/423 Jumper Blocks

BOARD JUMPERS



2.2 Post Power Up Procedures

2.2.1 How to Talk to the SPARC CPU-2CE On-Board Memory

At the > prompt type *n* to enter "New Mode" the FORTH/OpenBoot interpreter.

FORTH displays the *ok* prompt. Basic Assembly PROM commands are available to the SPARC CPU-2CE in this interpreter. Enter the following command at the OK prompt.

0 obmem 0 map-page <cr> to map in the first 4K bytes of main memory.

There is at least 16MB of onboard memory available to the SPARC CPU-2CE card. The first test of the SPARC CPU-2CE PROM validates correct operation of this memory. Use one of the three commands explained below:

2.2.1.1 Memory Test Command One

Perform a LOOP/READ that validates that the on-board memory is functional:

At the *ok* prompt type *100 50 dump <cr>*

RESULTS: If the PROM and the on-board memory are functional, the contents of the 80 bytes (50 hex) of memory starting at address 100 (hex) are displayed. If the PROM or the on-board memory is not working, you receive one of a number of possible error messages indicating a problem with the memory.

2.2.1.2 Memory Test Command Two

Perform a LOOP/WRITE that writes a number pattern to memory, validating memory and memory response:

At the *ok* prompt type *100 50 78 fill <cr>*

RESULTS: If the SPARC CPU-2CE is working correctly, the number pattern "78" is written to 80 bytes of memory starting at address 100, and you will receive the *ok* prompt. The fill command fills byte by byte. If the PROM or the on-board memory is not working correctly, you receive one of a number of possible error messages indicating a problem.

2.2.1.3 Memory Test Command Three

Perform a memory test that exercises the on-board memory. This test does not reside in the PROM, and must be keyed in at the FORTH prompt. (Memory must be mapped according to section 2.2.1)

Key in the following program:

```

: memory-test ( -- )
  150 100 do
    12345678 i 1!
    i 1@ dup
                                \from 0x100 to 0x150
                                \longword write
                                \longword read (result left
                                \on stack)

    12345678 <>
    if ." obs = " .
      ." exp = " 12345678 .
      ." adr = " i . cr
    then
    drop
  4 +loop
;

```

When this code has been correctly entered, you can perform the memory test:

```
memory-test <cr>
```

Also, the following command runs the Open Boot memory test:

```
test /memory or test-memory
```

This test is the same as the Open Boot POST test and will take approximately 4 minutes for 32 megabytes.

RESULTS: If the SPARC CPU-2CE is working correctly, the memory is erased and tested, and you will receive the *ok* prompt. If the PROM or the on-board memory is not working correctly, you receive one of a number of possible error messages indicating a problem.

Connect the cables for the additional devices you wish to test with the SPARC CPU-2CE: SCSI, Keyboard, Ethernet, Serial Ports A and B. (Make sure the power is off before connecting any cables to the CPU-2CE.)

2.2.2 Test the Ethernet Port

Test the Ethernet port by connecting a MAU (Medium Access Unit) to the Ethernet port and while in Forth/OpenBoot and typing:

```
test net <net>
```

2.2.3 How to Talk to the SPARC CPU-2CE Buses

Access to the devices available on the SPARC buses in general requires mapping, reading, and writing the device. This test is performed only when testing an SBus memory card. The procedure below can be used to map in memory. Devices are mapped in two stages:

2.2.3.1 Mapping Memory

Select unused segments, virtual address range and size. Map in the segments, e.g., `seg# = 0x80`, `va = 0x1000000`, `size = 0x4000`.

```
80 1000000 0x4000 map-segments <cr>
```

Select a physical address range and space (SBus Slot2). Map in the page e.g., `pa = 0xFC000000`, `space = SBus`.

```
FC000000 sbus 1000000 4000 map-pages <cr>
```

2.2.3.2 Accessing Memory

To dump out memory at the ok prompt type `1000 000 50 Dump <cr>`

2.2.4 How to Talk to the SBus

SBus devices are handled using the IDPROM onboard the SBus card. The IDPROM contains a driver for the SBus card which is read at boot time and interpreted by the Open PROM. During debug of an SBus device or it's driver a device in an SBus slot may be mapped in using `map-sbus`. (See the *Open PROM Toolkit User's Manual* for a description of the `map-sbus`).

2.3 Running PROM Diagnostics

Setting the diagnostic switch and resetting the card will cause the card to come-up into the extended selftests.

CAUTION: This command disables the monitor and keyboard and enables Serial Interface A as the I/O device.

```
type: setenv diag-switch? true <cr>
```

reset type: `reset` or `lift the reset switch` on the front panel. The board will exit automatically.

NOTE: You must have a tty device attached to Serial Port A. Setting the `diag-switch` to true puts messages to the serial port A console.

2.4 Running Functional Tests

The following tests are available for testing functional units on the CPU-2CE. These tests are automatically run at a reset or a power-up.

The functional test can be run individually by leaving the PROM monitor and entering the FORTH/OpenBoot interpreter.

At the > prompt, enter *n*.

At the ok prompt, enter one of the following commands.

```
test /memory or test-memory
test /sbus/le
test net
test floppy (requires floppy drive and Sun formatted FD formatted disk)
watch-clock
probe-scsi
```

NOTE: For a complete listing of diagnostics available from the PROM, type *help diag* or *help test*

2.5 Returning To Monitor Mode

To return to the > prompt from the OK prompt, enter the following:

```
ok old-mode <CR>
```

2.6 Diagnostics

This chapter describes the different types of diagnostic firmware and software tools available and how they are related.

2.6.1 Main Categories of Diagnostics

- Boot PROM diagnostics
 - Power-On Self-Test
 - On-Board Diagnostics
- Sundiag System Exerciser (SunOS)

This chapter will also briefly cover the Forth/OpenBoot Toolkit, which is an interactive command interpreter based on the Forth programming language. Additional information on the Sun Forth Toolkit can be found in the Open Boot PROM 2.0 Toolkit User's Guide in the SBus Developers Kit.

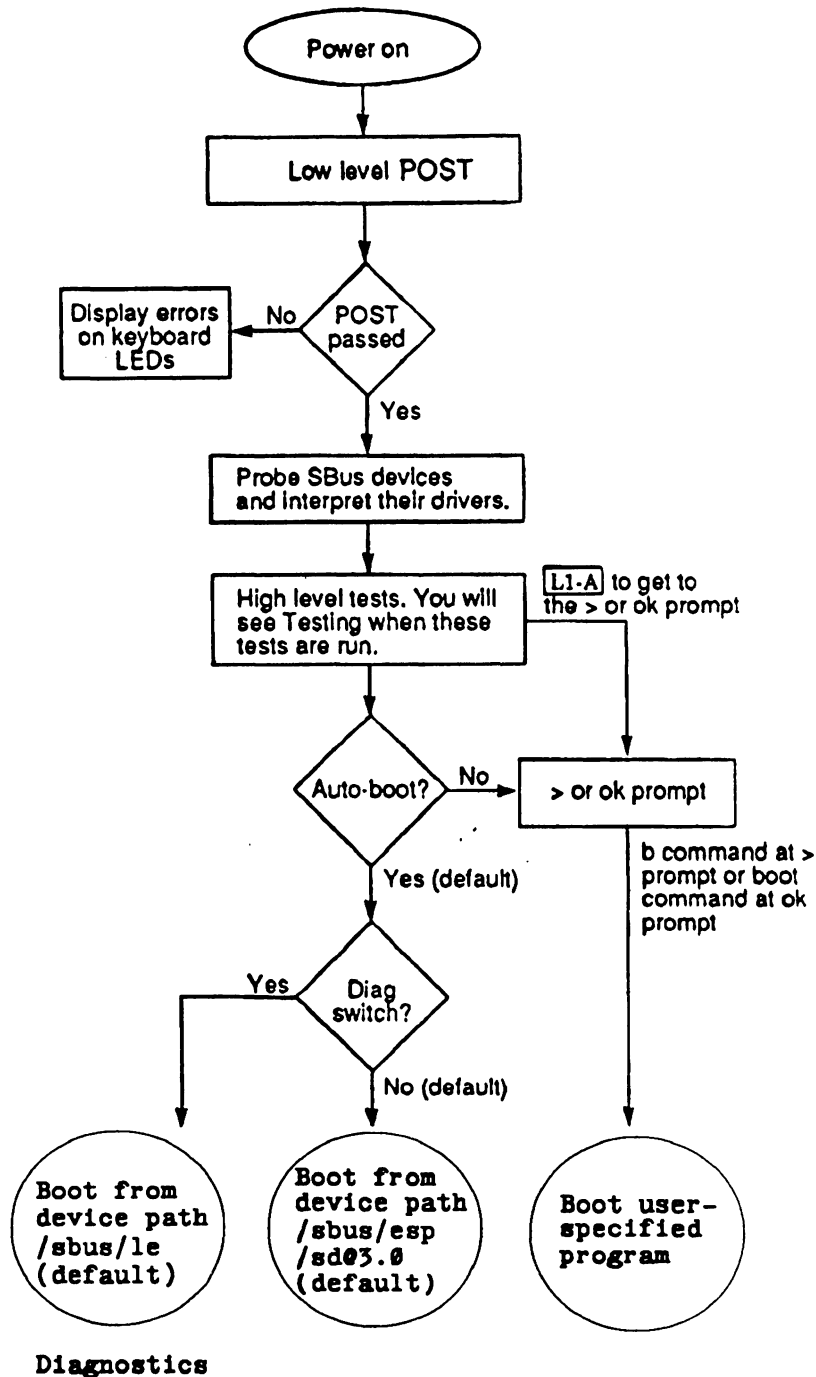
2.6.2 Diagnostic Selections

Table: 2.1 Diagnostic Tools

Diagnostic Tool	When to use
Power On Self Test (POST)	The POST code resides in the boot PROM and is driven automatically by a signal from the power supply when first powered on. The CPU performs the self-test. POST shows if there is a failure of the main components on the main logic board.
On Board Diagnostics	Individual test, ie memory, Ethernet, etc., available when in Forth Toolkit. Enter <i>n</i> from the <i>></i> prompt to enter the Forth Toolkit. The on board diagnostics reside in the boot PROM.
Sundiag System Exerciser	Using SunOS, it displays realtime use of system resources and peripherals. Sundiag will show if your system is functioning properly. If Sundiag fails, run the Power On Self Test.
Monitor	Monitor will be activated when the system crashes. Monitor is the <i>></i> prompt. Typing <i>b</i> boots, <i>c</i> resumes or continues program halted, or <i>n</i> to enter Forth Toolkit.
Forth Toolkit	Performs all functions available through the Monitor, except entering the Forth Toolkit i.e. changing the NVRAM parameters, resetting the system, running diagnostics, displaying system information, redirecting input and output, etc. There is more in this chapter and the Open Boot PROM 2.0 Toolkit User's Guide.

Figure: 2.2 POST

FIGURE 2.3 Power On Self Test Flow Chart



The Forth Toolkit offers an extensive set of functions for performing the following:

- Hardware Development
- Problem Determination (fault isolation)
- Software Development
- Debugging

This section will detail how various diagnostic tools work together in different power on modes.

NOTE: A terminal is needed to view these test results.

CAUTION- To run the On-Board Diagnostics the system must be halted. When the operating system or application has been booted, do not use the L1-A keys to halt the system. L1-A abruptly halts the system and may damage data files. As root halt the system by entering `/usr/etc/shutdown` or `/etc/fasthalt`.

- When the power is turned on, the lower level Power On Self Test (POST) code is executed from the Boot PROM.
- If failure occurs in POST, the type 4 keyboard LEDs will display the failure code. See The Power On Self Test in this section for more detailed information.
- If the POST passes, the system probes for SBus devices and interprets their drivers.
- High level test are performed next.
- Specialized diagnostic test can be performed after the high level test by activating the Monitor, indicated by the `>` prompt. The monitor is activated by pressing the *L1* and *A* keys simultaneously and waiting for the `>` prompt.
- If the autoboot switch parameter is set to false (not the default) the `>` or `ok` prompt will show.

The monitor with its prompt `>` is the default. It is possible to change the default to the Forth Toolkit with its prompt `ok` as an option, see the Open Boot PROM 2.0 Toolkit User's Guide.

- If the autoboot switch parameter is set to true (default), and the diagnostic switch parameter is set to false (default), SunOS boots from the default SCSI target 3 boot path:

```
/sbus/esp/sd@3,0
```

If the hard disk address is set to 0, then the boot parameter should be set to `/sbus/esp/sd@0,0`.

- For normal booting `boot /sbus/esp` will automatically find and boot the disk.

- If the autoboot switch parameter is set to true (default), and the diagnostics switch parameter is set to true (not the default), SunOS boots from the bootpath:

`/sbus/le`

- To boot user-specified programs, such as the SunDiagnostic Executive, you must be at the `>` or `ok` prompt. On Board Diagnostics section later in this chapter will detail how to obtain the `>` and `ok` prompts with `fasthalt`.

Table: 2.2 Diagnostic Switches

Autoboot Switch Parameter	Diagnostic Switch Parameter	Result
False	(Don't care)	<code>></code> or <code>ok</code> prompt
True	False	boot SunOS (vmunix) from SCSI ID 3 (<code>/sbus/esp/sd@3,0</code>)
True	True	boot SunOs (vmunix) from network*(<code>/sbus/le</code>)

* The boot parameters represented here are default settings. The defaults may be changed by following the procedures listed in the Open Boot PROM 2.0 Toolkit User's Guide.

2.6.3 Boot PROM Diagnostics

The diagnostics contained in the boot PROM include the following:

Power-On Self-Test

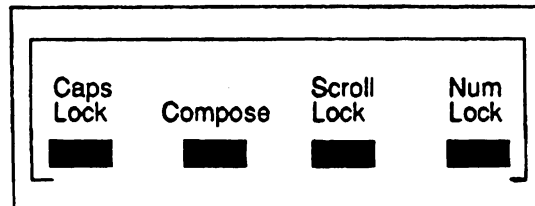
On-Board Diagnostics

2.6.3.1 POST

The Power On Self Test (POST) is the default mode. The POST consist of a sequence of test designed to test the major hardware components of the main logic board before SunOS is booted. Only major failures can be detected by POST. For additional more thorough diagnostics run the extended on-board diagnostics for items including Memory, Ethernet, diskette drives, etc.

Figure: 2.3 Keyboard LED Diagnostic Codes

Arrangement of Sun4 Keyboard LEDs



Sun4 Keyboard LED Diagnostic Code

LED Display Pattern	Unit	Meaning of Pattern
		Unassigned. Testing completed, SunOS is booted.
	Boot PROM	Bad checksum in boot PROM.
	NVRAM/TOD	NVRAM/Time-of-Day Clock failed.
	SPARC 2	SPARC 2 component failed.

2.6.3.2 On Board Diagnostics

1. Save all your work and quite all applications
2. As root, halt the system by entering

```
/usr/etc/fasthalt
```

The returning prompt will either be a default prompt `>` or the prompt `ok`. To change the prompt see the Open Boot PROM 2.0 Toolkit User's Guide.

If you see the `>` prompt, go to the next step. If you see the `ok` prompt go to step 4.

3. Enter `n` to enter the Forth/OpenBoot Toolkit.

The `ok` prompt signifies the Forth Toolkit mode.

4. Enter `help diag` to get a listing of on-board diagnostic test.
5. To return to the monitor `>` prompt, type the following command

```
old-mode
```

2.6.4 Sundiag System Exerciser

The Sundiag System exerciser verifies that the system is functioning properly. Sundiag runs under SunOs and displays real-time use of system resources and peripheral equipment such as Desktop Storage Packs and External Storage Modules.

Exerciser is shipped with SunOS. If it was selected during SunInstall (operating system loading) procedure, it can be run at any time. The file is found in the directory `/usr/diag/sundiag`. The file can also be loaded from tape or CD. You must also have Sunview and the file `userdiag` loaded on your disk. Become root on your system then type `sundiag`. See the Sundiag User's Guide for further information.

2.6.5 Monitor and Forth Toolkit

The Monitor is a basic diagnostic utility. If there is a problem with the operating system the Monitor will automatically start, indicated by the `>` prompt.

Table 2.3 Non-Volatile System Configuration Parameter Defaults

Parameter Name	Default Value
selfest-#megs	1
oem-logo	
oem-logo?	false
oem-banner	
oem-banner?	false
output-device	screen
input-device	keyboard
sbus-probe-list	0123
keyboard-click?	false
keymap	
ttyb-rts-dtr-off	false
ttyb-ignore-cd	true
ttya-rts-dtr-off	false
ttya-ignore-cd	true
ttyb-mode	9600,8,n,1,-
ttya-mode	9600,8,n,1,-
diag-file	
diag-device	net
boot-file	
boot-device	disk
auto-boot?	true
watchdog-reboot	false
fcode-debug?	false
local-mac-address?	false
use-nvramrc?	false
nvramrc	
screen-#columns	80
screen-#rows	34

sunmon-compat?	true
security-mode	none
security-password	
security-initiator-id	7
hardware-revision	
last-hardware-update	
testarea	0
mfg-switch?	false
diag-switch?	false

Table: 2.4 NVRAM VME Configuration Parameters

Parameter	Description	Def
vme-slavemap	Address space for system DVMA access	0
vme-a32map	Re-map VME address bits A[31:29]	0
vme-intena	Interrupt enable register	254
vme-mailbox	Select VME address for mailbox interrupt	0
vme-buslock	Enables atomic RMW	0
vm-server-slavemap	Client's server slavemap address	0
vm-server-addr	Client's server internet address	0
vm-ip-addr	Client's internet address	0
vme-rerun	rerun mailbox level register	0
vme-busloc	Bus locker register	0
next-prom	If true, jump to second PROM on exit from first	false

2.7 SPARC CPU-2CE NVRAM Parameters

This section lists and defines the NVRAM parameters unique to SPARC CPU-2CE architecture. Each parameter will be presented in the following format:

parameter [options][default]

parameter and a description of the parameter's function.

vme-slavemap [range:0-15][0]

vme-slavemap selects a one megabyte space for system DVMA access. vme-slavemap will select one of the first 16 MB of VME space.

vme-a32map [0x0,0x20,0x40,0x60,0x80,0xa0,0xc0,0xe0][0]

vme-a32map remaps VME address bits A[31:29] enabling full 4GB mapping.

vme-intena [0-255][254]

vme-intena selectively enables VME interrupt levels by setting a bit mask that corresponds to the Interrupt Enable Register. The default value of 0xfe enables all interrupts and disables Round-Robin Arbitration.

vme-mailbox [0-255][0]

vme-mailbox selects a VME address to be monitored. If enabled, generates a mailbox interrupt to the IU by way of an on-card interrupt. This mailbox detects accesses to A16 address space at a location programmed in the Mail Box register. No real memory is provided at this location, but the mailbox responds with a VME DTACK, acknowledging the access and generating a level 13 interrupt if the enable bit is set.

vme-buslock [0,1][0]

vme-buslock is a bus locker function that enables the CPU to do an Atomic Read-Modify-Write (RMW) to its on-card memory without being interrupted by an incoming RMW from another VMEbus master. This parameter should only be used in a multiprocessing environment.

vm-server-slavemap [0-15][0]

vm-server-slavemap sets the slavemap location of the client's server. See the **SunOS 4.1e Release Manual**, **ONC/VME** and the description of vme-slavemap above.

vm-server-address [internet address][0]

vm-server-address is the internet address of the client's server. This address is made up of four hexadecimal numbers expressed in decimal form. For example, the number 199.9.9.1 (all decimal numbers), converts to the hexadecimal value 0xc7090901. See the **SunOS 4.1e Release Manual**, **ONC/VME**.

vm-ip-addr [internet address][0]

vm-ip-addr is the internet address of the client. This address is made up of four hexadecimal numbers expressed in decimal form. For example, the number 199.9.9.1 (all decimal numbers), converts to the hexadecimal value 0xc7090901.

Table: 2.5 Front Panel

LEGEND	DESCRIPTION
SWLED (A,B)	2 software controlled LEDs
Reset	Resets the Board
Abort	Aborts Process
ENET	Ethernet 15 pin Micro-D connector
KBD	Keyboard, mouse circular DIN connector
Serial A	26 pin connector for serial ports
Serial B	26 pin connector for serial ports
AUDIO	Audio circular DIN connector
SCSI	SCSI-II 50 pin Micro-D connector

Table: 2.6 CPU-2CE Connectors

FUNCTION	Board Manufactures PN Front Panel	Cable Mate Mfg Part #
SCSI-II	AMP 749831-5	AMP 749621-5
Ethernet	ITT CANNON MDSM- 15PE-Z10	ITT CANNON MDSM- 15SC_Z11
Serial I/O	AMP 749830-2	AMP 749621-2
Audio	AMP 749232-1	AMP 750208-2
Keybd, Mouse	AMP 749232-1	AMP7502 08-2

Figure: 2.4 Font Panel

CPU-2CE FRONT PANEL

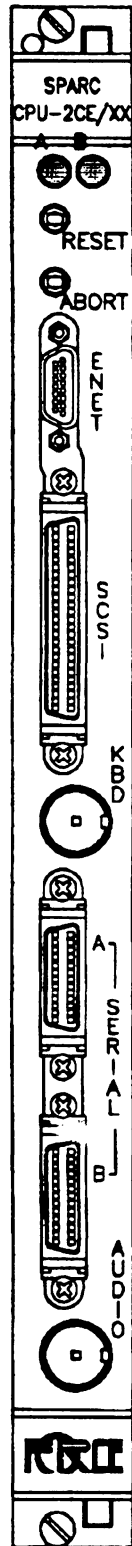


Figure: 2.5 Component Side

CPU-2CE Front Side

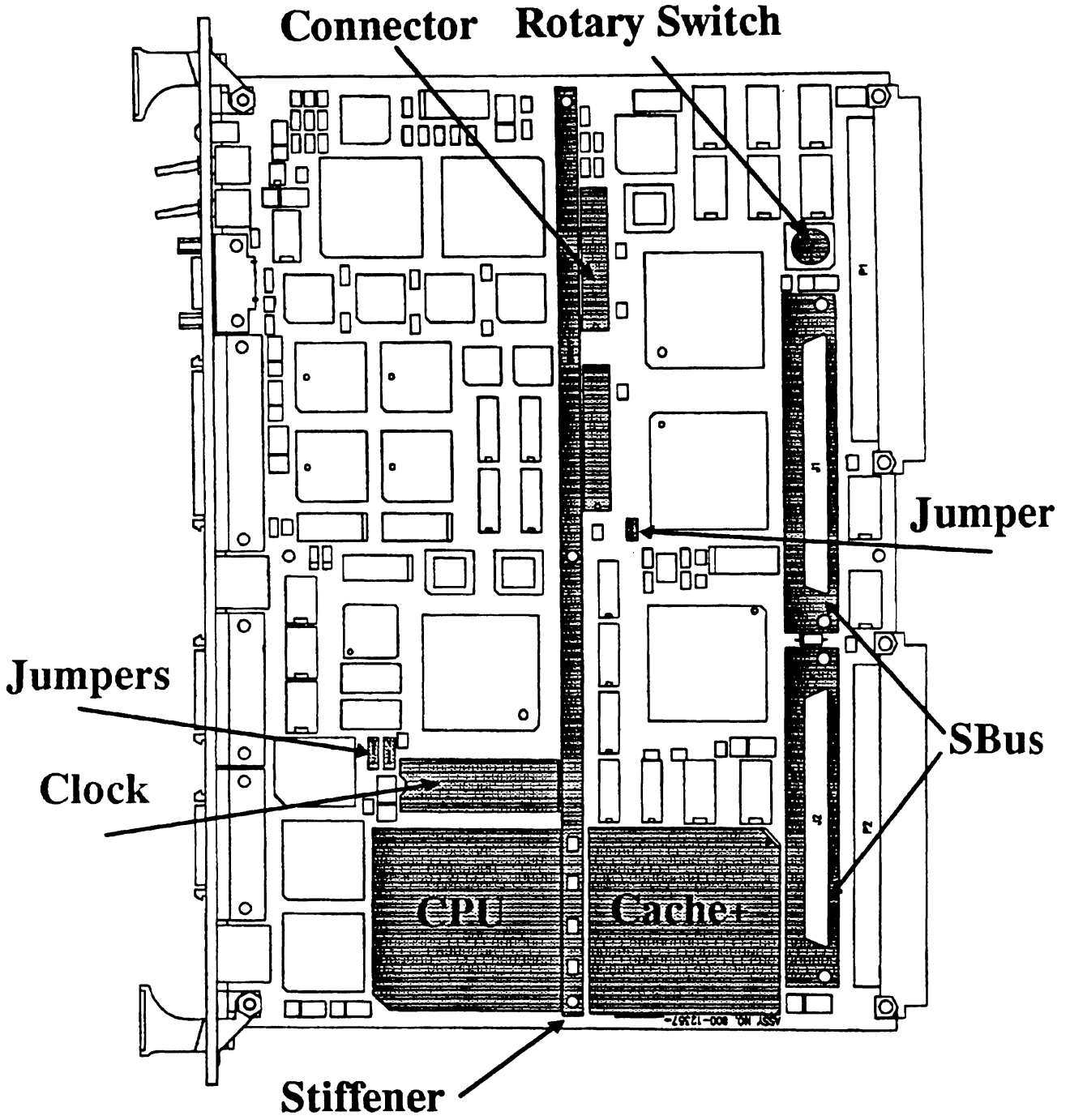


Figure: 2.6 Solder Side

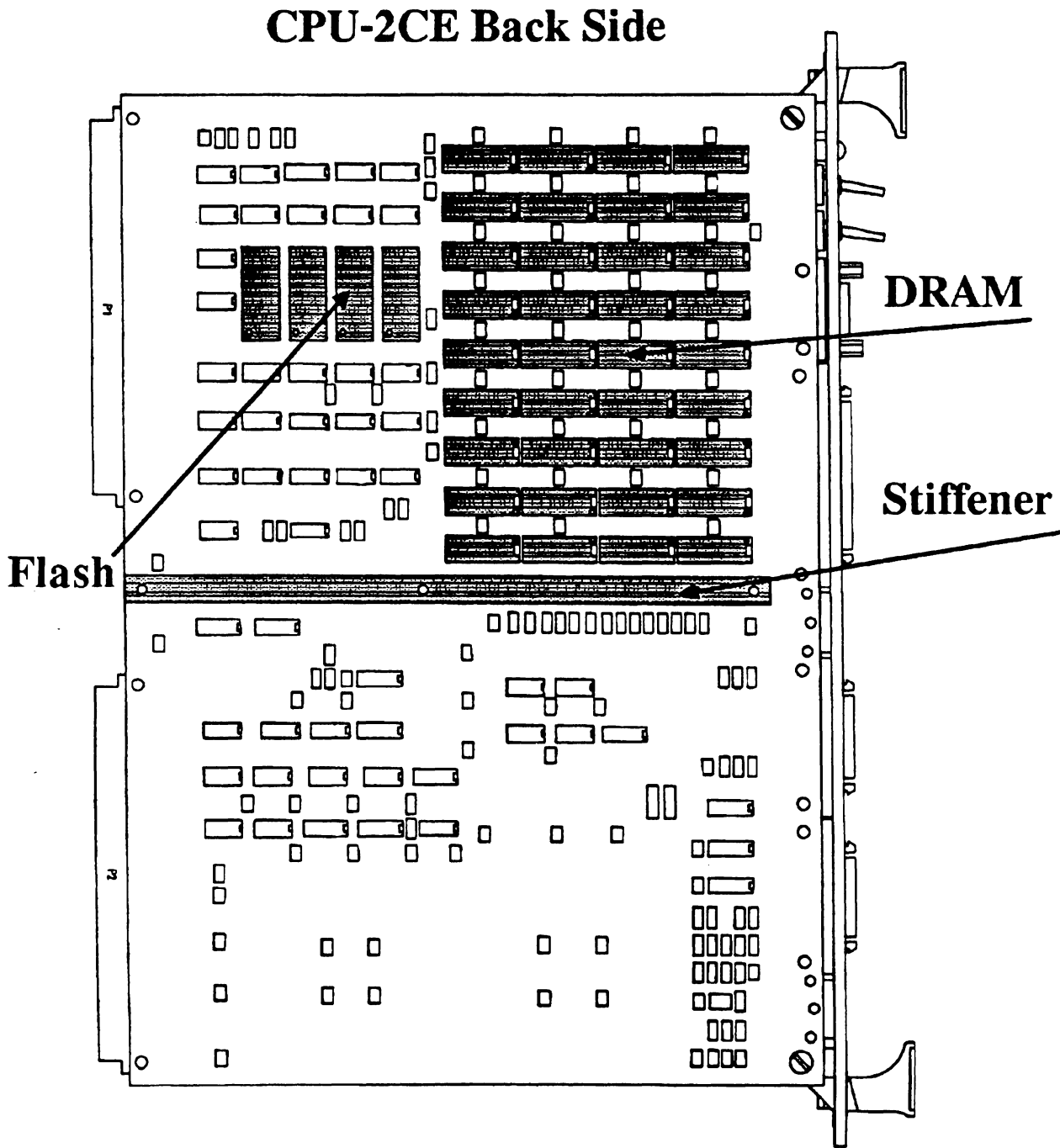
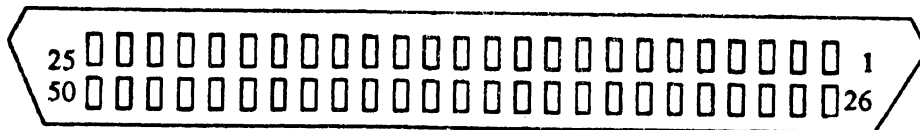


Table: 2.7 SCSI Pinout List

Type	Pin #	Comment	Type	Pin #	Comment
GND	1	Ground	SD0-	26	SCSI Data0-
GND	2	Ground	SD1-	27	SCSI Data1-
GND	3	Ground	SD2-	28	SCSI Data2-
GND	4	Ground	SD3-	29	SCSI Data3-
GND	5	Ground	SD4-	30	SCSI Data4-
GND	6	Ground	SD5-	31	SCSI Data5-
GND	7	Ground	SD6-	32	SCSI Data6-
GND	8	Ground	SD7-	33	SCSI Data7-
GND	9	Ground	SDP-	34	SCSI Parity-
GND	10	Ground	GND	35	Ground
GND	11	Ground	GND	36	Ground
NC	12	No Connect	NC	37	No Connect
NC	13	No Connect	TRMPWR	38	Term. Power (+5V DC, Fused, 3 Amps)
NC	14	No Connect	NC	39	No Connect
GND	15	Ground	GND	40	Ground
GND	16	Ground	ATN-	41	Attention-
GND	17	Ground	NC	42	No Connect
GND	18	Ground	BSY-	43	Busy-
GND	19	Ground	ACK-	44	Acknowledge-
GND	20	Ground	RST-	45	Reset-
GND	21	Ground	MSG-	46	Message-
GND	22	Ground	SEL-	47	Select-
GND	23	Ground	CD-	48	Command/Data-
GND	24	Ground	REQ-	49	Request-
GND	25	Ground	IO-	50	Input/Output-

Figure: 2.7 SCSI Connector and Pins (Front View)



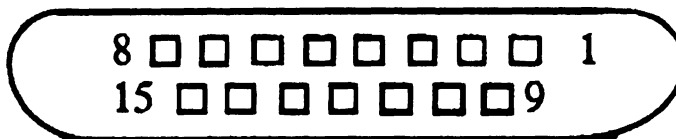
Ethernet Connector Pinout List

The following table is a pinout of the Ethernet connector. Figure 2.8 Shows the Ethernet connector and pin numbers.

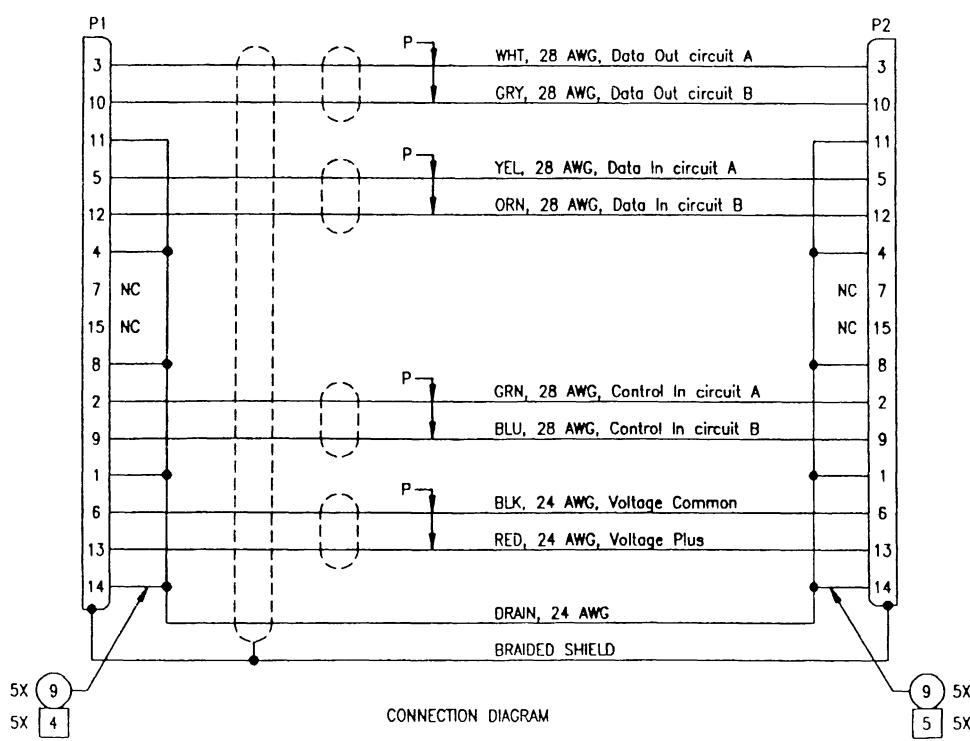
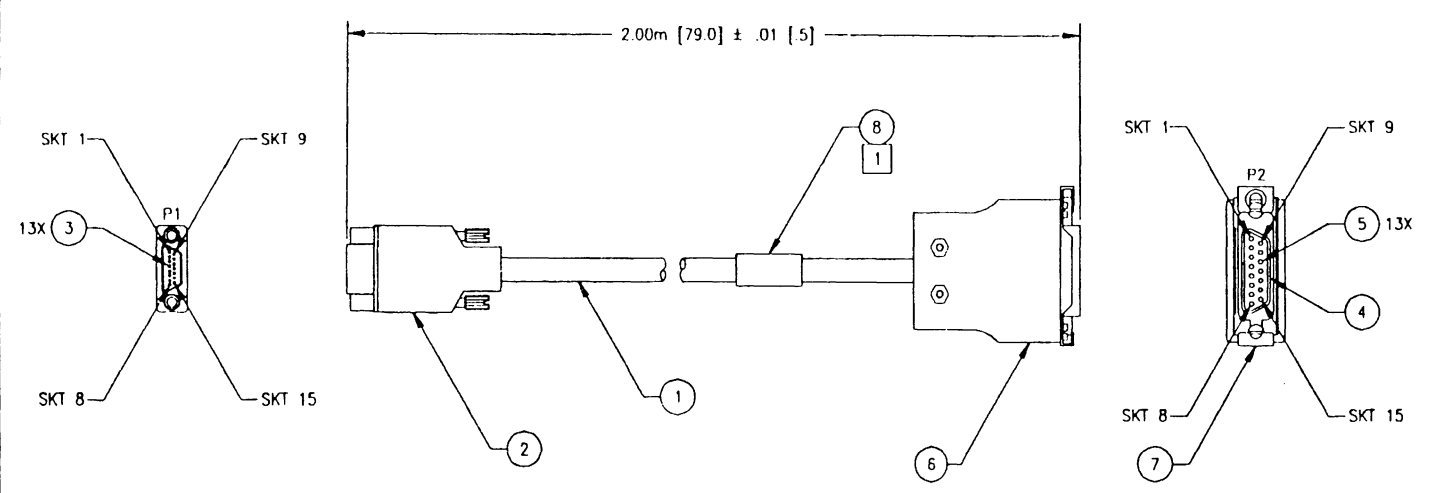
Table: 2.8 Ethernet Pinout List

Pin	Function
1	N.C.
2	Collision+
3	Transmit Data+
4	N.C.
5	Receive Data+
6	Ground
7	N.C.
8	N.C.
9	Collision-
10	Transmit Data-
11	N.C.
12	Receive Data-
13	+12VDC
14	N.C.
15	N.C.

Figure: 2.8 Ethernet Cable Connector and Pins (Front View)



REVISIONS					
DASH	REV	DESCRIPTION	DATE	PROVED	
101	A	PRE-PROD RLSE PER ECO 1375	JS	6/92	
101	A1	PROD RLSE PER ECO 1418	JS	8/92	<i>[Signature]</i>



- NOTES: UNLESS OTHERWISE SPECIFIED
- MARK ASSEMBLY PART NUMBER AND REV LEVEL WITH CONTRASTING PERMANENT COLOR ON ITEM 8 (LABEL) AND LOCATE NEAR CENTER OF CABLE.
 - DIMENSIONS ARE IN METERS, INCHES ARE IN [].
 - CABLE WIRING PAIR FOR "Control Out" SIGNALS NOT INCLUDED.
 - SOLDER ONE END OF ITEM 9 (WIRE) TO DRAIN WIRE OF ITEM 1 (CABLE), THEN ATTACH ITEM 3 (CONTACT) TO OTHER END OF ITEM 9.
 - SOLDER ONE END OF ITEM 9 (WIRE) TO DRAIN WIRE OF ITEM 1 (CABLE), THEN ATTACH ITEM 5 (CONTACT) TO OTHER END OF ITEM 9.

ITEM NO.	QTY	DESCR	FSCM NO.	PART OR IDENTIFYING NO.	DESCRIPTION	MATERIAL/SPECIFICATION
9	AR		9928		WIRE, BLACK, 28 AWG	BELDEN
8	1		TAG-22-100		LABEL, ADHESIVE	TYTON
7	1		DA51220-1		SLIDE LATCH KIT	ITTCANNON
6	1		DA121073-150		BACKSHELL, SHIELDED	ITTCANNON
5	13		D110238-478		CONN, SKT, CRIMP, 24-28 AWG	ITTCANNON
4	1		DMA155-A197-F0		CONN, DSUB, FEM, 15 PIN	ITTCANNON
3	13		MDS-S-T5		CONN, SKT, CRIMP, 26-28 AWG	ITTCANNON
2	1		MDSM-155C-211		CONN, MICRO-D, FEM, 15 PIN	ITTCANNON
1	2m		9903		CABLE, ROUND, B/C, 4 PR, 28(24) AWG, SHLD	BELDEN

PARTS LIST

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: DECIMALS .005 ANCHES ± 0.02 ± 0.30 ± 0.003

THIS DOCUMENT IS THE PROPERTY OF FORCE COMPUTERS, INC. AND CONTAINS INFORMATION WHICH IS CONFIDENTIAL AND PROPRIETARY TO FORCE COMPUTERS, INC. NO PART OF THIS DOCUMENT MAY BE COPIED, REPRODUCED OR DISCLOSED TO THIRD PARTIES WITHOUT THE WRITTEN CONSENT OF FORCE COMPUTERS, INC.

FORCE FORCE COMPUTERS INC. CAMPBELL, CALIFORNIA

CABLE ASSEMBLY, CPU2 ETHERNET

APPROVALS: DRAWN J. STODDEN DATE 6/92

FINISH: CHECKED: ISSUED:

SIZE C FSCM NO. 720-12257-101 DWG. NO. 720-12257-101 REV. A1

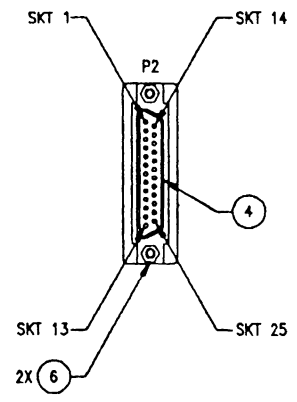
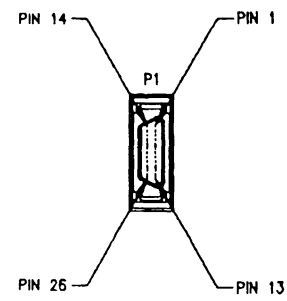
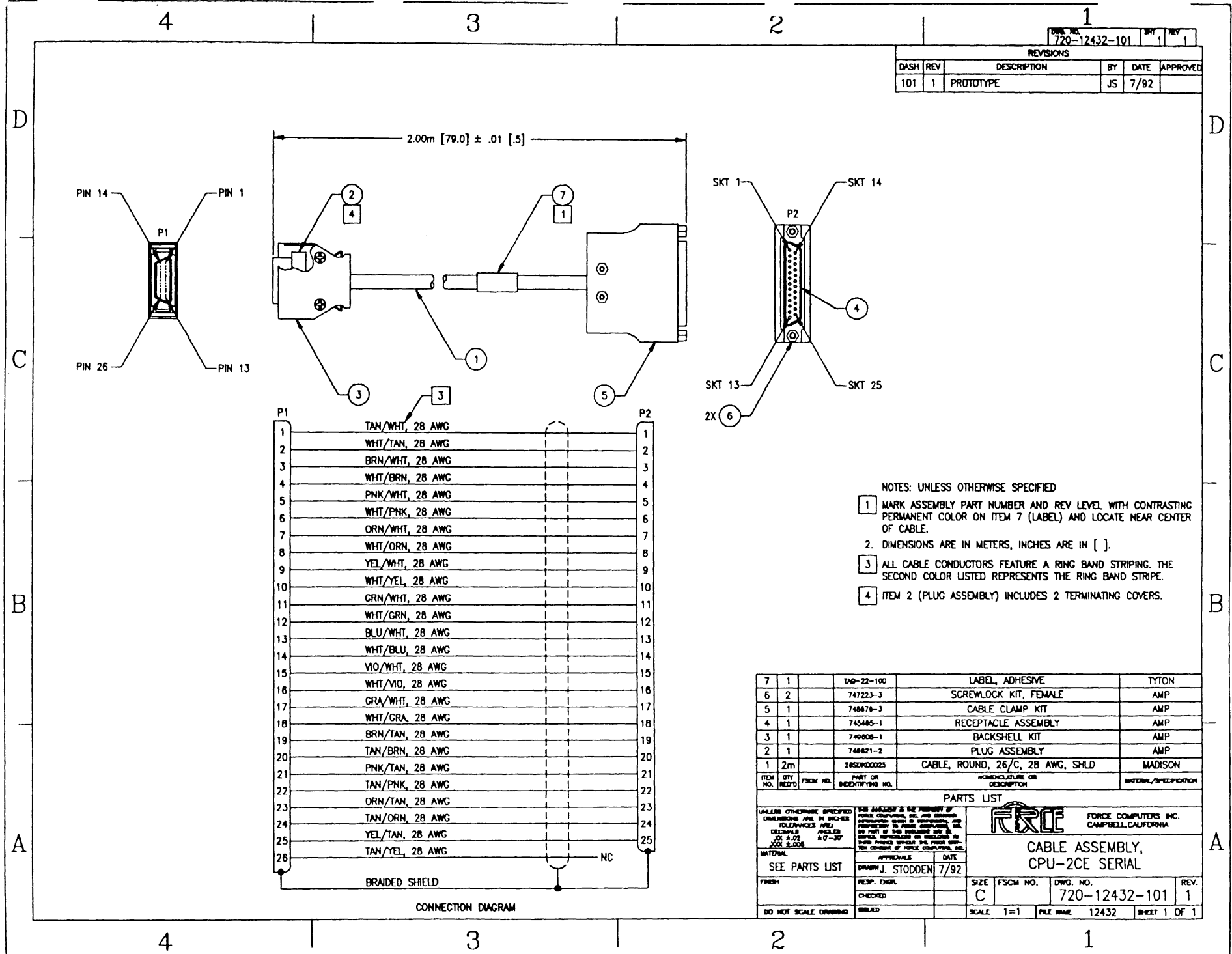
DO NOT SCALE DRAWING SCALE 1=1 FILE NAME 12257 SHEET 1 OF 1

Table: 2.9 Serial Pinout List

Pin	Transmitted Signals	Pin	Received Signals
2	TxD-Transmit Data	3	RxD- Receive Data
4	RTS-Request To Send	5	CTS-Clear to Send
7	Ground	6	SYNC*
20	DTR-Data Terminal Ready	8	DCD-Data Carrier Detect
24	TRXC-DTE Transmit Clock	15	TRXC-DCE Transmit Clock
		17	RTXC-DCE Receive Clock

* Connector casing is also grounded

REVISIONS					
DASH	REV	DESCRIPTION	BY	DATE	APPROVED
101	1	PROTOTYPE	JS	7/92	



P1		P2
1	TAN/WHT, 28 AWG	1
2	WHT/TAN, 28 AWG	2
3	BRN/WHT, 28 AWG	3
4	WHT/BRN, 28 AWG	4
5	PNK/WHT, 28 AWG	5
6	WHT/PNK, 28 AWG	6
7	ORN/WHT, 28 AWG	7
8	WHT/ORN, 28 AWG	8
9	YEL/WHT, 28 AWG	9
10	WHT/YEL, 28 AWG	10
11	CRN/WHT, 28 AWG	11
12	WHT/CRN, 28 AWG	12
13	BLU/WHT, 28 AWG	13
14	WHT/BLU, 28 AWG	14
15	VIO/WHT, 28 AWG	15
16	WHT/VIO, 28 AWG	16
17	GRA/WHT, 28 AWG	17
18	WHT/GRA, 28 AWG	18
19	BRN/TAN, 28 AWG	19
20	TAN/BRN, 28 AWG	20
21	PNK/TAN, 28 AWG	21
22	TAN/PNK, 28 AWG	22
23	ORN/TAN, 28 AWG	23
24	TAN/ORN, 28 AWG	24
25	YEL/TAN, 28 AWG	25
26	TAN/YEL, 28 AWG	25

NC

CONNECTION DIAGRAM

- NOTES: UNLESS OTHERWISE SPECIFIED
- 1 MARK ASSEMBLY PART NUMBER AND REV LEVEL WITH CONTRASTING PERMANENT COLOR ON ITEM 7 (LABEL) AND LOCATE NEAR CENTER OF CABLE.
 2. DIMENSIONS ARE IN METERS, INCHES ARE IN [].
 - 3 ALL CABLE CONDUCTORS FEATURE A RING BAND STRIPING. THE SECOND COLOR LISTED REPRESENTS THE RING BAND STRIPE.
 - 4 ITEM 2 (PLUG ASSEMBLY) INCLUDES 2 TERMINATING COVERS.

ITEM NO.	QTY REQ'D	FSCM NO.	PART OR IDENTIFYING NO.	HOMECULTURE OR DESCRIPTION	MATERIAL/SPECIFICATION
7	1		TAG-22-100	LABEL, ADHESIVE	TYTON
6	2		747223-3	SCREWLOCK KIT, FEMALE	AMP
5	1		748676-3	CABLE CLAMP KIT	AMP
4	1		745485-1	RECEPTACLE ASSEMBLY	AMP
3	1		749008-1	BACKSHELL KIT	AMP
2	1		748621-2	PLUG ASSEMBLY	AMP
1	2m		28SDK00025	CABLE, ROUND, 26/C, 28 AWG, SHLD	MADISON

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: DIMENSIONAL ANGLES JOE A.02 JOE S.005

SEE PARTS LIST

APPROVALS: DRAMIR J. STODDEN DATE 7/92

FINISH: CHECKED

DO NOT SCALE DRAWING

SIZE: C

SCALE: 1=1

FILE NAME: 12432

DATE: 720-12432-101

SHEET: 1 OF 1

FORCE COMPUTERS INC. CAMPBELL, CALIFORNIA

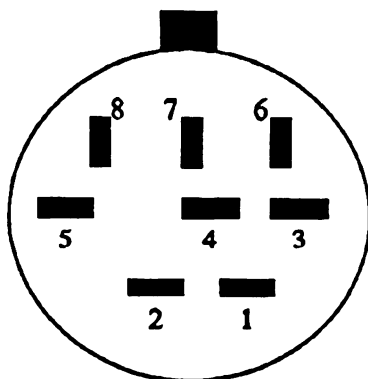
CABLE ASSEMBLY, CPU-2CE SERIAL

Table: 2.10 Audio Connector (Din 8)

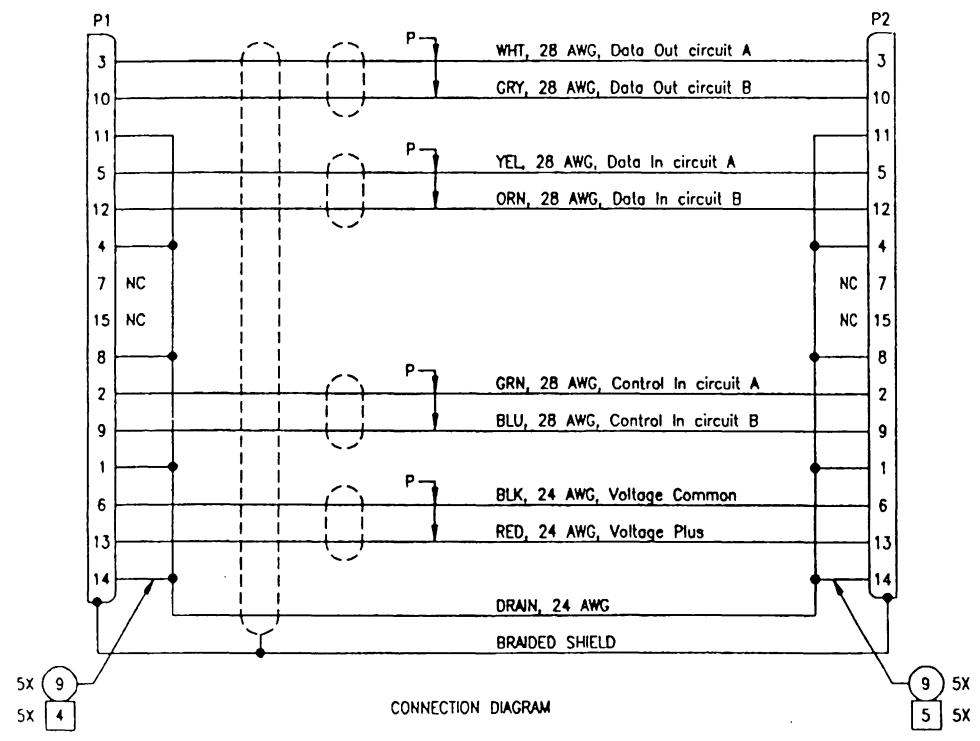
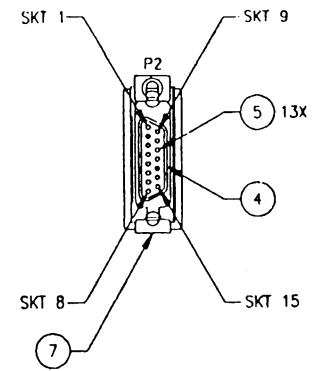
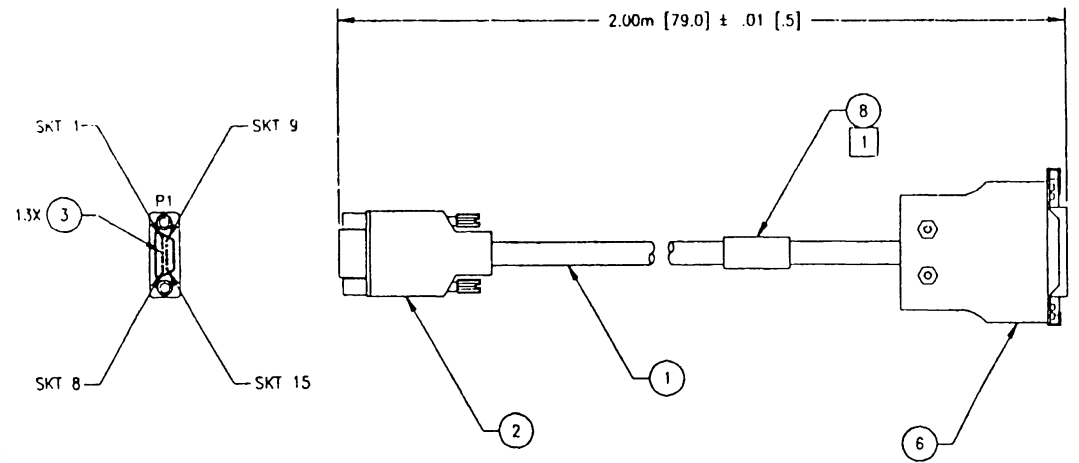
PIN	Description	
1	lin1	
2	lin2	
3	ain-	Audio in
4	lout1	
5	lout2	
6	ain+	Audio in
7	shield	Audio out
8	audio out	

*ISDN signals not supported

Figure: 2.11 Audio Connector



REVISIONS				
DASH	REV	DESCRIPTION	DATE	BY
101	A	PRE-PROD RLSE PER ECO 1375	JS	6/92

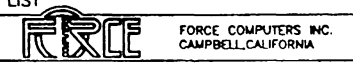


NOTES: UNLESS OTHERWISE SPECIFIED

- 1 MARK ASSEMBLY PART NUMBER AND REV LEVEL WITH CONTRASTING PERMANENT COLOR ON ITEM 8 (LABEL) AND LOCATE NEAR CENTER OF CABLE.
2. DIMENSIONS ARE IN METERS, INCHES ARE IN [].
3. CABLE WIRING PAIR FOR "Control Out" SIGNALS NOT INCLUDED.
- 4 SOLDER ONE END OF ITEM 9 (WIRE) TO DRAIN WIRE OF ITEM 1 (CABLE), THEN ATTACH ITEM 3 (CONTACT) TO OTHER END OF ITEM 9.
- 5 SOLDER ONE END OF ITEM 9 (WIRE) TO DRAIN WIRE OF ITEM 1 (CABLE), THEN ATTACH ITEM 5 (CONTACT) TO OTHER END OF ITEM 9.

ITEM NO.	QTY	REQ'D	FSCM NO.	PART OR IDENTIFYING NO.	MANUFACTURE OR DESCRIPTION	MATERIAL/SPECIFICATION
9	AR		9928		WIRE, BLACK, 28 AWG	BELDEN
8	1		TAG-22-100		LABEL, ADHESIVE	TYTON
7	1		DAS1220-1		SLIDE LATCH KIT	ITTCANNON
6	1		DA121073-150		BACKSHELL, SHIELDED	ITTCANNON
5	13		D110238-478		CONN, SKT, CRIMP, 24-28 AWG	ITTCANNON
4	1		DAA155-A197-FO		CONN, DSUB, FEM, 15 PIN	ITTCANNON
3	13		MDS-S-TS		CONN, SKT, CRIMP, 26-28 AWG	ITTCANNON
2	1		MDSM-155C-211		CONN, MICRO-D, FEM, 15 PIN	ITTCANNON
1	2m		9903		CABLE, ROUND, 8/C, 4 PR, 28(24) AWG, SHLD	BELDEN

PARTS LIST			
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: DECIMALS ANGLES .01 ± .02 .005 ± 0°-30°		SEE PARTS LIST	
MATERIAL		APPROVALS	DATE
DRAWN J. STODDEN		6/92	
FRESH	RES. ENGR.	CHECKED	ISSUED
SIZE	FSCM NO.	DWG. NO.	REV.
C		720-12257-101	A
DO NOT SCALE DRAWING	SCALE 1=1	FILE NAME 12257	SHEET 1 OF 1



CABLE ASSEMBLY, CPU2. ETHERNET

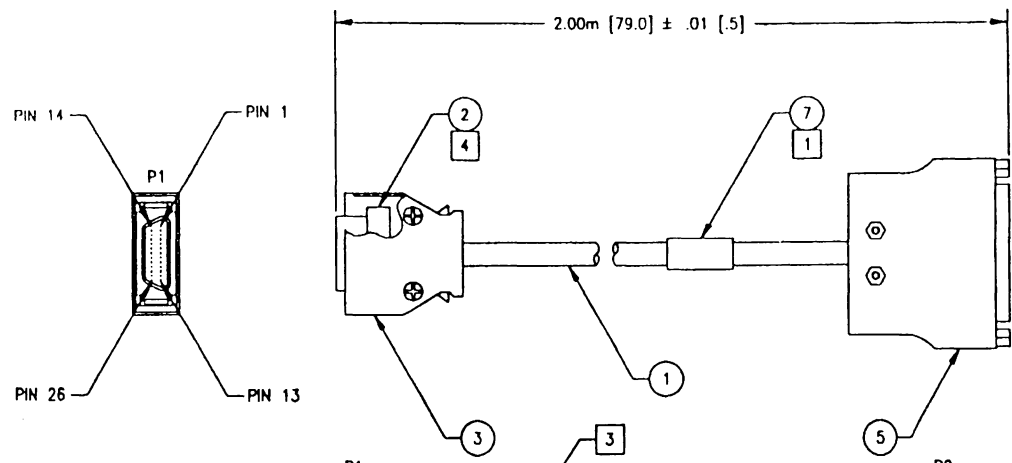
4

3

2

720-12432-101

REVISIONS					
DRAWING	REV	DESCRIPTION	BY	DATE	APPROVED
101	1	PROTOTYPE	JS	7/92	<i>K. Kelly</i>



P1			P2
1	TAN/WHT, 28 AWG		1
2	WHT/TAN, 28 AWG		2
3	BRN/WHT, 28 AWG		3
4	WHT/BRN, 28 AWG		4
5	PNK/WHT, 28 AWG		5
6	WHT/PNK, 28 AWG		6
7	ORN/WHT, 28 AWG		7
8	WHT/ORN, 28 AWG		8
9	YEL/WHT, 28 AWG		9
10	WHT/YEL, 28 AWG		10
11	GRN/WHT, 28 AWG		11
12	WHT/GRN, 28 AWG		12
13	BLU/WHT, 28 AWG		13
14	WHT/BLU, 28 AWG		14
15	VIO/WHT, 28 AWG		15
16	WHT/VIO, 28 AWG		16
17	GRA/WHT, 28 AWG		17
18	WHT/GRA, 28 AWG		18
19	BRN/TAN, 28 AWG		19
20	TAN/BRN, 28 AWG		20
21	PNK/TAN, 28 AWG		21
22	TAN/PNK, 28 AWG		22
23	ORN/TAN, 28 AWG		23
24	TAN/ORN, 28 AWG		24
25	YEL/TAN, 28 AWG		25
26	TAN/YEL, 28 AWG		25
	BRAIDED SHIELD		NC

CONNECTION DIAGRAM

- NOTES: UNLESS OTHERWISE SPECIFIED
- 1 MARK ASSEMBLY PART NUMBER AND REV LEVEL WITH CONTRASTING PERMANENT COLOR ON ITEM 7 (LABEL) AND LOCATE NEAR CENTER OF CABLE.
 - 2. DIMENSIONS ARE IN METERS, INCHES ARE IN [].
 - 3 ALL CABLE CONDUCTORS FEATURE A RING BAND STRIPING. THE SECOND COLOR LISTED REPRESENTS THE RING BAND STRIPE.
 - 4 ITEM 2 (PLUG ASSEMBLY) INCLUDES 2 TERMINATING COVERS.

ITEM NO.	QTY	REV	PART OR IDENTIFYING NO.	NOMENCLATURE OR DESCRIPTION	MATERIAL/SPECIFICATION
7	1		TAG-22-100	LABEL, ADHESIVE	TYTON
6	2		747223-3	SCREWLOCK KIT, FEMALE	AMP
5	1		748678-3	CABLE CLAMP KIT	AMP
4	1		745495-1	RECEPTACLE ASSEMBLY	AMP
3	1		749808-1	BACKSHELL KIT	AMP
2	1		749821-2	PLUG ASSEMBLY	AMP
1	2m		26SDK00025	CABLE, ROUND, 26/C, 28 AWG, SHLD	MADISON

PARTS LIST

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES. TOLERANCES ARE: DECIMALS ANGLES .XX ± .02 .X ± .01 .XXX ± .003		THIS DOCUMENT IS THE PROPERTY OF FORCE COMPUTERS, INC. AND CONTAINS INFORMATION WHICH IS CONFIDENTIAL AND PROPRIETARY TO FORCE COMPUTERS, INC. NO PART OF THIS DOCUMENT MAY BE COPIED, REPRODUCED OR DISCLOSED TO THIRD PARTIES WITHOUT THE PRIOR WRITTEN CONSENT OF FORCE COMPUTERS, INC.		FORCE FORCE COMPUTERS INC. CAMPBELL, CALIFORNIA	
MATERIAL: SEE PARTS LIST		APPROVALS: DRAWN J. STODDEN		DATE: 7/92	
FINISH: CHECKED		RESP. ENGR.		SIZE: C	
DO NOT SCALE DRAWING		ISSUED		SCALE: 1=1	
				FSCM NO. 720-12432-101	
				REV. 1	
				FILE NAME: 12432	
				SHEET 1 OF 1	

CABLE ASSEMBLY, CPU-2CE SERIAL

PRODUCT ERROR REPORT

HARDWARE/SOFTWARE/SYSTEMS

PRODUCT:	SERIAL NO.:
DATE OF PURCHASE:	ORIGINATOR:
COMPANY:	POINT OF CONTACT:
ADDRESS: _____ _____ _____	TELEPHONE: _____ EXT: _____
PRESENT DATE:	
<i>THIS AREA TO BE COMPLETED BY FORCE COMPUTERS:</i>	
DATE: _____	
PR#: _____	
RESPONSIBLE DEPT.:	
<input type="checkbox"/> ENGINEERING	
<input type="checkbox"/> MARKETING	
<input type="checkbox"/> PRODUCTION	
AFFECTED PRODUCT:	AFFECTED DOCUMENTATION:
<input type="checkbox"/> HARDWARE	<input type="checkbox"/> HARDWARE
<input type="checkbox"/> SOFTWARE	<input type="checkbox"/> SOFTWARE
<input type="checkbox"/> SYSTEM	<input type="checkbox"/> SYSTEM
ERROR DESCRIPTION: _____ _____ _____	

Please send this product error report to one of our nearest FORCE COMPUTERS offices:

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

FORCE COMPUTERS FRANCE S.A.R.L.
Le Volta
17-19 rue Jeanne Braconnier
F-92366 Meudon La Foret Cedex

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

FORCE COMPUTERS UK Ltd.
No. 1 Holly Court
3 Tring Road
Wendover
Buckinghamshire HP22 6PE
England

Reader Comment Card

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Please circle one number for each.					
• The manual is well organized.	1	2	3	5	6
• Instructions are complete.	1	2	3	5	6
• The manual is clearly written.	1	2	3	5	6
• Illustrations are clear and helpful.	1	2	3	5	6
• The manual contains enough illustrations.	1	2	3	5	6
• Layout and format enhance the manual's usefulness.	1	2	3	5	6
• This manual meets my overall expectations.	1	2	3	5	6

Please write additional comments, particularly if you disagree with a statement above. Use additional pages if you wish. The more specific your comments, the more useful they are to us.

Comments:

Optional Information-

Name : _____

Title: _____

Company: _____

Address: _____

City/State/Zip: _____

Phone: _____

Thank You

FOLD #1

[AFFIX
STAMP]

FORCE COMPUTERS
3165 Winchester Blvd.
Campbell, CA 95008

FOLD #2 THEN TAPE

(TAPE CLOSE)

Table: 2.11 Keyboard/Mouse Connector Pinout List

Pin #	Description
1	Ground
2	Ground
3	+5 VDC
4	Mouse In
5	Keyboard Out
6	Keyboard In
7	Ground*
8	+5 VDC
	All signals TTL Levels. +5V current-limited

* All signals TTL Levels. +5V current-limited. May be jumpered to Mouse Output.

Figure: 2.12 Keyboard/Mouse Connector

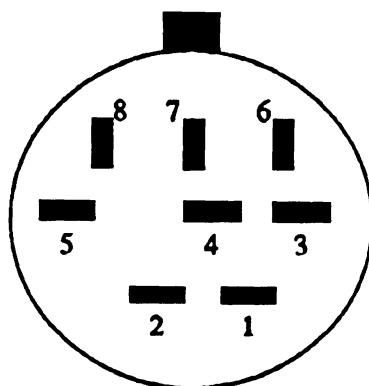


Table: 2.12 SBus Connector Pinout

Pin	Signal	Pin	Signal	Pin	Signal
1	Ground	33	PA[06]	65	D[18]
2	BR*	34	PA[08]	66	D[20]
3	Sel*	35	PA[10]	67	D[22]
4	IntReq[1]	36	err	68	Ground
5	D[00]	37	PA[12]	69	D[24]
6	D[02]	38	PA[14]	70	D[26]
7	D[04]	39	PA[16]	71	D[28]
8	IntReq[2]	40	Ack[1]	72	+5V
9	D[06]	41	PA[18]	73	D[30]
10	D[08]	42	PA[20]	74	Siz[1]
11	D[10]	43	PA[22]	75	Rd
12	IntReq[3]	44	Ack[32]	76	Ground
13	D[12]	45	PA[24]	77	PA[01]
14	D[14]	46	Ground	78	PA[03]
15	D[16]	47	Spare	79	PA[05]
16	IntReq[4]	48	-12V	80	+5V
17	D[19]	49	Clk	81	PA[07]
18	D[21]	50	BG*	82	PA[09]
19	D[23]	51	AS*	83	PA[11]
20	IntReq[5]	52	Ground	84	Ground
21	D[25]	53	D[01]	85	PA[13]
22	D[27]	54	D[03]	86	PA[15]
23	D[29]	55	D[05]	87	PA[17]
24	IntReq[6]	56	+5V	88	+5V
25	D[31]	57	D[07]	89	PA[19]
26	Siz[0]	58	D[09]	90	PA[21]
27	Siz[2]	59	D[11]	91	PA[23]
28	IntReq[7]	60	Ground	92	Ground
29	PA[00]	61	D[13]	93	Ground
30	PA[02]	62	D[15]	94	Ground
31	PA[04]	63	D[17]	95	Reset*
32	LErr*	64	+5V	96	+12V

Table: 2.13 P1 Bus Pinout List

Signal	Row A	Signal	Row B	Signal	ROW C
D(0)	1	BBSY*	1	D(8)	1
D(1)	2	BCLR*	2	D(9)	2
D(2)	3	ACFAIL*	3	D(10)	3
D(3)	4	BGIN(0)	4	D(11)	4
D(4)	5	BGOUT(0)	5	D(12)	5
D(5)	6	BGIN(1)	6	D(13)	6
D(6)	7	BGOUT(1)	7	D(14)	7
D(7)	8	BGIN(2)	8	D(15)	8
GND	9	BGOUT(2)	9	GND	9
SYSCLK	10	BGIN(3)	10	SYSFAIL*	10
GND	11	BGOUT(3)	11	BERR*	11
DS1*	12	BR(0)	12	SYSRESET*	12
DS0*	13	BR(1)	13	LWORD*	13
WRITE*	14	BR(2)	14	AM(5)	14
GND	15	BR(3)	15	A(23)	15
DTACK	16	AM(0)	16	A(22)	16
GND	17	AM(1)	17	A(21)	17
AS*	18	AM(2)	18	A(20)	18
GND*	19	AM(3)	19	A(19)	19
IACK*	20	GND	20	A(18)	20
IACKIN*	21	SERCLK	21	A(17)	21
IACKOUT*	22	GNDSERDAT*	22	A(16)	22
AM (4)	23	GND	23	A(15)	23
A(7)	24	IRQ(7)	24	A(14)	24
A(6)	25	IRQ(6)	25	A(13)	25
A(5)	26	IRQ(5)	26	A(12)	26
A(4)	27	IRQ(4)	27	A(11)	27
A(3)	28	IRQ(3)	28	A(10)	28
A(2)	29	IRQ(2)	29	A(9)	29
A(1)	30	IRQ(1)	30	A(8)	30
-12 VDC	31	+5 VSTBY	31	+12 VDC	31
+5 VDC	32	+5 VDC	32	+5 VDC	32

Table: 2.14 P2 Bus Pinout List

Signal	Row A	Signal	Row B	Signal	Row C
SCSI_DATA0	1	+5 VDC	1	NC	1
SCSI_DATA1	2	GND	2	fp4	2
SCSI_DATA2	3	P1_RETRY*	3	NC	3
SCSI_DATA3	4	P1_A(24)	4	fp8	4
SCSI_DATA4	5	P1_A(25)	5	drvsel	5
SCSI_DATA5	6	P1_A(26)	6	NC	6
SCSI_DATA6	7	P1_A(27)	7	NC	7
SCSI_DATA7	8	P1_A(28)	8	fp16	8
SCSI_DATA8	9	P1_A(29)	9	fp18	9
GND	10	P1_A(30)	10	fp20	10
SCSI_BP*	11	P1_A(31)	11	fp22	11
GND	12	GND	12	fp24	12
TERMPWR	13	+5 VDC	13	fp26	13
GND	14	P1_D(16)	14	fp28	14
GND	15	P1_D(17)	15	fp30	15
SCSI_CNTR2	16	P1_D(18)	16	fp32	16
GND	17	P1_D(19)	17	fp34ct	17
SCSI_CNTR1	18	P1_D(20)	18	feject	18
SCSI_CNTR3	19	P1_D(21)	19	NC	19
SCSI_CNTR5	20	P1_D(22)	20	NC	20
SCSI_CNTR6	21	P1_D(23)	21	NC	21
SCSI_CNTR0	22	GND	22	NC	22
SCSI_CNTR7	23	P1_D(24)	23	NC	23
SCSI_CNTR4	24	P1_D(25)	24	NC	24
SCSI_CNTR8	25	P1_D(26)	25	NC	25
spkr1	26	P1_D(27)	26	NC	26
led_out	27	P1_D(28)	27	NC	27
spkr2	28	P1_D(29)	28	NC	28
NC	29	P1_D(30)	29	NC	29
NC	30	P1_D(31)	30	NC	30
NC	31	GND	31	NC	31
NC	32	+5 VDC	32	NC	32

Expansion Connector Mechanical Layout

The Expansion Connector is an 8-pin header on 0.1" centers with the following pinout:

Table: 2.15 Expansion Connector Electrical Pinout

Pin #	Function	Signal Description
1	Pa26	SBus physical address line 26
2	ramclk1	A 20 MHz clock for a SS-2 ram controller. It has a 25-75 duty cycle.
3	ramsel1*	Second ram controller chip select. Asserted low whenever a virtual address translates to a legal page in Type0 address space between 0x8000000 and 0xFFFFFFFF (pa27 high).
4	Ground	Logic ground for the system
5	ParCS1*	Parity register chip select for second ram controller. Asserted low whenever a virtual address translates to a valid page in Type1 space with a physical address of 0xF4000008 - 0xF400000F.
6	RamClk	A 20 MHz clock for an SS-2 ram controller. It has a 75-25 duty cycle.
7	Pa25	SBus physical address line 25.
8	PErr*	Parity Error signal. This open drain signal is similar to the SBus signal LErr* except that it must meet SBus timings <i>with the data in error</i> , rather than one clock later.
9	Pa27	SBus physical address line 27
10	EXMP	Automatically changes state of the board when a memory SRX Board is installed.
11	GND	Ground
12	VCC	Voltage
13	GND	Ground
14	VCC	Voltage
15	GND	Ground
16	VCC	Voltage

Figure: 2.13 Sun Expansion Connector Pin Orientation

15	13	11	9	7	5	3	1
16	14	12	10	8	6	4	2

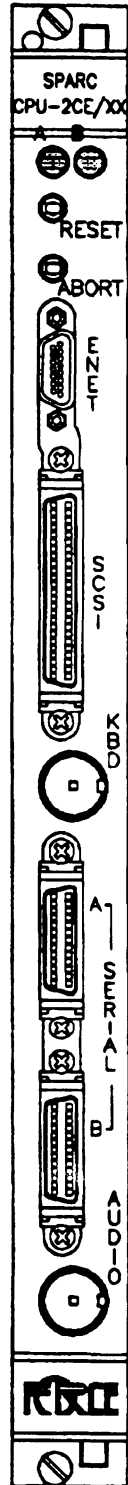
Section 3

HARDWARE

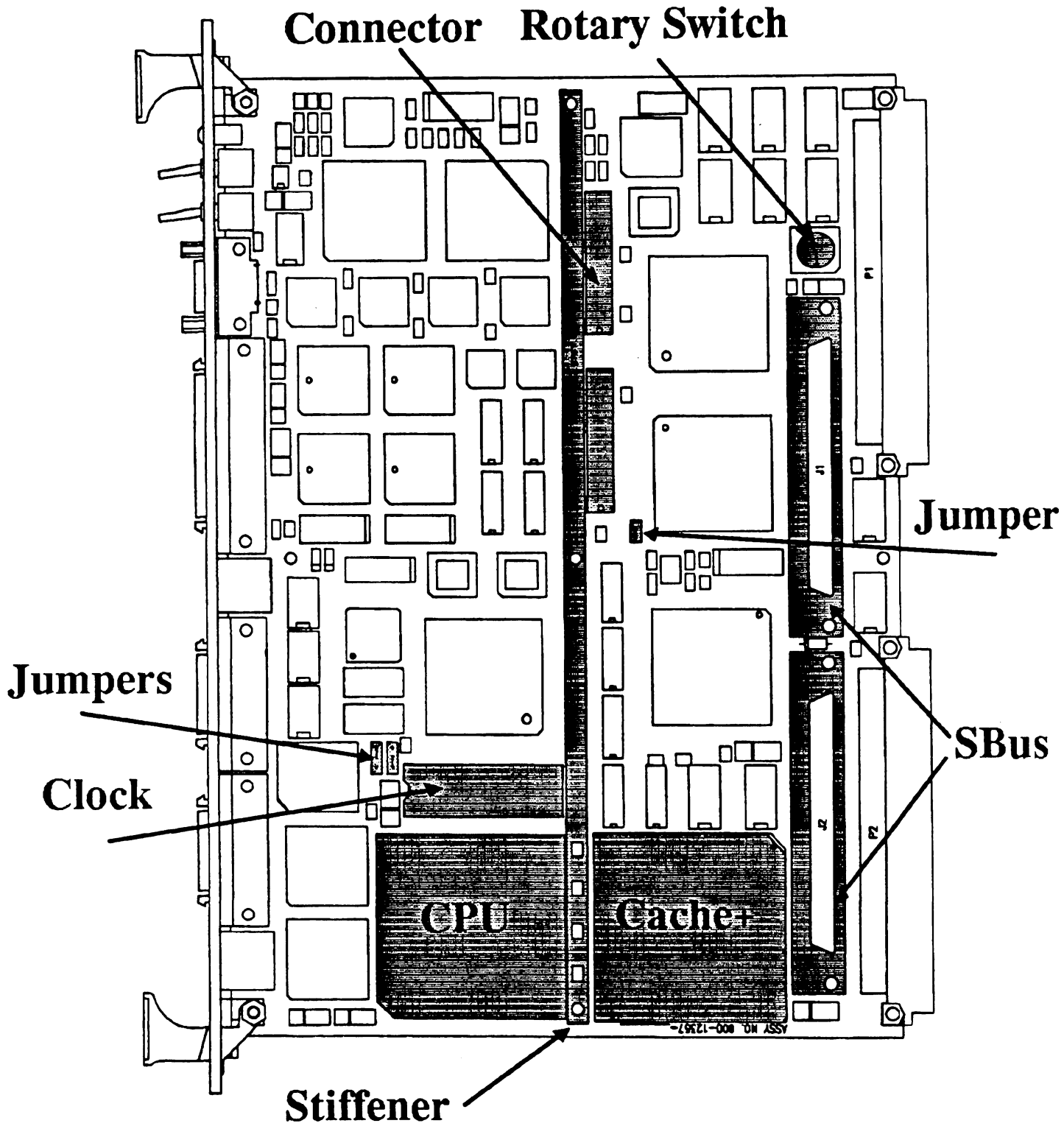
3.1 Card Landmarks

- Figure: 3.1 Shows the Front Panel
- Figure: 3.2 Shows the component-side of the SPARC CPU-2CE card with various call-outs pointing to the card landmarks.
- Figure: 3.3 Shows the solder side of the FORCE SPARC CPU-2CE

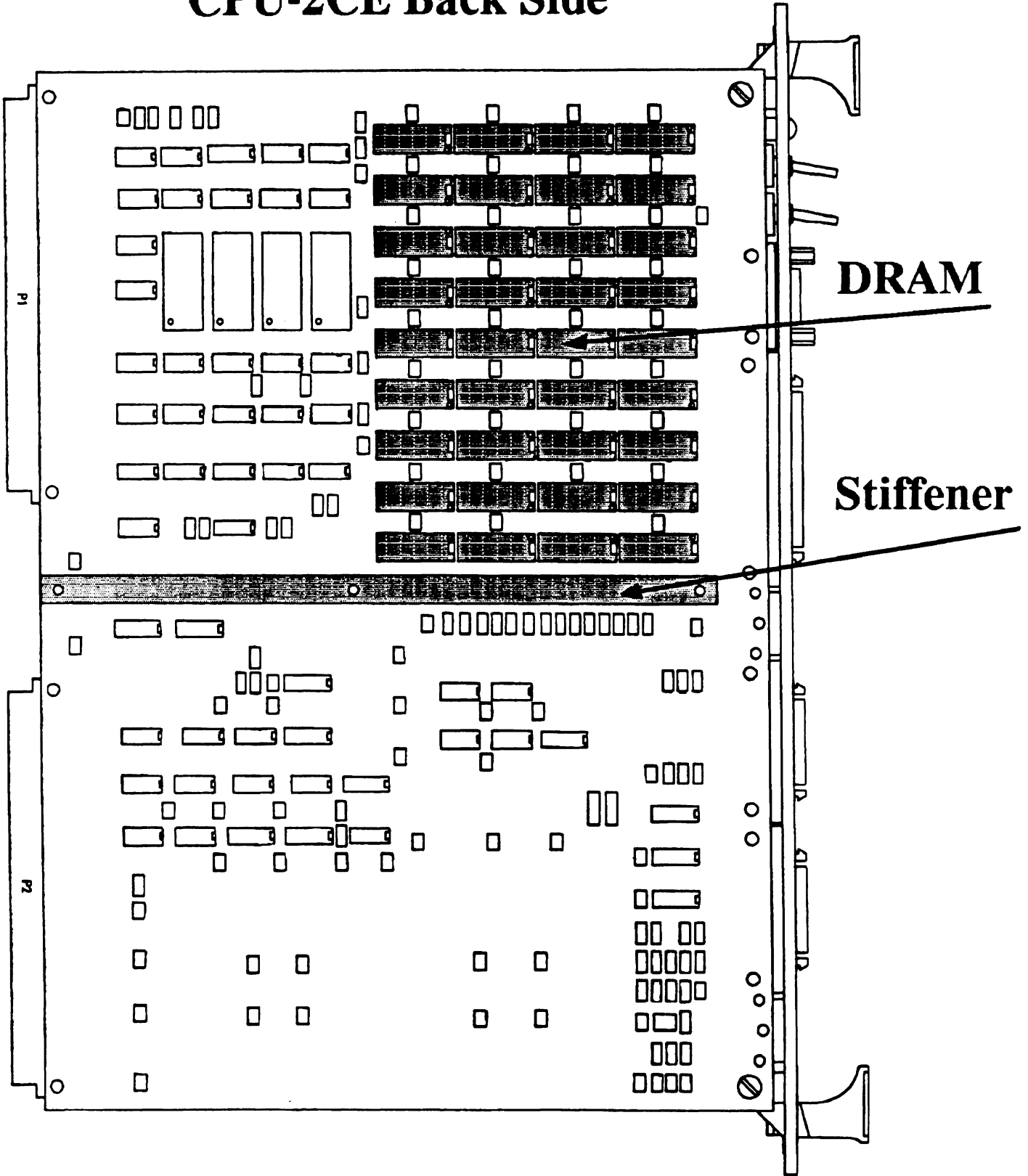
CPU-2CE FRONT PANEL



CPU-2CE Front Side



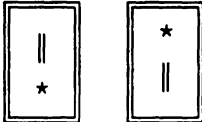
CPU-2CE Back Side



3.1.1 Card Jumpers

The only mechanical jumper blocks for the CPU-2CE are to configure the serial ports and to configure the VME bus system controller functions. The serial ports are linked so both ports are either RS-232-C or RS-423 with the default being RS-232-C. If RS-423 is needed, set the block before power up. Move the jumpers position to set RS-423. The VME default is slot one with the jumper in. See the example below and on the next page for the CPU-2CE defaults.

Serial RS232



VME SLOT 1



Figure: 3.4 RS-232/423 Jumper Blocks (Illustrations shows the default RS-232 setting)

BOARD JUMPERS

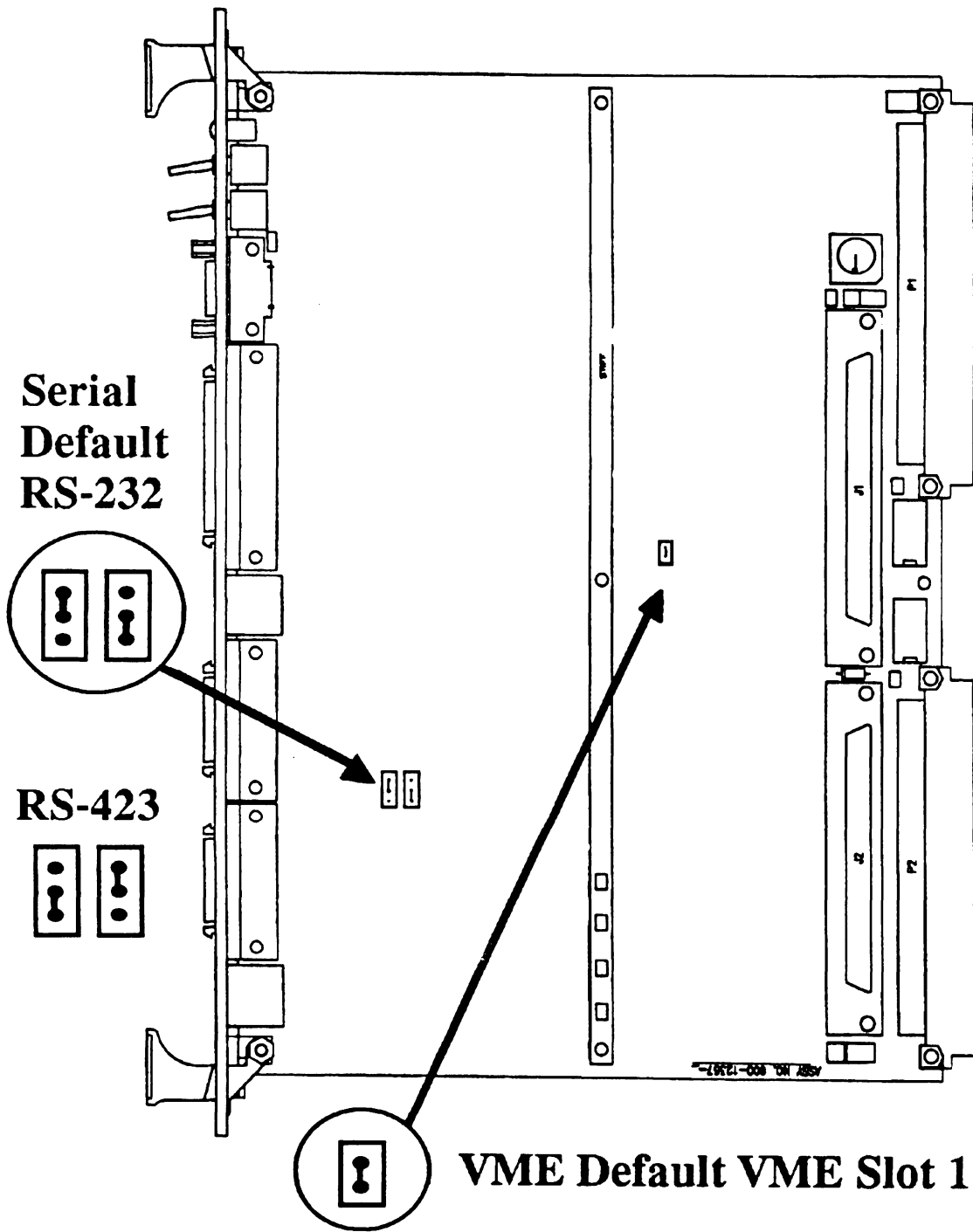
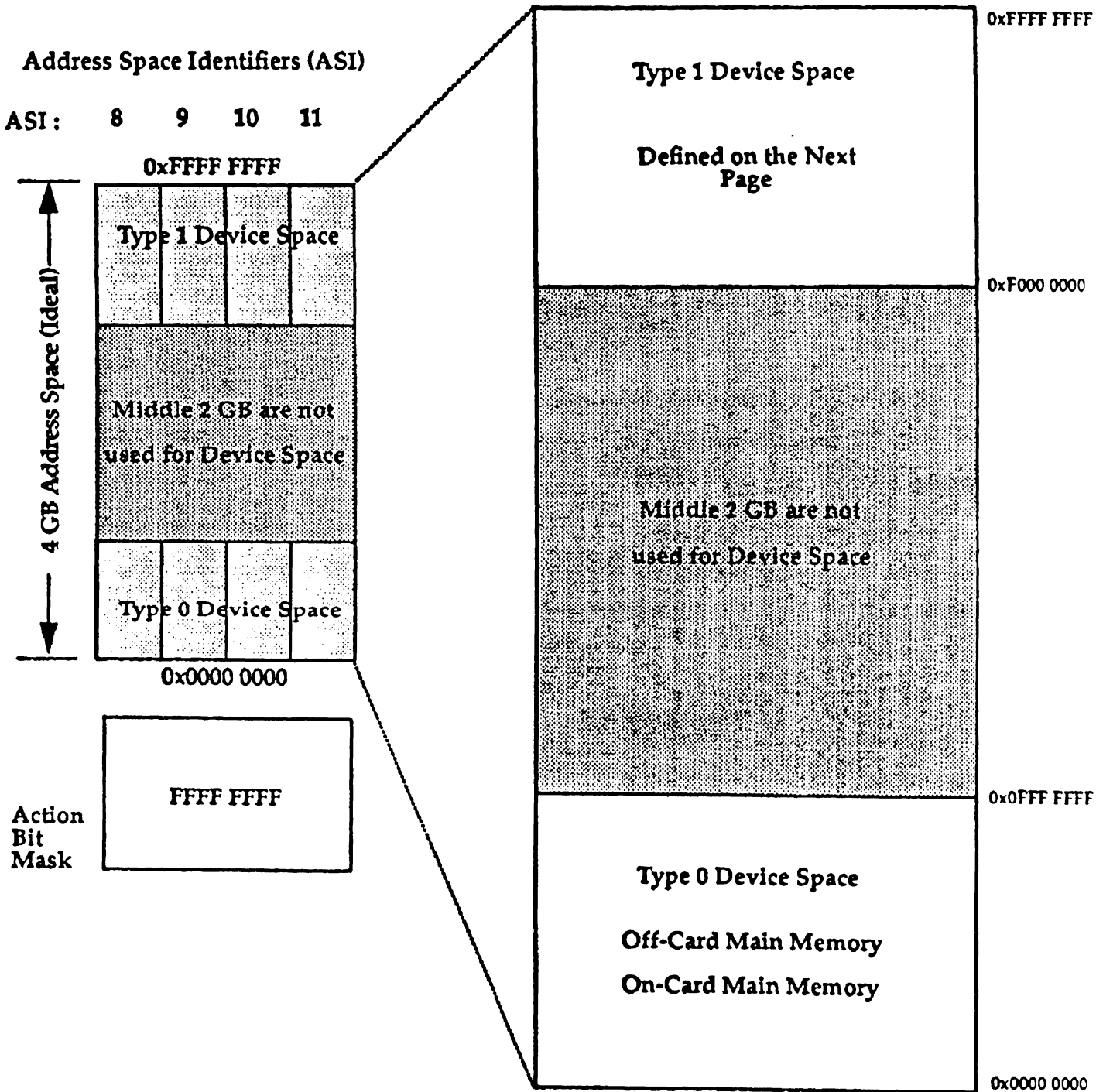


Figure: 3.5 SPARC/CPU-2CE ADDRESS SPACE VIRTUAL Physical Memory Map: ASI Program

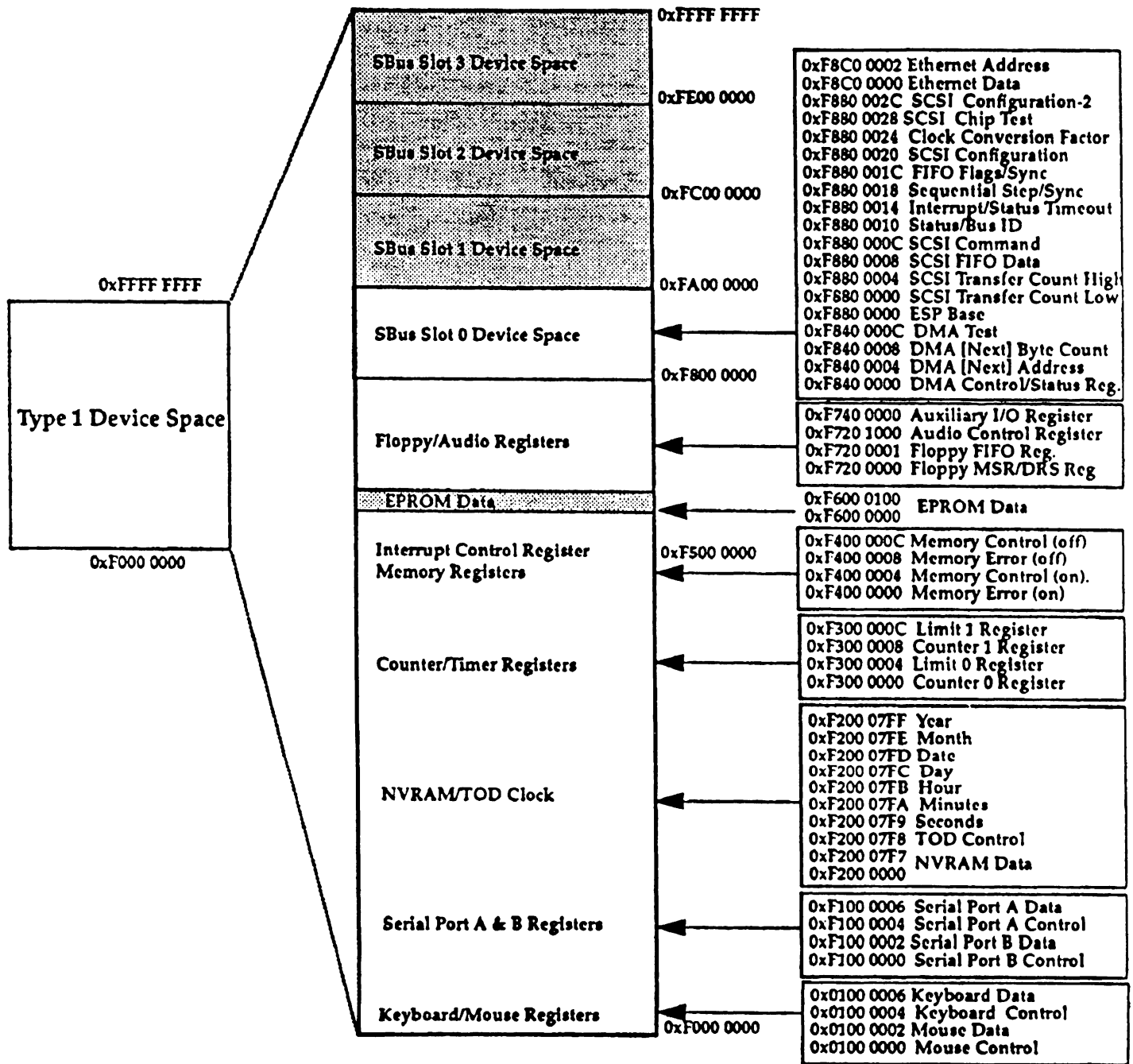
Figure - Physical Memory Map: Device Space



SPARCengine 2 User's Guide - December 1990

Figure: 3.6 Physical Memory Map (see page 64 for LED's, Hexswitch, and (Optional Feature - Flash maps)

Figure -- Physical Memory Map: Type 1 Device Space



Address Busses & Address Spaces

Virtual memory Layout is OS dependent. For SunOS 4.1.1 the information can be obtained from the .H files included with SunOS. This SunOS 4.1.1. example is from the file machdep.c but, this information will change for Sun Solaris and other operating systems.

Table: 3.1 Virtual Memory Layout		
SUN OS 4.1.1		
0xFFFFA000 _	interrupt reg	NPMEG-2
0xFFFF9000 _	memory error reg	
0xFFFF8000 _	eeprom, idprom, clock	
0xFFFF7000 _	counter regs	
0xFFFF6000 _	auxio reg	
0xFFF00000 _	DVMA IOPB memory	_ DVMA
0xFFEF0000 _	Ram Base	
0xFFE80000 _	Prom Base	
0xFFD00000 _	monitor	_ MONSTART
0xFFC00000 _	debugger	_ DEBUGSTART
	SEGTEMP2	
0xFFB80000 _	SEGTEMP	
	segkmap	
0xFF880000 _	NCARGS + MINMAPSIZE	_ Syslimit
0xFF000000 _	Sysmap and other kernel virt addresses	_ Sysbase
	unused	

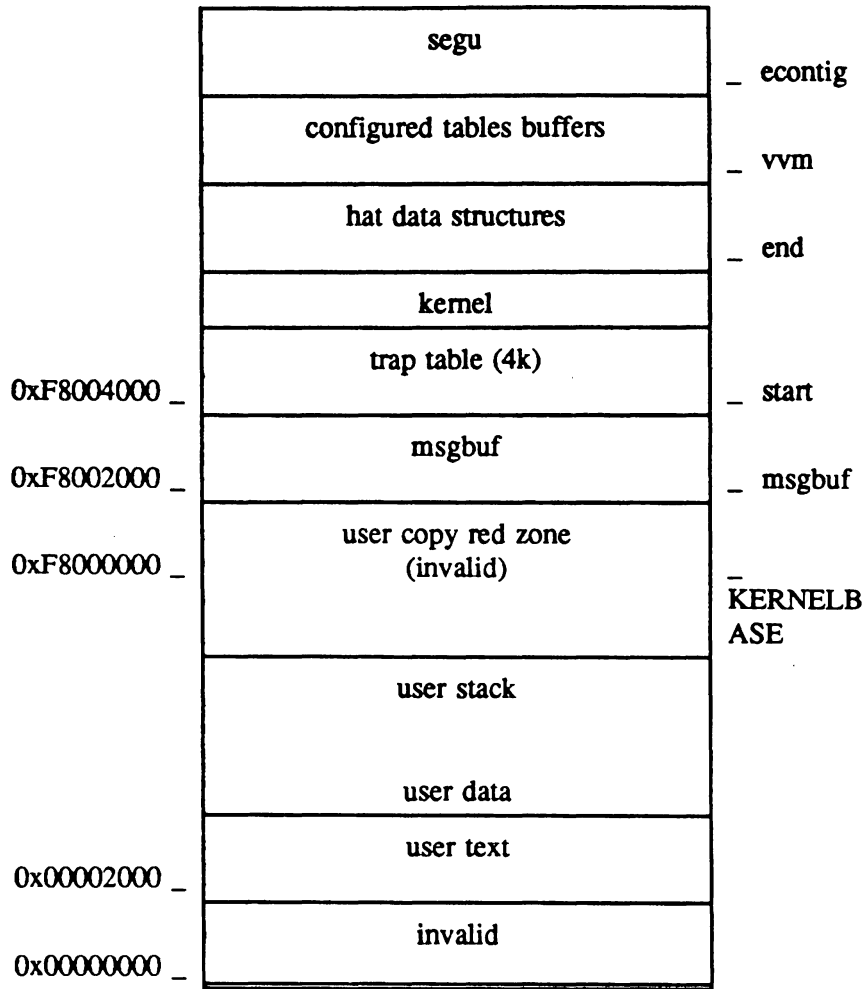


Table: 3.2 Address Space Indicator Contents

ASI#	Use	Address Space Type	Where Defined
0x0	Reserved	Control Space	-
0x1	Reserved	Control Space	-
0x2	System space	Control Space	System Control Space
0x3	Segment Map	Control Space	MMU Control Space
0x4	Page Map	Control Space	MMU Control Space
0x5	High Speed Segment Flush	Control Space	Main Memory Cache Control Space
0x6	High Speed Page Flush	Control Space	Main Memory Cache Control Space
0x7	High Speed Context Flush	Control Space	Main Memory Cache Control Space
0x8	User Instruction	Device Space	Device Space
0x9	Supervisor Instruction	Device Space	Device Space
0xA	User Data	Device Space	Device Space
0xB	Supervisor Data	Device Space	Device Space
0xC	Flush Cache by Segment	Control Space	Main Memory Cache Control Space
0xD	Flush Cache by Page	Control Space	Main Memory Cache Control Space
0xE	Flush Cache by Context	Control Space	Main Memory Cache Control Space
0xF	Hardware Virtual Flush	Control Space	Main Memory Cache Control Space

3.1.2 Device Space

Device Space consist of MMU-mapped main memory and I/O devices that are accessed through translation physical addresses.

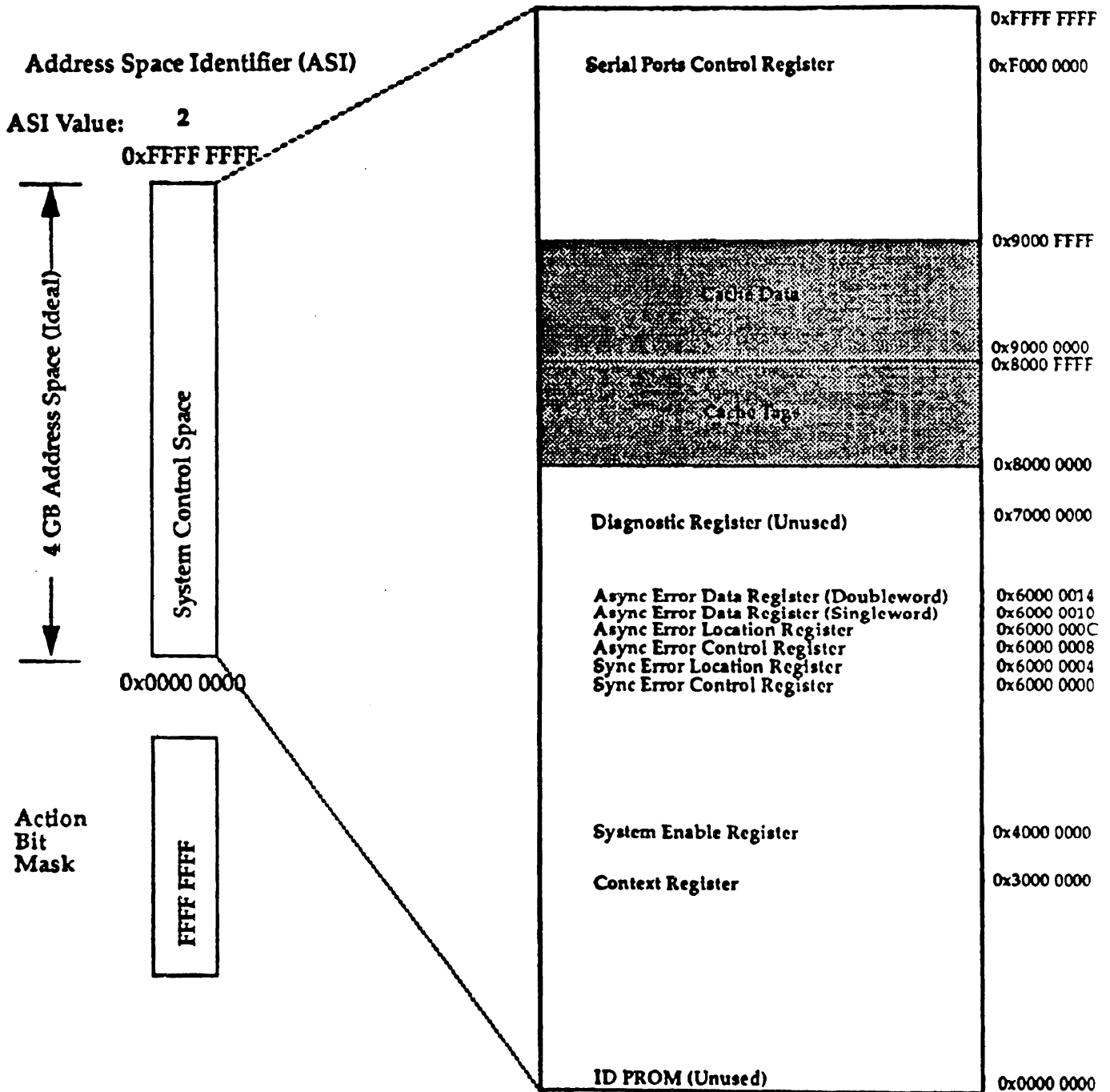
The CPU automatically sets the ASI bits correctly for accesses to device space user data/instruction and supervisor data/instruction.

3.1.3 Control Space

Control Space is all non-device spaces. Control Space is accessed only in supervisor mode. Control Space is sub-divided into System Control Space, Main memory Cache Control Space, Memory management Unit Control Space.

The System Control Space contains control and status registers, serial port bypass, and cache tags. The System Control Space accesses can only be done using alternative space instructions. System Control Space is not mapped.

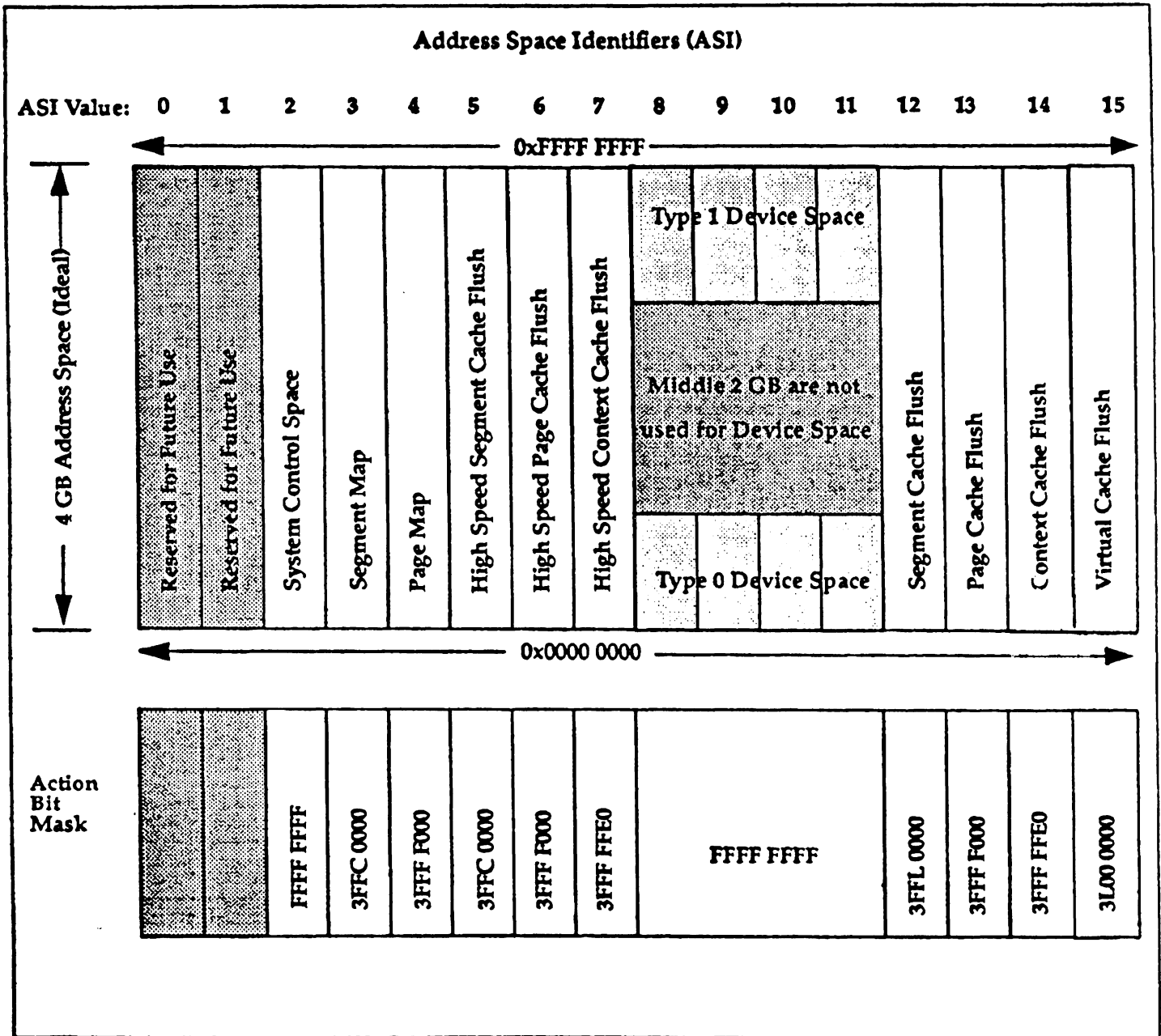
Figure - Physical Memory Map: System Control Space



Address Busses & Address Spaces

Figure: 3.7 Virtual/Physical Memory Map ASI Diagram

Figure - Virtual/Physical Memory Map: ASI Diagram



3.1.4 The Memory Map Page

Figure: 3.8 Type 1 Space (On-Board I/O) (obio)

		SBus Slot 0	
F0000000	Keyboard/Mouse	F8000000	IDPROM
		F8400000	DMA+
		F8800000	NCR 53C90A (SCSI)
F1000000	TTYA/TTYB	F8C00000	AMD 7990 (LANCE)
			SBus Slot 1
F2000000	TOD/NVRAM	FA000000	
			SBus Slot 2
F3000000	Counter/Timers	FC000000	
			SBus Slot 3
F4000000	Parity Ctrl RAM+Ctrl	FE000000	Flash EEPROM (Optional Feature)
F5000000	Internal Controller	FF000000	LED and Hex Switch
		FF800000	Reserved
F6000000	EPROM		
F7000000	i82072 (FLOPPY) AMD 79Ca30 (AUDIO) I/O Register		

Figure: 3.9 Type 0 Space (Main Memory) (obmem)

00000000	16 Megabytes On-board memory
01000000	32 Megabytes - 48 Megabytes Memory Expansion Board
08000000	32 MBytes SRX Board
0A000000	32 Mbytes SRX Memory Expansion
01000000	48 Mbytes SRX Memory Expansion

3.1.5 Interrupt Map

The SPARC CPU-2CE interrupt map is upwards compatible with the SPARCstation 2, but includes additional interrupts. The following table shows the preliminary interrupt table for the SPARC CPU-2CE. Multiple interrupt sources share some of the interrupt levels.

Table: 3.3 SPARC CPU-2CE Interrupt Priority Levels

Level	Source	Additional Source
15	Abort	
15	SYSFAIL	
15	Memory: Parity Error	ECC Uncorrectable Error
14		Counter/Timer 1
13	VME Level 7 VME Mailbox interrupt	Audio
12		Serial Ports
11	VME Level 6	Floppy Disk
10		Counter/Timer 0
9	VME Level 5	SBUS 7
8	VME Level 4	SBUS 6 (Video)
7		SBUS 5
6		SW IRQ6
5	VME Level 3	Ethernet, SBUS 4
4		SCSI SW IRQ4
3	VME Level 2	SBUS 3
2	VME Level 1	SBUS 2
1		SW IRQ1 SBUS 1

NOTE: If VME interrupt level 7 or 6 is needed with VME there is a conflict of the floppy and audio ports interrupts. Either the audio or floppy driver must be commented out of the `GENERIC.VME` and `GENERIC_SMALL.VME` to disable one. Remember to copy the files before modifying them. Search the `GENERIC.VME` and the `GENERIC_SMALL.VME` files in the `.conf` directory for `audioamd`. Use the `"#"` pound symbol at the head of the line to comment out either the audio or `fd` line. Then rebuild the kernel. See the `readme` file in the `.conf` directory for further details.

3.2 Custom Chips

The new ASICs are the Cache controller, DRAM controller, DMA controller, and the MMU chip. Further details on these ASICs are available in their respective chip specifications.

3.2.1 Cache+

The Cache+ ASIC consolidates the original Cache, Buffer, and the cache tags, all on one 224 pin chip. The Cache+/Sunray cpu core will operate at a clock rate between 33 and 40 MHz. The SBus will run at one-half the cpu clock rate. In order to balance the system somewhat (so that cache miss performance will not be terrible) a new RAM controller provides for 1-word-per-clock burst fills. In addition, a micro-TLB eliminates or reduces MMU translation time when the segment or page portions of the virtual address do not change. The Cache+ chip does *not* do parity checking; that function is moved to the RAM+ chip.

3.2.2 RAM+

The RAM+ chip is a DRAM controller and datapath chip for SBus systems. It uses 80 nsec DRAMs to give burst reads and writes at one word per clock. It does parity generation and checking. It has a 2-word write buffer to provide quick response to cache write-through operations. The RAM+ has a private RAM Data bus, isolating the DRAM data from SBus data. This lowers the load on the SBData lines, in addition to allowing the RAM+ chip to do buffered writes while releasing the SBus.

3.2.3 DMA+

The DMA+ chip is an update of the DMA chip, with optimizations to allow full speed SCSI operation along with ethernet on a 20 MHz SBus. It has an internal line buffer and does SBus operations one line at a time (where possible) in order to minimize SBus usage.

3.2.4 MMU+

The MMU+ gate array contains clock generation logic, the system parallel port, and the MMU datapath/decode logic. It is changed from the MMU chip in its decodes for the iodata bus (since the parity control register is no longer on the iodata bus).

3.2.5 SPARC CPU-2CE CACHE Controller

The Cache Controller controls the operation of the cache memory subsystem, provides a path to the main memory of the SBus, supports a single 64 Kbyte instruction/data cache, and monitors/controls the operation of the SBus. For the SBus it handles all arbitration, SBus watch functions, and fetching data and writing data back to memory.

The Cache Controller features:

- Virtual address, direct mapped instruction /data cache controller
- Supports 64 Kbyte write through cache, with 32-byte line size
- Integrates cache tags on-chip (2 K x 21 bits)
- Micro TLB improves performance to main memory or I/O devices
- Controls SBus reads and writes
- Supports three (full master/slave) SBus slots
- Automatically fills cache on cache misses
- Integrates write buffer logic and performs buffered writes that support up to two double words
- Performs cache flush comparisons
- Controls CPU byte packing
- Contains Virtual Address Error Latches
- Maintains copy of context registers
- Contains Bus Error register (both address and data)
- Monitors bus for unacknowledged transfers
- Generates system reset
- Tag support for 16 contexts
- Hardware assisted context, segment, page and virtual flushing

The Cache Controller functions:

The cache is organized as 2048 lines 32 bytes each. The cache tags are 21 bits wide and integrated into the cache controller chip.

Function of Write Through Cache Control

Lines are only loaded into the cache when the IU accesses (reads) a cachable memory location (Read Miss). Data is written into the Cache by the IU only if the location already resides there prior to the write. In all cases writes will be sent back to the main memory. Subsequent reads to the SBus are blocked until any pending writes are completed.

The cache design is implemented as a write-through cache to avoid halting the operation of the IU while the write to main memory occurs (2-entry cache write buffer has been implemented that allows the IU to continue operation) the IU is released early from a write only, if it is a cache hit, otherwise, the IU will be held until the write access will be finished.

Table: 3.4 SBus Bandwidth Estimation

Burst Length	Delays Included	Direction	Bandwidth
64 Byte	DRAM only	write	75 Mbytes/sec
64 Byte	DRAM only	read	67 Mbytes/sec
64 Byte	DRAM+arb+DVMA	write	61 Mbytes/sec
64 Byte	DRAM only	read	56 Mbytes/sec
32 Byte	DRAM only	write	71 Mbytes/sec
32 Byte	DRAM only	read	58 Mbytes/sec
32 Byte	DRAM+arb+DVMA	write	49 Mbytes/sec
32 Byte	DRAM+arb+DVMA	read	43 Mbytes/sec
16 Byte	DRAM only	write	53 Mbytes/sec
16 Byte	DRAM only	read	40 Mbytes/sec
16 Byte	DRAM+arb+DVMA	write	32 Mbytes/sec
16 Byte	DRAM+arb+DVMA	read	27 Mbytes/sec
4 Byte	DRAM+arb+DVMA	write	13 Mbytes/sec
4 Byte	DRAM+arb+DVMA	read	10 Mbytes/sec

3.3 SPARC Expansion Connector

3.3.1 Overview

SPARCstation-1, SPARCstation-1+, SPARCstation IPC, SPARCstation-2 and FORCE CPU-2CE all have an additional expansion connector beyond their SBus slots. These connectors are not identical, but do have some commonality between them. These connectors are *not* part of the SBus. They have different timing and loading requirements, and they are not guaranteed to be on any future Sun SBus machines. The purpose of this document is simply to allow access to these connectors (in the spirit of Openness). But first, this warning:

The expansion connector described in this document will continue to be supported throughout the product lifetimes of the specific machines listed. It will NOT necessarily be supported on any future SPARC Boards.

3.3.1.1 Intent of the Expansion Connector

The original intent of the SPARC Expansion Connector was for memory expansion without performance compromises. To this end, certain assumptions are made about devices which respond to the sel* signal on the Expansion Connector (such as support for burst fills). These requirements will be detailed below. In addition, these connectors are not identical from machine to machine. They were intended for internal use only, with minimal compatibility between them.

3.3.2 Reconciling the Differences

As there are some annoying differences between the expansion connector on various systems, this section will attempt to address methods of living with all of the flavors available.

3.3.2.1 Pa26

SBus address line 26 is not available on the SPARCstation-1 version of this connector. On the other hand, it *is* available on the SBus connector itself for SPARCstation-1's. Pin 46 of SS-1's SBus connector is pa26. On all other SPARCstations mentioned in this document, pin 46 of their SBus connectors are ground. Since Pin 1 of the expansion connector is ground on SS-1, the same functionality (with slightly later timing) can be achieved by OR-ing together pin 46 of the SBus connector and pin 1 of the expansion connector; one of the two will always be ground on these machines.

3.3.2.2 Pa25

SBus address line 25 is only available on SPARCstation-2's expansion connector (although it is also available on the SPARCstation-1 SBus connector). There is no way to depend upon pa25 across all versions of the SBus-based SPARCstations.

3.3.2.3 POK

Power OK is only available on SS-1, SS-1+, and SS-IPC. A better source for reset is the SBus `reset*` signal. The major difference is that `reset*` is asserted on watchdog and software resets, while `pok` is negated (low) only after (or during) a power failure.

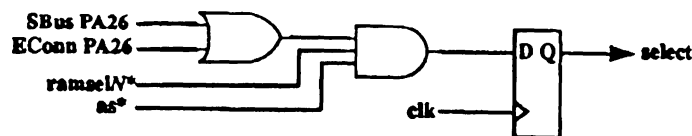
3.3.2.4 RamclkN

The `ramclk`'s are intended for a SPARCstation-2 memory controller. Do not put more than one CMOS load on either clock. It is vastly preferable that only the SBus `clk` signal be used.

3.3.2.5 Ramsel* vs. ramsel1*

`Ramsel1*` is "safe" in that no further decoding need be done; if you see `ramsel1*` and `as*` asserted, you are selected. On the other hand, `ramsel*` needs further decoding; you must see `ramsel*` asserted, `as*` asserted, and `pa26` high before selecting yourself. *Ignoring `pa26` will lead to bus conflicts with the on-board memory controllers.* The simplest resolution to these differences is simply to use `pa26` in all cases, and specify your board's address space for software purposes as `0xC000000 - 0xFFFFFFFF` (`pa26` and `pa27` both high).

Figure: 3.10 Safe Decode Circuit



3.3.2.6 ParityN vs parcs1*, ramclk1, pa25, and perr*

In order to work with both types of machines (SS-2's and those requiring parity) your controller board must have a mode bit which will call `ss2mode`. When not in `ss2mode`, the `parityN` signals must be driven on SBus reads (possibly with the value written on the last write to that address, possibly with parity values calculated on the fly with the data). On SBus writes, the `parityN` signals are driven by the cpu board, but are not intended to be used for error detection (although they could be).

When in `ss2mode`, (and after reset-- until you determine that you are *not* an SS2) you should not drive any of these pins, as you could cause a bus conflict with `parcs1*`, `ramclk1`, or `pa25`.

The `fcode` boot prom should be able to determine cpu type from the top byte of the `hostid` of the system. Use this information to decide how to set `ss2mode`.

3.4 SCSI Interface

3.4.1 SCSI Definition

The Small Computer System Interface (SCSI) uses the NCR 53C90A controller chip and is augmented by the DMA+ chip with a data path of 8 bits. It features full compatibility with SCSI Specification, ANSI x3.131/1986. This allows the SPARC CPU-2CE to directly control and access a wide variety of mass storage devices such as hard disks, tapes, and CD-ROMs.

Up to seven external SCSI devices can be connected to the SPARC CPU-2CE. The SCSI bus is buffered and terminated so SCSI device connections are simple and straightforward, fully supporting synchronous and asynchronous operations. SCSI termination is self sensing. Just plug a terminated device on and the board will adjust termination automatically. For versatility and flexibility, the SCSI bus on the SPARC CPU-2CE is routed to both the front panel via a 50-pin SCSI-II connector and to the board's P2 User I/O pins.

3.4.2 SCSI Performance

The NCR 53C90 Extended SCSI Processor timing is as follows:

Synchronous Mode Speed:

Absolute Maximum:	5.0 MB per second*
Typical:	2.5 MB per second

Asynchronous Mode Speed:

Absolute Maximum:	3.0 MB per second
Typical:	1.75 MB per second

* Note: The current maximum transfer rate in synchronous mode when using SUNOS (version 4.1.1b) is 4.0 Mbytes per second.

3.4.3 SCSI Addresses

The following table lists the physical addresses used to access the SCSI registers for byte loads and stores.

Table: 3.5 SCSI Register Addresses	
Address	Description
0xF0800000	Transfer Count Low
0xF0800004	Transfer Count High
0xF0800008	FIFO Data
0xF080000C	Command
0xF0800010	Status/Bus ID
0xF0800014	Interrupt/Status Timeout
0xF0800018	Sequential Step/Synchronization Transfer Period
0xF080001C	FIFO Flags/Synchronization Offset
0xF0800020	Configuration
0xF0800024	Clock Conversion Factor (write only)
0xF0800028	Chip Test (Extended SCSI Processor test use only)
0xF080002C	Chip (Extended SCSI Processor) Configuration-2

NOTE: Byte accesses must be performed even though the addresses are all fullword-aligned.

3.4.4 SCSI Interrupt

The SCSI Interrupt is Level 3.

3.4.5 Interface Programming

Since the SCSI controller uses the DMA controller to perform the actual transfer of data to and from memory, the two devices must be programmed together. One possible algorithm is as follows:

```
scsi_start()
{
    /*start an operation on the SCSI*/
    lock data pages into contiguous virtual memory;
    DMA_address_register = starting virtual address;
    setup SCSI registers (except for "go");
    DMA_control_status_register = (EN__DMA|INT_EN|(other bits));
    start SCSI;
    /*The SCSI will interrupt us when it is done.*/
}

scsi_interrupt()
{
    /*must drain DMA on a read from disk/write to memory*/
    if (last operation==READ){
        DMA_control_status_register=(DRAIN);
    }
}
```

For a detailed description of the SCSI registers, see the NCR 53C90 Data Sheet.

3.5 Ethernet Interface

3.5.1 Ethernet Interface Definition

The Ethernet controller is the AMD AM7990 LANCE chip. The data is translated for the Ethernet by the AM7992 Serial Interface Adapter (SIA). The LANCE connects directly to the DMA, over unique Ethernet interface signals. The DMA will provide all necessary buffering and arbitration functions to allow the Ethernet chip to access main memory. The DMA Ethernet interface contains a 1 word (32bit) pack/unpack register with consistency control logic. Consistency control ensures that all data written by the Ethernet chip gets to main memory in a deterministic manner.

The LANCE uses multiplexed address and data, so the DMA demultiplexes them internally. The address supported by the LANCE is 24 bits; as per the SPARCstation specification the upper 8 bits of the 32 bit virtual address are driven to 0xFF. The LANCE has a 16 bit data path which can accommodate 8 or 16 bit accesses, by the use of byte masking capabilities.

3.5.1.1 Ethernet Transmits and Receives

When it is in transmit mode, the AM7990 LANCE transfers data from the current buffer in LANCE to a register called a Silo. The output of the Silo is serialized and then goes to the Serial Interface Adapter (AM7992). The output of the Serial Interface Adapter is connected to the Ethernet lines through a transformer.

In the receive mode, the Serial Interface Adapter (AM7992) receives data from the Ethernet cable. It transfers the data to the Silo in the LANCE. The LANCE transfers the data from the silo to the correct buffer in local memory.

3.5.1.2 Ethernet Buffer

The LANCE Controller utilizes up to 128 Kilobytes of memory for buffers. These buffers are allocated from the SPARC CPU-2CE's onboard parity memory.

3.5.2 Ethernet Performance

Maximum Transfer Rate: 10 Mbits per second

Maximum Observed Rate: 7.5 Mbits per second

3.5.3 Ethernet Interrupt

Processor Level 5, SBus level 4

SPARC CPU-2CE can transfer data from the Ethernet into the on-card buffer memory at a maximum (theoretical) rate of 10 megabits/sec.

3.5.4 Ethernet Addresses

The Ethernet Controller chip is in SBus slot 0 address space at 0xF0C00000. The following table lists the address and location of its internal registers:

Table: 3.6 Ethernet Registers	
Address	Description
0xF0C00000	Register Data Port (RDP)
0xF0C00002	Register Address Port (RAP)

For additional information, see the *AMD Am7990 Data Sheet*.

3.5.5 Ethernet Interface Programming

```

/* Initializing on-card ethernet chip          */
/*Abstract:                                     */
variable          *                               */
*                               */
diagnostic.      *                               */
*Algorithm:      *                               */
acquisition      *                               */
*                               */
*                               */
data            *                               */
*                               */
*Routines called:   ini_print, ini_read*        */
*                               */
*****
*/

long le_sys_init(argc,argv)
    int      argc ;
    char     *argv[] ;
{
    extern int      byteswap() ;

    extern short   DSPL_ON;
    extern short   err_dsp;
    extern char    home_adr[6];
    extern short   INTR_ON ;
    extern etraddr myaddr ;
    extern char    net_data [];
    extern short int ringsize,bufsize;
    extern int     totbuf;
    u_short *rdpp ; /* ptr to reg data port */
    u_short *rapp ; /* ptr to reg adr port */

    long          mrc ;
    long          rc ;
    extern int     slot;
    int           i;

    rc = PASS ;

```

```

/* init etherbase_va array */
for (i=0; i<= NUM_SLOTS; i++)
etherbase_va[i] = 0;

ether.etherbase_size = (int)ROUNDUP(sizeof(LE_BLOCK), PAGESZ);
etherbase_va[0] = ether.dvma_start ;

/* Initializing on-card ethernet chip
*/
/* set remaining chip/diagnostic ether
mem variables */
ether.txbuf_n = ONE << TLENMAX ;
ether.rxbuf_n = ONE << RLENMAX ;
ether.txbuf_size = 1024 ; /* may need to tweek this */
ether.rxbuf_size = 1024 ; /* may need to tweek this */

/* map on-card ethernet control block
& buffers here */
/* find the first available block for
control block */
for(ether.etherbase_va=etherbase_va[0],mrc=FAIL;
PASSED(rc) && FAILED(mrc);)
{
    if (FAILED(mrc = execmap(&ether.etherbase_va,
&ether.etherdum_pa,
ether.etherbase_size,
ether.ether_flags)))
    {
        /* increment va by a page */
ether.etherbase_va += PAGESZ;
        /* will we go beyond dvma space ? */
if(((u_long)(ether.etherbase_va+ether.etherbase_size))
>= ((u_long)(ether.dvma_start+ether.dvma_size)) )
        {
            rc = mrc ;
            exec_log(ERROR, ALL_DST,
"le_sys_init(le%d): can not map on-card
etherbase{%d}@0x%x\n",
testing_LE,rc,ether.etherbase_va) ;
        }
    }
}
/* for */
etherbase_va[0] = ether.etherbase_va;
ether.ethermin_va = ether.etherbase_va + ether.etherbase_size;

rdpp = (u_short *) (ethernet_va[0]) ; /* ptr to reg data port */
rapp = (u_short *) (ethernet_va[0] + 2) ; /* ptr to reg adr port */
*rapp = 0; /* Select CSR 0 */
/* set the stop bit in csr0 to
inactivate the lance chip */
*rdpp = (*rdpp) | 0x0004;

for (i=1; i<=slots[0]; i++)
{
    /* Initializing 2nd ethernet chip. */
/* map 2nd ethernet control block &
buffers here */
/* find the first available block for
control block */
for(ether.etherbase_va=etherbase_va[i-1]+ether.etherbase_size,mrc=FAIL;
PASSED(rc) && FAILED(mrc);)
{
    if (FAILED(mrc = execmap(&ether.etherbase_va,
&ether.etherdum_pa,
ether.etherbase_size,
ether.ether_flags)))
    {
        /* increment va by a page */

```

```

ether.etherbase_va += PAGESZ;
/* will we go beyond dvma space ? */
if(((u_long) (ether.etherbase_va+ether.etherbase_size))
    >= ((u_long) (ether.dvma_start+ether.dvma_size)) )
{
    rc = mrc ;
    exec_log(ERROR,ALL_DST,
             "le_sys_init(le%d): can not map 2nd
etherbase{%d}@0x%x\n",
             testing_LE,rc,ether.etherbase_va) ;
}
}
}
/* for */
etherbase_va[i] = ether.etherbase_va;
ether.ethermin_va += ether.etherbase_size;
rdpp = (u_short *) (ethernet_va[i]) ; /* ptr to reg data port */
rapp = (u_short *) (ethernet_va[i] + 2) ; /* ptr to reg adr port */
*rapp = 0; /*Select CSR0 */
/* set the stop bit in csr0 to
inactivate the lance chip */
*rdpp = (*rdpp) | 0x0004; /* set stop bit in csr0 */
} /* for */

totbuf = BUFSPC ;
bufsize = 1024;
ringsize = TLENMAX;

/* copy ethernet address from myaddr
to home_adr */
bcopy(myaddr,home_adr,sizeof(etraddr)) ; /* 1 to 1 copy */

net_data[0] = '\0'; net_data[1] = '\0'; net_data[2] = '\0';
net_data[3] = '\0'; net_data[4] = '\0'; net_data[5] = '\0';
net_data[6] = '\0'; net_data[7] = '\0'; net_data[8] = '\0';
net_data[9] = '\0'; net_data[10] = 'T'; net_data[11] = 'h';
net_data[12] = 'i'; net_data[13] = 's'; net_data[14] = ' ';
net_data[15] = 'i'; net_data[16] = 's'; net_data[17] = ' ';
net_data[18] = 'a'; net_data[19] = ' '; net_data[20] = 't';
net_data[21] = 'e'; net_data[22] = 's'; net_data[23] = 't';
net_data[24] = ' '; net_data[25] = 'o'; net_data[26] = 'f';
net_data[27] = ' '; net_data[28] = 't'; net_data[29] = 'h';
net_data[30] = 'e'; net_data[31] = ' '; net_data[32] = 'L';
net_data[33] = 'A'; net_data[34] = 'N'; net_data[35] = 'C';
net_data[36] = 'E'; net_data[37] = ' '; net_data[38] = 'E';
net_data[39] = 'm'; net_data[40] = 'u'; net_data[41] = 'l';
net_data[42] = 'a'; net_data[43] = 't'; net_data[44] = 'o';
net_data[45] = 'r';

err_dsp = SET; /* error display flag */
DSPL_ON = 0; /* By default, no debug msgs */
INTR_ON = 0 ;

go_reset();

return(rc) ;
} /* le_sys_init */

```

3.6 SBus Interface

3.6.1 Introduction

The SBus is a non-proprietary I/O expansion bus developed by Sun Microsystems for some new Sun systems and card products. The bus resides in type 1 device space.

3.6.2 SBus Connections

The SBus connector is located on the component side of the SPARC CPU-2CE card, near the P2 connector.

PROCESS TRAP LEVELS	SOURCE	TRAP TYPE	TRAP VECTOR ADDRESS
15	Async SBus Error	31	1F
14	Timer 1	30	1E
13	Audio Interrupt	29	1D
12	On-board Serial Ports	28	1C
11	Floppy Disk Controller	27	1B
10	Timer 0	26	1A
9	SBus IRQ 7	25	19
8	SBus IRQ 6	24	18
7	SBus IRQ 5	23	17
5	Ethernet Cntrl/SBus IRQ 4	21	15
4	SWI	20	14
3	SCSI Controller/SBus IRQ 3	19	13
2	SBus IRQ 2	18	12
1	SWI/SBus IRQ 1	17	11

The column marked "Level" is the Interrupt Level which is encoded to the SPARC main processor, or Integer Unit, by the MMU. The source to the MMU is listed under "Source."

3.6.3 SBus Slot Addresses

Address bits PA[27:26] select one of two SBus slots, and bits PA[24:0] are available for addressing devices on the SBus card. Bit PA[25] is not used.

The following table shows how bits PA[27:26] select the SBus slots.

Table: 3.8 SBus Slot Addresses		
PA[27:26]	Slot	Address Range
00	SBus slot 0	0xF0000000 to 0xF1FFFFFF
01	SBus slot 1	0xF4000000 to 0xF5FFFFFF
10	SBus slot 2	0xF8000000 to 0xF9FFFFFF
11	SBus slot 3	0xFE000000 to 0xFFFFFFFF

SBus slot 0 and 3 are not physical slots; they are occupied by devices residing on the SPARC CPU-2CE card.

SBus slot 1 and 2 on the SPARC CPU-2CE are physical slots, where cards such as the SBus video cards can be utilized.

The SBus device addresses are described in the following sections.

3.6.4 SBus Slot 0 Devices

The SBus Slot 0 devices are physically on the CPU card. SBus slot 0 is in type 1 space at a base address of 0xF0000000. The base address contains the card ID number, DMA registers, SCSI registers, and Ethernet registers.

Table: 3.9 SBus Slot 0 Addresses	
Offset	Description
0xF0000000	Card ID (0xFE810101)
0xF0400000	DMA Registers:
	Control/Status Byte Count Diagnostics
0xF0800000	SCSI Registers:
	Transfer Count Register FIFO Register Command Register Status Register Select/Reselect Bus ID Register Sequence Step Synchronous Transfer Period Register FIFO Flags Register Synchronous Offset Register Configuration Register Clock Conversion Register Test Register
0xF0C00000	Ethernet Registers:
	Data Port Address Port

NOTE: The slot 0 space from 0xF2000000 to 0xF3FFFFFF is not used.

3.6.4.1 Card ID

The card ID (data 0xFE810101 at location 0xF0000000) is a line of Forth code that identifies the card in SBus slot 0 to the operating system (in this case, the CPU card identifies itself).

3.6.4.2 DMA Registers

The DMA register space contains four registers, addressed by fullword loads and stores.

Address	Description
0xF0400000	DMA Control/Status Register
0xF0400004	DMA Address Register
0xF0400008	DMA Byte Count Register
0xF040000C	DMA Diagnostic Register

DMA Control/Status Register

The DMA control/status register has the following format:

D[31:27]- DEV_ID

These read-only bits contain the data 0x0B1000. They identify the Ethernet device type.

D[27:15]

These bits are unused; they read back as 0's.

D14- TC

This read-only bit indicates the that the byte counter has expired (decremented to 0).

D13- EN_CNT

This read/write bit enables the DMA byte count register.

D[12:11]- BYTE_ADDR

These two read-only bits indicate the next byte number to be accessed.

D10- REQ_PEND

This read-only bit is set when the DMA interface is active. Note that RESET and FLUSH must not be asserted when this bit is active.

D9- EN_DMA

This read/write bit is set to enable DMA activity and reset to disable it.

D8- WRITE

This read/write bit is set for DMA from memory to device (write) and reset for DMA from device to memory (read).

D7- RESET

Setting this read/write bit causes a hardware reset. ERR_PEND, PACK_CNT, INT_EN, FLUSH, DRAIN, WRITE, EN_DMA, REQ_PEND, EN_CNT, and TC are all set to zero. RESET remains set, and must be reset to resume operation.

D6- DRAIN

Setting this read/write bit forces remaining pack register bytes to be drained to memory. It resets itself.

D5- FLUSH

This write-only bit forces PACK_CNT and ERR_PEND to zero. It always reads as zero.

D4- INT_EN

This read/write bit enables interrupts when it is set.

D[3:2]- PACK CNT

This read only field provides the number of bytes in the pack register.

D1- ERR_PEND

This read-only bit is set when a memory exception occurs, and is reset by setting FLUSH. DMA activity stops until it is reset.

D0- INT_PEND

This read-only bit is set when TC=1, or when an external device raises an interrupt.

DMA Address Register

The DMA address register is located at address 0xF0400004. It holds the virtual address of the DMA transfer. Initially, it should be loaded with the virtual address where the DMA will start. As the DMA proceeds, both the DMA byte count register and bits [23:0] of the DMA address register are decremented.

Bits [31:24] of the DMA address register indicate which 16-megabyte portion of virtual memory is accessed. These bits are latched; they do not change as the DMA proceeds.

DMA Byte Count Register

When the EN_CNT bit in the DMA control/status register is set, bits [23:0] of this register keep a count of the number of bytes transferred by the DMA circuits. It should be loaded with the total number of bytes to be transferred; it decrements towards zero as the bytes are transferred. When it reaches zero, it sets TC and INT_PEND in the DMA status/control register.

Note that bits [31:24] are hardwired to zero.

DMA Diagnostic Register

This register is not implemented at this time.

3.6.5 SBus Slot 1 Devices and Slot 2 Devices

Any device plugged into an actual SBus connector on the CPU card is an SBus device.

As of the print date of this manual, the SPARC CPU-2CE supports two video cards, color and monochrome, that can be plugged into the SBus connector on the CPU card. These cards are interchangeable at the card level, but require different cables and monitors. The two cards have different registers. For specific information on these cards, please refer to **The SBus Color & Monochrome Video Cards User's Manual**.

For all other SBus cards, refer to the technical documentation and manuals provided with the card.

The Sbus supports all SBus devices that Sun's SPARCstation 2 supports.

3.6.5.1 SPARC Assembly Language Example

```
/*
 * sbus.s
 */
#define sbud_id          0x00010000/* sbus ID virtual address
*/
#define sbus_phy        0xf0000000/* sbus ID physical address
```

```

*/
#define TYPE1_ATTRIBUTES                                0xd4000000/* valid, writable, don't
cache */

        .seg      "text"
        .global  Sbus_id
!-----
! Read SBus slot 0 card ID register
Sbus_id:
    save      %sp, -MINFRAME,%sp! preserve calling indow
    set       sbus_id, %o0          ! Virtual address to be mapped.
    set       TYPE1_ATTRIBUTES, %o2      ! Page table entry
attributes.
    set       sbus_phy, %o3          ! Physical address to be mapped.

    call      _pmap_memory          ! Map in specified block of memory.
    mov       %o0, %o1              ! Upper virtual address to be mapped.
    set       sbus_virt, %i15        ! Address of SBus register.
    ld        [%o0],%i14            ! Read Card ID status
    nop
    ret
    restore

/*
 * PMAP.S
 */
#define SEGINCR                                0x40000/* offset between adjacent
segments */
#define SGSHIFT                                18/* LOG2(NBSG) */
#define ASI_SM                                  0x3/*segment map */
#define PAGINCR                                0x2000/* offset between adjacent pages
*/
#define PGSHIFT                                13/* LOG2(NBPG) */
#define ASI_PM                                  0x4/* page map */

        .seg      "text"
        .global  _pmap_memory

/*
 * Synopsis: status=pmap_memory(low_va, hi_va, attributes, low_pa);
 * status          : (int) don't care
 * low_va          (%i0): (unsigned long) initial virtual
address
 * hi_va           (%i1): (unsigned long) final virtual
address
 * attributes      (%i2) : (unsigned long) page table entry
attributes
 * low_pa         (%i3): (unsigned long) initial
physical address
 *
 * Function "pmap_memory" (physical map memory) linearly maps the range of
 * virtual addresses specified by 'low_va' and 'hi_va', beginning at physical
 * address 'low_pa', with page table entry attributes 'attributes'.
 */
_pmap_memory:
    save      %sp, -MINFRAME,%sp
    /*
     * Fill in segment table entries for a linear mapping of main memory.
     */
    set       SEGINCR, %i13          ! Adjacent segment table entry offset.
    sub       %i13, 0x1, %i14        ! Mask for segment table addressing.
    andn     %i0, %i14, %i12        ! Adjust initial virtual address.
3:    srl     %i2, SGSHIFT, %i10     ! Segment table entry (pmsg number).
    dd       %i0, [%i2]ASI_SM       ! Set segment table entry.
    add      %i2, %i13, %i12        ! Increment address by one segment.
    cmp      %i2, %i1              ! If current virtual address is not
    bgt      4f
    nop

```

```

bleu          3b          ! greater than the final virtual
nop          ! address, continue.
4:
/*
 * Fill in page table entries for a linear mapping of main memory.
 */
set          PAGINCR, %i3    ! Adjacent page table entry offset.
sub          %i3, 0x1, %i4   ! Mask for page table addressing.
andn         %i0, %i4, %i2   ! Adjust initial virtual address.
/*
 * Generate the initial page table entry.
 */
andn         %i3, %i4, %i1   ! Adjust initial physical address.
5: srl          %i1, PGSHIFT, %i0 ! Page number of page table entry.
or           %i0, %i2, %i0   ! Combine attributes and page number.
sta          %i0, [%i2]ASI_PM ! Set page table entry.
add          %i1, %i3, %i1   ! Increment physical address by a
page.
add          %i2, %i3, %i2   ! Increment virtual address by a page.
cmp          %i2, %i1       ! If current virtual address is not
bgt          6f
nop
bleu          5b          ! greater than the final virtual
nop          ! address, continue.
6:
/*
 * Mapping completed.
 */
ret
restore

```

3.6.5.2 Forth Example

```
f000.0000 constant SBUS-SLOT0      \ physical address
0001.0000 constant SBUS-ID        \ virtual address

SBUS-SLOT0 obio SBUS-ID map-page   \ map into type1 (on-card I/O) space
SBUS-ID l@ .                       \ fetch and display sbus-id
```

NOTE: SBus slot 0 contains the DMA, Ethernet, and SCSI chips. Slot 1 & 2 contains SBus cards.

3.6.5.3 C/SunOS Example

```
/*
 * caddr_t map_SBus_address(u_int p_addr, u_int num_pages)
 *
 * Inputs: Physical address and the # of pages to map.
 * Returns: Virtual address of the mapped page(s).
 */

caddr_t map_SBus_address(p_addr, num_pages)
u_int p_addr;                /* physical address on the SBus to map
*/
u_int num_pages;            /* # of pages to allocate/map */
{
    long                v_pgnum;
    u_int               v_addr;
    u_int               offset;
    u_int               pg_tbl_ent;

/*
 * Save the offset from the page boundary so that
 * we can apply it to the virtual address once the
 * page is mapped in.
 */
    offset = p_addr & MMU_PAGEOFFSET;

/*
 * Create a page table entry that maps the physical address
 * "p_addr" into SBus space.
 */

    pg_tbl_ent = PGT_OBIO | PG_NC | btop(p_addr);

/*
 * Allocate virtual pages that we can use from the
 * kernel map of available virtual addresses.
 */
    v_pgnum = rmalloc(kernelmap, (long)btoc(num_pages));
    if (v_pgnum == 0) /* no addresses available from kernelmap */
        return (caddr_t)0;

/*
 * Convert the virtual page # to a virtual address
 */
    v_addr = (u_int)kmxtob(v_pgnum);

/*
 * Map the physical address into our allocated virtual address.
 */
    segkmem_mapin(&kseg, v_addr, num_pages, PROT_READ | PROT_WRITE,
                  pg_tbl_ent, 0);

/*
 * Return to the caller of this function the virtual address of
 * the mapped SBus physical address.
 */
    return (caddr_t)(v_addr | offset);
} /* end of map_SBus_address */
```

3.7 RAM+

3.7.1 FEATURES

- Provides all DRAM timing and refresh cycles.
- Controls 64 Mbytes.
- Generates and checks byte-wide (or word-wide?) parity.
- Supports 32-byte and 8-byte SBus sizes.

3.7.2 PIN DESCRIPTION

The RAM+ chip is built in a 196-pin plastic flat pack. It has *X* outputs, *Y* bidirectional pins, *Z* inputs, *A* VCC pins, and *B* Grounds.

3.7.2.1 Sbus Interface

clk	(TLCHT) System clock. This signal runs at up to 20 MHz with a 50% duty cycle. (Perhaps: for clockrates greater than 20 MHz, the configuration register must be set to two-clocks-per-word mode.)
sb_d[31:0]	(BD8) Sbus data. On read cycles, this data is guaranteed valid on the clock following an "ack."
sb_pa[27:0]	(TLCHT) SBus physical address. It is guaranteed to be valid before the clock edge in which sb_as_ is sampled low.
sb_siz[2:0]	(TLCHT) SBus size control signals. The RAM+ chip supports an extension of the normal SBus size codes for 8-byte and 32-byte bursts.
sb_rd	(TLCHT) SBus read control signal. This signal is asserted (high) for read cycles and negated (low) for write cycles.
sb_ack32_	(BT4) SBus acknowledge for 32-bit word size. Tri-state output. It is asserted (low) for one clock for each word transferred.
sb_merr_	(BT4) SBus Memory Error. Asserted after data to report parity error.
sb_as_	(TLCHT) SBus address strobe. This signal is asserted (low) by the system controller for the duration of an SBus cycle.
selram_	(TLCHT) Ram select input. This signal is asserted (low) to indicate that a DRAM access is taking place. It is asynchronous when sb_as_ is not asserted. (It only guarantees a setup to a clock edge when as_ is also seen asserted.) As long as parity is internally enabled, all accesses are checked. There is no "par_en_" signal on the RAM+ chip.

par_cs_ (TLCHT) Parity register select input. (There is a problem currently in that the MMU chip asserts `ioden_` during `par_cs_` and `pio_sel_` cycles.)

totals: inputs: 36, outputs: 2 , bidirects: 32.

3.7.2.2 DRAM Interface

ram_d[31:0] (BD4) DRAM data bus. Driven during write cycles, latched during reads on the rising edge of `cas_`. Each signal drives 4 pins.

ma[10:0] (BT4) Multiplexed DRAM address bus. These 11 address lines contain the addresses to be used by the DRAM at the next falling edge of `ras_` or `cas_`. They will be buffered externally. DRAMs expect 0 nsec setup time and 10 nsec hold time wrt `ras_` and `cas_`.

we_ (BT4) Write Enable outputs to all DRAM banks. Assumed to be externally buffered..

ras_[15:0] (BT8) Row Address Strobe outputs to DRAM banks, one for each byte. Each signal drives 9 pin.

cas_[15:0] (BD8) Column Address Strobe outputs to DRAM banks, one for each byte. Each signal drives 9 pins. These signals use bidirectional pads because the `cas_` signal is read back in to provide the clock for the read data register.

p[3:0] (BD8) Parity input/output. Each signal drives 4 pins.

totals:
inputs: 0 , outputs 28 , bidirects: 52.

3.7.2.3 Power/Ground

totals: 41 VCC: x , Ground: $41 - x$.

3.7.2.4 LSI Logic Buffer Naming Conventions

Input pads are generally TLCHT's. Outputs are generally BTx's where x is the number of milliamps of drive capacity. BDx's are bidirectional pads with x milliamps of drive.

Table: 3.11 Address Ranges

<u>Address Range/Bank Pin</u>	<u>Usage</u>
0000000 - 3FFFFFF / 0	DRAM (parity-protected)
8000000 - BFFFFFF / 1	"
4000000 - 7FFFFFF / 0	Video Registers
C000000 - FFFFFFF / 1	"

0000000	RAM Bank 0
1000000	Ram Bank 1
2000000	RAM Bank 2
3000000	RAM Bank 3
4000000	Video

- RAM Banks may be made up of 1 Mbit or 4 Mbit DRAMs, providing up to 64 Mbytes of memory.

- When a bank is filled with 1 Mbit DRAMs, the memory contents are "shadowed" four times.

- If a bank is not populated with DRAM, the RAM+ chip still responds with an ack32_ to all requests. No errors (other than possible parity errors) are returned.

- The base address of the Frame Buffer is programmable.

3.7.3 SBus INTERFACE

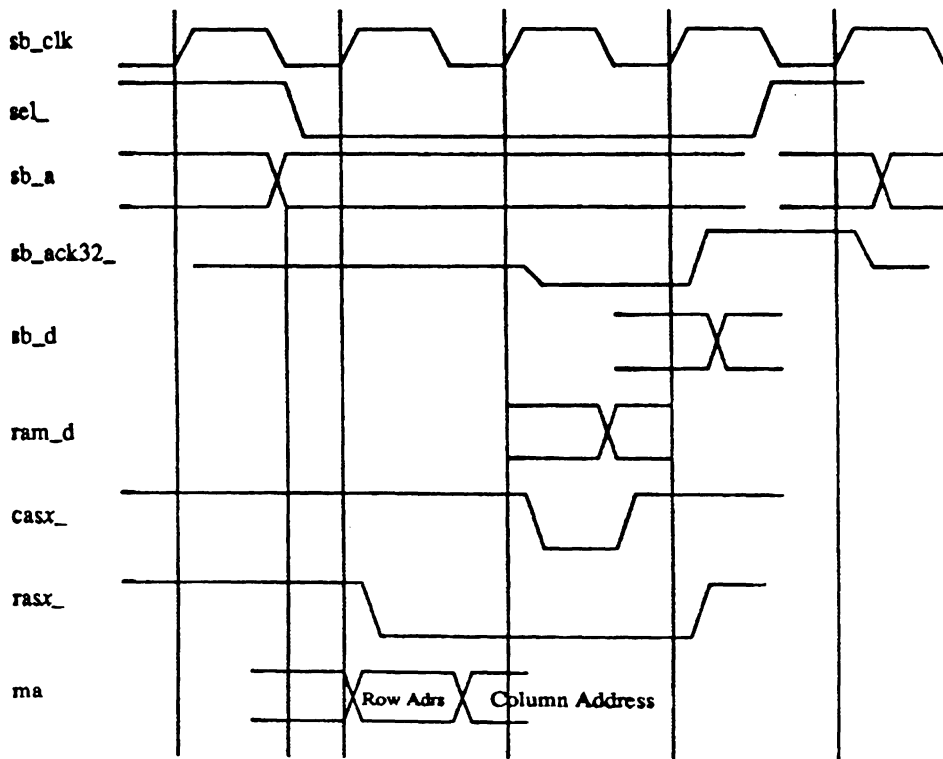
3.7.3.1 DRAM Access

All DRAM timing diagrams assume no contention for the DRAM. If the SBus request is lower priority than another requestor, waitstates will be added. Writes will only be delayed if the Write Buffer is full. DRAM and Video Register accesses are initiated by seeing both sb_as_ and ramsel_ asserted on the rising edge of an sb_clk. In the following diagrams, the logical AND of sb_as_ and ramsel_ is shown as sel_.

• Word Read

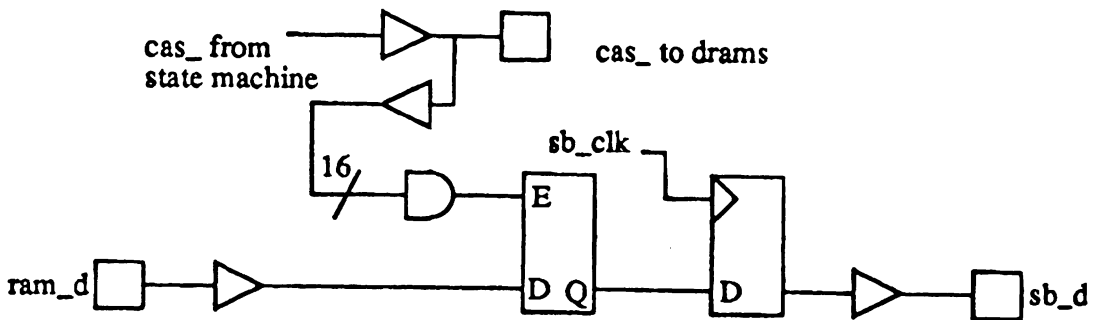
RAM+ reads are done in the "fast mode" of the RAM chip. sb_rd is high:

Figure: 3.11 Word Read



The following diagram shows the capture of data from the ram data bus to the sb data bus:

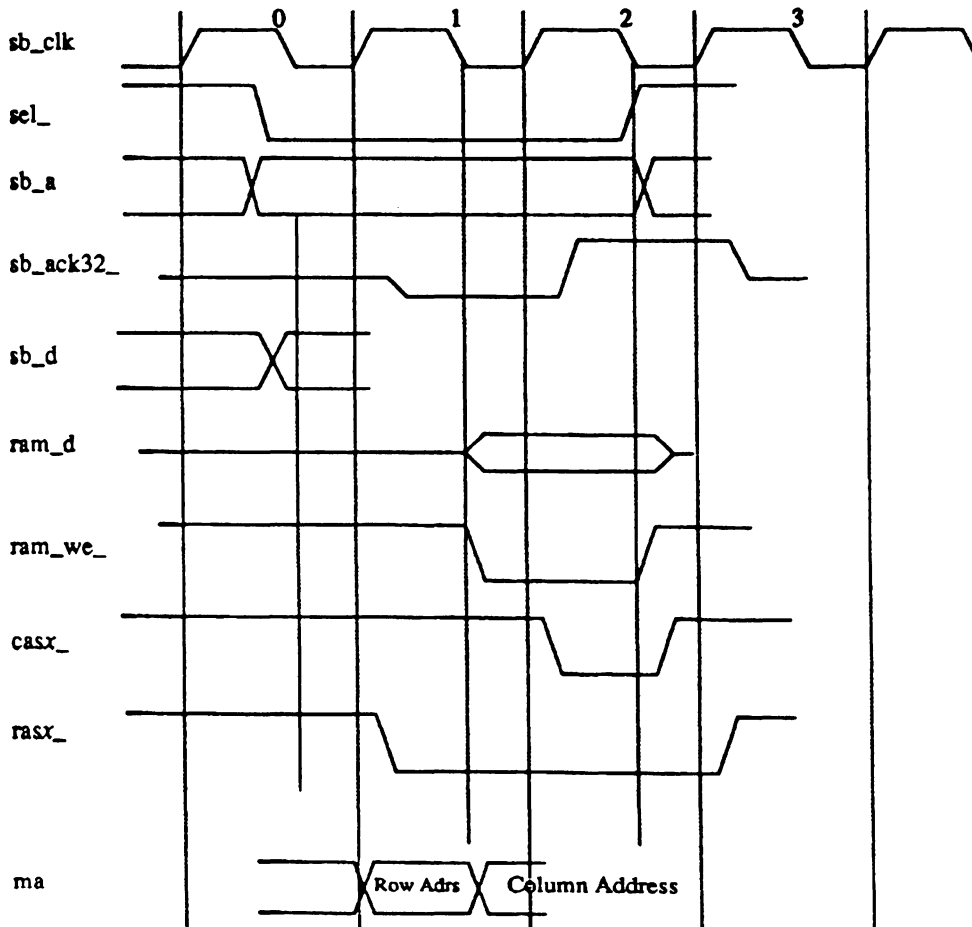
Figure: 3.12 Word Read 2



• Word Write - Buffered

Writes go directly into the write buffer and release the bus. **sb_rd** is low:

Figure: 3.13 Word Write

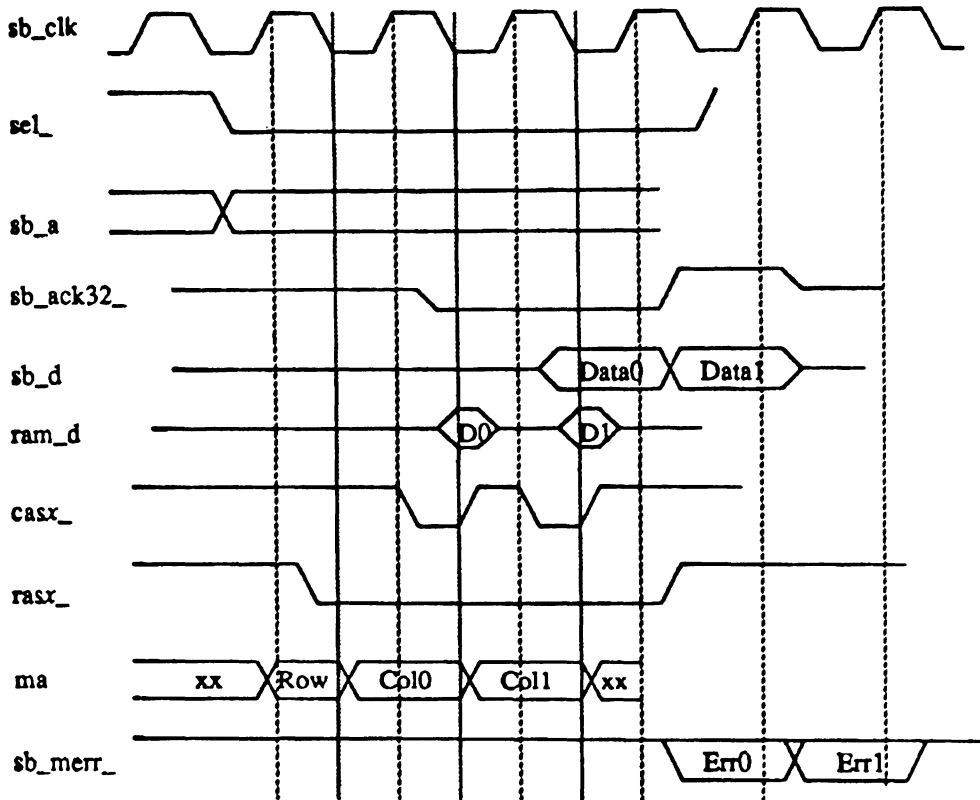


The data on the **sb_d** bus is sampled on the same clock as **sb_as_** and **ramsel_**. Data on the **ram_d** bus is driven on the falling edge of **sb_clk**. (It is possible that this cycle will need to be extended one clock such that the **ram_d** bus is driven on the falling edge of cycle 2, rather than cycle 1. This would not slow down the **ack32_** response, however. (This would be because of writebuffer fallthrough time, and also because the burst writes must wait until cycle 2 for their first **cas**, as well.)

• Burst Read

Only an 8-byte Burst Read is shown. For a 16-byte burst, add two more clocks. For a 32-byte burst, add 6 more clocks. `sb_rd` is high:

Figure: 3.14 Burst Read

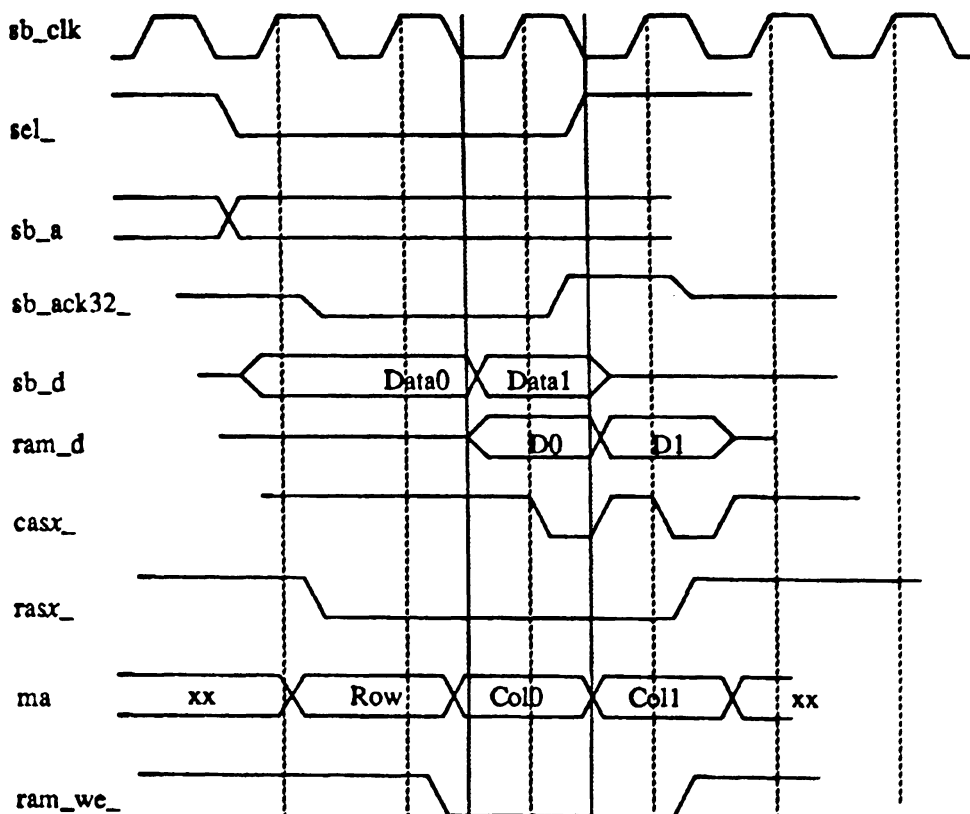


SBus addresses will not change during a burst access. The `ma` address lines must increment themselves, wrapping around $\text{mod}(8)$ for 8-byte transfers, $\text{mod}(16)$ for 16-bytes transfers, and $\text{mod}(32)$ for 32-byte transfers. (In reality, no 8-byte transfer will ever take place *except* on a longword boundary. 16 and 32-byte transfers may take place on any word boundary.)

• Burst Write

Only an 8-byte Burst Write is shown, and the write buffer is not full. `sb_rd` is low:

Figure: 3.15 Burst Write



3.7.3.2 Address Mapping

The row and column address bits for the RAM+ chip are compatible with the RAM chip in its 4Mbit mode. No 1Mbit mode is available.

sb_a	ma
a0	byte
a1	byte
a2	col0
a3	col1
a4	col2
a5	col3
a6	col4
a7	col5
a8	col6
a9	col7
a10	col8
a11	col9
a12	row0
a13	row1
a14	row2
a15	row3
a16	row4
a17	row5
a18	row6
a19	row7
a20	row8
a21	row9
a22	row10
a23	col10
a24	set0
a25	set1
a26	ram_/reg
a27	bank

Byte[1:0] specifies which byte within the word will be accessed.
Set[1:0] specifies which of the four ras/cas sets will be accessed.
Ram_/reg is low to access DRAM, high to access video registers.
Bank is the bank strapping pin.

3.7.4 DETAILED TIMING

3.7.4.1 DRAM Timing

The ram+ chip is designed to work with 80 nsec 1Mb and 4Mb fast page mode DRAMs at 20 MHz SBus clock, and 100 nsec DRAMs at 16.67 MHz. The crossover point between the two DRAMs is tbd.

DRAM Errors

When memory parity errors are reported (either as a result of processor or DVMA activity) a *physical* address is reported, along with the contents of the memory error register. The following table maps physical addresses and error register bits to SIMM locations.

Table: 3.12 DRAM maps physical addresses to SIMM Locations

Address Base	Error Register Bits-----		
	8 (D[7:0])	4 (D[15:8])	2 (D[23:16])1 (D[31:24])
0x0	U0322	U0307U0309	U0311
0x1000000	U0321	U0308U0310	U0312
0x2000000	U0320	U0315U0314	U0313
0x3000000	U0316	U0317U0318	U0319

DRAM

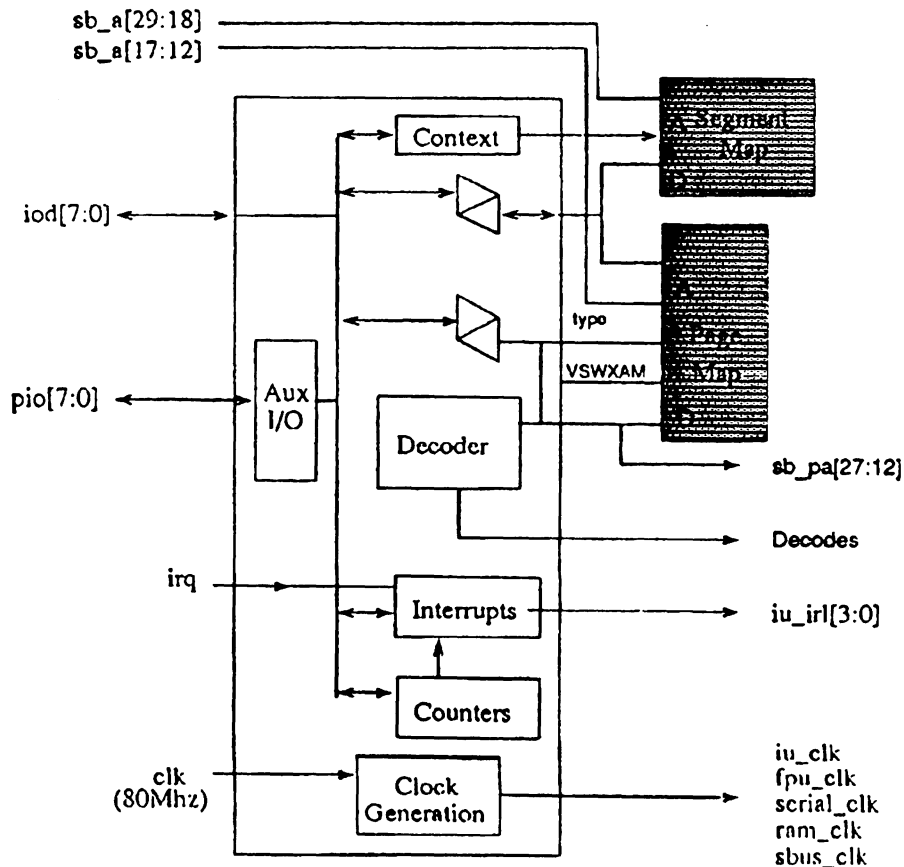
For further details on the DRAM subsystem of, see the *RAM+ Controller Specification*.

3.8 The MMU+

3.8.1 Features

- Provides data paths to Sun-4 SRAM MMU
- Updates MMU statistics word
- Provides decodes and timing strobes for Type-1 devices
- Prioritizes 15 levels of interrupts
- 4-bit context register allows 16 contexts
- Two counters generate high-resolution periodic interrupts
- Crystal oscillator for serial ports
- Assumes 20 nsec page and segment RAMS: one clock write cycles.
- Supports up to 256 PMEGS

Figure: 3.16 Segment Map & Page Map



3.8.2 Pin Description

The MMU+ is built in a 160-pin plastic flat pack. It has 40 *X* outputs, 48 *Y* bidirectional pins, 36 *Z* inputs, 17 *A* VCC pins, and 19 *B* grounds.

3.8.2.1 SBus Interface

sb_clk	(DRVT4) System clock. This signal runs at up to 20 MHz with a 50% duty cycle.
sb_a[3:0]	(TLCHT) Low-order sbus address lines used for internal byte and register selects.
sb_rd	(TLCHT) High during sbus read cycles, low during write cycles.
sb_as_	(TLCHT) SBus Address Strobe. Low during valid SBus cycles.
sb_reset_	(TLCHT) Low to reset SBus devices.
sb_ack8_	(BT4) SBus 8-bit acknowledge. Used to terminate an SBus cycle when the addressed device is 8 bits wide.
iod[7:0]	(BD4) 8-bit buffered extension of SBus for on-board data. Used to access on-chip registers and MMU RAMs.
pio[7:0]	(BD4TOD) Pseudo-bidirectional parrel port pins. to use as inputs, the controlling register should be programmed with 1's. To use as out puts, the pin should have a pullup resistor added externally.
ctl[2:0]	(TLCHT) Encoded control space selects from Cache+ chip. Private signal from cache chip.
devspc_	(TLCHT) Low for device space accesses, High for control space accesses. Private signal from cache chip.
user_	(TLCHT) Low for user space accesses, high for supervisor space accesses.
sb_bg[3:0]	(TLCHT) Any one of these signals is Low during DVMA accesses. Forces CID output to zero.
devspc_	(TLCHT) Low for device space accesses, High for control space accesses. Private signal from cache chip.
user_	(TLCHT) Low for User Space accesses, High for Supervisor Space accesses. Private signal from cache chip.
sb_bg[3:0]_	(TLCHT) Any one of these signals is Low during DVMA accesses. Forces CID output to zero.

3.8.2.2 MMU Interface

cid[3:0]	(BT2) Context ID to Segment Map RAMs.
pmeg[7:0]	(BD4TD) Page Map Entry Group. Data bits to/from Segment Map RAMs; used for programming Segment Map.
pa[27:12]	(BD4T) SBus Physical Address bits to/from Page Map RAMs. Used for programming Page Map and as inputs for decodes.
mmu[v,w,s,x]	(BD4T) MMU Valid, Write-allowed, Supervisor-only, don't-cache bits to/from Page Map RAMs. Used to program Page Map and as inputs for permission checks.

<code>mmu_typ[1:0]</code>	(BD4T) MMU Type bits to/from Page Map RAMs. Used to program Page Map and as inputs for decodes. The type bits are essentially extensions of the physical address output where Type 0 is on-board RAM and Type1 is IO and SBus. Types 2 and 3 are not used on this newer design, but are supported for VME-based machines.
<code>mmu[a,m]</code>	(BD4T) MMU Accessed and Modified bits to/from Page Map RAMs. Used to program Page Map and updated during all device-space accesses. Any access to a page marks the <code>a</code> bit to one, while any successful write to a page marks the <code>m</code> bit to one.
<code>sm_wr_</code>	(BT4) Segment Map Write Enable.
<code>pm_wr[2:0]_</code>	(BT4) Page Map Write Enables.

3.8.2.3 Decodes

<code>ramsel_[1.0]</code>	(BT2) DRAM select. <i>Asynchronous Decode</i>
<code>kbm_rd_</code>	(BT2) Read Strobe for 85C30. Asserted during reads and during reset. Synchronization hold-off is guaranteed.
<code>kbm_wr_</code>	(BT2) Write Strobe for 85C30. Asserted during writes and during reset. Synchronization hold-off is guaranteed.
<code>scc_rd_</code>	(BT2) Read Strobe for 85C30. Asserted during reads and during reset. Synchronization hold-off is guaranteed.
<code>scc_wr_</code>	(BT2) Write Strobe for 85C30. Asserted during writes and during reset. Synchronization hold-off is guaranteed.
<code>tod_cs_</code>	(BT2) Chip-select signal (active low) for MK48T08 TOD/NVRAM chip.
<code>par_cs_[1.0]</code>	(BT2) Chip-select signal (active low) for parity control/status register in RAM2 chip
<code>eprom_rd_</code>	(BT2) Read/Output Enable for EPROM accesses.
<code>fd_rd_</code>	(BT2) Active-low floppy controller read strobe.
<code>rd_wr_</code>	(BT2) Active-low floppy controller write strobe.
<code>iosel_</code>	(BT2) Select for Type 1 device
<code>sb_sel[3:0]_</code>	(BT2) SBus selects <i>Asynchronous decodes</i> .
<code>iod_en_</code>	(BT2) iod bus enable. Asserted low whenever an access takes place on the iod bus.

sb_merr_	(TLCHTN) sbus memory error/async error signal. Used to generate non-maskable interrupt when an async error occurs. Edge-sensitive. External Pullup Required.
irq[13:11,9:1]_	(SCHMITCN) Interrupt request inputs. External Pullup Required.
iu_irl[3:0]	(BT2) Encoded Interrupt Requests to Integer Unit. Positive-true; zero indicates no interrupts pending, 0xF indicates a level-15 (non-maskable) interrupt is pending.

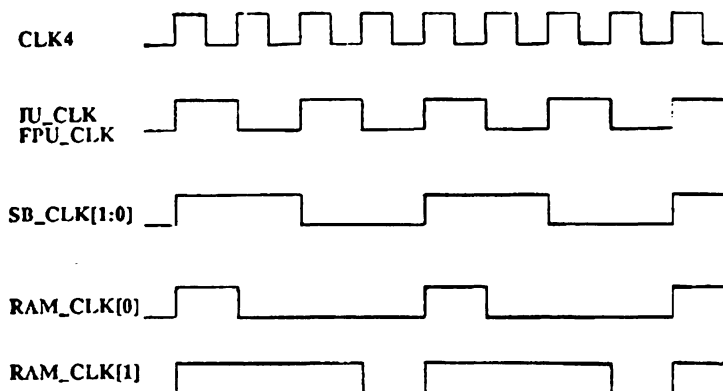
3.8.2.4 Miscellaneous

const_clk	(TLCHT) <i>x</i> -MHz clock used for counter-timers.
XSO/XSI	Oscillator input/output pins for 19.66 MHz xtal.
sclk4	(BT2) Serial port clock; XSI divided by four.
pio[7:0]	(BD4TOD) Pseudo-bidirectional parallel port pins. To use as inputs, the controlling register should be programmed with 1's. To use as outputs, the pin should have a pullup resistor added externally.
od_	(TLCHTU)
para 32 in, 32 out, 48 bidirect... 112 total.	(BT1) Parametric output for testing only.

3.8.3 Clock Generation

3.8.3.1 General Description

The MMU+ chip generates all system level clocks required for the IU, FPU, SBus and Ram+ controller. The input clock frequency is four times the desired Sbus frequency. The clock logic will generate a divide by two output for the IU and FPU. The clock is further divided (by four with respect to the input) to provide the Sbus and ram controller clocks. While the Sbus clocks are 50-50 duty cycle, the ram controller clocks are 25-75 and 75-25.

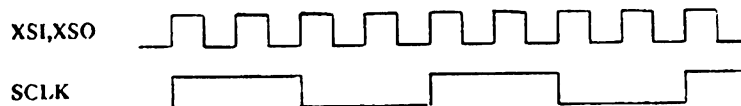


3.8.3.2 Constant Clock

A constant clock input is used to provide a 100nS clock to the counter-timers. Since the main clock and SBus clock frequencies are subject to change, this input is required.

3.8.3.3 Serial Clock

The MMU+ chip has a serial clock oscillator circuit used for generation of a 4.91 Mhz, 50-50 duty cycle clock. This clock is used by the 85C30 serial chips. The primary frequency of the crystal is 19.66Mhz.



3.8.4 Decodes

3.8.4.1 Control Space

- CTL mappings

The `devspc_` signal from the cache+ chip chooses between device space and control space accesses. Device space accesses are accessed with physical addresses provided by the MMU RAMs, while control space accesses are identified by virtual addresses (decoded from the IU by the cache+ chip onto the `ctl[n]` lines). When `devspc_` is high, the `ctl` lines have the following meaning:

Table: 3.13 CTL mappings

ctl	Device
0	Cache+ internal (no action)
1	Reserved for VME IACK
2	Context Register
3	Diagnostic Register (unused)
4	SCC Bypass
5	Segment Map
6	Page Map
7	EPROM Bypass

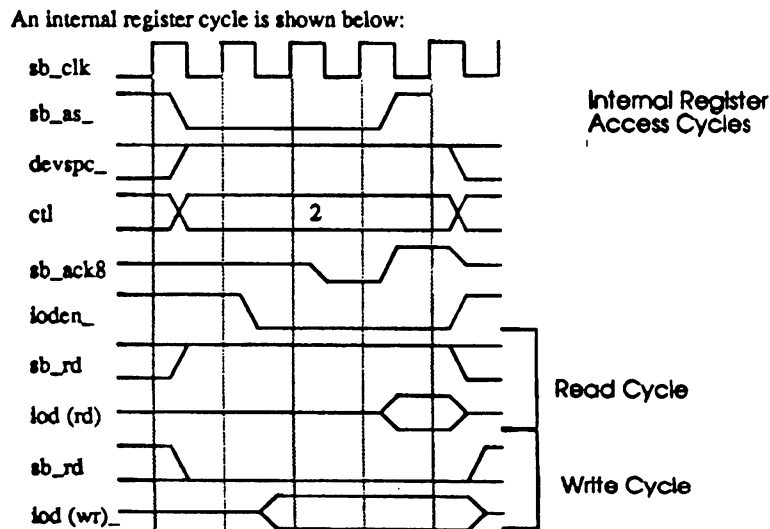
• Context Register Access

Writing to or reading from ASI 2, VAdr 0x30000000 causes the cache+ chip to decode for context register accesses. The MMU+ chip will reply with an `sb_ack8_`. Only byte writes or reads should be done. Only the least significant four bits of the byte are used as a context register. Writes to this address will go to both copies of the context register: in the MMU+ chip (and to the MMU SRAMs) and in the cache+ chip (to be used for cache tag comparisons). Reads from this location will come from the MMU+ chip, while reads from ASI 2, VAdr 0x30000001 will get data from the cache+ chip. This difference in read behavior is for diagnostic purposes only.

All internal register accesses use the `iod[7:0]` bus to transfer data. This bus is simply a buffered version of `sb_d[31:24]`. It is enabled with `ioden_low`.

An internal register cycle is shown below:

Figure: 3.17 Internal Register Cycle



• SCC Bypass

Writing to or reading from ASI 2, VAdr 0xF0000000 will bypass the MMU and provide access to the SCC. See SCC/Keyboard-Mouse accesses for further timing details. `ctl[2:0] = 0x4` for these accesses.

• EPROM Bypass

Supervisor Instruction accesses made while the system is in *boot* state will cause the cache+ chip to operate in control space with `ctl[2:0] = 0x7`. These cycles will be forced to EPROM accesses on reads, and no device will be selected on writes.

• Segment Map Access

Writing to or reading from ASI 3 will access the segment map RAM(s). The location accessed is specified by the combination of the current context register value and virtual address lines `sb_a[29:18]`.

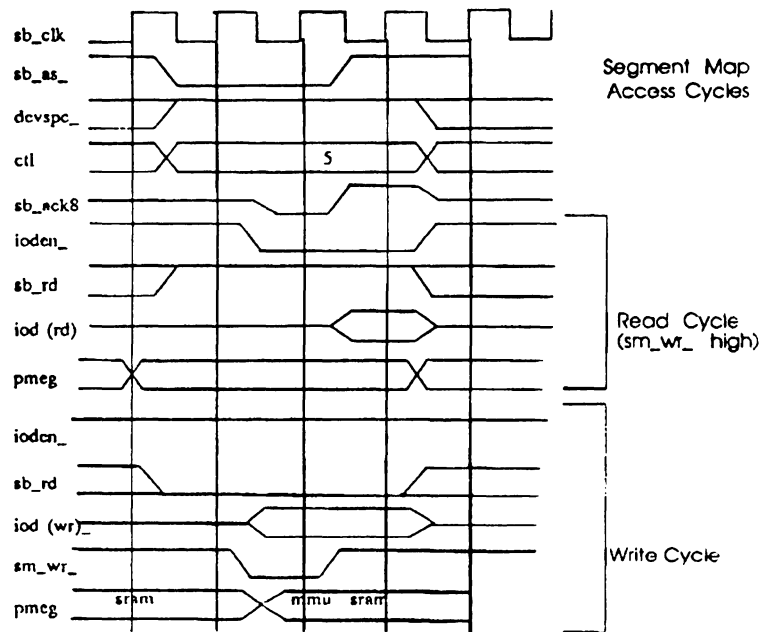
The cache+ chip enforces the restriction that the top three address lines, `a[31:29]` must either all be ones or all be zeros.

The number of contexts supported (8 or 16) is system-dependent. The mmu+ chip supports up to 16 contexts.

Segment Map write cycles cause the `iod[7:0]` value to be driven onto the `pmeg[7:0]` lines. The output of the Page Map is ignored for this (and all control space) cycle.

The `sb_a[29:18]` and `ctxt[3:0]` signals are valid by the clock cycle in which `sb_as_` is asserted:

Figure: 3.18 Clock Cycle



The `sm_wr_` signal is asserted for one clock. The first half-clock is used to allow the segment ram to get off the `pmeg` databus. During the second half-clock the mmu drives the value from the iodata bus onto the `pmeg` bus. The mmu continues to drive this data for the half-clock after `sm_wr_` is negated to allow for hold time. During this clock, both the segment map RAM and the mmu+ will be driving the same data onto the `pmeg` bus. The `sb_a[29:18]` lines must not change in the clock following `sb_as_` negation on segment map access cycles.

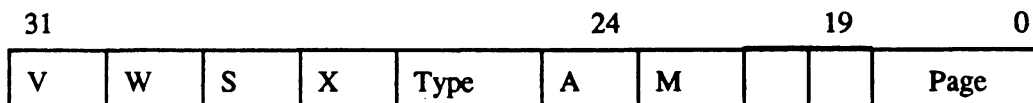
Critical Path: `ioden_` to `iod[7:0]` valid to data out on `pmeg[7:0]`. Must all happen 40 nsec after `sb_clk` rising edge.

Only byte accesses, with 1s address bits zero, should be made to the Segment Map.

• Page Map Access

Page map access is extended by 4 bits on the CPU-2CE for VME only. Writing to or reading from ASI 4 will access the Page Map RAMs. The location accessed is selected by the pmeq value (selected by the current context and sb_a[29:18]) concatenated with sb_a[17:12]. Timing is identical to the Segment Map cycles, except that for word accesses, four cycles will be done (because the mmu+ will respond with sb_ack8_). The page map value is defined as follows:

Page Type Entry



• Page Type Bits

Bits 31 through 16 of the page map entry provide the following information about the page: 27 and 26 Type bits, which select device space access as shown in the following table.

Table: 3.14 Page Type Bit Definitions

Bit 27	Bit 26	Type
0	0	Main memory (type 0)
0	1	I/O space (type 1)
1	0	VME D16 Port (type 2)
1	1	VME D32 Port (type 3)

3.8.4.2 Device Space

Table: 3.15 Device Space		
Type	Address	Device
0	00XXXXXX to 07XXXXXX	Ramsel [0] (on-board DRAM)
0	08XXXXXX to 0FXXXXXX	Ramsel [1] (off-board DRAM)
1	F0X00XXX	Keyboard/Mouse
1	F1X00XXX	SCC
1	F2X0XXXX	TOD, NVRAM
1	F3X00XXX	Counters
1	F4X00XX0	Parity Control/Status [0]
1	F4X00XXX	Parity Control/Status [1]
1	F5X00XXX	Interrupt Control
1	F6XXXXXX	EPROM
1	F7200XXX	Floppy
1	F7201XXX	Audio
1	F7400XXX	Aux I/O
1	F8XXXXXX	SBSel Zero
1	F9XXXXXX	
1	FAXXXXXX	SBSel One
1	FBXXXXXX	
1	FCXXXXXX	SBSel Two
1	FDXXXXXX	
1	FEXXXXXX	FLASH (Optional Feature)
1	FFXXXXXX	LED's and Hex Switch

SBus Select Zero is for on-board DMA on the CPU-2CE. Both Floppy and Audio use the `fd_rd_` and `fd_wr_` signals. The use of `sb_a[12]` to distinguish between the two devices is done at the board level, not in the mmu+ chip.

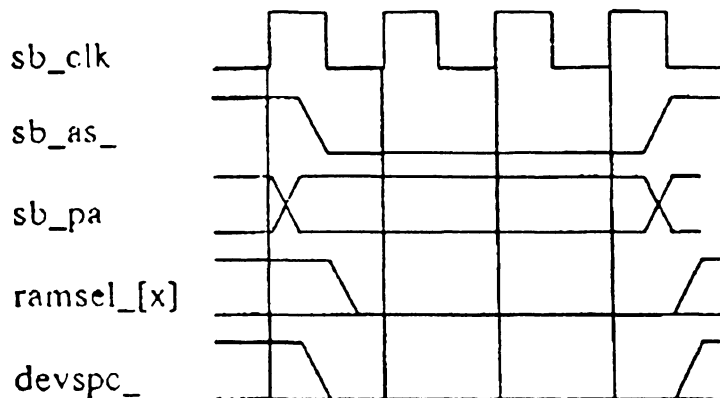
For Interrupt and Counter accesses, see the details in following chapters. `Devspc_` is assumed low in the following diagrams.

- SBus or DRAM Access

SBus and DRAM decodes are different from any other mmu+ decode. In these cases, the mmu+ does not provide any sbus acknowledge (the sbus device or ram controller is expected to do that) and the select signal is totally asynchronous. The sbus device or ram controller must look at both its select signal and `sb_as_` in order to decode itself.

The following diagram shows an SBus or DRAM access:

Figure: 3.19 SBus or DRAM Access



`Sb_ack{8,32}_` is not shown; it is generated by the slave device. The `sb_pa` lines also include the mmu permissions and statistics bits. The `ramsel_` output could as easily be an `sb_seln_` signal. See the note on statistics updates regarding how quickly the physical address may change to guarantee correct operation on statistics update cycles.

• TOD, EPROM, Floppy Access

The 48T08, EPROM, and Floppy Controller/Audio chips have identical timings.

The EPROM and 48T08 are assumed to have a 200 nsec access time.

Figure: 3.20 Scan EPROM Access

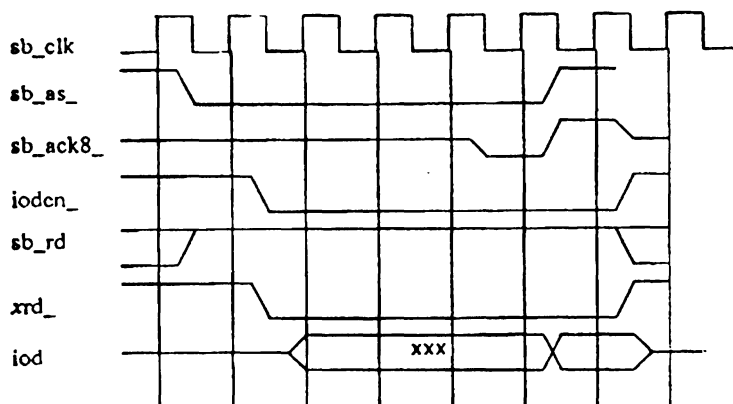
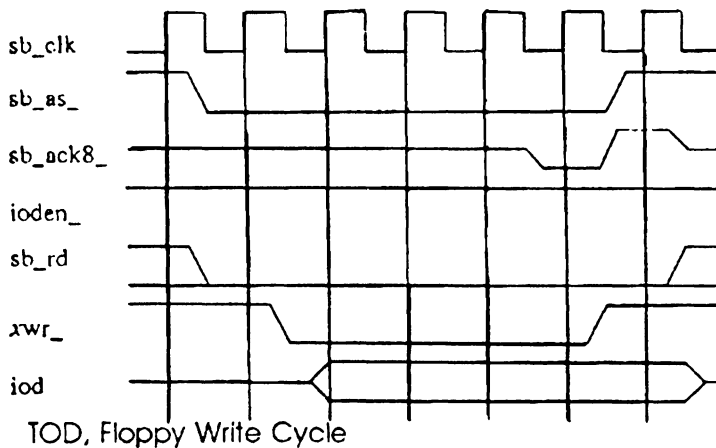


Figure: 3.21 2nd EPROM Access

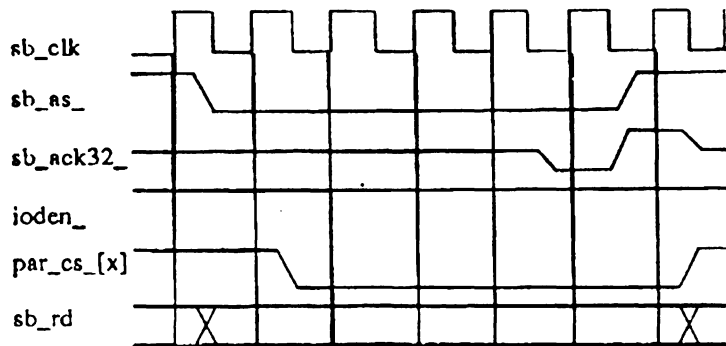


Note that the **wr_** strobe (or **cs_**, in the case of the TOD chip) is asserted *before* the data is valid. The **iod** bus makes setup to the trailing edge of the **wr_** strobe. In addition, the minimum **wr_** low time for the Am79C30 is 200 nsec. This is guaranteed by the fact that LSI's process goes low faster than high. Writes to the EPROM will return an **sb_ack8_** but will not enable any chip.

• Parity Register Access

Access of the parity control register does *not* cause `ioden_` to be asserted, since the register is in the ram+ gate array on the `sb_d` bus, rather than the `iod` bus.

Figure: 3.22 Parity Register Access

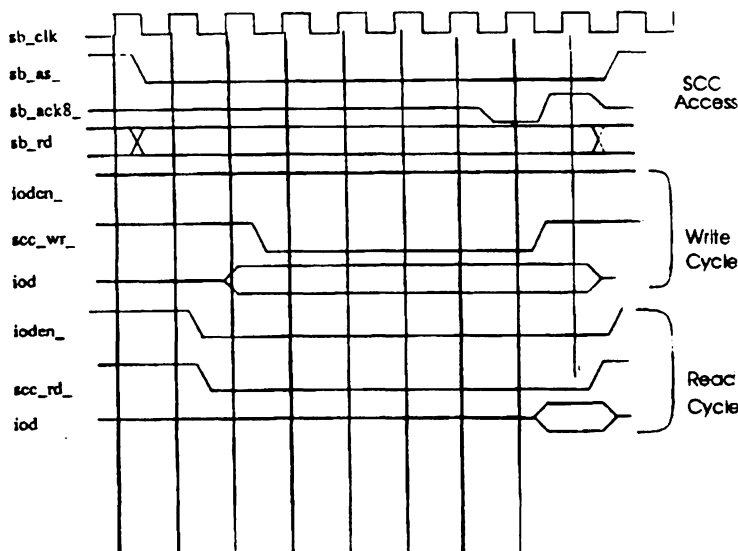


• SCC/KB Mouse Access

The 85C30 Serial Communications Controllers used for the serial ports and for the Keyboard/Mouse ports have longer access times than other 8-bit peripherals. In addition, the mmu+ chip guarantees 1 uSecond of hold-off time between accesses.

A 6 MHz 85C30 is assumed, with 180 nsec rd-datavalid delay, 200 nsec rd width, 300 nsec Adr-to-rd-datavalid, 200 nsec wr width, 10 nsec write-data-valid-to-write-asserted delay.

Figure: 3.23 SCC/KB Mouse Access



3.8.5 Protection Checking

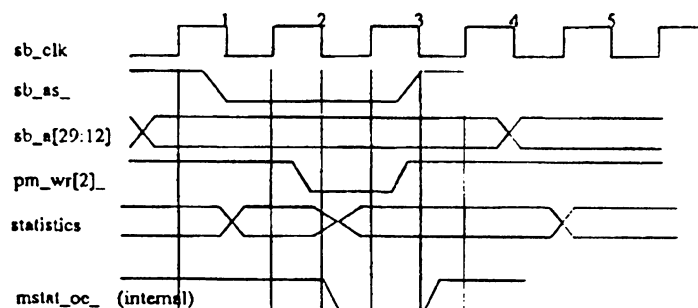
An access violation may occur on any device space cycle. When `sb_as_` is recognized along with any access violation, no statistics update cycle will take place, and no device will be selected. Error reporting is done by the cache+ chip. The following rules must be satisfied to allow a cycle to take place:

- if `user_` is low, `mmu_s` must be low
- if `sb_rd` is low or `ctl[1]` is high, `mmu_w` must be high. `ctl[0]` must be low.
- `Ctl[0]` is used by the cache+ chip to report an address violation in which `iu_a[31:29]` were not either all ones or all zeroes. `Ctl[1]` is used to report a load-store cycle, which must not take place if the page is not writable.

3.8.6 Statistics Updates

Every successful device-space access will cause a statistics update cycle to take place to the current page map entry. A read-modify-write cycle is done on the statistics byte (`pm_wr[2]_`). The `mmu_a` bit is set to one, and the `mmu_m` bit is OR-ed with one if this is a read cycle. Because of this cycle, the cache+ chip must guarantee that the high-order virtual address lines (`sb_a[29:12]`) does not change for 2 clocks after the assertion of `sb_as_`.

Figure: 3.24 Statistics Updates



As with the normal page map and segment map writes, the page map write enable signal is asserted in clock 2, and the mmu+ drives the statistics bits in the second half of clock 2 and the first half of clock 3. The write-enable signal is negated in clock 3. In order to provide address hold time, the input addresses may not change until clock 4.

3.8.7 Interrupts

The Interrupt Register provides for software generation of interrupts and allows the CPU to disable all interrupts or only certain ones. It is cleared on `sb_reset_`, and has the following fields:

31							24
Enabled 14	Reserved	Enabled 10	Enabled 8	SWI 6	SWI 4	SWI 1	Enable Any

Address Type 1, Physical Address 0xF500 0000

- All `irq[13:1]_` inputs may be asynchronous to the system clock.
- *Software* interrupts may be generated on levels 6, 4, and 1 by writing a 1 into bits 27, 26, or 25 when interrupts are enabled (bit 24 high).
- Level 15 interrupts (maskable only by the **Enable Any** bit) are captured from the `sb_merr_` input. `sb_merr_` is ignored on *cpu read* cycles, when the Cache+ chip would be generating a synchronous error. These cycles are identified by all `sb_bgn_` negated and a one-clock-delayed version of `sb_rd` asserted.
- Writing a zero to the **Enable Interrupts** bit (24) clears any pending level 15 interrupt.
- Writing a zero to any of the **Enable** bits in the Interrupt Register only masks out that levels interrupt. It does not clear the source, with the exception of level 15 requests. This is different from the Sun-4 architecture, in that the periodic interrupts at levels 10 and 14 must be cleared by accessing their respective Limit Registers (see "Counters" below).

See the *SUN Calvin System Specification* for a list of mappings between interrupt levels and interrupting devices.

3.8.8 Counters

The MMU+ has two microsecond-resolution counter/timers which provide periodic interrupts at levels 10 and 14. There are two counter registers and two limit registers. Interrupts occur when they are enabled in the Interrupt Register and the counter reaches the value preset in its corresponding limit register.

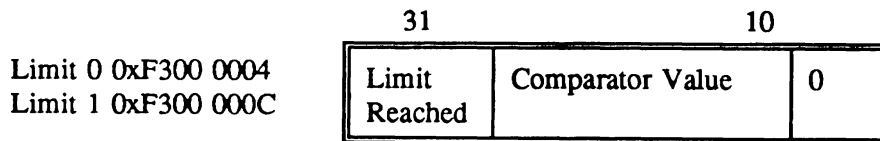
- Each counter is set to 1.0 uSec on `sb_reset_` and free-runs.
- The least significant 10 bits of each counter always read as zeroes to allow for future expansion to sub-microsecond resolution.
- When a counter reaches the value set in its limit register, it is set back to 1.0 on the following increment, and the **Limit Reached** bit is set.
- Counter 0 interrupts at Level 10 when enabled.
- Counter 1 interrupts at Level 14 when enabled.

- Reading a Limit register clears its **Limit Reached** bit, clearing the pending interrupt. *Clearing the level 10 or 14 bit in the Interrupt register does not clear a pending interrupt; it only masks it.*
- Each Limit register is set to all 0's on `sb_reset_`, allowing the counter to freerun.
- Writing to a Limit Register will set its corresponding Counter back to 1.0 uSec.
- Both counters may be read at any time, and are guaranteed by hardware not to increment during a word read operation.
- Both counters are separately writable for testing purposes. They should not be written in normal operation as results would be unpredictable.

Figure: 3.25 Counter



Figure: 3.26 2nd Counter



3.8.9 I/O Register

An 8-bit parallel port exists at Type 1 Physical Address 0xF740 0000. It is a pseudo-bidirectional port which can read the value at its pins and has open-drain outputs. To use an individual bit as an input, it should be always written with a "1" to avoid contention on that pin.

- All bits are set to 1's on `sb_reset_`.
- See the *SUN Calvin System Specification* for the functional mappings of the bits.
- Bits which may be used as outputs must have an external pullup resistor.

3.9 P2 Bus Interface Overview

Illustrations of the P1/P2 connectors of the SPARC CPU-2CE card can be found in Section 2. The P1 and P2 connectors are identical.

3.9.1 P1/P2 Connector Numbering

The CPU-2CE follows standard VME pin numbering of 3 rows of 32 pins on each connector. See section 2 for the pin signals.

3.10 Serial Interface A and B

3.10.1 Serial Interface A and B Device Address

The address of Serial Interface A is 0xE2000000.

On-Board Input/Output (OBIO)

3.10.1.1 RS232/RS423 Jumper Selection

The serial ports are RS-232-C or RS-423 with the default being RS-232-C. If RS-423 is needed, set the block before power up. Move the jumpers position to set RS-423. See the next page for the CPU-2CE defaults. The two ports are linked so both must be either RS-232-C or RS-423. See section 2 for pin signals.

3.10.2 Serial Interface A/B Definition

The A serial port interface supports asynchronous RS-423 with full modem control lines. For most applications, the interface will be directly compatible with RS-232 equipment as well. In addition to the RS-423 interface logic, four of the signals are brought out to separate connector pins via RS-423 compatible drivers and receivers. Transmit Data (TxD), Receive Data (RxD), Request to Send (RTS) and Clear to Send (CTS) signals are provided for use in electrically noisy environments or where longer cable lengths are required.

3.10.3 Serial Interface A/B Performance

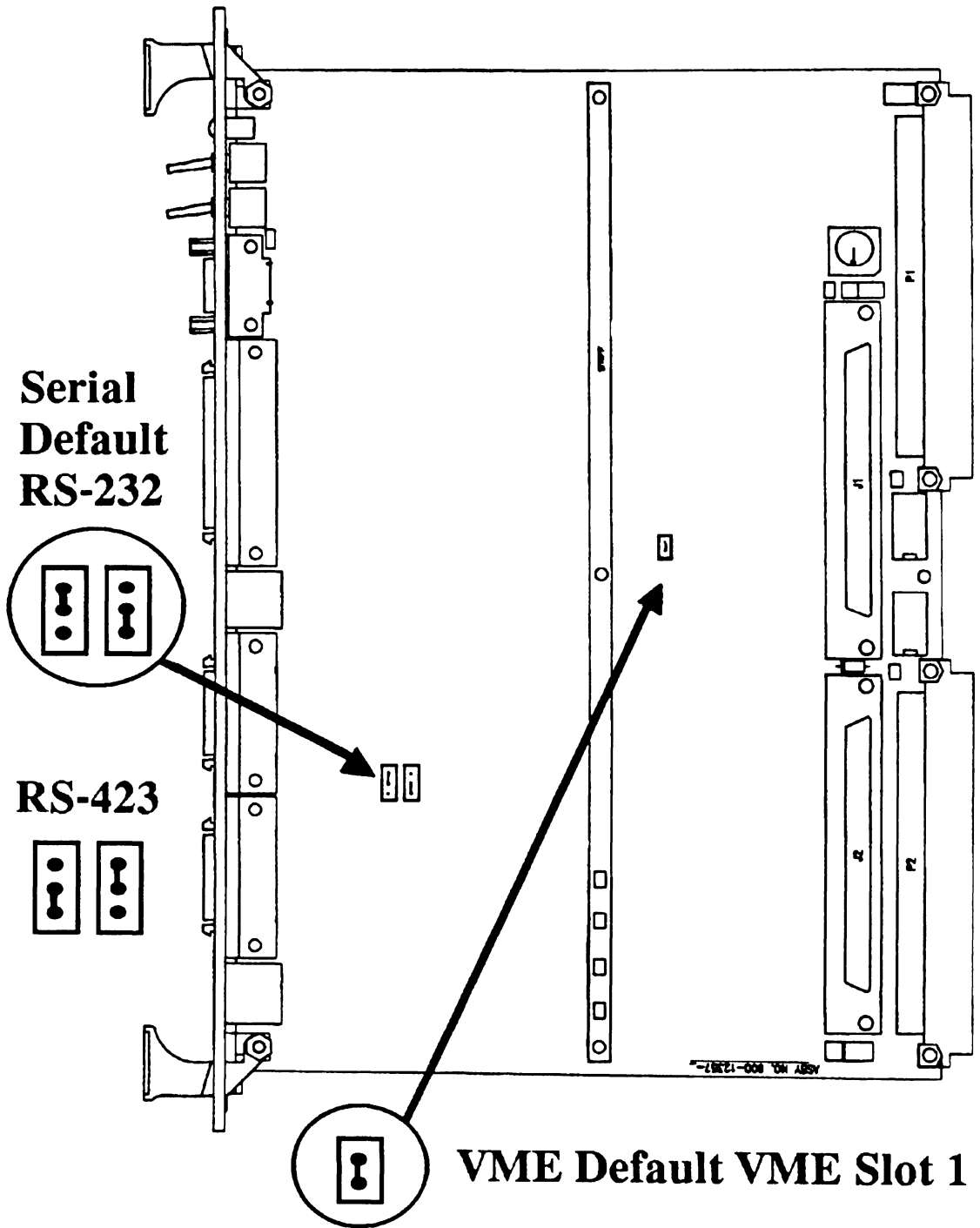
Asynchronous Speed: 19.2 Kbaud

The maximum baud rate supported by the Boot PROM on the SPARC CPU-2CE is 19.2K bits/second and assumes a 1/16 bit clock divisor. Faster bit rates can be programmed for custom applications. Refer to the Z8530 SCC Technical Manuals for more details. For reference, the clock input to the SCC for baud rate generation is 4.9152 MHz.

The A/B serial port interface does support Synchronous Serial Communications. RS-232 limits transmission rates in asynchronous mode to 20K bits/second, and RS-423 limits data rates to 100K bits/second.

Figure: 3.27 RS-232/423 Jumper Blocks

BOARD JUMPERS



3.11 Keyboard/Mouse Interface

3.11.1 Keyboard/Mouse Device Address

The keyboard/mouse address is 0xF0000000.

On-Board Input/Output (OBIO)

3.11.2 Keyboard/Mouse Interface Definition

The keyboard/mouse serial interfaces are implemented with a Z8530 Synchronous Serial Communications Controller. The SCC features two programmable serial channels with built in baud-rate generators. The clock input to the SCC for baud-rate generation is 4.9152 MHz and is independent of the IU clock.

Reset of the keyboard/mouse SCC is forced at power-up by asserting both read and write strobes simultaneously.

Mouse and keyboard data are transmitted and received over connector J1101 (DB-15). All data signals are TTL-compatible and cannot be direct-connected to RS-232, RS-423, or other non-TTL compatible equipment.

3.11.3 Specifications

The keyboard/mouse interfaces are designed to connect to Sun type-3 and type-4 keyboards. The Boot PROM will interrogate the interface upon power-up to determine if a keyboard is present. If not, the PROM expects a terminal to be connected to the Serial A port.

NOTE: The keyboard/mouse interface is intended for use with Sun keyboards only; custom serial expansion via this interface requires Boot PROM modifications.

Section 4

APPENDICES TO THE HARDWARE USER'S MANUAL

The CPU Card Schematic Diagrams

This appendix provides schematic diagrams of the key interfaces in the SPARC CPU-2CE for better understanding.

All of the included schematics and drawings are listed below:

DWG Schematics

Serial Cable Drawing

Ethernet Cable Drawing

Reader Comment Card

Product Error Report

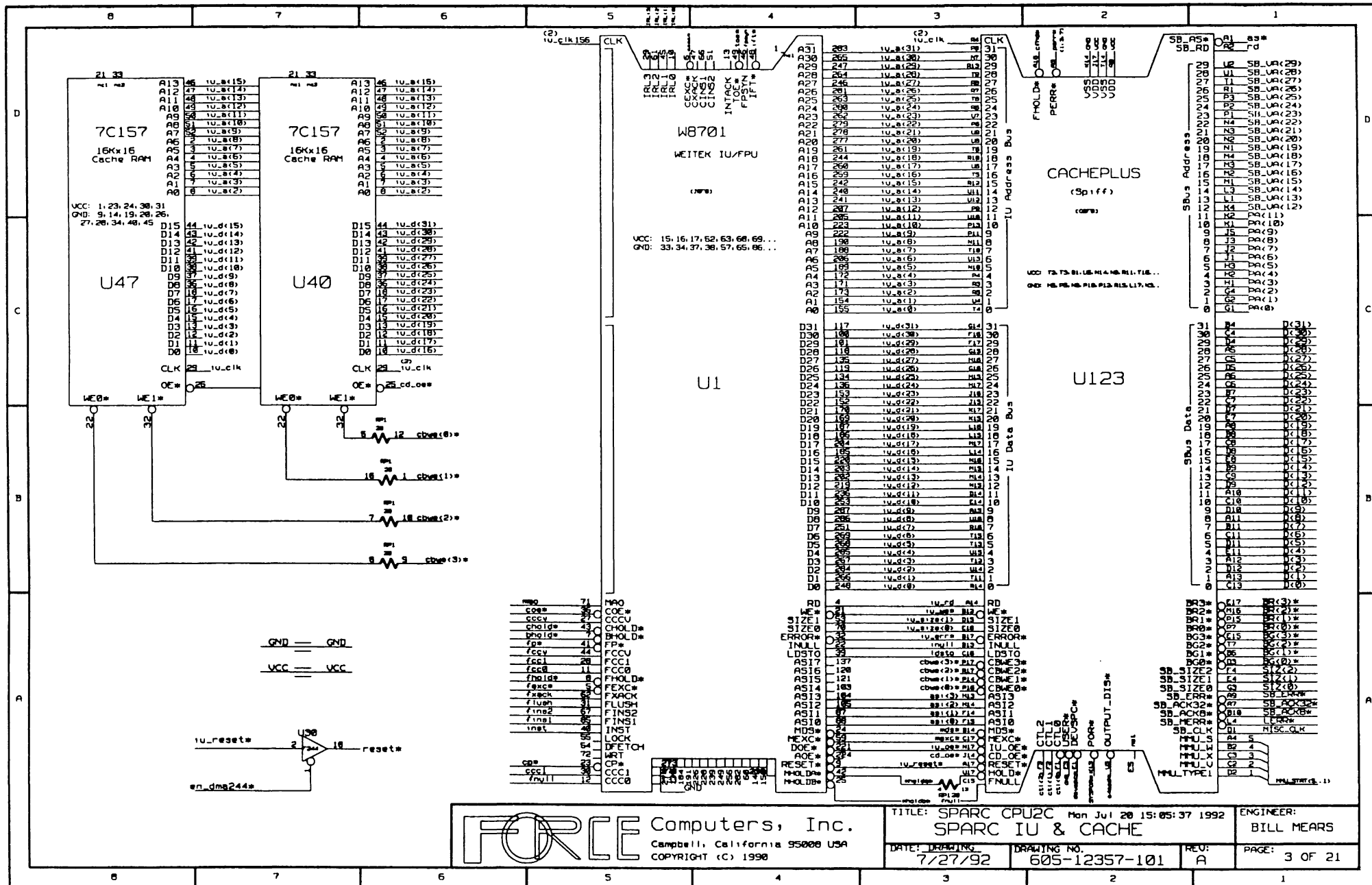
COVER SHEET 2

LEDGEND

- 3 IU & CACHE
- 4 MMU & MMUPLUS
- 5 MMU EXTENSIONS EPROM & TOD
- 6 RAMPLUS & EXPANSION CONNS
- 7 16 MBYTE RAM ARRAY
- 8 SLOT0: SCSI & ENET
- 9 SCSI II CON & VME P1 & P2
- 10 VME INF
- 11 VME INF
- 12 VME INF
- 13 VME INF
- 14 FLASH EEPROM
- 15 SERIAL INF
- 16 FLOPPY / AUDIO
- 17 RESET / DIAGNOSTIC LEDS
- 18 SBUS CONNECTORS
- 19 PULLUPS & TP'S
- 20 DECOUPLERS
- 21 VME RECEIVE BUFFERS & EXTRA STUFF ADDED AT THE LAST MINUTE

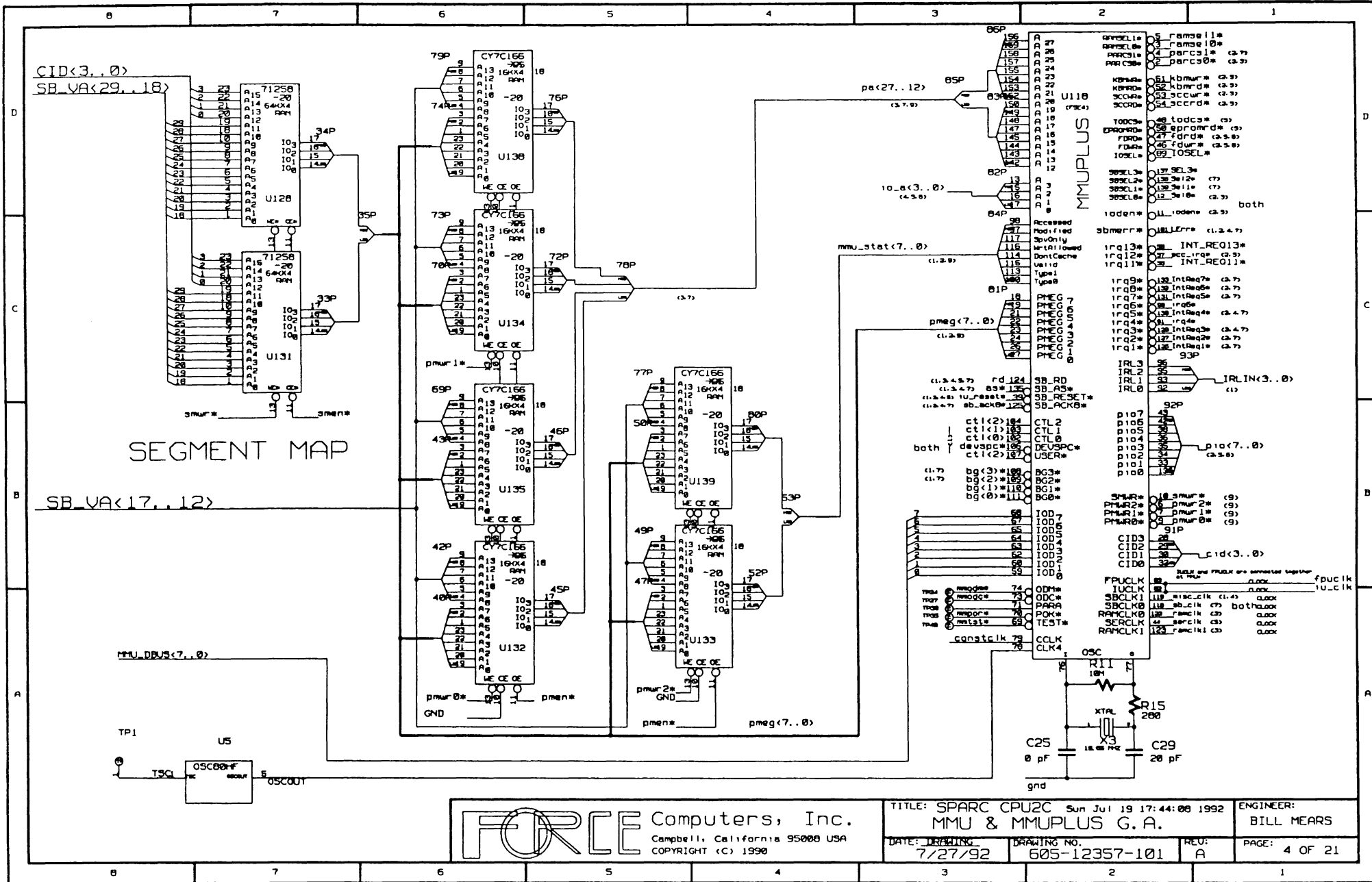
FORCE Computers, Inc.
Campbell, California 95008 USA
COPYRIGHT (C) 1990

TITLE: SPARC CPU2C Tue Jun 2 16:02:35 1992		ENGINEER: BILL MEARS	
DATE: <u>DRAWING</u> 7/27/92	DRAWING NO. 605-12357-101	REV: A	PAGE: 2 OF 21



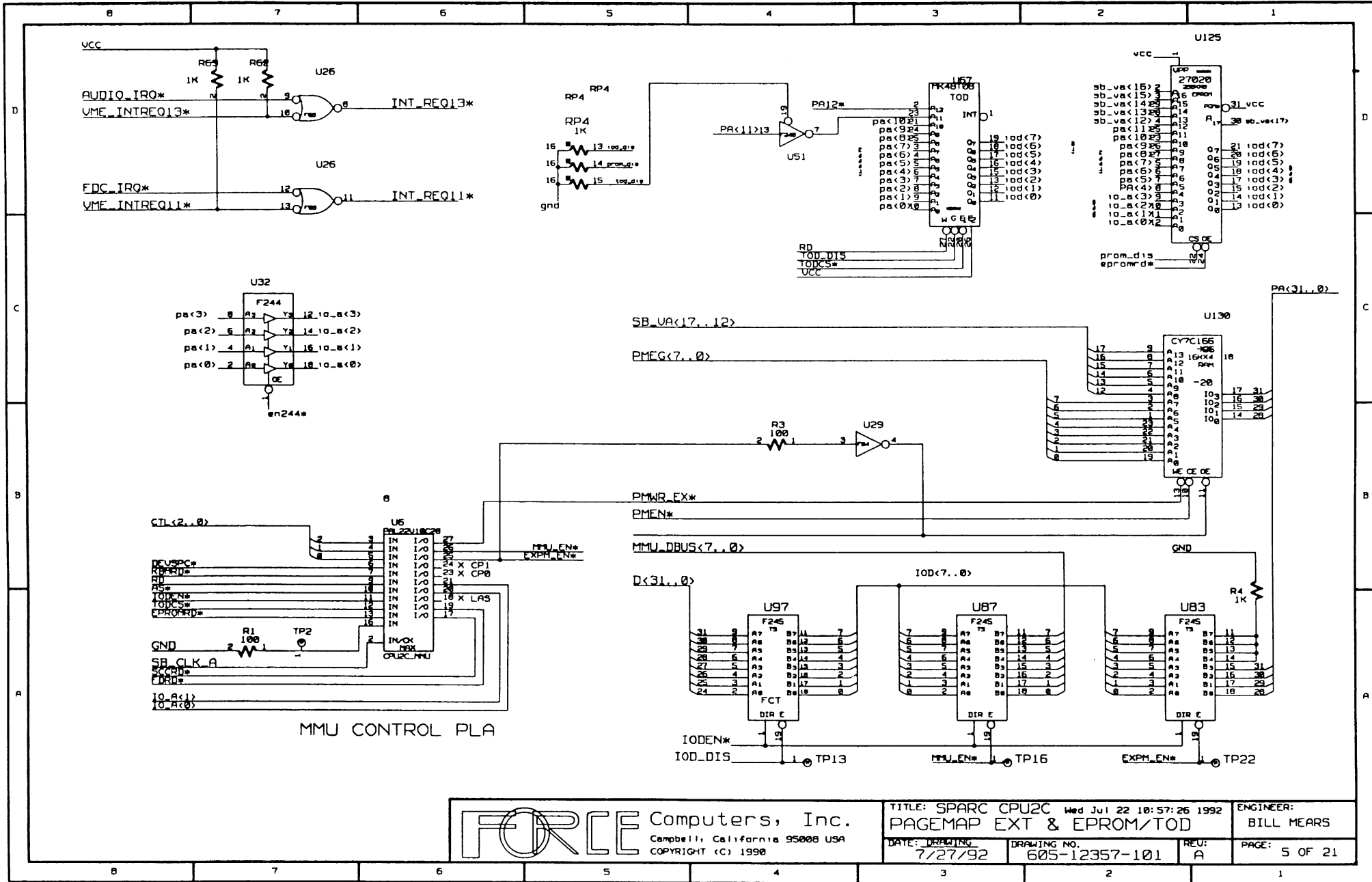
FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C Mon Jul 20 15:05:37 1992
 SPARC IU & CACHE
 DATE: DRAWING 7/27/92 DRAWING NO. 605-12357-101 REV: A
 ENGINEER: BILL MEARS
 PAGE: 3 OF 21



FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU20 Sun Jul 19 17:44:08 1992		ENGINEER: BILL MEARS	
MMU & MMUPLUS G.A.			
DATE: <u>REVISION</u> 7/27/92	DRAWING NO. 605-12357-101	REU: A	PAGE: 4 OF 21



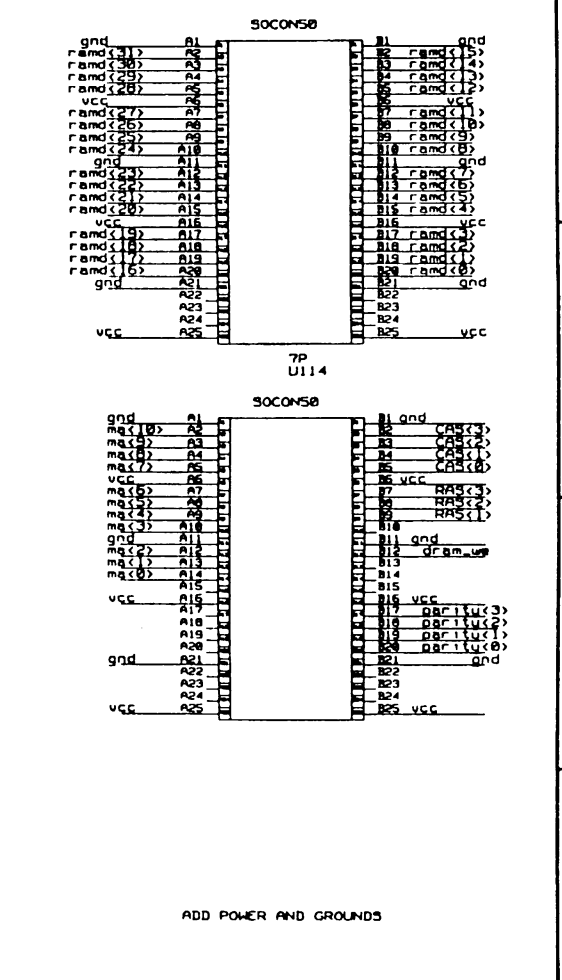
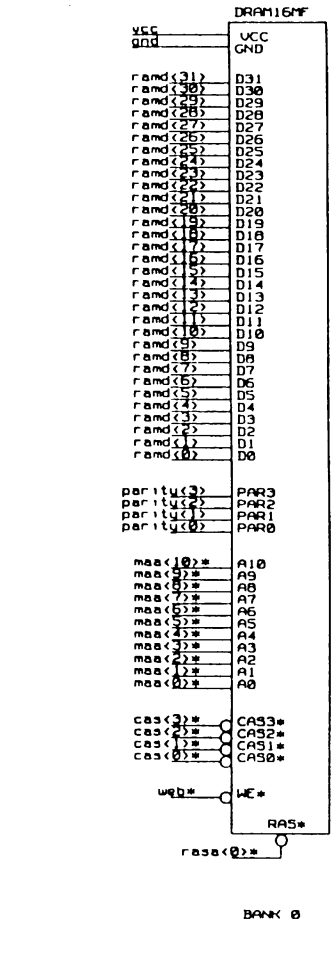
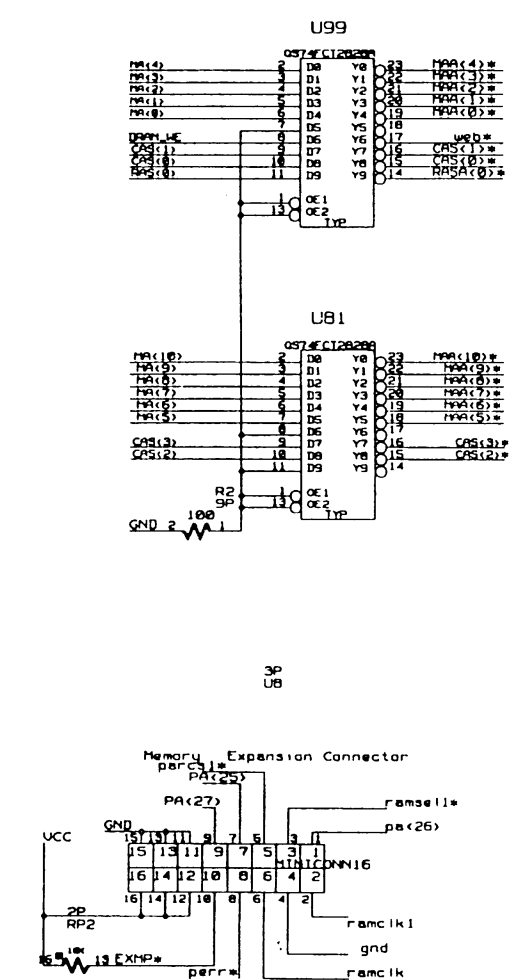
DRAM BLOCK
DETAILS ON PAGE 7

8P
U115

RAMPLUS

ramclk	18	RAM CLK	U112
ramclk1	50	RAM CLK1	U112
pa<25>	24	3ba26	
pa<25>	25	3ba25	
pa<24>	26	3ba24	
pa<23>	27	3ba23	
pa<22>	28	3ba22	
pa<21>	29	3ba21	
pa<20>	30	3ba20	
pa<19>	31	3ba19	
pa<18>	32	3ba18	
pa<17>	33	3ba17	
pa<16>	34	3ba16	
pa<15>	35	3ba15	
pa<14>	36	3ba14	
pa<13>	37	3ba13	
pa<12>	38	3ba12	
pa<11>	39	3ba11	
pa<10>	40	3ba10	
pa<9>	41	3ba09	
pa<8>	42	3ba08	
pa<7>	43	3ba07	
pa<6>	44	3ba06	
pa<5>	45	3ba05	
pa<4>	46	3ba04	
pa<3>	47	3ba03	
pa<2>	48	3ba02	
pa<1>	49	3ba01	
pa<0>	50	3ba00	
d<31>	41	3bd31	
d<30>	42	3bd30	
d<29>	43	3bd29	
d<28>	44	3bd28	
d<27>	45	3bd27	
d<26>	46	3bd26	
d<25>	47	3bd25	
d<24>	48	3bd24	
d<23>	49	3bd23	
d<22>	50	3bd22	
d<21>	51	3bd21	
d<20>	52	3bd20	
d<19>	53	3bd19	
d<18>	54	3bd18	
d<17>	55	3bd17	
d<16>	56	3bd16	
d<15>	57	3bd15	
d<14>	58	3bd14	
d<13>	59	3bd13	
d<12>	60	3bd12	
d<11>	61	3bd11	
d<10>	62	3bd10	
d<9>	63	3bd09	
d<8>	64	3bd08	
d<7>	65	3bd07	
d<6>	66	3bd06	
d<5>	67	3bd05	
d<4>	68	3bd04	
d<3>	69	3bd03	
d<2>	70	3bd02	
d<1>	71	3bd01	
d<0>	72	3bd00	
3b_3a3	37	3b_3a3	
3b_3a2	38	3b_3a2	
3b_3a1	39	3b_3a1	
3b_3a0	40	3b_3a0	
ramclk1	50	ramclk1	
parc32	21	3b_ack32	
parc31	22	3b_ack31	
parc30	23	3b_ack30	
parc29	24	3b_ack29	
parc28	25	3b_ack28	
parc27	26	3b_ack27	
parc26	27	3b_ack26	
parc25	28	3b_ack25	
parc24	29	3b_ack24	
parc23	30	3b_ack23	
parc22	31	3b_ack22	
parc21	32	3b_ack21	
parc20	33	3b_ack20	
parc19	34	3b_ack19	
parc18	35	3b_ack18	
parc17	36	3b_ack17	
parc16	37	3b_ack16	
parc15	38	3b_ack15	
parc14	39	3b_ack14	
parc13	40	3b_ack13	
parc12	41	3b_ack12	
parc11	42	3b_ack11	
parc10	43	3b_ack10	
parc9	44	3b_ack9	
parc8	45	3b_ack8	
parc7	46	3b_ack7	
parc6	47	3b_ack6	
parc5	48	3b_ack5	
parc4	49	3b_ack4	
parc3	50	3b_ack3	
parc2	51	3b_ack2	
parc1	52	3b_ack1	
parc0	53	3b_ack0	

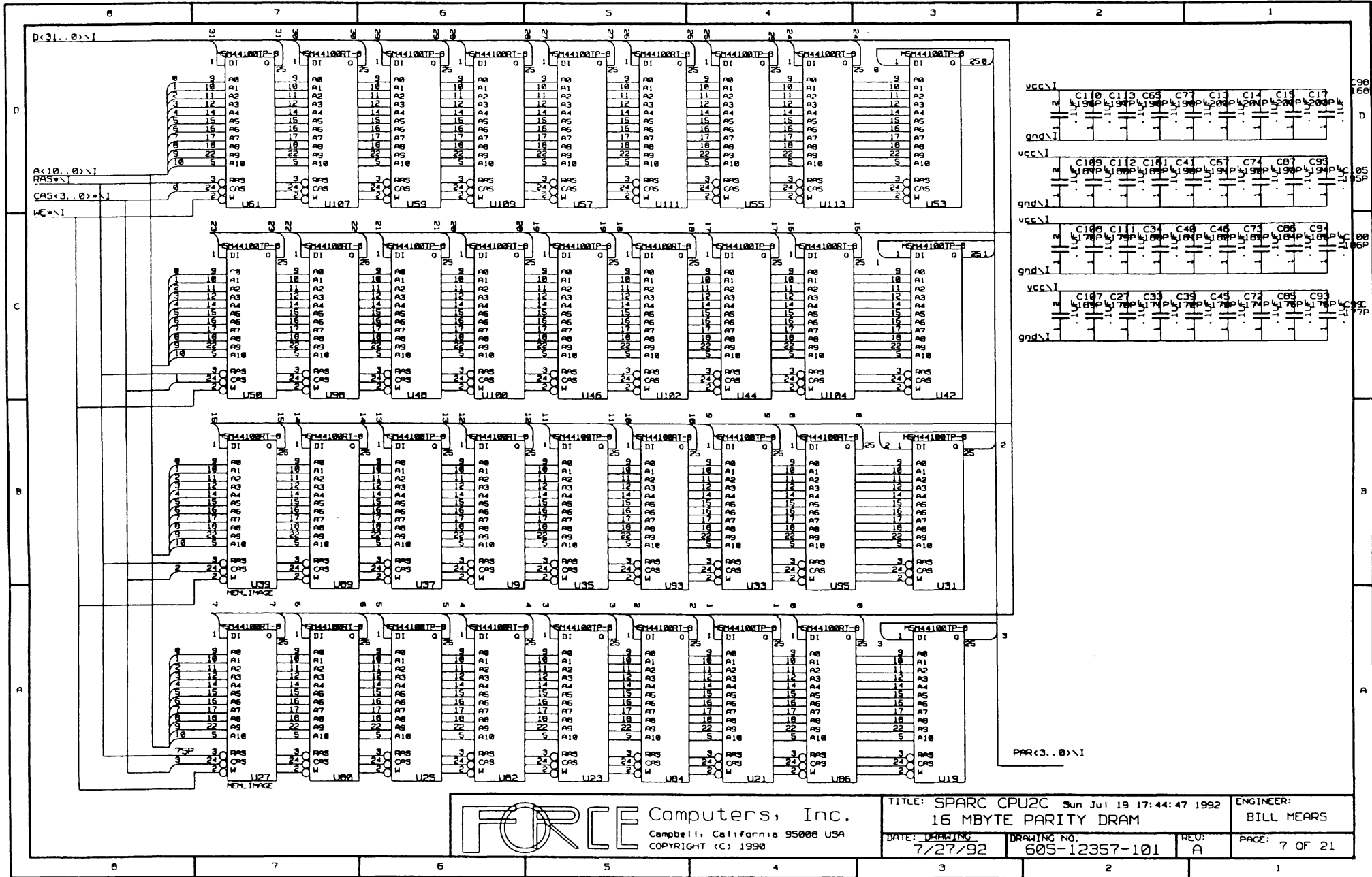
ma10	128	ma<10>
ma09	129	ma<9>
ma08	130	ma<8>
ma07	131	ma<7>
ma06	132	ma<6>
ma05	133	ma<5>
ma04	134	ma<4>
ma03	135	ma<3>
ma02	136	ma<2>
ma01	137	ma<1>
ma00	138	ma<0>
ca33	127	ca<3>
ca32	128	ca<2>
ca31	129	ca<1>
ca30	130	ca<0>
ra33	139	ra<3>
ra32	140	ra<2>
ra31	141	ra<1>
ra30	142	ra<0>
we	133	dram_we
rd31	180	rd<31>
rd30	181	rd<30>
rd29	182	rd<29>
rd28	183	rd<28>
rd27	184	rd<27>
rd26	185	rd<26>
rd25	186	rd<25>
rd24	187	rd<24>
rd23	188	rd<23>
rd22	189	rd<22>
rd21	190	rd<21>
rd20	191	rd<20>
rd19	192	rd<19>
rd18	193	rd<18>
rd17	194	rd<17>
rd16	195	rd<16>
rd15	196	rd<15>
rd14	197	rd<14>
rd13	198	rd<13>
rd12	199	rd<12>
rd11	200	rd<11>
rd10	201	rd<10>
rd09	202	rd<9>
rd08	203	rd<8>
rd07	204	rd<7>
rd06	205	rd<6>
rd05	206	rd<5>
rd04	207	rd<4>
rd03	208	rd<3>
rd02	209	rd<2>
rd01	210	rd<1>
rd00	211	rd<0>



VCC & GND on connector to respective planes
INTENDED FOR SRX CARD

FORCE Computers, Inc.
Campbell, California 95008 USA
COPYRIGHT (C) 1990

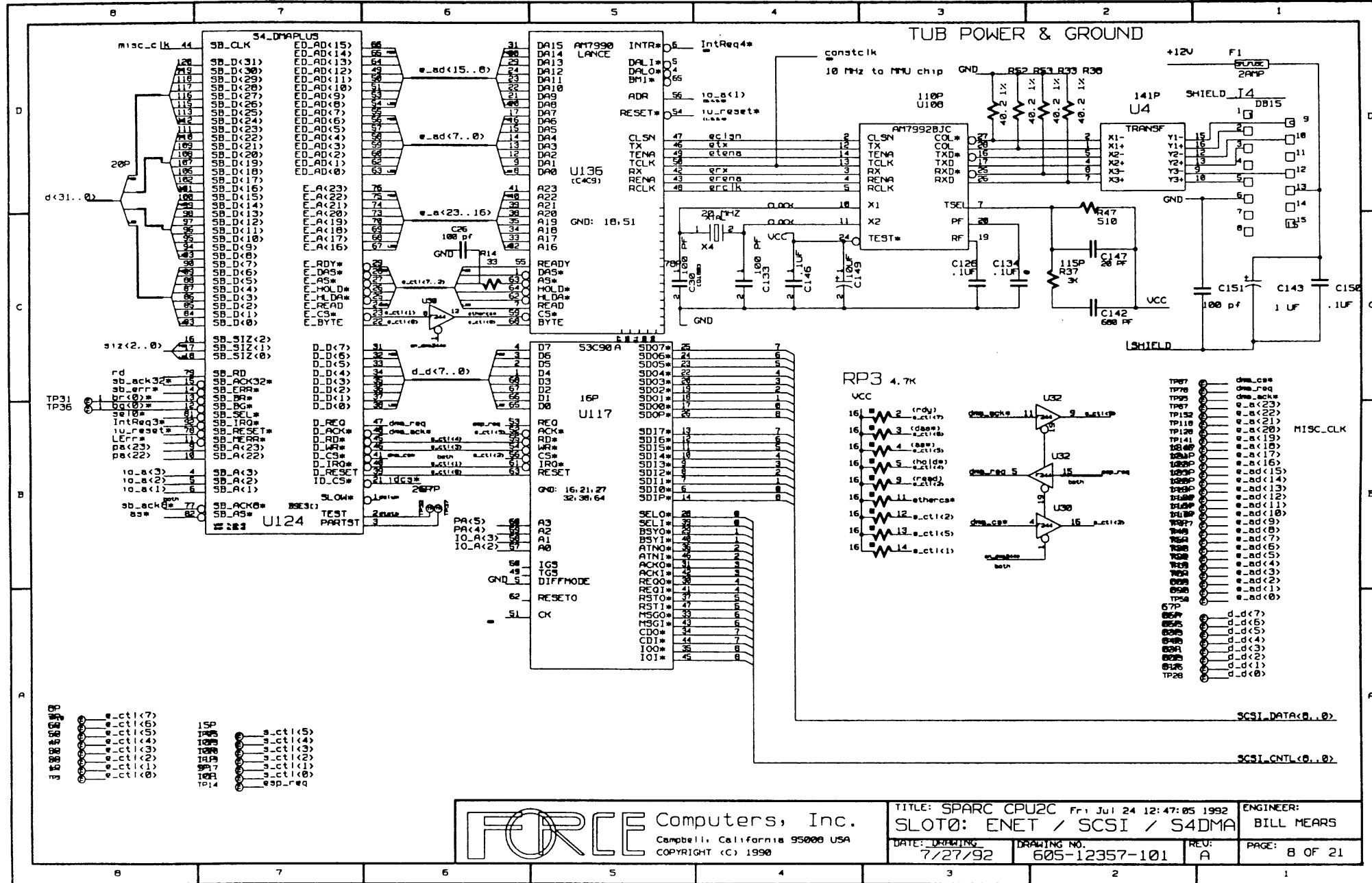
TITLE: SPARC CPU2C Mon Jul 20 15:58:18 1992		ENGINEER: BILL MEARS	
RAMPLUS & EXPANSION CONN'S		PAGE: 6 OF 21	
DATE: 7/27/92	DRAWING NO: 605-12357-101	REV: A	



FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C Sun Jul 19 17:44:47 1992
 16 MBYTE PARITY DRAM
 DATE: DRAWING 7/27/92
 DRAWING NO. 605-12357-101

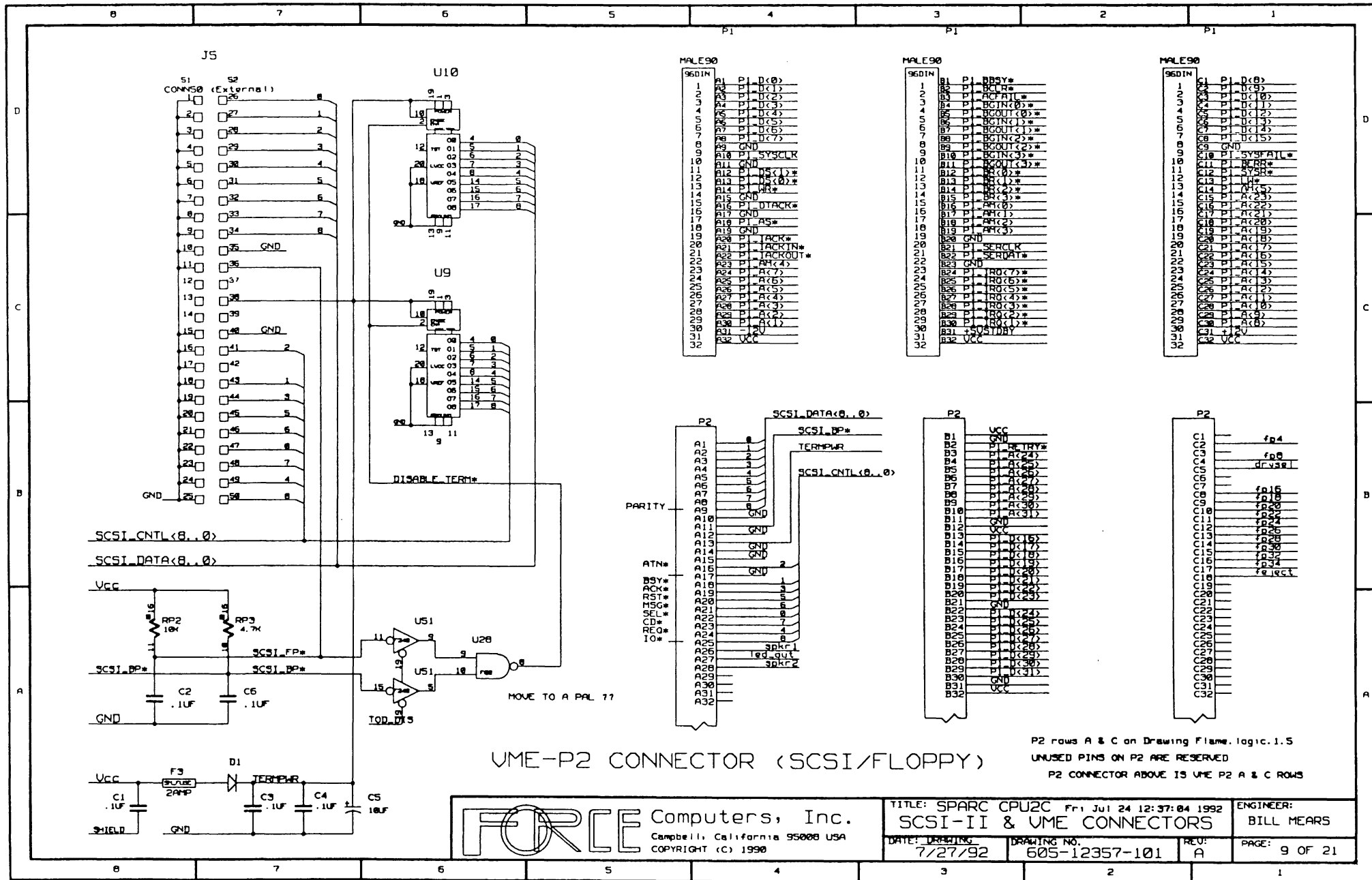
ENGINEER: BILL MEARS
 REV: A
 PAGE: 7 OF 21



FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C Fri Jul 24 12:47:05 1992
 SLOT0: ENET / SCSI / S4DMA
 DATE: DRAWING 7/27/92 DRAWING NO. 605-12357-101 REV: A

ENGINEER: BILL MEARS
 PAGE: 8 OF 21



MALE90

96DIN	A1	P1	D<0>
1	A2	P1	D<1>
2	A3	P1	D<2>
3	A4	P1	D<3>
4	A5	P1	D<4>
5	A6	P1	D<5>
6	A7	P1	D<6>
7	A8	P1	D<7>
8	A9	GND	
9	A10	P1	SYSLCK
10	A11	GND	
11	A12	P1	DS<1>*
12	A13	P1	DS<0>*
13	A14	P1	IR*
14	A15	GND	
15	A16	P1	UTACK*
16	A17	GND	
17	A18	P1	AS*
18	A19	GND	
19	A20	P1	TACK*
20	A21	P1	TACKOUT*
21	A22	P1	AR<4>
22	A23	P1	AR<3>
23	A24	P1	AR<2>
24	A25	P1	AR<1>
25	A26	P1	AR<0>
26	A27	P1	AR<4>
27	A28	P1	AR<3>
28	A29	P1	AR<2>
29	A30	P1	AR<1>
30	A31	P1	AR<0>
31	A32	VCC	

MALE90

96DIN	B1	P1	DBSY*
1	B2	P1	BCLR*
2	B3	P1	BCFRT*
3	B4	P1	BCFNC<0>*
4	B5	P1	BCOUT<0>*
5	B6	P1	BCIN<1>*
6	B7	P1	BCOUT<1>*
7	B8	P1	BCFNC<2>*
8	B9	P1	BCOUT<2>*
9	B10	P1	BCIN<3>*
10	B11	P1	BCOUT<3>*
11	B12	P1	BC<0>*
12	B13	P1	BC<1>*
13	B14	P1	BC<2>*
14	B15	P1	BR<3>*
15	B16	P1	AR<0>
16	B17	P1	AR<1>
17	B18	P1	AR<2>
18	B19	P1	AR<3>
19	B20	GND	
20	B21	P1	SEARCK
21	B22	P1	SERDAT*
22	B23	GND	
23	B24	P1	RO<7>*
24	B25	P1	RO<6>*
25	B26	P1	RO<5>*
26	B27	P1	RO<4>*
27	B28	P1	RO<3>*
28	B29	P1	RO<2>
29	B30	P1	RO<1>
30	B31	P1	RO<0>
31	B32	VCC	

MALE90

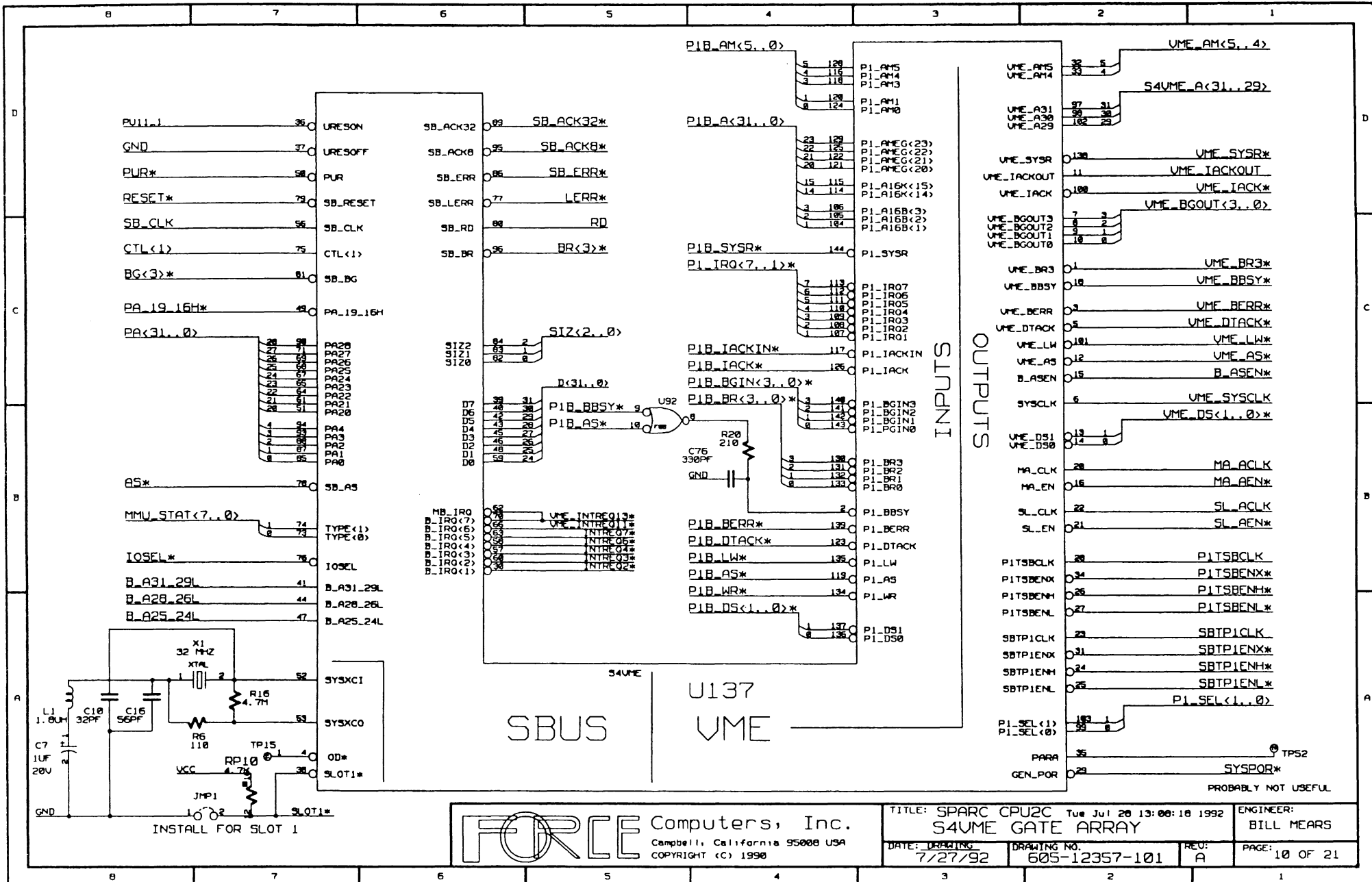
96DIN	C1	P1	D<0>
1	C2	P1	D<1>
2	C3	P1	D<2>
3	C4	P1	D<3>
4	C5	P1	D<4>
5	C6	P1	D<5>
6	C7	P1	D<6>
7	C8	P1	D<7>
8	C9	GND	
9	C10	P1	SYSPATL*
10	C11	P1	BRAR*
11	C12	P1	SYSH*
12	C13	P1	IR*
13	C14	P1	AR<4>
14	C15	P1	AR<3>
15	C16	P1	AR<2>
16	C17	P1	AR<1>
17	C18	P1	AR<0>
18	C19	P1	AR<4>
19	C20	P1	AR<3>
20	C21	P1	AR<2>
21	C22	P1	AR<1>
22	C23	P1	AR<0>
23	C24	P1	AR<4>
24	C25	P1	AR<3>
25	C26	P1	AR<2>
26	C27	P1	AR<1>
27	C28	P1	AR<0>
28	C29	P1	AR<4>
29	C30	P1	AR<3>
30	C31	P1	AR<2>
31	C32	VCC	

VME-P2 CONNECTOR (SCSI/FLOPPY)

P2 rows A & C on Drawing Frame, logic 1.5
 UNUSED PINS ON P2 ARE RESERVED
 P2 CONNECTOR ABOVE IS VME P2 A & C ROWS

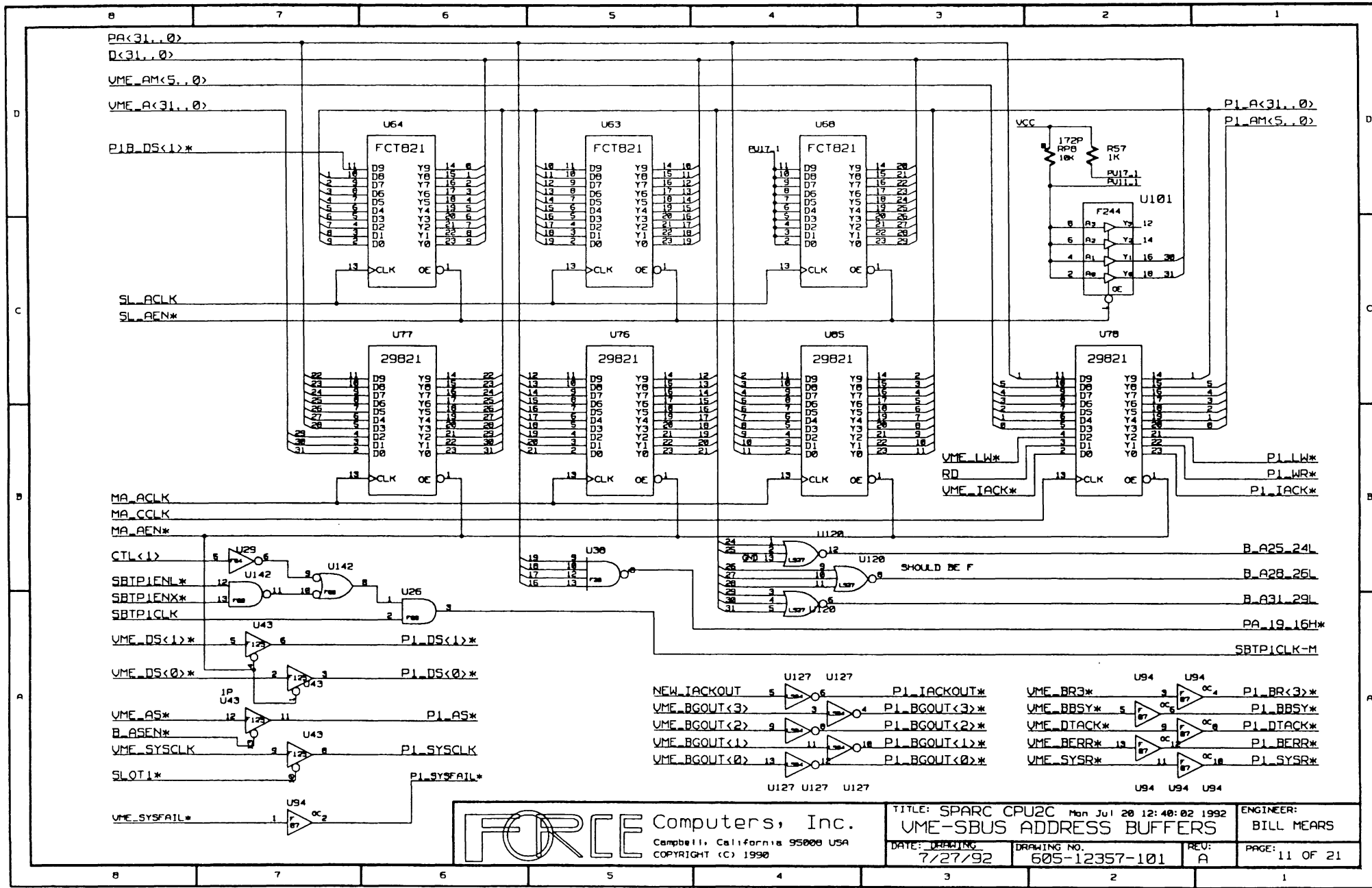
FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C SCSI-II & VME CONNECTORS	DATE: DRAFTING 7/27/92	ENGINEER: BILL MEARS	REU: A
Fri Jul 24 12:37:04 1992	DRAWING NO. 605-12357-101	PAGE: 9 OF 21	



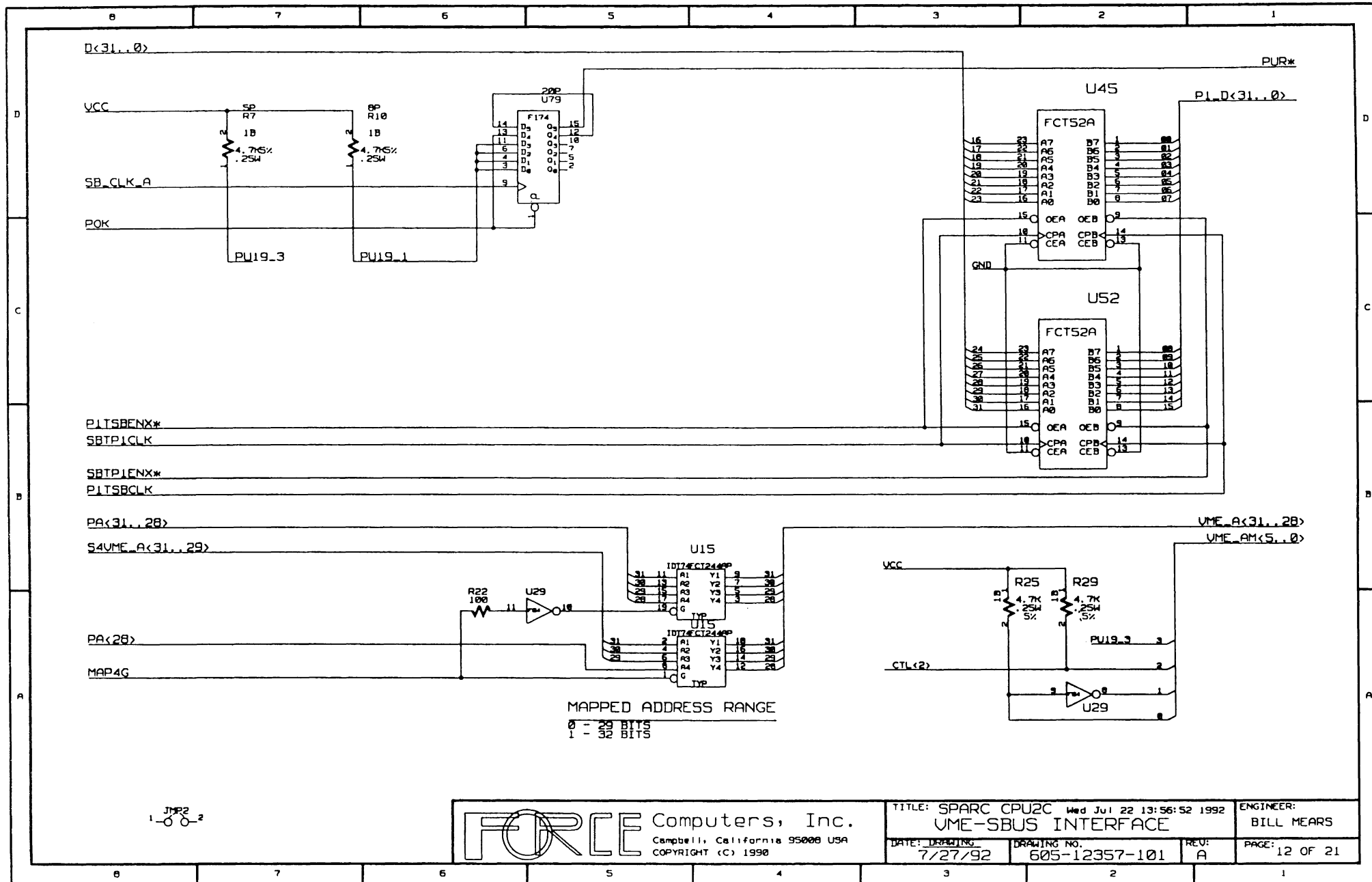
FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C S4UME GATE ARRAY
 DATE: DRAWING 7/27/92
 DRAWING NO. 605-12357-101
 REV: A
 Tue Jul 26 13:00:18 1992
 ENGINEER: BILL MEARS
 PAGE: 10 OF 21



FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

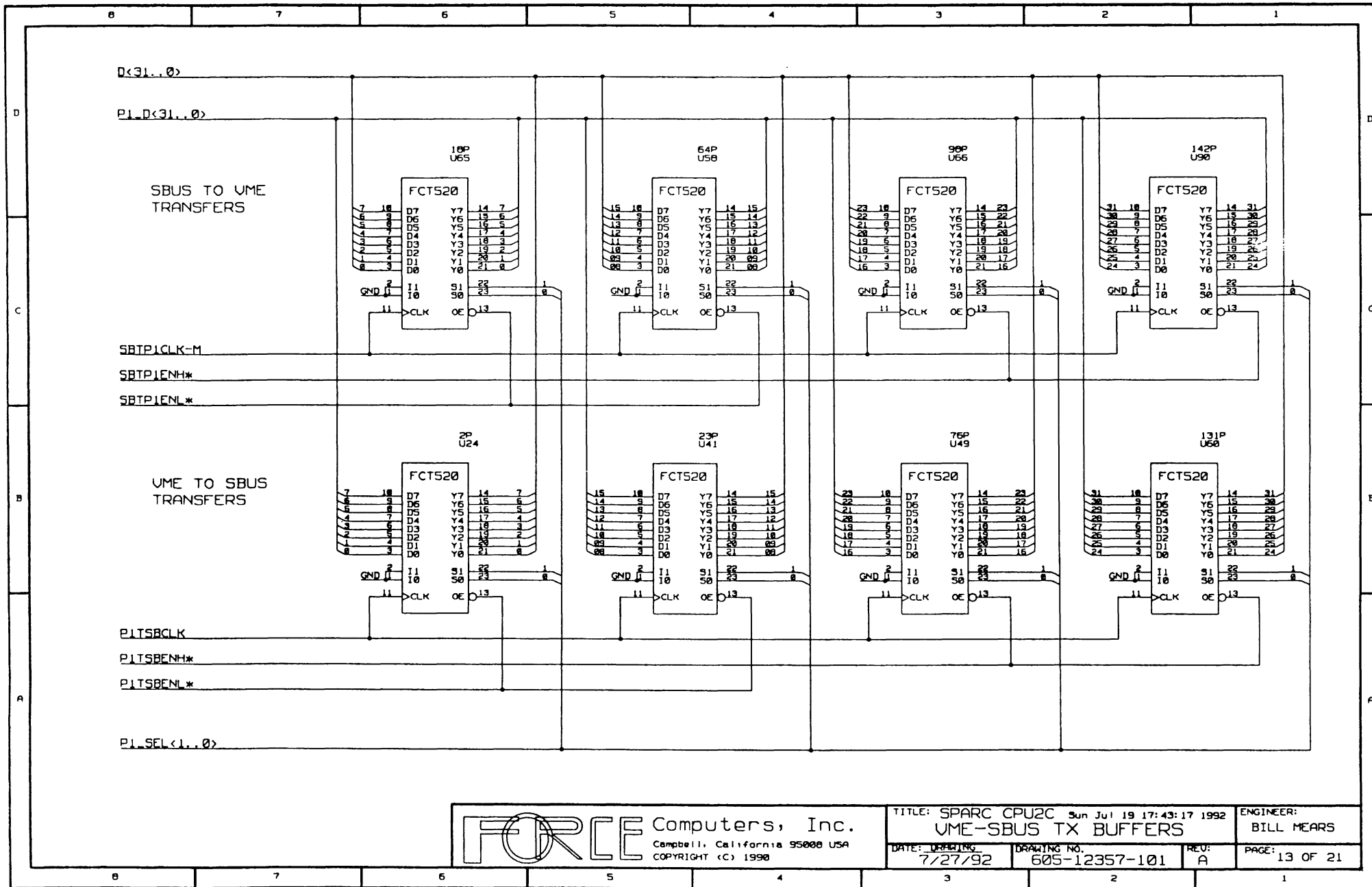
TITLE: SPARC CPU2C Mon Jul 28 12:48:02 1992		ENGINEER: BILL MEARS	
DATE: DRAFTING 7/27/92		DRAWING NO. 605-12357-101	
REV: A		PAGE: 11 OF 21	

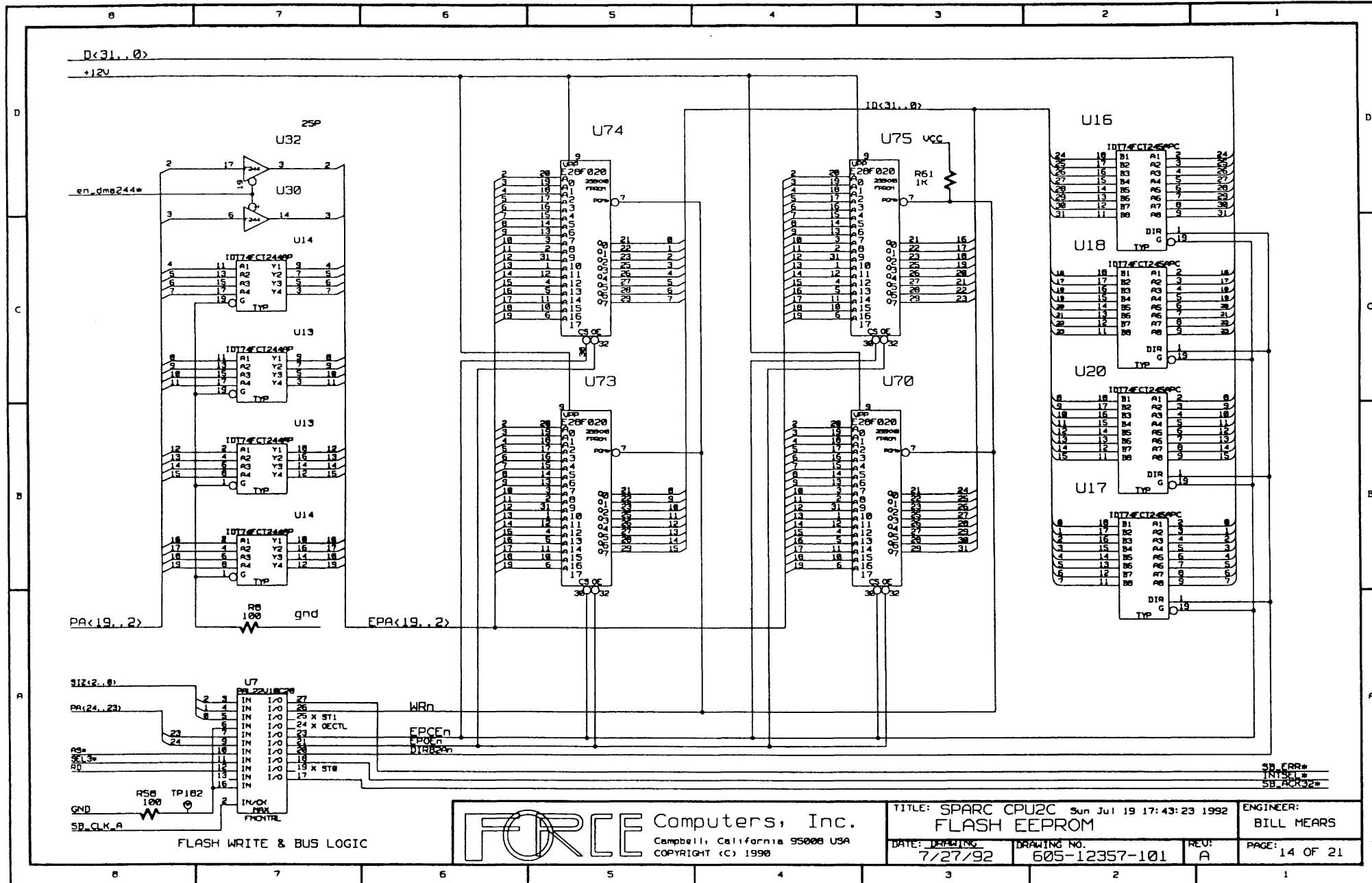


JMR2
1-0-0-2

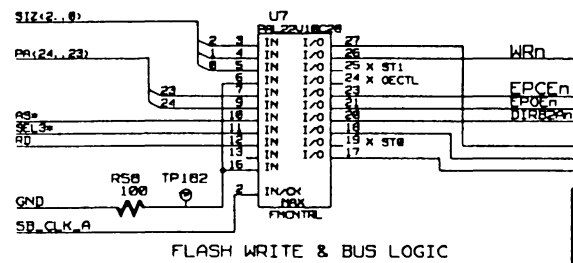
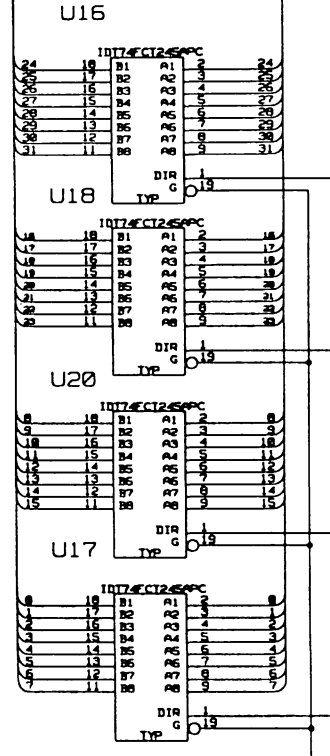
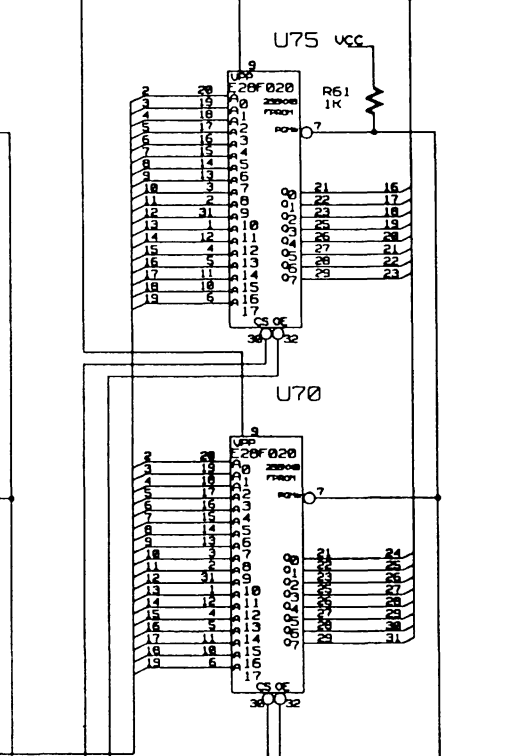
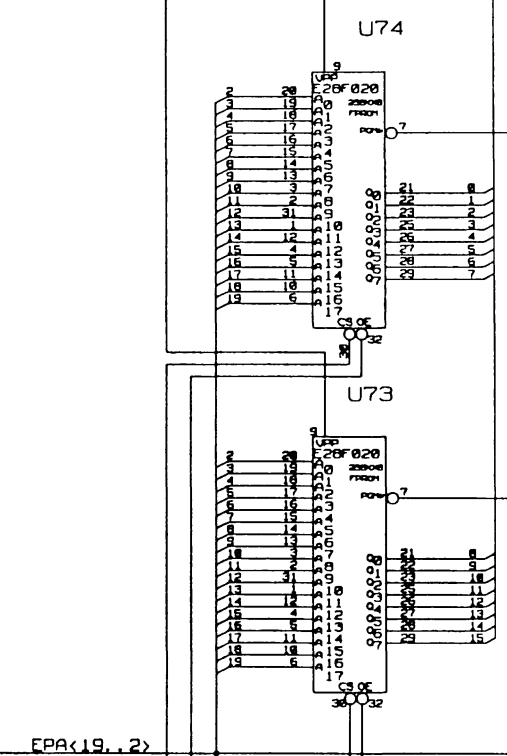
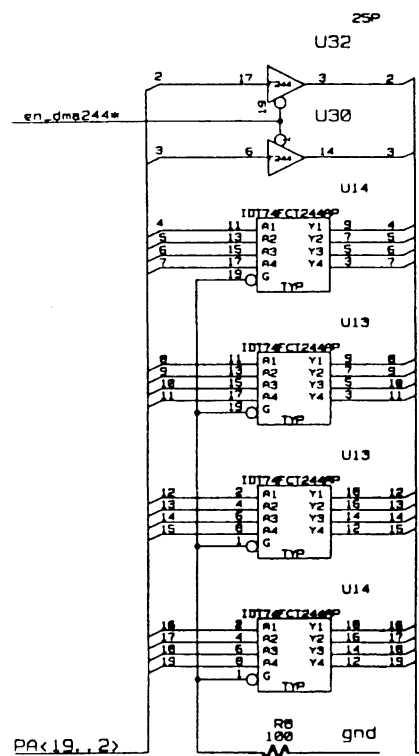
FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C Wed Jul 22 13:56:52 1992		ENGINEER: BILL MEARS	
VME-SBUS INTERFACE			
DATE: DRAFTING 7/27/92	DRAWING NO. 605-12357-101	REV: A	PAGE: 12 OF 21





D<31..0>
+12V



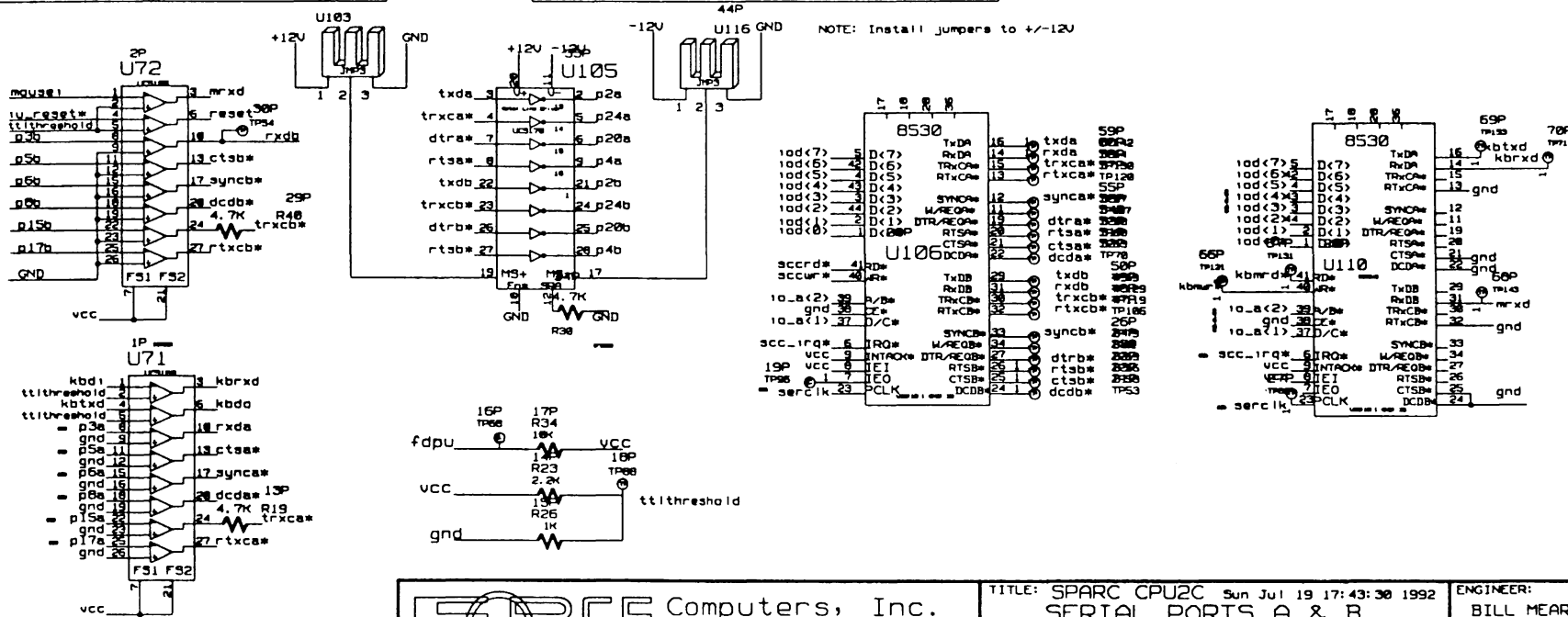
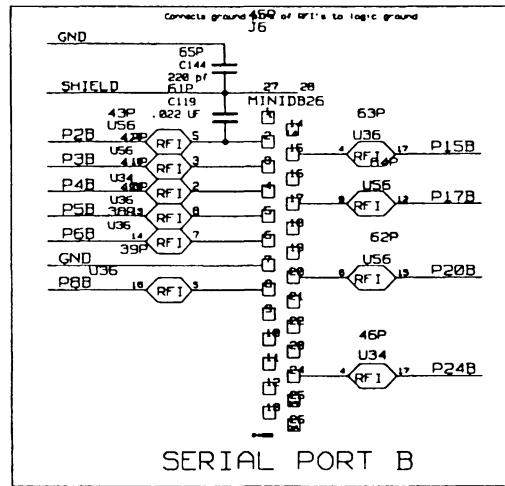
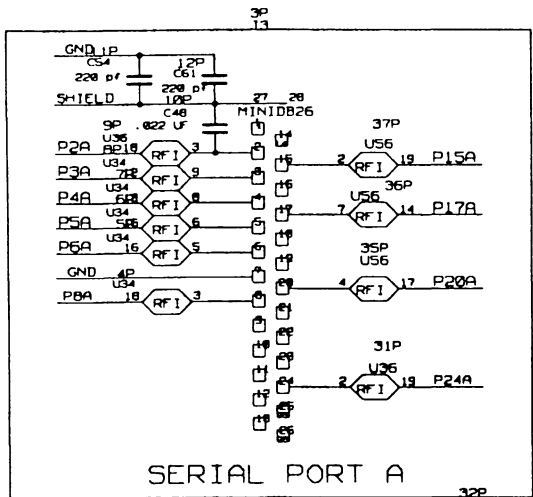
FORCE Computers, Inc.
Campbell, California 95008 USA
COPYRIGHT (C) 1990

TITLE: SPARC CPU2C Sun Jul 19 17:43:23 1992
FLASH EEPROM

DATE: DRAFTING 7/27/92
DRAWING NO. 605-12357-101
REV: A

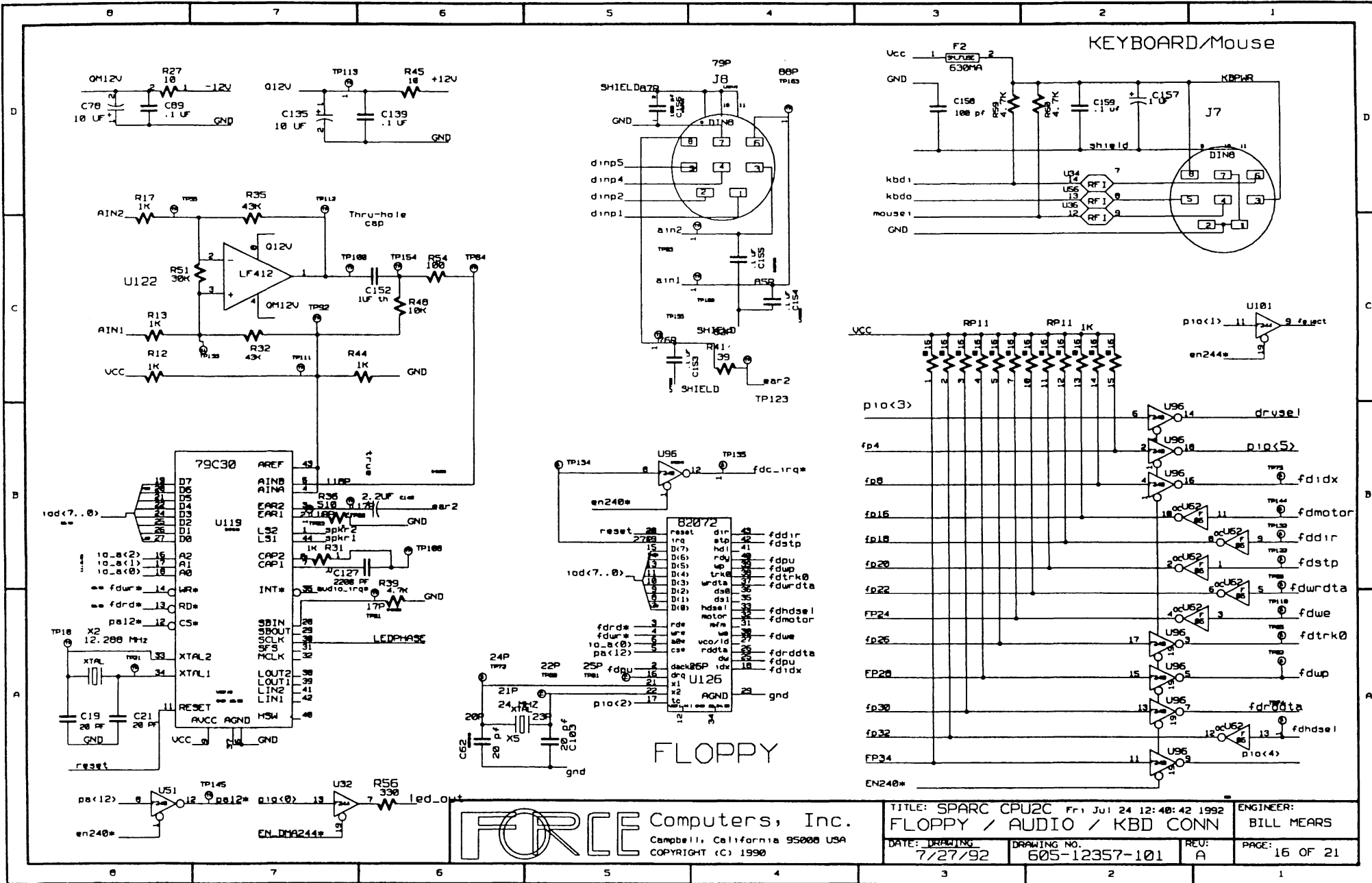
ENGINEER: BILL MEARS
PAGE: 14 OF 21

SB_EPR#
INT#1
SB_ACR32#



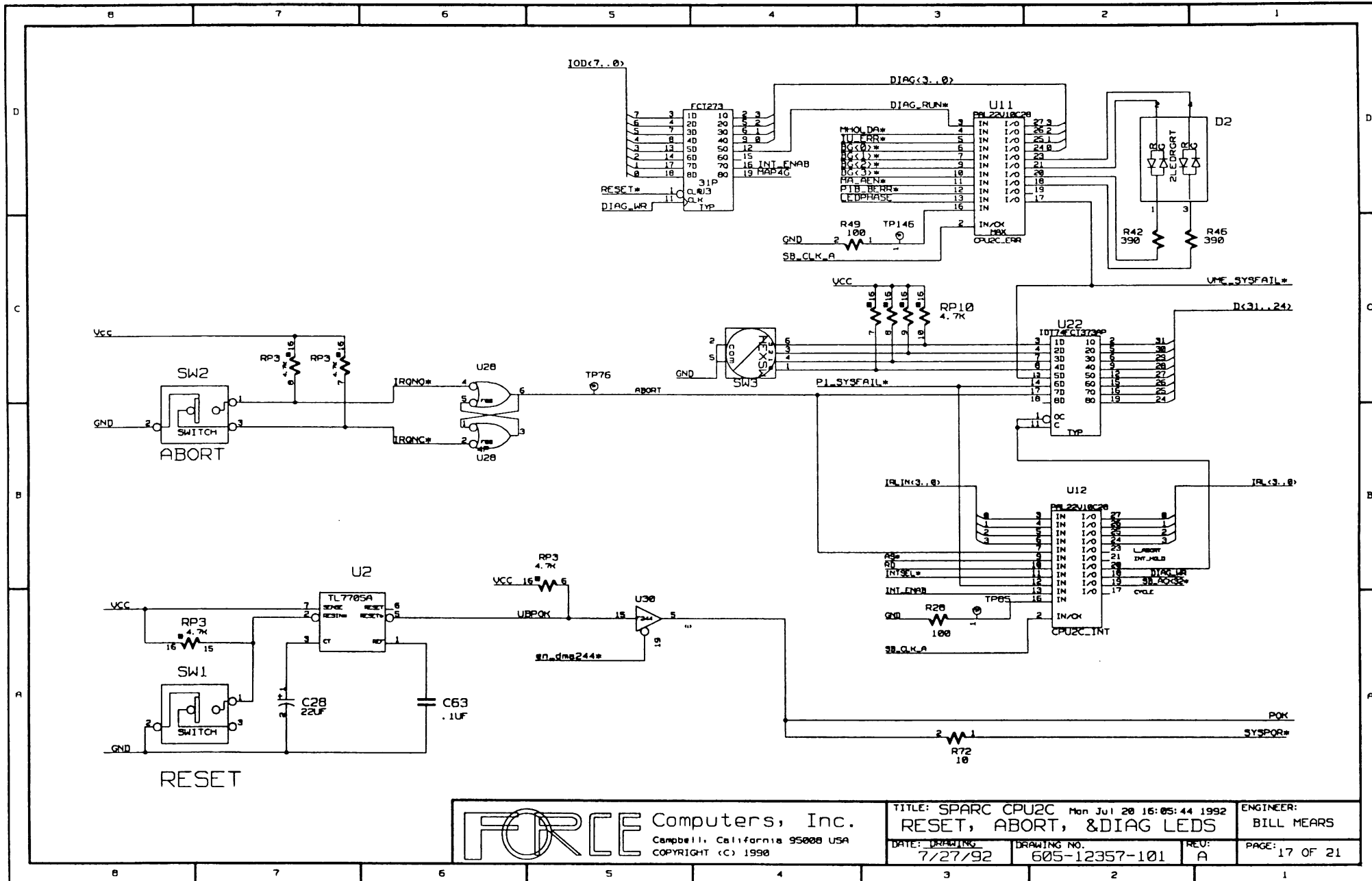
FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C SERIAL PORTS A & B		Sun Jul 19 17:43:30 1992		ENGINEER: BILL MEARS
DATE: DRAWING 7/27/92	DRAWING NO. 605-12357-101	REV: A	PAGE: 15 OF 21	



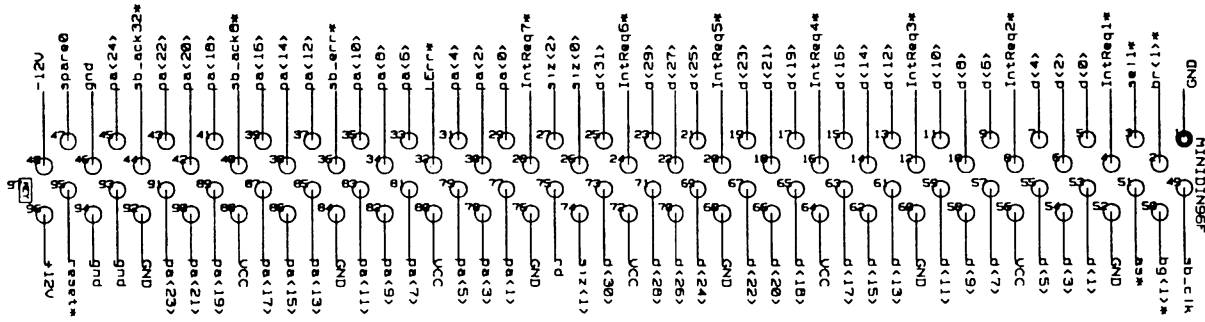
TITLE: SPARC CPU2C		Fri Jul 24 12:40:42 1992		ENGINEER:	
FLOPPY / AUDIO / KBD CONN				BILL MEARS	
DATE: DRAWING	DRAWING NO.	REV:	PAGE:		
7/27/92	605-12357-101	A	16 OF 21		

FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

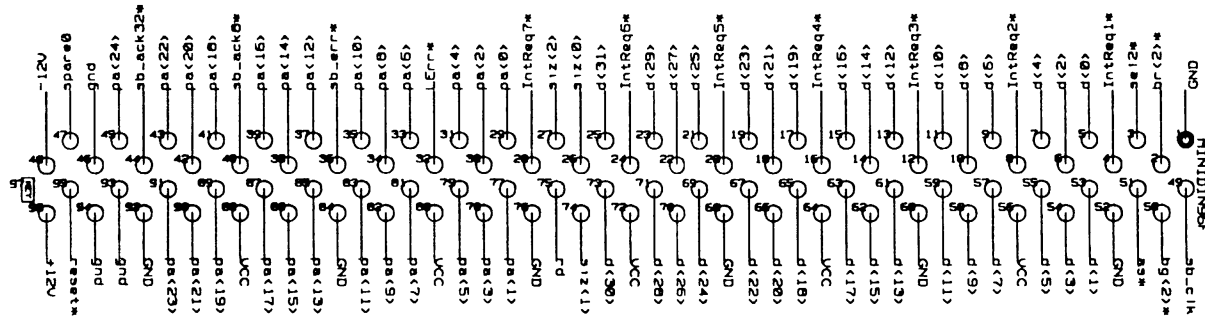


FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C RESET, ABORT, & DIAG LEDs		Mon Jul 28 16:05:44 1992		ENGINEER: BILL MEARS	
DATE: DRAWING 7/27/92	DRAWING NO. 605-12357-101	REV: A	PAGE: 17 OF 21		

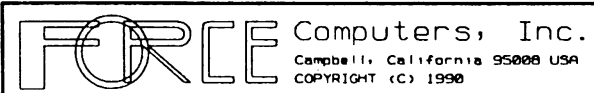


2P
J1
SBus Slot 1

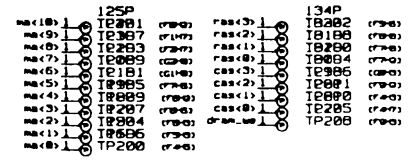
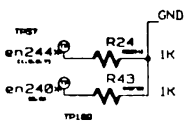
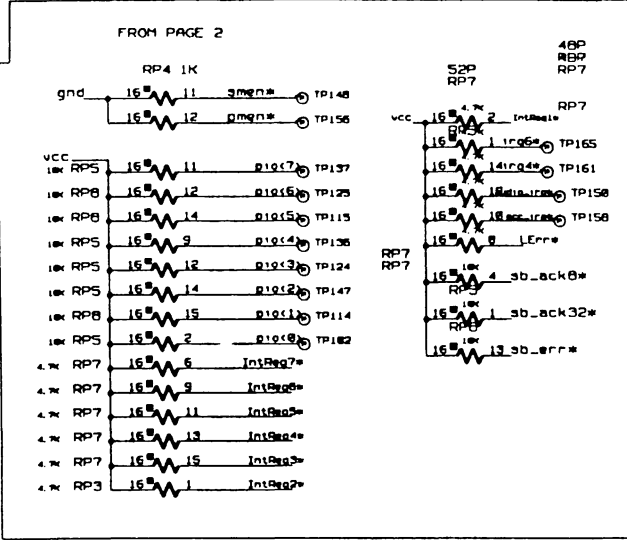


1P
J2
SBus Slot 2

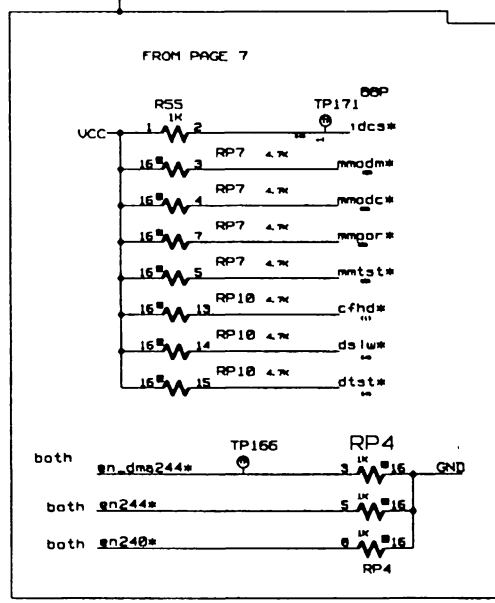
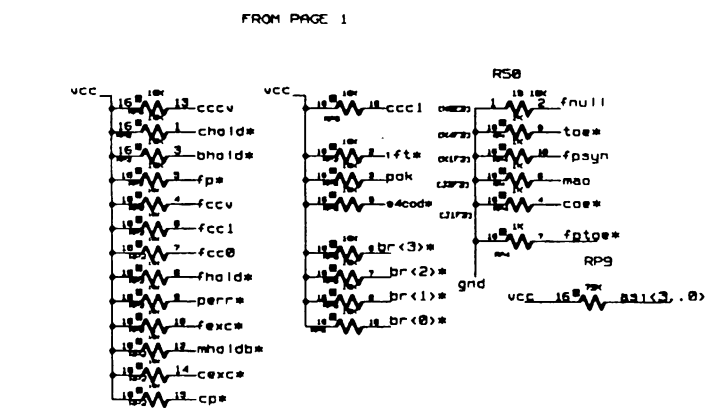
Note: SBus Connectors only get 25 bits of physical address.
(Pins for pa<27..25> are grounded.)



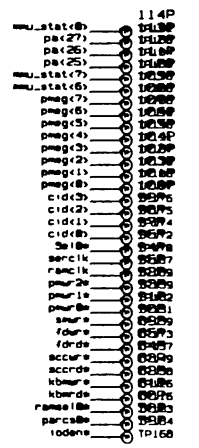
TITLE: SPARC CPU2C Sun Jul 19 17:43:46 1992		ENGINEER: BILL MEARS	
SBUS CONNECTORS			
DATE: DRAWING 7/27/92	DRAWING NO. 605-12357-101	REV: A	PAGE: 18 OF 21



PAGE 3 TP'S

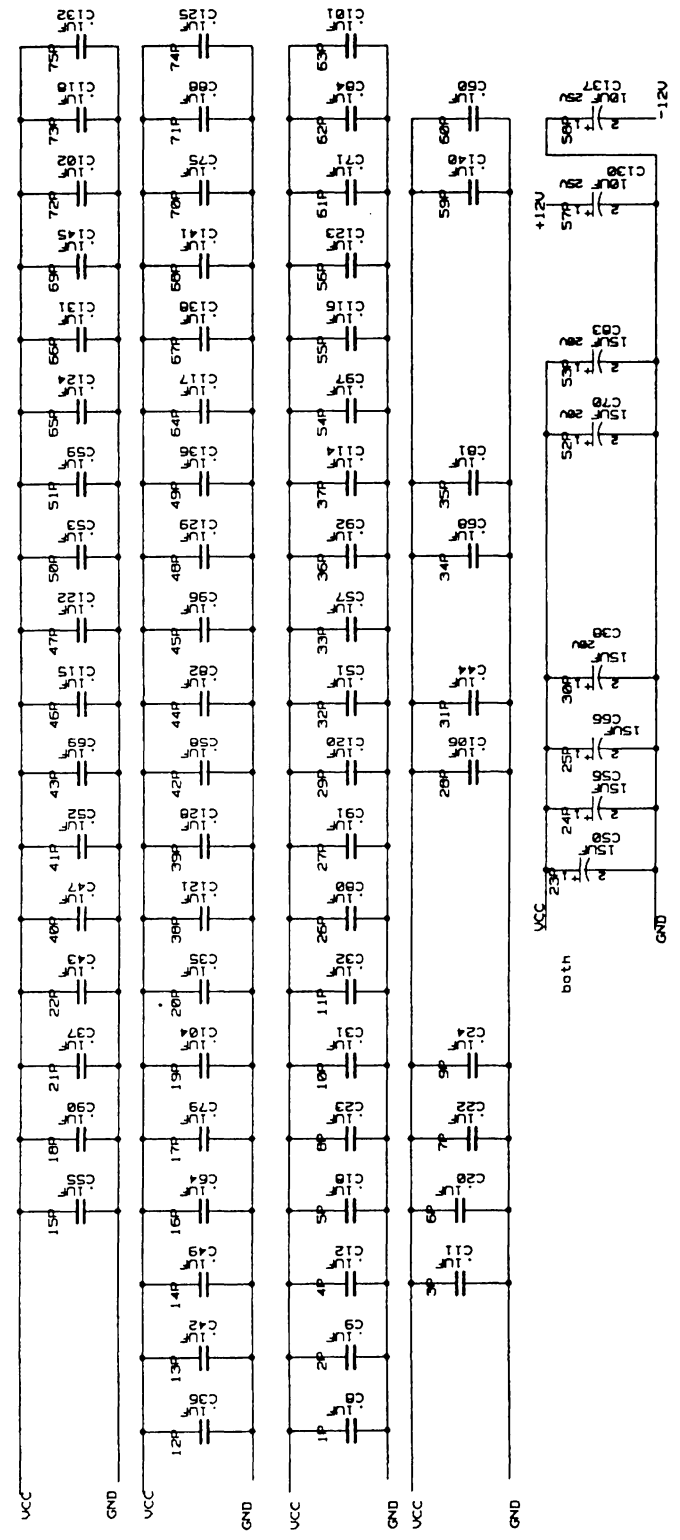


PAGE 2 TP'S



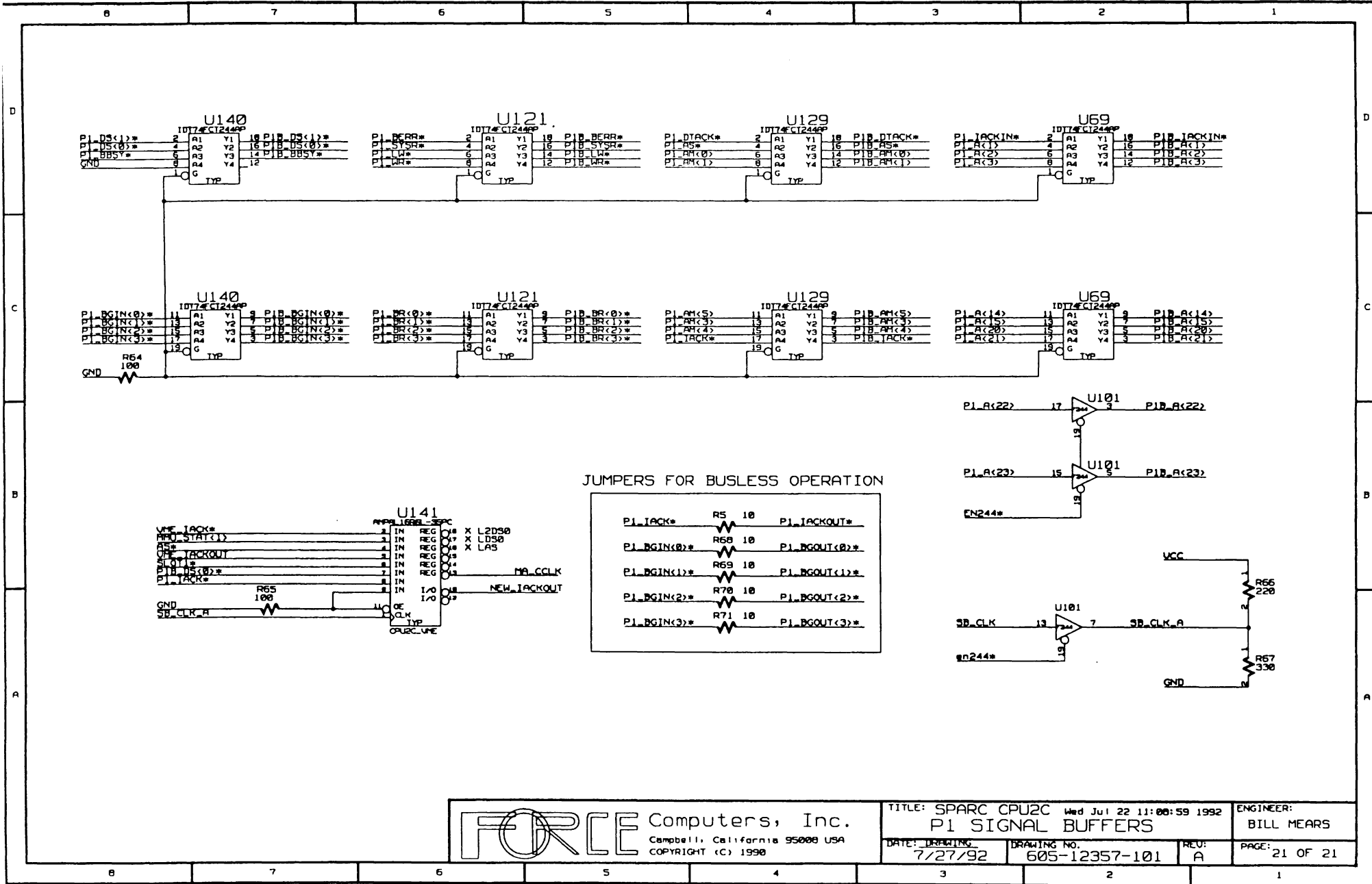
FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990

TITLE: SPARC CPU2C Man Jul 20 12:40:12 1992
 PULLUPS
 ENGINEER: BILL MEARS
 DATE: DRAWING 7/27/92
 DRAWING NO. 605-12357-101
 REV: A
 PAGE: 19 OF 21



TITLE: SPARC CPU2C DECOUPLERS
 DATE: 7/27/92
 DRAWING NO. 605-12357-101

ENGINEER: BILL HEARS
 REV: A
 FORCE Computers, Inc.
 Campbell, California 95008 USA
 COPYRIGHT (C) 1990



JUMPERS FOR BUSLESS OPERATION

P1_IACK*	R5	10	P1_IACKOUT*
P1_BGIN<0>*	R68	10	P1_BGOUT<0>*
P1_BGIN<1>*	R69	10	P1_BGOUT<1>*
P1_BGIN<2>*	R70	10	P1_BGOUT<2>*
P1_BGIN<3>*	R71	10	P1_BGOUT<3>*

Section 5**ABSTRACT COPIES OF DATA SHEETS**

SPARC Processor Weitek W8701

SCSI NCR53C90A

AM7990 LANCE

DRAM

Audio Interface AMD 79C30

Floppy Controller INTEL 82072

FLASH INTEL 28F020 (Optional Feature)

Serial I/O AMD 85C30 and Keyboard Mouse

Line Driver Signetics NE5170 Receiver NE5180

NVRAM Mostek 48T08 RTC-NVRAM

VME

W8701 INTEGRATED SPARC PROCESSOR TECHNICAL OVERVIEW

ADVANCE DATA

February 1991

This technical overview gives basic technical information about WEITEK's W8701 SPARC processor, which combines an integer unit and a floating-point coprocessor on a single chip.

Contents	
Features	1
Description	1
Signal Description	2
DC Specifications	4
AC Specifications	5
Pin Configuration	8
Package Dimensions	9
Ordering Information	9
Sales Offices	back cover

W8701 Integrated SPARC Processor Technical Overview
Advance Data
February 1991

Copyright © WEITEK Corporation 1991
All rights reserved

WEITEK Corporation
1060 East Arques Avenue
Sunnyvale, California 94086
Telephone (408) 738-8400

WEITEK is a trademark of WEITEK Corporation.

WEITEK Corporation assumes no responsibility for errors in this document, and retains the right to make changes at any time, without notice. Please contact your sales office to obtain the latest specifications before placing your order.

Written by Tovy Kanter-Henderson
Edited by Robert Plamondon
Line drawings by Tovy Kantor-Henderson and Robert Plamondon

Printed in the United States of America
92 91 6 5 4 3 2 1

DOC 9101

Features

single-chip pin-compatible replacement for the Cypress CY7C601 integer unit and either the WEITEK WTL 3171 or the Cypress CY7C602 floating-point coprocessor

Lower power and real estate requirements for both current and new designs

Higher floating-point performance (comparison)

Complete binary software compatibility with the Cypress (CY7C601/602), the Cypress-WEITEK (CY7C601/WTL 3171), and the LSI Logic L64811 IU/L68414 FPU chip sets

Available for bus clock speeds of 25, 33, and 40 MHz

Description

The W8701 is a single-chip replacement for the Cypress CY7C601/CY7C602 SPARC™ RISC chips. It contains a SPARC processor and floating-point coprocessor on a single die. The W8701 can be used as a production upgrade to current designs by plugging the W8701 into the CY7C601 processor socket, and leaving the coprocessor socket empty. This will provide higher throughput, lower power requirements, and reduce cost while maintaining full software compatibility.

The W8701 is an implementation of the version 7.0 SPARC architecture. It provides complete compatibility with the Cypress SPARC chip set or the Cypress-WEITEK combination chip set.

This document covers the features unique to the W8701 and provides complete electrical and mechanical specifications.

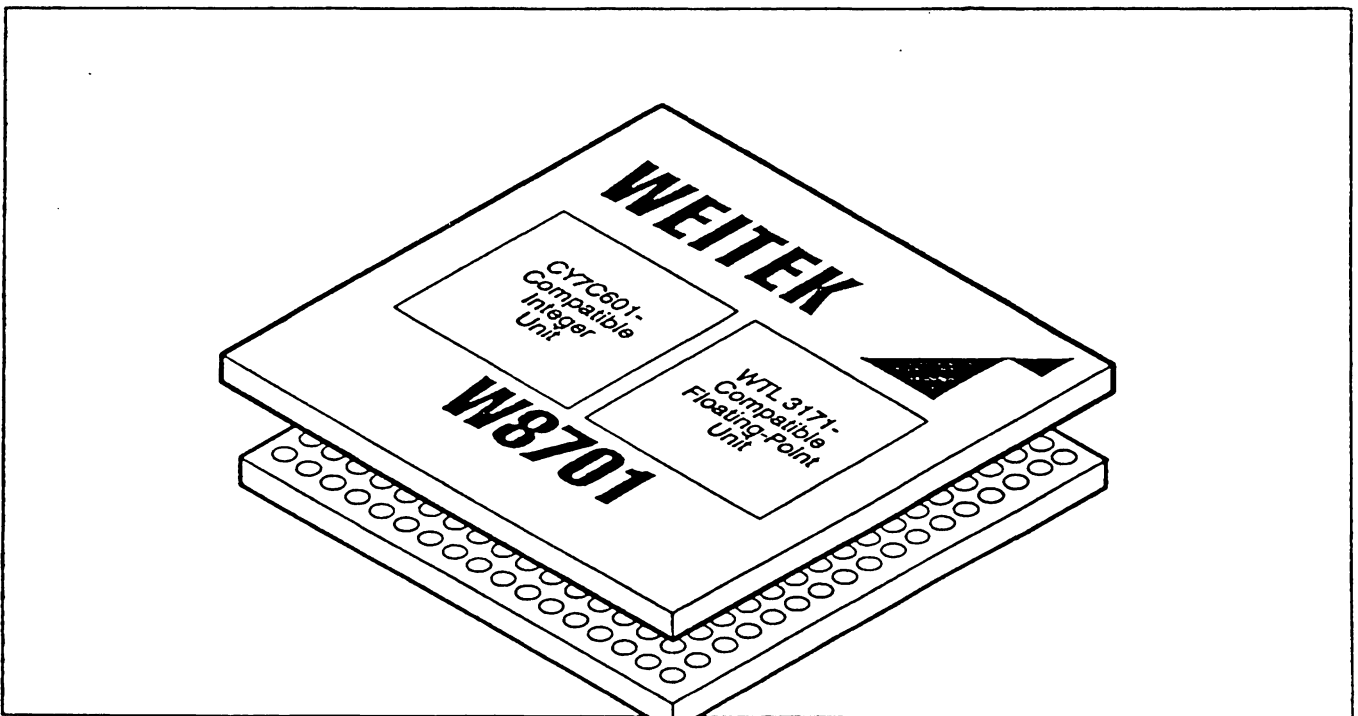


Figure 1. The W8701 SPARC processor: a single-socket chip set

Signal Description

In this chapter we will describe the W8701's external signal interface.

- Memory subsystem interface signals
- Interrupt and control signals
- Integer to floating-point unit signals
- Power and clock signals

NOTATIONAL CONVENTIONS

Signals that are active low will be marked with a suffixed dash (-) after the signal name. All other signals are active high. "Active" and "asserted" are equivalent terms, as are "inactive" and "de-asserted." When signals are shown as ones and zeroes, a "1" always indicates a voltage near VCC, while a "0" indicates a voltage near GND.

Signal	Description	Type
A[31:0]	Address bits	Tri-state Output
AOE-	Address Output Enable	Input
ASI[17:0]	Address Space ID	Tri-state Output
BHOLD-	Hold signal from I/O bus	Input
COE-	Control Output Enable	Output
D[31:0]	Data	Tri-state
DOE-	Data Output Enable	Input
DXFER	Data Transfer	Tri-state Output
FNULL	Floating-point unit null cycle	Output
IFT-	Instruction cycle flush trap	Input
NULL	Integer unit null cycle	Output
LDSTO	Load and store (atomic)	Tri-state output
LOCK	Bus lock (multi-cycle instruction)	Tri-state output
MAO	Memory address Output select	Input
MDS-	Memory data strobe	Input
MEXC-	Memory exception	Input
MHOLDA-	Hold signal from memory	Input
MHOLDB-	Hold signal from memory	Input
RD	Read access	Tri-state output
SIZE[1:0]	Bus transaction size	Tri-state output
WE-	Write enable	Tri-state output
WRT	Write Tag	Tri-state output

Figure 2. Memory system interface signals

Signal Description, continued

Signal (Pin Name)	Description	Type
ERROR-	Internally generated error	Output
INTACK	Interrupt acknowledge	Output
IRL[3:0]	Interrupt request level	Input
RESET-	Reset input	Input
TOE-	Test output enable	Input

Figure 3. Interrupt and control signals

Signal (Pin Name)	Description	Type
FINS1	Floating-Point Inst. Buffer 1	Output
FINS2	Floating-Point Inst. Buffer 2	Output
FLUSH	Flush Floating-point Unit	Output
FXACK	FPU Exception Acknowledge	Output
INST	Instruction Fetch	Output

Figure 4. Integer unit to floating-point unit signals

Signal (Pin Name)	Description	Type
FCC[1:0]	Floating-Point CC bits	Output
FCCV	Floating-Point CC bits valid	Output
FEXC-	Floating-Point Exception	Output
FHOLD-	Hold signal from FPU	Output

Figure 5. Floating-point unit to integer unit signals

Signal (Pin Name)	Description	Type
CLK	Main Clock	Input
VDDI	Main Internal VCC	Power
VDDO	Output Driver VCC	Power
VDDT	Input Circuit VCC	Power
VSSI	Main Internal GND	Power
VSSO	Output Driver GND	Power
VSST	Input Circuit GND	Power

Figure 6. Power and clock signals

DC Specifications

ABSOLUTE MAXIMUM RATINGS

Parameter	Range
Storage Temperature	-65 °C to + 150 °C
Ambient Temperature with Power Applied	-55 °C to + 125 °C
Supply Voltage to Ground Potential ¹	-0.5 VDC to +7.0 VDC
DC Voltage Applied to Outputs in High Z state	-0.5 VDC to +7.0 VDC
DC Input Voltage	-3.0 VDC to +7.0 VDC
DC Output Low sink Current	30 mA
DC Input Current	-10 mA to +10 mA
Ambient Temperature* (T _a)	0 °C to +70 °C
Supply Voltage (V _{cc})	+4.5 VDC to + 5.5 VDC
Power Dissipation (all outputs floating)	2.2 watts

* Ambient temperature is defined as the case temperature just before power is applied
 All power and ground pins must be connected before power is applied.

Figure 7. Absolute maximum ratings

DC CHARACTERISTICS

Parameter	Description	Test Conditions	Minimum	Maximum	Units
V _{OH}	Output High Voltage	V _{cc} = Min, I _{OH} = -2.0 mA	2.4		V
V _{OL}	Output Low Voltage	V _{cc} = Min, I _{OH} = 8.0 mA		0.5	V
V _{IH}	Input High Voltage		2.1	V _{cc}	V
V _{IL}	Input Low Voltage		-3.0	0.8	V
I _{IH}	Input High Current	V _{cc} = Max, V _{IN} = V _{CC}		+10	μA
I _{IL}	Input Low Current	V _{cc} = Max, V _{IN} = V _{SS}		-10	μA
I _{oz}	Output Leakage Current	V _{cc} = Max, V _{IN} = V _{SS} ≤ V _{OUT} ≤ V _{CC}	-40	+40	μA
I _{cc}	Supply Current	V _{CC} = Max, f = 40 MHz		450	mA
I _{cc}	Supply Current	V _{CC} = Max, f = 33 MHz		400	mA

Figure 8. DC characteristics

PIN CAPACITANCE

Parameter	Parameter Description	Value
C _{IN}	Input Capacitance (V _{CC} = 5.0 VDC, T _A = 25 °C, f = 1 MHz)	10 pF maximum
C _{OUT}	Output Capacitance (V _{CC} = 5.0 VDC, T _A = 25 °C, f = 1 MHz)	12 pF maximum
C _{IO}	I/O pin Capacitance (V _{CC} = 5.0 VDC, T _A = 25 °C, f = 1 MHz)	15 pF maximum

Figure 9. Pin capacitance

ADVANCE DATA
February 1991

AC Specifications

NOTATIONAL CONVENTIONS

The individual AC specifications are not numbered; they are identified by the signal name and a parameter type from the table in figure 10. Future editions will use WEITEK's usual notation for AC specifications.

Signal	Definition
T_{do}	Propagation delay time of an output referenced to a given clock edge
T_{ho}	Hold time of an output referenced to a given clock edge
T_{si}	Setup time of an input referenced to a given clock edge
T_{hi}	Hold time of an input referenced to a given clock edge
T_{off}	Turn off time for a tri-state output driver after the rising edge of DOE-
T_{on}	Turn on time for a tri-state output driver after the falling edge of DOE-
CLK+	The rising edge of clock CLK
CLK-	The falling edge of clock CLK
C_{LOAD}	Output Load Capacitance: A load capacitance of 50 pF is assumed

All times are given in ns. Clock references are made with respect to the 1.5 VDC level of the clock. HOLD signal T_{do} from CLK+ must be 8 ns or less @ 33 MHz and 7ns or less @ 40 Mhz. $t_4 = t_1 \leq 3ns$. Inputs switch 0 to 3.0 VDC

Figure 10. AC timing specification conventions

OUTPUT SIGNALS

Pin Name	Parameter	Ref Clock Edge	33 MHz		40 MHz	
			MAX	MIN	MAX	MIN
A[31:0]	T_{do}/T_{ho}	CLK+	24	7	20	7
ASI[7:0]	T_{do}/T_{ho}	CLK+	24	7	20	7
SIZE[1:0]	T_{do}/T_{ho}	CLK+	24	7	20	7
RD	T_{do}/T_{ho}	CLK+	24	7	20	7
LDSTO	T_{do}/T_{ho}	CLK+	24	7	20	7
LOCK	T_{do}/T_{ho}	CLK+	24	7	20	7
WE-	T_{do}/T_{ho}	CLK+	24	7	20	7
WRT	T_{do}/T_{ho}	CLK+	24	7	20	7
D[31:0] Out	T_{do}/T_{ho}	CLK+	15	4	13	4
DXFER	T_{do}/T_{ho}	CLK+	23	2	19	2
ERROR	T_{do}/T_{ho}	CLK+	15	3	13	3
FNULL	T_{do}/T_{ho}	CLK+	13	3	11	3
INTACK	T_{do}/T_{ho}	CLK+	15	3	13	3
INULL	T_{do}/T_{ho}	CLK+	13	3	11	3

Figure 11. W8701 output timings, relative to CLK

AC Specifications, continued

OUTPUTS WITH RESPECT TO HOLD AND MAO

Pin Name	Parameter	Ref Clock Edge	33 MHz		40 MHz	
			Min	Max	Min	Max
A/CTL ¹	T_{dn}/T_{hn}	HOLD+/-	0	15	0	12
A/CTL ²	T_{dn}/T_{hn}	³ MAO+/-	2	14	2	12
DXFER	T_{dn}/T_{hn}	² HOLD+/-	0	15	0	12
DXFER	T_{dn}/T_{hn}	² HOLD+/-	0	0	0	0

¹ A/CTL signals include: A[31:0], AS[17:0], SIZE[1:0], RD, LDSTO, LOCK, WE-, and WRT
² The HOLD in "Reference Edge" above includes MHOLDA-, MHOLDB-, BHOLD-, and FHOLD
³ MAO may be deleted

Figure 12. W8701 outputs with respect to HOLD and MAO

INPUTS WITH RESPECT TO CLK

Pin Name	Parameter	Ref Edge	33	33	33	33	40	40	40	40
			MHz	MHz	MHz	MHz	MHz	MHz	MHz	MHz
			T_{Si} min	T_{Hi} min	T_{Si} max	T_{Hi} max	T_{Si} min	T_{Hi} min	T_{Si} max	T_{Hi} max
D[31:0] In	T_{si}/T_{hi}	CLK+	2	5			2	4		
MEXC-	T_{si}	CLK+	11	1			10	1		
MHOLDA- ⁴	T_{si}	CLK+	4	5			3	4.5		
MDS-	T_{si}	CLK+	4	5			3	4.5		
IRL[3:0]	T_{si}	CLK+			10	5			8	4
RESET-	T_{si}	CLK+			10	3			8	2
AOE- ⁵	T_{si}				11	19			9	17
TOE-	T_{hi}									

⁴ This signal description includes MHOLDB-, and BHOLD-
⁵ This signal description includes COE-, and DOE-

Figure 13. W8701 inputs (setup and hold times) with respect to CLK

AC Specifications, continued

CLOCK INPUT SPECIFICATIONS

Parameter	Description	Min-Max	33 MHz	40 MHz	Unit
T_{cyl}	Clock Period	min	30	25	ns
T_{cjh}	Clock High Time	min	13	11	ns
T_{cjl}	Clock Low Time	min	13	11	ns
T_{cr}	Clock Rise Time	max	1	1	V/ns
T_{cf}	Clock Fall Time	max	1	1	V/ns

Figure 14. Clock input specifications

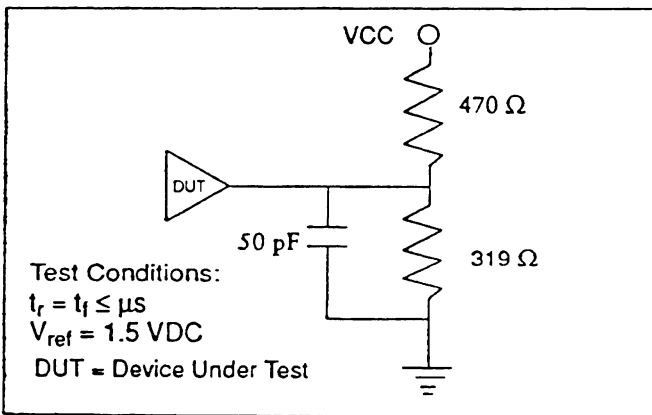


Figure 15. W8701 output loading circuitry

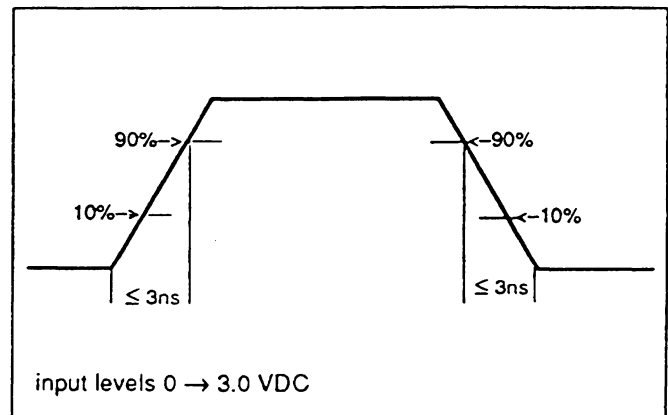


Figure 16. Clock input timing diagram

Pin Configuration


	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T	U	
17	VCC	GND	TO2	VDD	FINS1	GND	D26	D24	D23	D21	D19	D17	DOE-	VCC	VCC	GND	GND	17
16	VCC	GND	GND	FINS2	VCC	D29	D28	D27	D22	D20	D18	D14	D15	VCC	VCC	GND	GND	16
15	VCC	ERROR-	TOE-	TO1	GND	D30	D31	D25	VCC	VCC	D16	D13	D12	D11	D10	GND	D9	15
14	GND	FLUSH	IFT-	GND	VCC	GND	VCC	GND	GND	GND	GND	VCC	GND	GND	GND	D6	D8	14
13	INTACK	TI3	TDO	GND	W8701 SPARC Processor 207-pin PGA Top View Cavity Down									VCC	D7	D5	D4	13
12	FNULL	IRL3	TMS	VDD										VCC	VCC	D3	D2	12
11	FCC0	FCC1	IRL1	FXACK										GND	GND	D1	A31	11
10	IRL0	NC	FCCV	IRL2										VCC	D0	A30	GND	10
9	RESET-	GND	NC	GND										GND	A29	A28	A26	9
8	FHOLD-	MHOLDB	MHOLDA	MEXC-										VCC	A27	A25	A24	8
7	BHOLD-	MDS-	NC	VDD										GND	VCC	A23	A22	7
6	NC	NC	INST	GND										GND	A18	A19	A21	6
5	FEXC-	NULL	LDSTO	VDD										GND	GND	A17	A20	5
4	RD	WE-	GND	LOCK										WRT	GND	VCC	GND	GND
3	GND	GND	GND	DXFER	MA0	AS10	AS12	VCC	GND	CLK	A2	A8	A12	AOE-	A16	VCC	VCC	3
2	VCC	GND	COE-	SIZE1	SIZE0	AS11	AS13	AS15	GND	A0	A4	A5	A6	A10	A14	VCC	GND	2
1		VCC	GND	VCC	VCC	GND	AS14	AS16	AS17	A1	A3	A7	A11	A9	GND	GND	GND	1
	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T	U	

Figure 17. Pin configuration for the W8701

ADVANCE DATA
February 1991

Package Dimensions

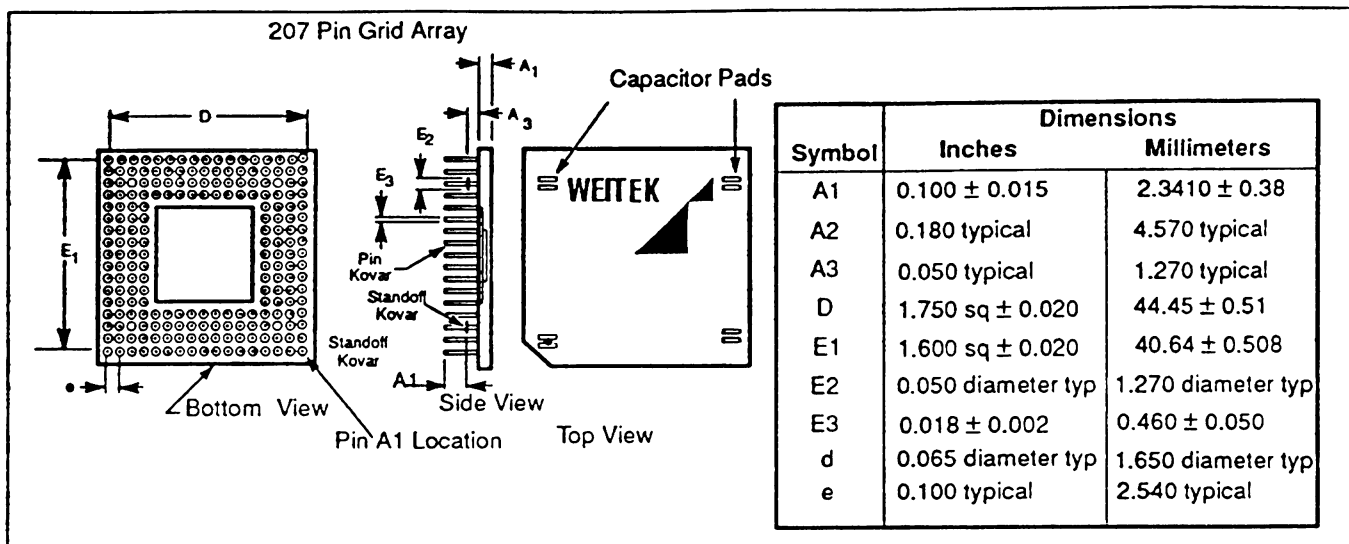


Figure 18. W8701 Physical Dimensions and package information

Ordering Information

Package Type	Speed Grade	Temperature Range (case)	Order Number
207-pin ceramic PGA	25 MHz	0-85°C	W8701-025-GCD
207-pin ceramic PGA	33 MHz	0-85°C	W8701-033-GCD
207-pin ceramic PGA	40 MHz	0-85°C	W8701-040-GCD

Figure 19. Ordering information

NCR 53C90A, 53C90B

Figure 2. NCR 53C90 and 53C90A Pin Configurations

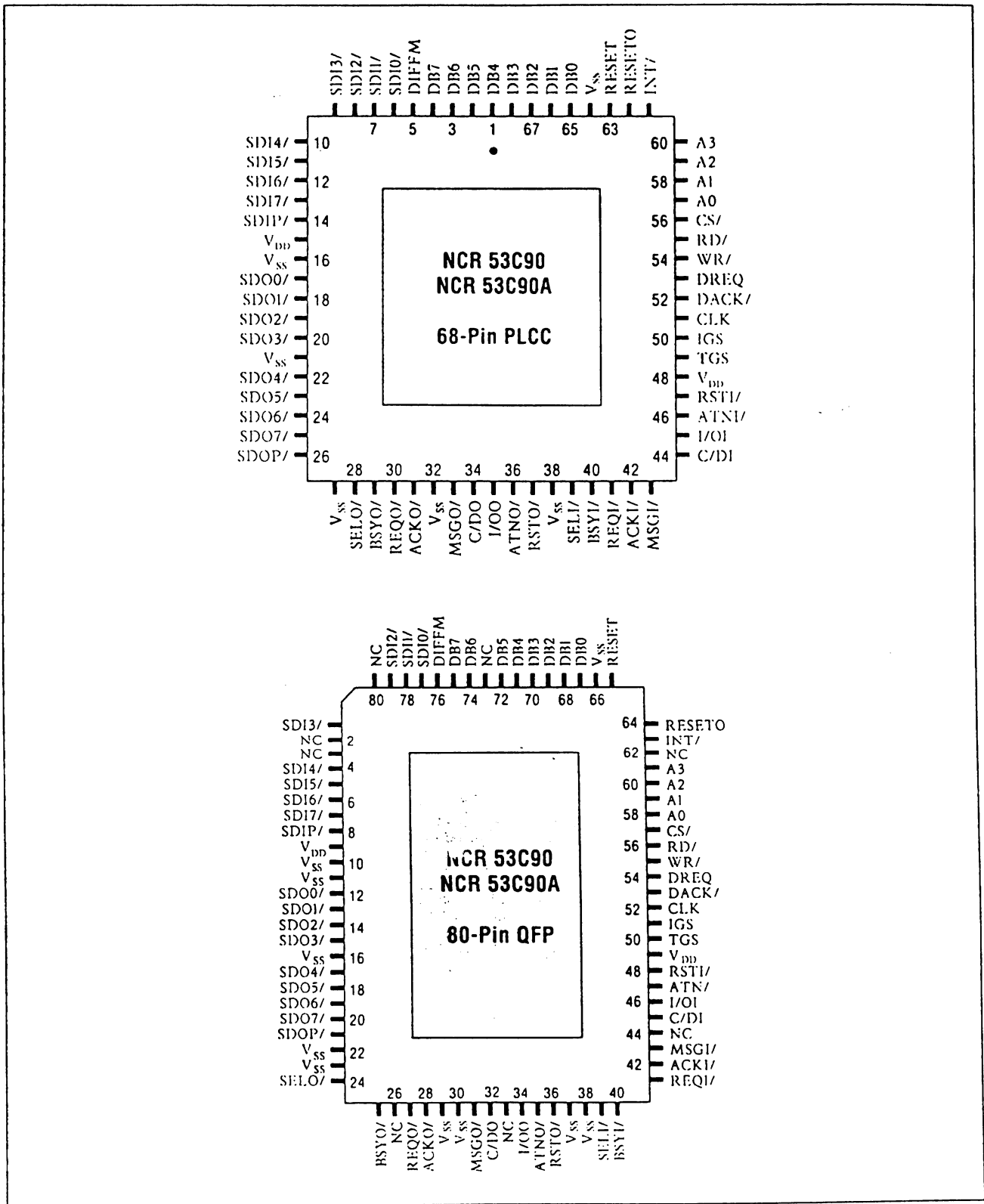


Table 1. Host Processor and DMA Interface Pins – PLCC Package

Pin	Signal	Type	Description
4-1, 68-65	DB7-DB0	B	Active-high data bus connected to the DMA controller, CPU and buffer memory. Each pad contains a pull-up to V_{DD} (12.5K minimum).
63	RESET	I	Active-high chip reset. Reset must be asserted for two CLK periods, minimum, after the voltage on the power pins has reached V_{DD} min. This input must not be connected to RESET0.
62	RESET0	O	Active-high reset output. This output is always asserted when the RESET input is true OR may be asserted when the SCSI reset signal is active if bit 6 of the Config 1 register is cleared and the host has not serviced the interrupt (generated because of SCSI reset) within 1-2 ms (depending on CLK frequency and clock conversion factor). Refer to <i>Bus Initiated Reset</i> .
61	INT/	O	Active-low, open drain interrupt signal to the microprocessor. It is latched on the rising edge of CLK and may be cleared by reading the interrupt register or by a host hardware reset, or by a host software reset (but not by a SCSI reset). This output cannot be disabled internally.
60-57	A3-A0	I	Active-high address bus which specifies one of the ASPs internal registers for reading or writing. Used with CS/, ignored with DACK/.
56	CS/	I	Active-low chip-select signal that enables access to the ASPs internal registers. CS/ accesses any register, including the FIFO, while DACK/ accesses only the FIFO. CS/ and DACK/ must never be active at the same time.
55	RD/	I	Active-low read signal that enables ASP data onto DB7-DB0. CS/ or DACK/ must also be active.
54	WR/	I	Active-low write signal that strobes DB7-DB0 data into the ASP. CS/ or DACK/ must also be active.
53	DREQ	O	Tri-state active-high DMA request to the DMA controller. DREQ will be true as long as the FIFO has at least one byte to send to memory, or has room to receive at least one byte from memory, depending on data direction.
52	DACK/	I	Active-low DMA acknowledge from the DMA controller. DACK/ accesses the FIFO only, while CS/ accesses any register, including the FIFO. CS/ and DACK/ must never be active at the same time. DACK/ must toggle true then false for every byte transferred. Refer to <i>DREQ Hi Z Bit in Config 2</i> .
51	CLK	I	Square wave clock that generates internal chip timing. The maximum frequency is 25 MHz, with a 35% to 65% duty cycle. The minimum frequency required for asynchronous SCSI transmission is 10 MHz. The minimum frequency required for synchronous transmission is 12 MHz. The synchronous transmission rate is equal to the CLK input period divided by the value in the synchronous transfer period register. The asynchronous transfer rate is indirectly affected by CLK frequency. Refer to <i>Data Transfer Rate</i> .

Table 2. SCSI Bus Interface – PLCC Package

Pin	Signal	Type	Description
6-13, 14	SDI0/-SDI7/, SDIP/	B	Schmitt trigger, active-low SCSI data/parity bus. In single-ended mode (DIFFM = 0) these inputs are SCSI data bus signals. In differential mode (DIFFM = 1) these are bi-directional data and parity signals for external SCSI bus transceivers.
17-20, 22-25 26	SDO0/-SDO7/ SDOP	O O	48 mA, open drain SCSI data parity bus. In single-ended mode (DIFFM = 0) these outputs are active-low SCSI data signals. In differential mode (DIFFM = 1) these outputs are used to control the direction of external differential transceivers, with high meaning output to the SCSI bus, low meaning input from the SCSI bus.
28	SELO/	O	48 mA, open drain SCSI select signal. In single-ended mode this output is active-low. In differential mode it is active-high.
29	BSYO/	O	48 mA, open drain SCSI busy signal. In single-ended mode, this output is active-low. In differential mode it is active-high.
30	REQO/	O	48 mA, open drain, active-low SCSI request signal. This output is only asserted when the ASP is in target mode.
31	ACKO/	O	48 mA, open drain, active-low SCSI acknowledge signal. This output is only asserted when the ASP is in initiator mode.
33-35	MSGO/, C/DO, I/OO	O	48 mA, open drain, active-low SCSI phase signals. These outputs are only asserted when the ASP is in target mode.
36	ATNO/	O	48 mA, open drain, active-low SCSI attention signal. This output is only asserted when the ASP is in initiator mode. Several ASP commands will set ATN. It is also asserted when the ASP detects an incoming parity error if parity checking is enabled.
37	RSTO/	O	48 mA, open drain SCSI reset signal. In single-ended mode this output is active-low. In differential mode it is active-high. The ASP drives this signal true only when the host writes the SCSI bus reset command to the command register. The pulse length is 25-40 us, depending on CLK frequency and clock conversion factor. Refer to <i>Bus Initiated Reset</i> .
39	SELI/	I	Schmitt trigger, active-low SCSI select input.
40	BSYI/	I	Schmitt trigger, active-low SCSI busy input.
41	REQUI/	I	Schmitt trigger, active-low SCSI request input.
42	ACKI/	I	Schmitt trigger, active-low SCSI acknowledge input.
43	MSGI/	I	Schmitt trigger, active-low SCSI message input.
44	C/DI	I	Schmitt trigger SCSI control/data input.
45	I/OI	I	Schmitt trigger SCSI input/output input.
46	ATNI/	I	Schmitt trigger, active-low SCSI attention input.
47	RSTI/	I	Schmitt trigger, active-low SCSI reset signal. When this input is true, the ASP will automatically disconnect from the SCSI bus. If bit 6 in the Config 1 register is zero, the ASP will interrupt the host. If the interrupt is not serviced within 1-2 ms, the ASP will reset its host processor. Refer to <i>Bus Initiated Reset</i> .
50	IGS	O	Active-high initiator group select signal. This pin is high whenever the ASP is in initiator mode. It is used in differential mode to enable the initiator signals (ACKO/, ATNO/). When low, the ASP should be receiving these signals.

NCR 53C90A, 53C90B

Table 3. Power and Ground Pins

PLCC Pin Number	QFP Pin Number	Signal	Description
15, 48	9, 49	V _{DD}	+5 Volt power input.
16, 21, 27, 32, 38, 64	10, 11, 16, 22, 23, 29 30, 37, 64	V _{SS}	Ground. NCR recommends a ground plane be used.

Table 4. 53C90 and 53C90A only – PLCC Package

Pin	Signal	Type	Description
49	TGS	O	Active-high target group select signal. This pin is high whenever the ASP is in target mode. It is used in differential mode to enable the target signals (REQO/, MSGO/, C/DO, I/OO). When low, the ASP should be receiving these signals.
5	DIFFM	I	Differential mode enable. When this pin is grounded, the ASP operates in single-ended mode, with separate SCSI data input and output buses. When this pin is high, the ASP operates in differential mode, with bi-directional SCSI data on the SDI pins and active-high differential transceiver enables on the SDO pins.

Table 5. 53C90B only – PLCC Package

Pin	Signal	Type	Description
49	TGS/DIFFM	B	<p>This pin is sampled during reset to put the chip in differential mode or single-ended mode, it then switches function to become the TGS output. An internal pull-up pulls it high for single-ended mode, while an external 1K pull-down will place the C90B in differential mode.</p> <p>There are two kinds of reset, power-up reset and running-reset. On power-up, the state of this pin is sampled as the voltage on the power pins rises through 3 volts (approximately). A running-reset is a host hardware reset that occurs sometime later. A running-reset will sample this pin until the 10th rising edge of CLK after RESET goes true (high). This is enough time for the internal pull-up to pull it high. Whatever state it's in on the 10th clock will determine the mode. The TGS function will be disabled until the 12th rising edge of CLK after RESET first goes true and RESET is false.</p> <p>When RESET is false (low), and 12 clocks have occurred since RESET first went true, this pin becomes the TGS output, an active-high target group select signal. This pin is high whenever the ASP is in target mode. It is used in differential mode to enable the target signals (REQO/, MSGO/, CDO/, IOO/). When low, the ASP should be receiving these signals.</p>
5	DBP	B	Active-high data bus parity for host, DMA, and memory data bus. Four bits in Config 1 and Config 2 control parity generation and checking.

Differences from 53C90

- Supports three-byte message exchange SCSI-2 tagged-queuing
- Added select with ATN3 command
- Added target DMA abort command
- Added interrupt polling bit
- Added second configuration register
- Improved immunity to cable impedance mismatches and improper termination
- Tri-state DMA request output
- Cut leakage current on SCSI input pins when powered off
- Relaxed register timings
- Relaxed DMA timings
- Relaxed CLK duty cycle
- Lengthened read data access time
- NOP required less often

Functional Description

The ASC SCSI data bus has a set of inputs and a set of outputs. This allows the ASC to be used in either single-ended mode or differential mode. In single-ended mode, the inputs are usually connected to the outputs on the circuit board. In differential mode, the SDI (SCSI Data Input) pins become bi-directional data pins, while the SDO (SCSI Data Output) pins become enable signals for the differential transceivers. Separate enables are required, because during arbitration two data bus signals must be outputs while the other six must be inputs. Two signals, TGS and IGS, control the direction of the external transceivers, allowing the ASC to dynamically switch between initiator and target roles.

The ASC has a command set that allows it to perform common SCSI sequences at hardware speed without host intervention. Its on-chip FIFO may be accessed simultaneously by the SCSI bus and either the host processor or the host DMA controller. All command, data, status, and message bytes pass through the FIFO on their way to or from the SCSI bus. Most ASC commands have two versions: DMA and non-DMA. When DMA instructions are used, data will pass between memory and the SCSI bus with the FIFO acting as temporary storage when the DMA channel is temporarily shut down by a higher priority event such as DRAM refresh.

The FIFO also helps speed execution during non-DMA transfers. For example, in initiator role, the host processor will load the CDB (Command Descriptor Block) and optionally one or three message bytes into the FIFO, issue one of several selection commands and wait for an interrupt. The ASC will wait for bus-free, arbitrate for the bus again and again until it acquires it, send the message bytes followed by the CDB, then generate an interrupt. Meanwhile, a multi-tasking host may continue with other tasks.

In target role, the host processor will enable selection, then wait for an interrupt. Eventually, an initiator will select the ASC and will then automatically step through the arbitration, selection, and command phases before generating an interrupt. When the interrupt occurs, the entire command descriptor block will be in the FIFO along with any message bytes sent by the initiator. Combination commands such as these, are identified with the *sequence* suffix in the *Table of ASC Commands*.

NCR 53C90A, 53C90B

After selection phase has been successfully completed, the ASC may transfer bytes in any of the SCSI information phases whether operating in initiator or a target role. The ASC supports disconnect/reselect in both initiator and target roles, making high performance multi-threaded systems easy to implement.

The ASC may transfer data phase bytes across the bus synchronously, at speeds up to 5 MB/S, or asynchronously at speeds up to 6 MB/S. Refer to *Data Transfer Rate*. The difference between the two is transparent to the user except that the synchronous offset and the synchronous transfer period registers must be programmed prior to synchronous data transfer. The default, after hardware or software reset, is asynchronous transmission.

Data phase bytes will usually be transferred using DMA. The host processor will program an external DMA controller, program the ASC transfer count, issue an ASC data transfer command (there are several), then wait for an interrupt. The DMA controller and the ASC will transfer all the data without host processor intervention.

To end the SCSI transaction, the ASC target will place a status byte and a message byte in the FIFO, then issue a single command (there are two to choose from) which will cause the ASC to first assert status phase, send the first byte, assert message in phase, send the second byte, disconnect from the SCSI bus (after the initiator releases ACK (Acknowledge)) and interrupt the host processor.

The end of a SCSI transaction is similar for an ASC initiator except that it receives two bytes into its FIFO. The initiator prevents the target from disconnecting by holding ACK asserted on the bus while the host processor examines the status and message bytes. If both bytes are good, the message accepted command is used to instruct the ASC to release ACK, which allows the target to disconnect which causes the initiator to interrupt its host and report the disconnect. If the status and message bytes are not good, the host should first issue the set ATN (Attention) command before issuing the message accepted command. This instructs the ASC to assert ATN before releasing ACK, which should cause the target to request message out phase rather than disconnect.

Bus Initiated Sequences

- Selection
- Reselection
- SCSI bus reset

Selection or reselection sequences occur in the disconnected state when the ASC is selected or reselected by another initiator or target, if the enable selection or reselection command had previously been received by the ASC.

In addition to responding to bus initiated events, the ASC may initiate a bus event by using one of several selection or reselection commands. If one of these commands starts executing, *it will clear enable selection/reselection* after arbitration has been won. Normally the host processor will have 250 ms (ANSI recommended selection time-out period) after the chip disconnects from the bus to re-enable bus initiated events. If the time-out is exceeded, an initiator or target which is attempting to connect to the ASC, may time-out and abort.

If, on the other hand, the bus initiated event occurs before the command starts executing, the FIFO and command register will be cleared, and any further writes by the host processor will be ignored until the interrupt register is read. Since a selection/reselection command requires that something be placed in the FIFO, these bytes will be lost, as will any command written to the command register. The interrupt handler that services a selection/reselection command will have to examine the bits in the interrupt register to determine if the ASC selected another device, or if it was selected by another device. The former case will cause a function complete interrupt, the latter case will cause a selection/reselection interrupt.

Table 6. ASC Register Set

Address (hex)	Read	Write
0	Transfer counter LSB	Transfer count LSB
1	Transfer counter MSB	Transfer count MSB
2	FIFO	FIFO
3	Command	Command
4	Status	Destination bus ID
5	Interrupt	Select/reselect timeout
6	Sequence step	Synchronous period
7	FIFO flags/sequence step	Synchronous offset
8	Configuration 1	Configuration 1
9	NCR reserved	Clock conversion factor
A	NCR reserved	Test mode
B	Configuration 2	Configuration 2

Register Set

Some ASC registers have different meanings during reads than writes. When CS/ is true, the register being accessed is determined by either RD/ or WR/ together with the address pins A0-3. The FIFO may be accessed using either CS/ or DACK/ together with RD/ or WR/. Address pins A0-A3 are ignored when DACK/ is active, but must be driven when CS/ is active.

Transfer Count (Write address 0, 1)

These two registers together form a 16-bit transfer count for DMA operations. Transfer count specifies the number of bytes that are to be transferred over the SCSI bus. Values written to these two registers will be stored internally and loaded into the transfer counter by any DMA command. These values remain unchanged while the transfer counter decrements. Thus, successive blocks of equal size may be transferred without reprogramming the count. They may be reprogrammed any time after the previous DMA operation has started, whether it has finished or not. Zero specifies a maximum length count (65536). These registers are not changed by any reset; their states are unpredictable after power-up.

Transfer Counter (Read address 0, 1)

A read from these two addresses will return the value currently in the counter. DMA commands use the counter to terminate a transfer. Any DMA command will load count into the counter. A DMA NOP 80h will load the counter while the non-DMA NOP 00h will not.

With one exception, non-DMA commands do not use the counter. The exception is when the ASC has been selected, it decodes the group code field of the CDB (Command Descriptor Block), loads the counter with the number of bytes in the CDB, then decrements once for every byte received.

The transfer counter decrements on the leading edge of:

Target	Decrement by
Data in phase	DACK/
Data out phase	REQO/

Initiator	Decrement by
Synchronous data in	DACK/
Asynchronous data in	ACKO/
Data out	DACK/

NCR 53C90A, 53C90B

Note that DACK/ can decrement the counter even if RD/ or WR/ do not go true. False DACK/s can cause the counter to get out of sync with the data stream, leading to subtle errors that are difficult to trace. When false DACK/s are expected to interfere with a temporarily suspended DMA operation, the DREQ Hi-/ bit in Config 2 should be set.

FIFO Register (Read/write address 02)

The FIFO is a 16 by 9-bit first-in-first-out buffer between the SCSI bus and memory. It is accessible by the host processor at this address. It is also accessible by an external DMA controller and by the SCSI bus. The DMA may access the FIFO by asserting DACK/ together with either RD/ or WR/. When accessed by CS/, the address bits must be valid. When accessed by DACK/, the address bits are ignored. The bottom FIFO element and the FIFO flags are initialized to zero during hardware reset, software reset chip and at the beginning of bus initiated selection or reselection. The contents of the rest of the FIFO are not changed by any reset, but when the flags are zero, successive FIFO reads will always access the bottom register.

Command Register (Read/write address 03)

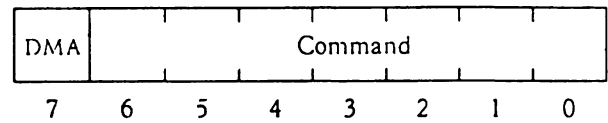
The command register is a two deep 8-bit read/write register used to give commands to the ASC. Up to two commands may be stacked in the command register. The second command may be written before the ASC completes (or even starts) the first. Reset chip, reset SCSI bus and target stop DMA execute immediately, all others wait for the previous command to complete. The last executed (or executing) command will remain in the command register and may be read by the host processor. Reading the command register has no effect on its contents. The command register will be cleared by any of the following conditions:

- Hardware, software or SCSI bus reset
- SCSI bus disconnect
- Bus-initiated selection or reselection
- Select command
- Reconnect command if ATN is set
- Select or reselect time-out
- Target terminate command
- Parity error detected in target mode

- Assertion of ATN in target mode
- Any phase change in initiator mode
- Illegal command

If two commands are placed in the command register, two interrupts may result. If the first interrupt is not serviced before the second finishes, the second interrupt is stacked behind the first. When the interrupt register is read by the host to service the first interrupt, the contents status register, sequence step register, and interrupt register will change to describe the second interrupt.

Figure 5. Command Register (Read/write address 03)



Bit 7 (Enable DMA)

When bit 7 is not set, the command is a non-DMA instruction. When it is set, the command is a DMA instruction. DMA instructions will load the internal byte counter with the value in the transfer count register (without changing the count register) then transfer data until that count decrements to zero. If the transfer terminates prematurely, the bits in the status, sequence step, and interrupt registers will indicate why.

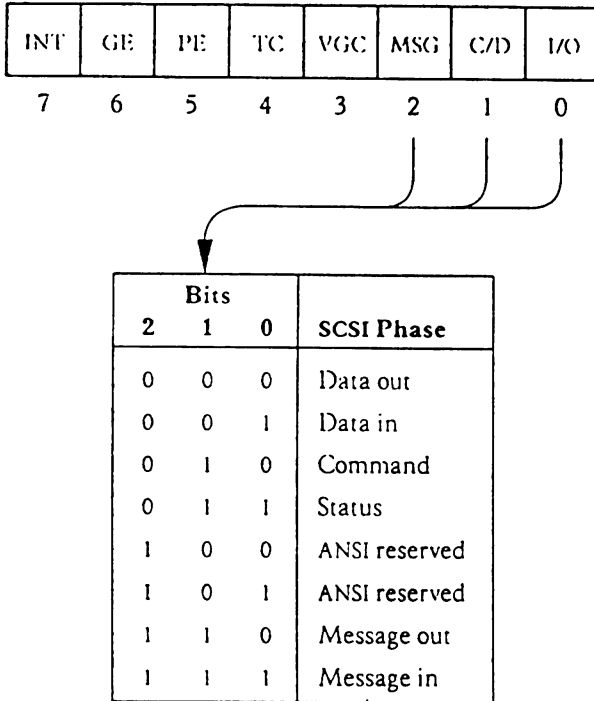
Bits 6-0 (Command code)

The ASC commands are shown in Table 19. Bits 4, 5 and 6 specify a mode group. Commands from the miscellaneous group may be issued at any time. Commands from the disconnected, target or initiator groups will only be accepted by the ASC if it is in the same mode as the command when it falls to the bottom of the command FIFO. Otherwise, an illegal command interrupt will be generated. For example, after hardware or software reset, the ASC will be in the disconnected state. A command from either the target group or the initiator group will cause an illegal command interrupt. An enable selection or reselection command by itself will not change modes. However, if another SCSI device then selects the ASC, it will be in the target state; if another device reselects the ASC, it will then be in the initiator state. Similarly, any select command will place the ASC in initiator mode, while the reselect sequence command will place the ASC in target mode.

Status Register (Read address 04)

The status register contains important flags that indicate various conditions. All but the phase bits are latched. The phase bits are live indicators of the state of the SCSI bus. All the latched bits except the terminal count are cleared by reading the interrupt register.

Figure 6. Status Register (Read address 04)



Bit 7 (Interrupt)

This bit is set whenever the ASC drives the INT output true. It may be polled. It is buffered from the actual output, so that in wired-OR (shared interrupt) designs, this bit will indicate whether the ASC is attempting to interrupt the host processor. This bit is reserved by NCR in the 53C90. Hardware reset or software reset chip or a read from the interrupt register will release an active INT signal and also clear this bit.

Bit 6 (Gross error)

This bit is set when one of the following has occurred:

- The top of the FIFO is overwritten
- The top of the command register has been overwritten
- Direction of DMA transfer is opposite to the direction of the SCSI transfer
- An unexpected phase change in initiator role during synchronous data phase

Gross error does not cause an interrupt, it may be detected only while servicing another interrupt. The bit is cleared by reading the status register if the interrupt output is asserted. It will also be cleared by hardware reset, or software reset chip (but not SCSI reset).

Bit 5 (Parity error)

This bit will be set if parity checking is enabled in the Config 1 register and the ASC detects a SCSI parity error on incoming command, data, status or message bytes. It will be cleared by reading the interrupt register if the interrupt output is asserted. Hardware reset or software reset chip will clear this bit (but not SCSI reset).

Bit 4 (Terminal count)

This bit is set when the transfer counter decrements to zero. It resets when the transfer count is loaded. Since a DMA NOP 80h command will load the transfer counter, it will also clear this bit. Note that a non-DMA NOP 00h will not load the counter and will not clear this bit. Reading the interrupt register will not clear this bit. Hardware reset or software reset chip will clear it (but not SCSI reset).

Bit 3 (Valid group code)

The name of this bit has changed from transfer complete in the 53C90 to valid group code in the 53C90A and 53C90B; but its function remains the same.

When the ASC is selected, it decodes the group code field in the first byte of the command descriptor block. If the group code matches one defined in ANSI X3.131-1986, this bit will be set. An undefined group code (designated reserved by the ANSI committee) leaves it not set. If the SCSI-2 bit is set in the Config 2 register, group 2 commands will be recognized as ten-byte commands and the bit will be set. If the SCSI-2 bit is cleared, group 2 commands will be treated as reserved commands. Groups 3 and 4 are always treated as reserved commands. A reserved group command will cause the ASC to request 6 command bytes. The ASC recognizes group 6 as six-byte vendor unique commands and group 7 as 10-byte vendor unique commands. The valid group code bit will be cleared by reading the interrupt register if the interrupt output is asserted. It will also be cleared by hardware reset or software reset chip (but not by SCSI reset).

NCR 53C90A, 53C90B

DC Electrical Characteristics

Absolute Maximum Stress Ratings

Parameter	Symbol	Pins	Test Conditions	Min	Max	Unit
Storage temperature	T_{STG}	-	-	-55	150	°C
Supply voltage	V_{DD}	-	-	-0.5	7.0	V
Input voltage	V_{IN}	-	-	$V_{SS}-0.5$	$V_{DD}+0.5$	V
Latch-up current	I_{LUP}	-	$-2V < V_{IN} < +8V$	±100	-	mA
Electrostatic discharge	ESD	-	Human body model	-	-	-
		SCSI pins	100 pF at 1.5K ohms	±6000	-	V
		Other pins	100 pF at 1.5K ohms	±3000	-	V

Conditions that exceed the absolute maximum stress limits may destroy the device.

Conditions that exceed the operating limits may cause the device to function incorrectly.

Operating Conditions

Parameter	Symbol	Pins	Test Conditions	Min	Max	Unit
Supply voltage	V_{DD}	-	-	4.75	5.25	V
Supply current	I_{DD}	-	Static*	-	10	mA
Supply current	I_{DD}	-	Dynamic	-	50	mA
Ambient temperature	T_A	-	-	0	70	°C

* Static means: all inputs at V_{SS} , all outputs floating, and all bi-directional pins configured as inputs.

Inputs

Parameter	Symbol	Pins	Test Conditions	Min	Max	Unit
Input high voltage	V_{IH}	-	-	2.0	$V_{DD}+0.5$	V
Input low voltage	V_{IL}	-	-	$V_{SS}-0.5$	0.8	V
Input leakage current	I_{IN}	Non-SCSI	$0 < V_{IN} < V_{DD}$ $4.75 \leq V_{DD} \leq 5.25$	-10	10	µA
Hysteresis	V_{H}	BSYI/, SELI/, REQI/, ACKI/, RSTI/	-	400	-	mV
Input low leakage	I_{IL}	SCSI	$V_{IN} = 0.5; 0 \leq V_{DD} \leq 5.5$	-10	0.0	µA
Input high leakage	I_{IH}	SCSI	$V_{IN} = 2.7; 0 \leq V_{DD} \leq 5.5$	0.0	10	µA
Capacitance	C_{IN}	-	-	-	10	pF

Outputs

Parameter	Symbol	Pins	Test Conditions	Min	Max	Unit
Output high voltage	V_{OH}	DREQ, IGS,	$I_{OH} = -2 \text{ mA}$	2.4	V_{DD}	V
Output low voltage	V_{OL}	DREQ, IGS, INT/	$I_{OL} = 4 \text{ mA}$	V_{SS}	0.4	V
Output high voltage	V_{OH}	RESTO	$I_{OH} = -4 \text{ mA}$	2.4	V_{DD}	V
Output low voltage	V_{OL}	RSTO/, SEL0, ATNO/, MSGO/, ACKO/, REQO/, SDOP/, BSYO/, C/D, I/O, SDO7-0	$I_{OL} = 48 \text{ mA}$	V_{SS}	0.5	V
Hi Z state leakage	I_{OZ}	-	$0 < V_{OUT} < V_{DD}$	-10	10	μA
Capacitance	C_{OUT}	-	-	-	10	pF

Bi-Directional Pins

Parameter	Symbol	Pins	Test Conditions	Min	Max	Unit
Input high voltage	V_{IH}	-	-	2.0	$V_{DD} + 0.5$	V
Input low voltage	V_{IL}	-	-	$V_{SS} - 0.5$	0.8	V
Output high voltage	V_{OH}	SDI7-0, DBI5-0, DBP1-0, PAD7-0	$I_{OH} = -2 \text{ mA}$	2.4	V_{DD}	V
Output low voltage	V_{OL}	SDI7-0, DBI5-0, DBP1-0, PAD7-0	$I_{OL} = 4 \text{ mA}$	V_{SS}	0.4	V
Output high voltage	V_{OH}	TGS	$I_{OH} = -4 \text{ mA}$	2.4	V_{DD}	V
Output low voltage	V_{OL}	TGS	$I_{OL} = -8 \text{ mA}$	V_{SS}	0.4	V
Input current, low	I_{IL}	TGS	-	-600	0	μA
Input current, low	I_{IN}	SDI7-0	$0 < V_{IN} < V_{DD}$	-10	10	μA
Input current, low	I_{IL}	DBI5-0, DBP1-0, PAD7-0	$V_{IN} = V_{IL}$	-400	0	μA
Input current, high	I_{IH}	DBI5-0, DBP1-0, PAD7-0, TGS	$V_{IN} = V_{IH}$	0	10	μA
Hi Z pull-up current	I_{PU}	DBI5-0, DBP1-0, PAD7-0	-	100	400	μA
Capacitance	C_{IO}	-	-	-	10	pF

NCR 53C90A, 53C90B

AC Electrical Characteristics

The AC characteristics described herein apply over the voltage range $V_{DD} = 4.75 - 5.25$ V and the temperature range $0^\circ - 70^\circ$ C. Chip output timing is based on simulation under

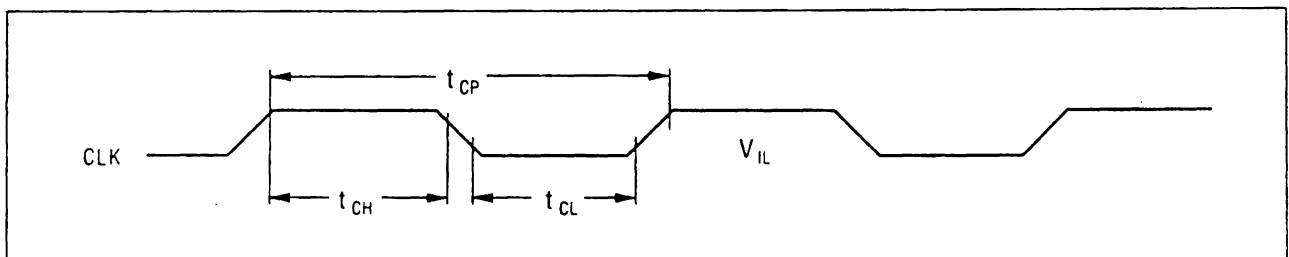
worst case conditions (4.75 V, 70° C) and the following pad termination:

Signal Name	Output Load
RESET0, DREQ, TGS, IGS, SDIP/, SDI7/-SDIO/	50 pF
DB7-0	85 pF
INT/	50 pF; 1K pull-up
RST0/, SEL0/, BSY0/, ATNO/, MSG0/, CDO/ IO0/, REQ0/, ACK0/, SDO7-0/, SDOP	200 pF; 110 pullup, 165 pulldown

System Interface

All timings in this specification are taken from the 10% and 90% points with respect to the specified V_{OL} and V_{OH} of the waveforms.

Clock



Parameter	Symbol	Minimum	Maximum	Units
Clock period	t_{CP}	40	100	ns
Clock frequency, asynchronous	t_{CPA}	10*	25	MHz
Clock frequency, synchronous	t_{CPS}	12*	25	MHz
Clock high	t_{CH}	14.58	-	ns
Clock low	t_{CL}	14.58	-	ns
Synchronization latency = $t_{CP} + t_{CL}$	t_{CS}	-	-	-

- * These minimum numbers required to comply with ANSI SCSI specification. For synchronous SCSI data transfers, the clock inputs must also meet the following requirements:

$$2t_{CP} \cdot t_{CL} \geq 9792 \text{ ns} \quad \text{and} \quad 2t_{CP} + t_{CH} \geq 9792 \text{ ns}$$



Am7990

Local Area Network Controller for Ethernet (LANCE)

DISTINCTIVE CHARACTERISTICS

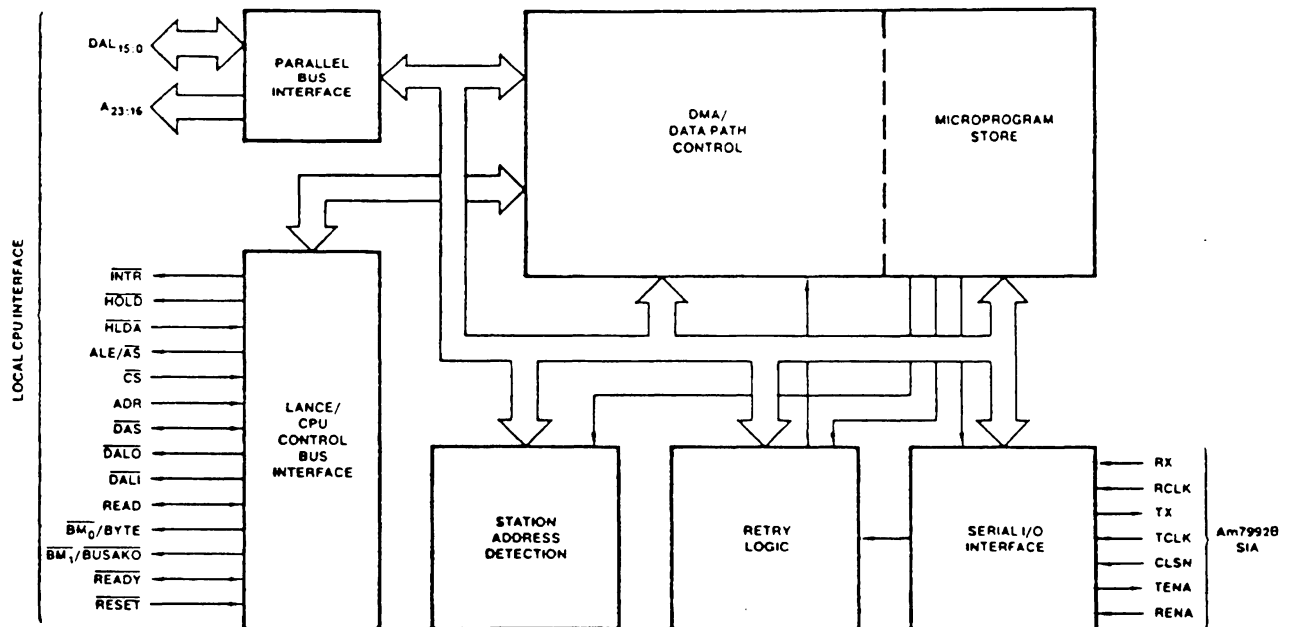
- Compatible with Ethernet and IEEE-802.3 10Base5 Type A, and 10Base2 Type B, "Cheapernet")
- Easily interfaced to 8086, 68000, Z8000™, LSI-II™ microprocessors
- On-board DMA and buffer management, 48 byte FIFO
- 24-bit wide linear addressing (Bus Master Mode)
- Network and packet error reporting
- Back-to-back packet reception with as little as 4.1 μ sec interpacket gap time
- Diagnostic Routines
 - Internal/external loop back
 - CRC logic check
 - Time domain reflectometer

GENERAL DESCRIPTION

The Am7990 Local Area Network Controller for Ethernet (LANCE) is a 48-pin VLSI device designed to greatly simplify interfacing a microcomputer or minicomputer to an IEEE-802.3/Ethernet Local Area Network. The LANCE, in conjunction with the Am7992B Serial Interface Adapter (SIA), Am7996 Transceiver, and closely coupled local memory and microprocessor, is intended to provide the

user with a complete interface module for an Ethernet network. The Am7990 is designed using a scaled N-Channel MOS technology and is compatible with a variety of microprocessors. On-board DMA, advanced buffer management, and extensive error reporting and diagnostics facilitate design and improve system performance.

BLOCK DIAGRAM

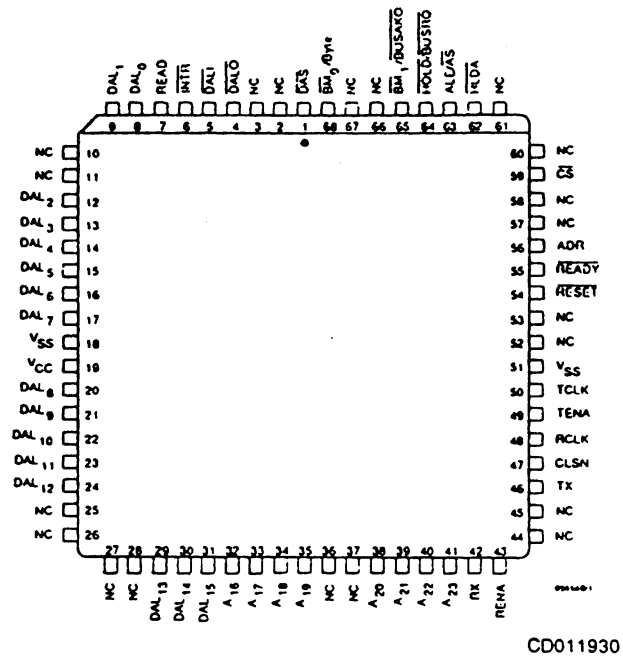
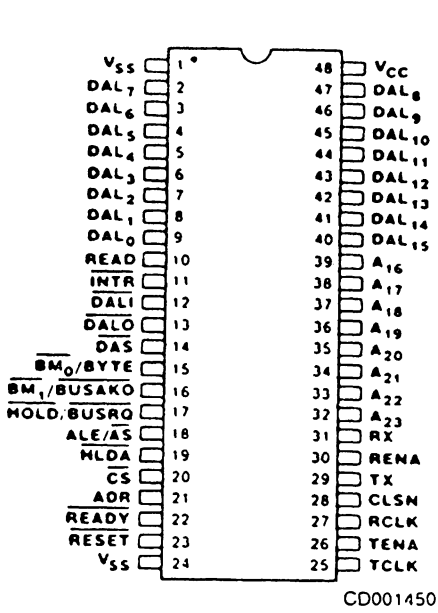


BD002063

RELATED AMD PRODUCTS

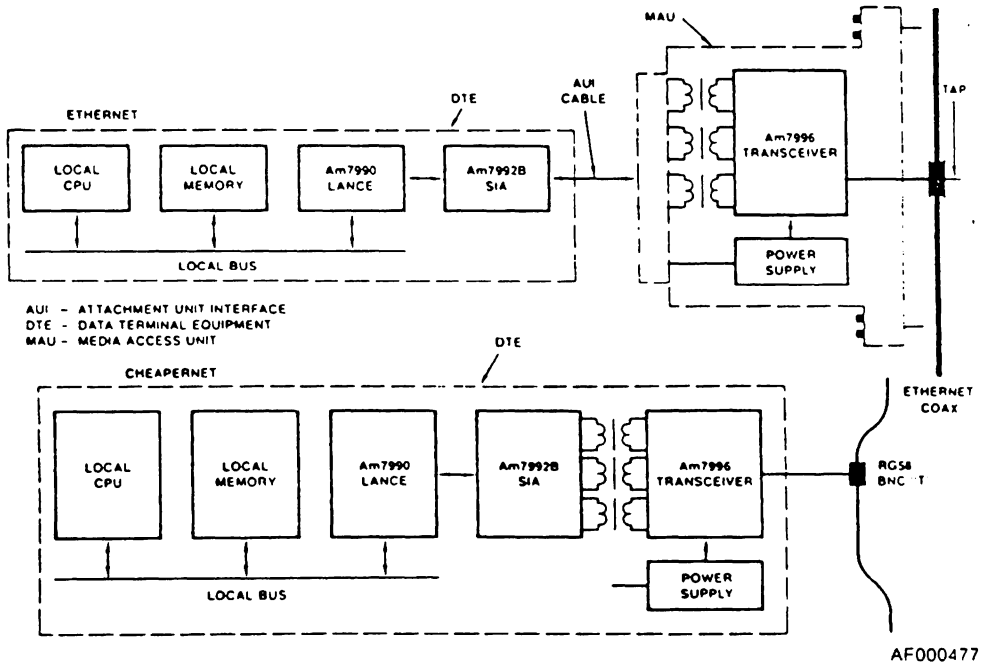
Part No.	Description
Am7992B	Serial Interface Adaptor (SIA)
Am7996	IEEE-802.3/Ethernet/Cheapernet Transceiver
Am79C900	Integrated Local Area Communications Controller

CONNECTION DIAGRAMS Top View



Note: Pin 1 is marked for orientation.

TYPICAL ETHERNET/CHEAPERNET NODE



PIN DESCRIPTION

DAL₀₀ - DAL₁₅ Data/Address Lines (Input/Output, Three-State)

The time multiplexed Address/Data bus. During the address portion of a memory transfer, DAL₀₀ - DAL₁₅ contains the lower 16 bits of the memory address. The upper 8 bits of address are contained in A₁₆ - A₂₃.

During the data portion of a memory transfer, DAL₀₀ - DAL₁₅ contains the read or write data, depending on the type of transfer.

The LANCE drives these lines as a Bus Master and as a Bus Slave.

A₁₆ - A₂₃ High Order Address Bus (Output Three-State)

Additional address bits to access a 24-bit address. These lines are driven as a Bus Master only.

READ (Input/Output, Three-State)

Indicates the type of operation to be performed in the current bus cycle. This signal is an output when the LANCE is a Bus Master.

High - Data is taken off the DAL by the LANCE.

Low - Data is placed on the DAL by the LANCE.

The signal is an input when the LANCE is a Bus Slave.

High - Data is placed on the DAL by the LANCE.

Low - Data is taken off the DAL by the LANCE.

\overline{BM}_0 /BYTE, \overline{BM}_1 / \overline{BUSAKO} (Output, Three-state)

The two pins are programmable through bit (00) of CSR₃.

\overline{BM}_0 , \overline{BM}_1 - If CSR₃ (00) BCON = 0

PIN 15 = \overline{BM}_0 (Output Three-state) (48-Pin DIPs)

PIN 16 = \overline{BM}_1 (Output Three-state) (48-Pin DIPs)

\overline{BM}_0 , \overline{BM}_1 (Byte Mask). This indicates the byte(s) on the DAL are to be read or written during this bus transaction. The LANCE drives these lines only as a Bus Master. It ignores the Byte Mask lines when it is a Bus Slave and assumes word transfers.

Byte selection using Byte Mask is done as described by the following table.

\overline{BM}_1	\overline{BM}_0	
LOW	LOW	Whole Word
LOW	HIGH	Upper Byte
HIGH	LOW	Lower Byte
HIGH	HIGH	None

BYTE, \overline{BUSAKO} - If CSR₃ (00) BCON = 1

PIN 15 = BYTE (Output Three-state) (48-Pin DIPs)

PIN 16 = \overline{BUSAKO} (Output) (48-Pin DIPs)

Byte selection may also be done using the BYTE line and DAL₀₀ line, latched during the address portion of the bus cycle. The LANCE drives BYTE only as a Bus Master and ignores it when a Bus Slave selection is done (similar to \overline{BM}_0 , \overline{BM}_1).

Byte selection is done as outlined in the following table.

BYTE	DAL ₀₀	
LOW	LOW	Whole Word
LOW	HIGH	Illegal Condition
HIGH	LOW	Lower Byte
HIGH	HIGH	Upper Byte

\overline{BUSAKO} is a bus request daisy chain output. If the chip is not requesting the bus and it receives HLDA, \overline{BUSAKO} will

be driven LOW. If the LANCE is requesting the bus when it receives HLDA, \overline{BUSAKO} will remain HIGH.

Byte Swapping

In order to be compatible with the variety of 16-bit microprocessors available to the designer, the LANCE may be programmed to swap the position of the upper and lower order bytes on data involved in transfers with the internal FIFO.

Byte swapping is done when BSWP = 1. The most significant byte of the word in this case will appear on DAL lines 7-0 and the least significant byte on DAL lines 15-8.

When BYTE = H (indicating a byte transfer) the table indicates on which part of the 16-bit data bus the actual data will appear.

Whenever byte swap is activated, the only data that is swapped is data traveling to and from the FIFO.

Signal Line	Mode Bits	
	BSWP = 0 and BCON = 1	BSWP = 1 and BCON = 1
BYTE = L and DAL ₀₀ = L	Word	Word
BYTE = L and DAL ₀₀ = H	Illegal	Illegal
BYTE = H and DAL ₀₀ = H	Upper Byte	Lower Byte
BYTE = H and DAL ₀₀ = L	Lower Byte	Upper Byte

\overline{CS} Chip Select (Input)

Indicates, when asserted, that the LANCE is the slave device of the data transfer. \overline{CS} must be valid throughout the data portion of the bus cycle. \overline{CS} must not be asserted when HLDA is LOW.

ADR Register Address Port Select (Input)

When LANCE is slave, ADR indicates which of the two register ports is selected. ADR LOW selects register data port; ADR HIGH selects register address port. ADR must be valid throughout the data portion of the bus cycle and is only used by the LANCE when \overline{CS} is LOW.

ALE/ \overline{AS} Address Latch Enable (Output, Three-State)

Used to demultiplex the DAL lines and define the address portion of the bus cycle. This I/O pin is programmable through bit (01) of CSR₃.

As ALE (CSR₃ (01), ACON = 0), the signal transitions from a HIGH to a LOW during the address portion of the transfer and remains LOW during the data portion. ALE can be used by a Slave device to control a latch on the bus address lines. When ALE is HIGH, the latch is open, and when ALE goes LOW, the latch is closed.

As \overline{AS} (CSR₃ (01), ACON = 1), the signal pulses LOW during the address portion of the bus transaction. The LOW-to-HIGH transition of \overline{AS} can be used by a Slave device to strobe the address into a register.

The LANCE drives the ALE/ \overline{AS} line only as a Bus Master.

\overline{DAS} Data Strobe (Input/Output Three-State)

Defines the data portion of the bus transaction. \overline{DAS} is high during the address portion of a bus transaction and low during the data portion. The LOW-to-HIGH transition can be

used by a Slave device to strobe bus data into a register. \overline{DAS} is driven only as a Bus Master.

\overline{DALO} Data/Address Line Out (Output, Three-State)

An external bus transceiver control line. \overline{DALO} is asserted when the LANCE drives the DAL lines. \overline{DALO} will be LOW only during the address portion of a READ transfer. It will be LOW for the entire transfer if the transfer is a WRITE. \overline{DALO} is driven only when LANCE is a Bus Master.

\overline{DALI} Data/Address Line In (Output, Three-State)

An external bus transceiver control line. \overline{DALI} is asserted when the LANCE reads from the DAL lines. It will be LOW during the data portion of a READ transfer and remain HIGH for the entire transfer if it is a WRITE. \overline{DALI} is driven only when LANCE is a Bus Master.

$\overline{HOLD}/\overline{BUSRQ}$ Bus Hold Request (Output, Open Drain)

Asserted by the LANCE when it requires access to memory. \overline{HOLD} is held LOW for the entire ensuing bus transaction. The function of this pin is programmed through bit (00) of CSR_3 . Bit (00) of CSR_3 is cleared when \overline{RESET} is asserted.

When CSR_3 (00) $BCON = 0$

PIN 17 = \overline{HOLD} (Output Open Drain and input sense) (48-Pin DIPs)

When CSR_3 (00) $BCON = 1$

PIN 17 = \overline{BUSRQ} (I/O Sense, Open Drain) (48-Pin DIPs)

If the LANCE wants to use the bus, it looks at $\overline{HOLD}/\overline{BUSRQ}$; if it is HIGH the LANCE can pull it LOW and request the bus. If it is already LOW, the LANCE waits for it to go inactive-HIGH before requesting the bus.

\overline{HLDA} Bus Hold Acknowledge (Input)

A response to \overline{HOLD} . When \overline{HLDA} is LOW in response to the chip's assertion of \overline{HOLD} , the chip is the Bus Master.

During bus master operation the LANCE waits for \overline{HLDA} to be deasserted 'HIGH' before reasserting \overline{HOLD} 'LOW'. This insures proper bus handshake under all situations.

\overline{INTR} Interrupt (Output Open Drain)

An attention signal that indicates, when active, that one or more of the following CSR_0 status flags is set: $BABL$, $MERR$, $MISS$, $RINT$, $TINT$ or $IDON$. \overline{INTR} is enabled by bit 06 of CSR_0 ($INEA = 1$). \overline{INTR} remains asserted until the source of Interrupt is removed.

RX Receive (Input)

Receive Input Bit Stream.

TX Transmit (Output)

Transmit Output Bit Stream.

TENA Transmit Enable (Output)

Transmit Output Bit Stream enable. When asserted, it enables valid transmit output (TX).

RCLK Receive Clock (Input)

A 10-MHz square wave synchronized to the Receive data and only active while receiving an Input Bit Stream.

CLSN Collision (Input)

A logical input that indicates that a collision is occurring on the channel.

RENA Receive Enable (Input)

A logical input that indicates the presence of carrier on the channel.

TCLK Transmit Clock (Input)

10-MHz clock.

\overline{READY} (Input/Output, Open Drain)

When the LANCE is a Bus Master, \overline{READY} is an asynchronous acknowledgement from the bus memory that it will accept data in a WRITE cycle or that it has put data on the DAL lines in a READ cycle.

As a Bus Slave, the LANCE asserts \overline{READY} when it has put data on the DAL lines during a READ cycle or is about to take data off the DAL lines during a write cycle. \overline{READY} is a response to \overline{DAS} and will return High after \overline{DAS} has gone High. \overline{READY} is an input when the LANCE is a Bus Master and an output when the LANCE is a Bus Slave.

\overline{RESET} Reset (Input)

Bus Request Signal. Causes the LANCE to cease operation, clear its internal logic, and enter an Idle state with the stop bit of CSR_0 set. It is recommended that a 3.3 k Ω pullup register be connected to this pin.

V_{CC} Power supply pin +5 volts $\pm 5\%$.

It is recommended that a 0.1- μF and a 10- μF decoupling capacitor be used between V_{CC} and V_{SS} .

V_{SS} Ground.

Pin 1 and 24 (48-Pin DIPs) should be connected together externally, as close to the chip as possible.

PROGRAMMING

This section defines the control and Status Registers and the memory data structures required to program the Am7990 (LANCE).

Programming the Am7990 (LANCE)

The Am7990 (LANCE) is designed to operate in an environment that includes close coupling with a local memory and a microprocessor (HOST). The Am7990 LANCE is programmed by a combination of registers and data structures resident within the LANCE and in memory. There are four Control and Status Registers (CSRs) within the LANCE which are programmed by the HOST device. Once enabled, the LANCE has the ability to access memory locations to acquire additional operating parameters.

The Am7990 has the ability to do independent buffer management as well as transfer data packets to and from the Ethernet. There are three memory structures accessed by the Chip:

1. Initialization Block – 12 words in contiguous memory starting on a word boundary. It also contains the operating parameters necessary for device operation. The initialization block is comprised of:

- Mode of Operation
- Physical Address
- Logical Address Mask
- Location to Receive and Transmit Descriptor Rings
- Number of Entries in Receive and Transmit Descriptor Rings

2. Receive and Transmit Descriptor Rings – Two ring structures, one each for incoming and outgoing packets. Each entry in the rings is 4 words long and each entry must start on a quadword boundary. The Descriptor Rings are comprised of:

- The address of a data buffer
- The length of that data buffer
- Status information associated with the buffer

3. Data Buffers – Contiguous portions of memory reserved for packet buffering. Data buffers may begin on arbitrary byte boundaries.

In general, the programming sequence of the LANCE may be summarized as:

1. Programming the LANCE's CSRs by a host device to locate an initialization block in memory. The byte control, byte addressing, and address latch enable modes are defined here also.
2. The LANCE loading itself with the information contained within the initialization block.
3. The LANCE accessing the descriptor rings for packet handling.

Control and Status Registers

There are four Control and Status Registers (CSRs) resident within the chip. The CSRs are accessed through two bus addressable ports, an address port (RAP) and a data port (RDP).

Accessing the Control and Status Registers

The CSRs are read (or written) in a two step operation. The address of the CSR to be accessed is written into the address

port (RAP) during a bus slave transaction. During a subsequent bus slave transaction, the data being read from (or written into) the data port (RDP) is read from (or written into) the CSR selected in the RAP.

Once written, the address in RAP remains unchanged until rewritten.

To distinguish the data port from the address port, a discrete I/O pin is provided.

ADR I/O Pin	Port
L	Register Data Port (RDP)
H	Register Address Port (RAP)

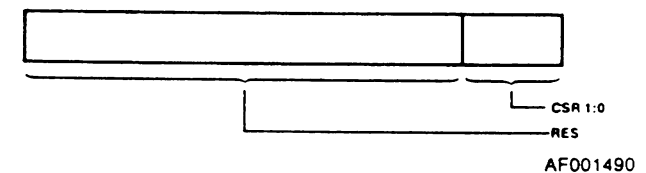
Register Data Port (RDP)



Bit	Name	Description
15:00	CSR Data	Writing data into RDP writes the data into the CSR selected in RAP. Reading the data from the RDP reads the data from the CSR selected in RAP. CSR ₁ , CSR ₂ and CSR ₃ are accessible only when the STOP bit of CSR ₀ is set.

If the STOP bit is not set while attempting to access CSR₁, CSR₂ or CSR₃, the LANCE will return READY, but a READ operation will return undefined data. WRITE operation is ignored.

Register Address Port (RAP)



Bit	Name	Description
15:02	RES	Reserved and read as zeroes.
01:00	CSR(1:0)	CSR address select. READ/WRITE. Selects the CSR to be accessed through the RDP. RAP is cleared by Bus RESET.

CSR(1:0)	CSR
00	CSR ₀
01	CSR ₁
10	CSR ₂
11	CSR ₃

Am7992B

Serial Interface Adapter (SIA)

Advanced
Micro
Devices

DISTINCTIVE CHARACTERISTICS

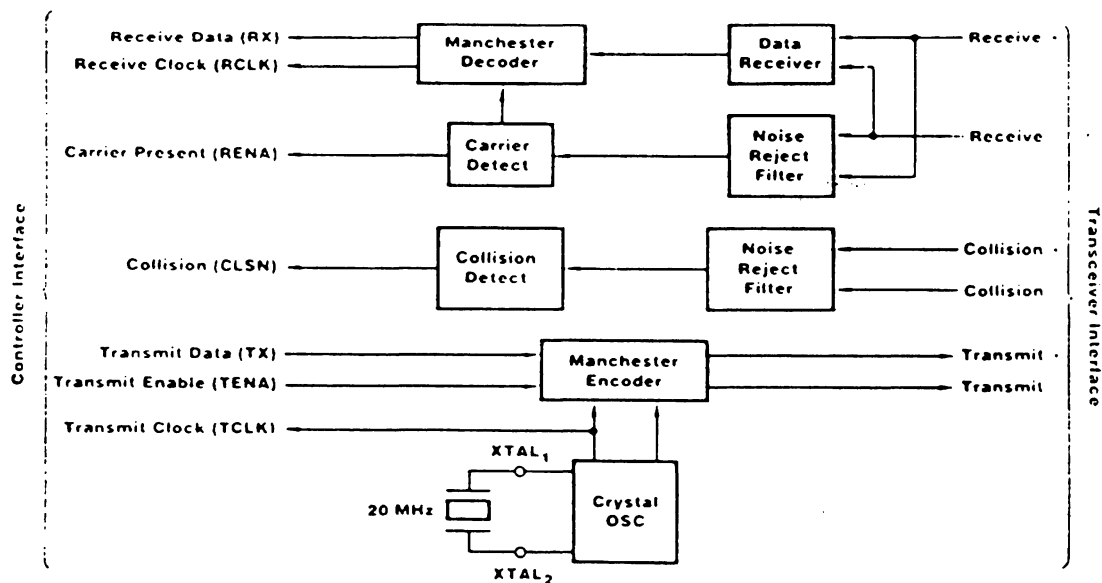
- Compatible with Ethernet/CheaperNet/IEEE-802.3 specifications
- Crystal/TTL oscillator controlled Manchester Encoder
- Manchester Decoder acquires clock and data within four bit times with an accuracy of ± 3 ns
- Guaranteed carrier and collision detection squelch threshold limits
 - Carrier/collision detected for inputs greater than -275 mV
 - No carrier/collision for inputs less than -175 mV
- Input signal conditioning rejects transient noise
 - Transients < 10 ns for collision detector inputs
 - Transients < 20 ns for carrier detector inputs
- Receiver decodes Manchester data with worst case ± 19 ns of clock jitter (at 10 MHz)
- TTL compatible host interface
- Transmit accuracy $\pm 0.01\%$ (without adjustments)

GENERAL DESCRIPTION

The Am7992B Serial Interface Adapter (SIA) is a Manchester Encoder/Decoder compatible with IEEE-802.3, Cheapernet and Ethernet specifications. In an IEEE-802.3/Ethernet application, the Am7992B interfaces the Am7990 Local Area Network Controller for Ethernet (LANCE) to the Ethernet transceiver cable, acquires clock and data within

four bit times, and decodes Manchester data with worst case ± 19 ns phase jitter at 10 MHz. SIA provides both guaranteed signal threshold limits and transient noise suppression circuitry in both data and collision paths to minimize false start conditions.

BLOCK DIAGRAM



BD002071

RELATED PRODUCTS

Part No.	Description
Am7990	Local Area Network Controller for Ethernet (LANCE)
Am7996	IEEE-802.3/Ethernet/CheaperNet/Transceiver
Am79C900	Integrated Local Area Communications Controller (ILACC)

PIN DESCRIPTION

CLSN Collision (Output, TTL Active HIGH)

Signals at the Collision \pm terminals meeting threshold and pulse width requirements will produce a logic HIGH at CLSN output. When no signal is present at Collision \pm , CLSN output will be LOW.

RX Receive Data (Output)

A MOS/TTL output, recovered data. When there is no signal at Receive \pm and $\overline{\text{TEST}}$ is HIGH, RX is HIGH. RX is actuated with RCLK and remains active until RENA is deasserted at the end of message. During reception, RX is synchronous with RCLK and changes after the rising edge of RCLK. When $\overline{\text{TEST}}$ is LOW, RX is enabled.

RENA Receive Enable (Output, TTL Active HIGH)

When there is no signal at Receive \pm RENA is LOW. Signals meeting threshold and pulse width "on" requirements will produce a logic HIGH at RENA. When RENA is HIGH, Receive \pm signals meeting threshold and pulse width "off" requirements will produce a LOW at RENA.

RCLK Receive Clock (Output)

A MOS/TTL output, recovered clock. When there is no signal at Receive \pm and $\overline{\text{TEST}}$ is HIGH, RCLK is LOW. RCLK is activated 1/4 bit time after the second negative Manchester preamble clock transition at Receive \pm , and remains active until end of message. When test is LOW, RCLK is enabled and meets minimum pulse width specifications.

TX Transmit (Input)

TTL-compatible input. When TENA is HIGH, signals at TX meeting setup and hold time to TCLK will be encoded as normal Manchester at Transmit+ and Transmit-.

TX HIGH: Transmit+ is negative with respect to Transmit- for first half of data bit cell.

TX LOW: Transmit+ is positive with respect to Transmit- for first half of data bit cell.

TENA Transmit Enable (Input)

TTL-compatible input. Active HIGH data encoder enable. Signals meeting setup and hold time to TCLK will allow encoding of Manchester data from TX to Transmit+ and Transmit-.

TCLK Transmit Clock (Output)

MOS/TTL output. TCLK provides symmetrical HIGH and LOW clock signals at data rate for reference timing of data to be encoded. It also provides clock signals for the controller chip (Am7990 - LANCE) and an internal timing reference for receive path voltage controlled oscillators.

Transmit+ Transmit (Outputs)

Transmit-

A differential line output. This line pair is intended to operate into terminated transmission lines. For signals meeting setup and hold time to TCLK at TENA and TX, Manchester clock and data are outputted at Transmit+ / Transmit-. When operating into a 78 Ω terminated transmission line, signaling meets the required output levels and skew for both Ethernet and IEEE-802.3 drop cables.

Receive+ Receiver (Inputs)

Receive-

A differential input. A pair of internally biased line receivers consisting of a carrier detect receiver with offset threshold and noise filtering to detect the line activity, and a data recovery receiver with no offset for Manchester data decoding.

Collision+ Collision (Inputs)

Collision-

A differential input. An internally biased line receiver input with offset threshold and noise filtering. Signals at Collision \pm have no effect on data-path functions.

TSEL Transmit Mode Select (Output, Open Collector; Input, Sense Amplifier)

TSEL LOW: Idle transmit state Transmit+ is positive with respect to Transmit-.

TSEL HIGH: Idle transmit state Transmit+ and Transmit- are equal, providing "zero" differential to operate transformer coupled loads.

When connected with an RC network, TSEL is held LOW during transmission. At the end of transmission the open collector output is disabled, allowing TSEL to rise and provide a smooth transmission from logic HIGH to "zero" differential idle. Delay and output return to zero are externally controlled by the RC network at TSEL and Transmit \pm load inductance.

X₁, X₂ Biased Crystal Oscillator (Input)

X₁ is the input and X₂ is the bypass port. When connected for crystal operation, the system clock which appears at TCLK is half the frequency of the crystal oscillator. X₁ may be driven from an external source of two times the data rate.

RF Frequency Setting Voltage Controlled Oscillator (V_{CO}) Loop Filter (Output)

This loop filter output is a reference voltage for the receive path phase detector. It also is a reference for timing noise immunity circuits in the collision and receive enable path. Nominal reference V_{CO} gain is 1.25 TCLK frequency MHz/V.

PF Receive Path V_{CO} Phase-Lock Loop Filter (Input)

This loop filter input is the control for receive path loop damping. Frequency of the receive V_{CO} is internally limited to transmit frequency $\pm 12\%$. Nominal receive V_{CO} gain is 0.25 reference V_{CO} gain MHz/V.

$\overline{\text{TEST}}$ Test Control (Input)

A static input that is connected to V_{CC} for Am7992B/Am7990 operation and to Ground for testing of Receive \pm path threshold and RCLK output high parameters. When $\overline{\text{TEST}}$ is grounded, RX is enabled and RCLK is enabled except during Clock acquisition when RCLK is HIGH.

GND₁ High Current Ground

GND₂ Logic Ground

GND₃ Voltage Controlled Oscillator Ground

V_{CC1} High Current and Logic Supply

V_{CC2} Voltage Controlled Oscillator Supply

FUNCTIONAL DESCRIPTION

The Am7992B Serial Interface Adapter (SIA) has three basic functions. It is a Manchester Encoder/line driver in the transmit path, a Manchester Decoder with noise filtering and quick lock-on characteristics in the receive path, and a signal detect/converter (10 MHz differential to TTL) in the collision

path. In addition, the SIA provides the interface between the TTL logic environment of the Local Area Network Controller for Ethernet (LANCE) and the differential signaling environment in the transceiver cable.



M5M44100J, L-8, -10

FAST PAGE MODE 4194304-BIT(4194304-WORD BY 1-BIT) DYNAMIC RAM

DESCRIPTION

This is a family of 4194304-word by 1-bit dynamic RAMs, fabricated with the high performance CMOS process, and ideal for large-capacity memory systems where high speed, low power dissipation, and low costs are essential. The use of quadruple-layer polysilicon process combined with silicide technology and a single-transistor dynamic storage stacked capacitor cell provide high circuit density at reduced costs. Multiplexed address inputs permit both a reduction in pins and an increase in system densities.

FEATURES

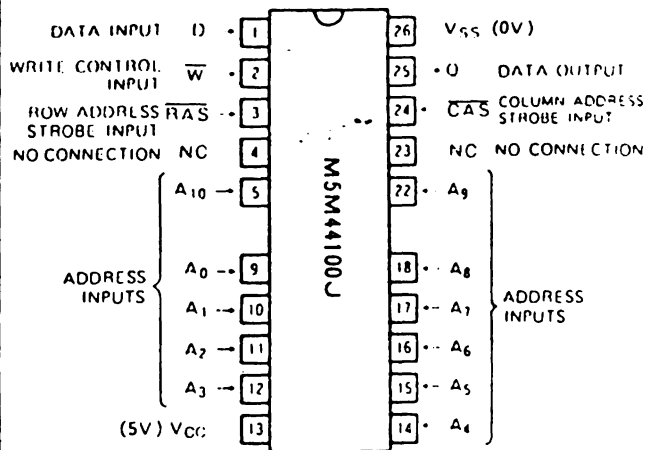
Device name	$\overline{\text{RAS}}$ access time (max ns)	$\overline{\text{CAS}}$ access time (max ns)	Address access time (max ns)	Cycle time (min ns)	Power dissipation (typ mW)
M5M44100 _L -8	80	20	40	160	415
M5M44100 _L -10	100	25	50	190	350

- Standard 26 pin SOJ, 20 pin ZIP
- Single 5V±10% supply
- Low standby power dissipation
5.5mW (Max) CMOS Input level
- Low operating power dissipation
M5M44100J, L-8 522.5mW (Max)
M5M44100J, L-10 467.5mW (Max)
- Fast-page mode (2048 bits random access), Read-modify-write, $\overline{\text{RAS}}$ -only refresh, $\overline{\text{CAS}}$ before $\overline{\text{RAS}}$ refresh, Hidden refresh capabilities
- Early-write operation gives common I/O capability
- All inputs, output TTL compatible and low capacitance
- 1024 refresh cycles every 16.4ms ($A_0 \sim A_9$)
- 512K word x 8 bit test mode capability

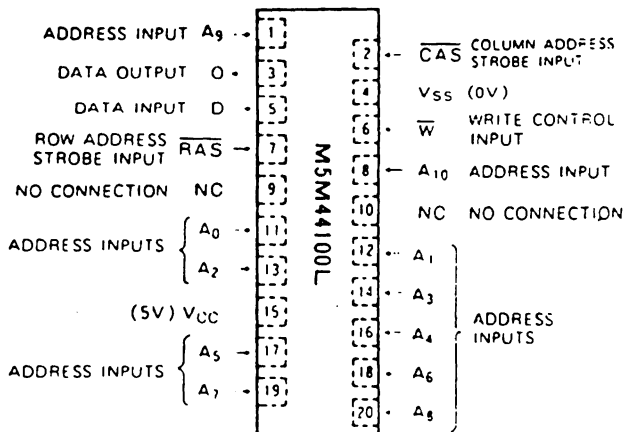
APPLICATION

Main memory unit for computers, Microcomputer memory, Refresh memory for CRT

PIN CONFIGURATION (TOP VIEW)



Outline 26P0Z (SOJ)



Outline 20P5L-B (ZIP)

FAST PAGE MODE 4194304-BIT(4194304-WORD BY 1-BIT) DYNAMIC RAM

FUNCTION

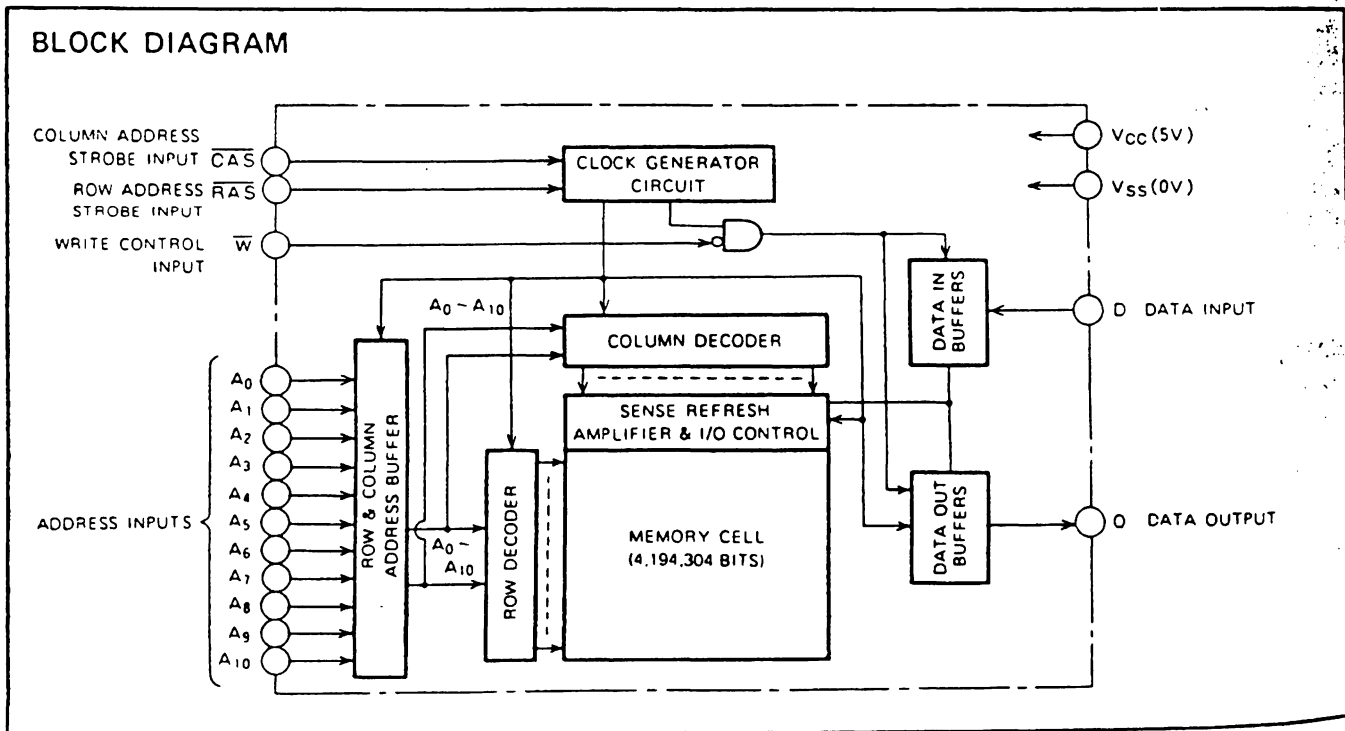
The M5M44100J, L provide, in addition to normal read, write, and read-modify-write operations, a number of other functions, e.g., fast-page mode, $\overline{\text{RAS}}$ -only refresh, and delayed-write. The input conditions for each are shown in Table 1.

Table 1 Input conditions for each mode

Operation	Inputs						Output	Refresh	Remark
	$\overline{\text{RAS}}$	$\overline{\text{CAS}}$	$\overline{\text{W}}$	D	Row address	Column address	O		
Read	ACT	ACT	NAC	DNC	APD	APD	VLD	YES	Fast page mode identical
Write (Early write)	ACT	ACT	ACT	VLD	APD	APD	OPN	YES	
Write (Delayed write)	ACT	ACT	ACT	VLD	APD	APD	IVD	YES	
Read-modify-write	ACT	ACT	ACT	VLD	APD	APD	VLD	YES	
$\overline{\text{RAS}}$ -only refresh	ACT	NAC	DNC	DNC	APD	DNC	OPN	YES	
Hidden refresh	ACT	ACT	DNC	DNC	DNC	DNC	VLD	YES	
$\overline{\text{CAS}}$ before $\overline{\text{RAS}}$ refresh	ACT	ACT	DNC	DNC	DNC	DNC	OPN	YES	
Standby	NAC	DNC	DNC	DNC	DNC	DNC	OPN	NO	

Note ACT active, NAC nonactive, DNC don't care, VLD valid, IVD invalid, APD: applied, OPN open

BLOCK DIAGRAM





Am79C30A/32A

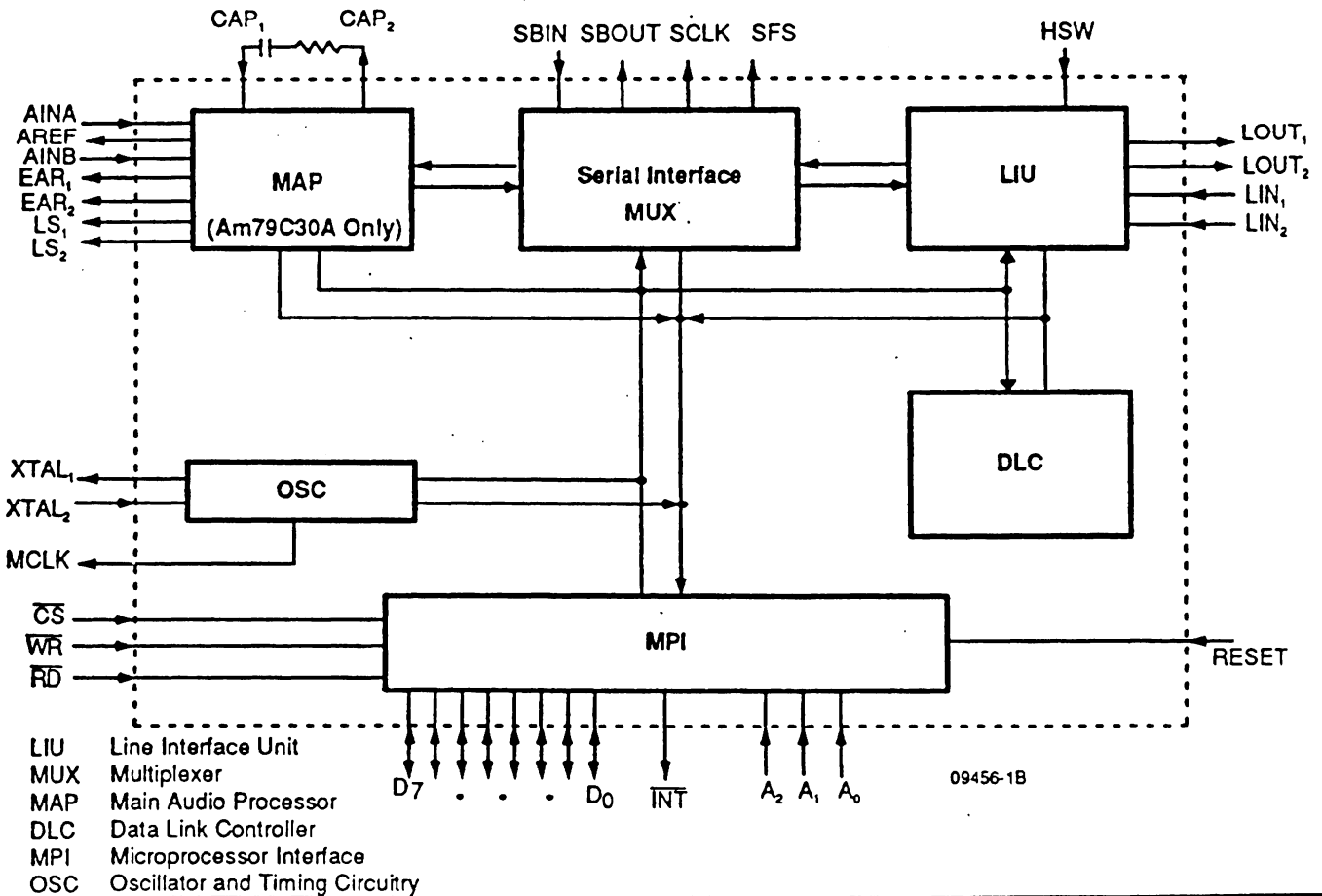
Digital Subscriber Controller (DSC)
ISDN Data Controller (IDC)

Advanced
Micro
Devices

DISTINCTIVE CHARACTERISTICS

- Combines CCITT I.430 S/T Interface transceiver, D-channel LAPD processor, and audio processor (DSC only) in a single chip
- Interrupt-driven microprocessor interface
- CMOS technology, TTL compatible
- 'S' or 'T' Interface Transceiver
 - Level 1 Physical Layer Controller
 - Supports point-to-point, short or extended passive bus configurations
 - Multiframe support
- D-channel Processing Capability
 - Flag generation/detection
- Audio Processing Capability (DSC only)
 - CRC generation/checking
 - Zero insertion/deletion
 - Four 2-byte address detectors
 - Random number generation
 - 8-byte transmit and receive FIFOs
 - Dual audio inputs
 - Earpiece and loudspeaker drivers
 - Filter/codec with A/mu selection
 - Programmable gain and equalization filters
 - Programmable sidetone level
 - Programmable DTMF, single tone, and ringer tone generation

BLOCK DIAGRAM



Publication# 09893 Rev. D Amendment /0
Issue Date: March 1989

GENERAL DESCRIPTION

The Am79C30A Digital Subscriber Controller (DSC) and Am79C32A ISDN Data Controller (IDC), shown in the block diagram, provide the Terminal Equipment access to the ISDN. The Am79C30A/32A is compatible with the CCITT I-Series recommendations at the 'S' reference point allowing the user of the device to design TEs which conform to the international ISDN standards.

The Am79C30A/32A provides a 192 kbps full duplex digital path between the TE located in the subscriber's premises and the NT or PABX line card over 4-wires. The Am79C30A/32A separates the bit stream into the B1- (64 kbps), B2- (64 kbps) and D- (16 kbps) channels. The B-channels are routed to different sections of the Am79C30A under user control. The D-channel is partially processed in the Am79C30A/32A and passed to the microprocessor for further processing.

The transmission rate of 192 kbps provides a 48-bit frame every 250 μ s for framing and maintenance. The frame structure provides for frame synchronization and multiple terminal contention resolution as described in the CCITT I-series recommendations. Both point-to-point and point-to-multipoint connections are supported.

The Am79C30A can be used as a voice telephone, a digital data terminal, or a voice and data terminal. The Am79C32A can be used as a digital data terminal.

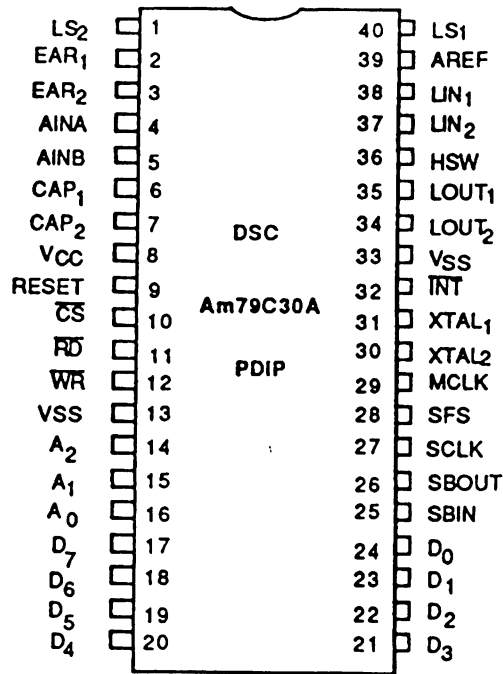
The audio processor in the Am79C30A, shown in the block diagram, uses Digital Signal Processing (DSP) to implement the codec and filter functions. The audio processor interfaces to a speaker, an earpiece, and two separate audio inputs. In the receive and transmit paths the user may program gain or alter the frequency response. The audio processor is not available in the Am79C32A.

A serial port gives the user access to the B-channels of the Am79C30A/32A multiplexer. This serial port may be used by data terminals and provides, with additional circuitry, access to the CCITT 'R' reference point.

The Am79C30A/32A is controlled via an interrupt driven microprocessor bus interface by an external microprocessor. Using this interface, the microprocessor processes the D-channel information and programs the Am79C30A/32A accordingly. This includes programming a multiplexer within the Am79C30A/32A to route the B-channels as specified by the D-channel control information. The microprocessor can interrogate and program the Am79C30A/32A via its mode, status, and error registers.

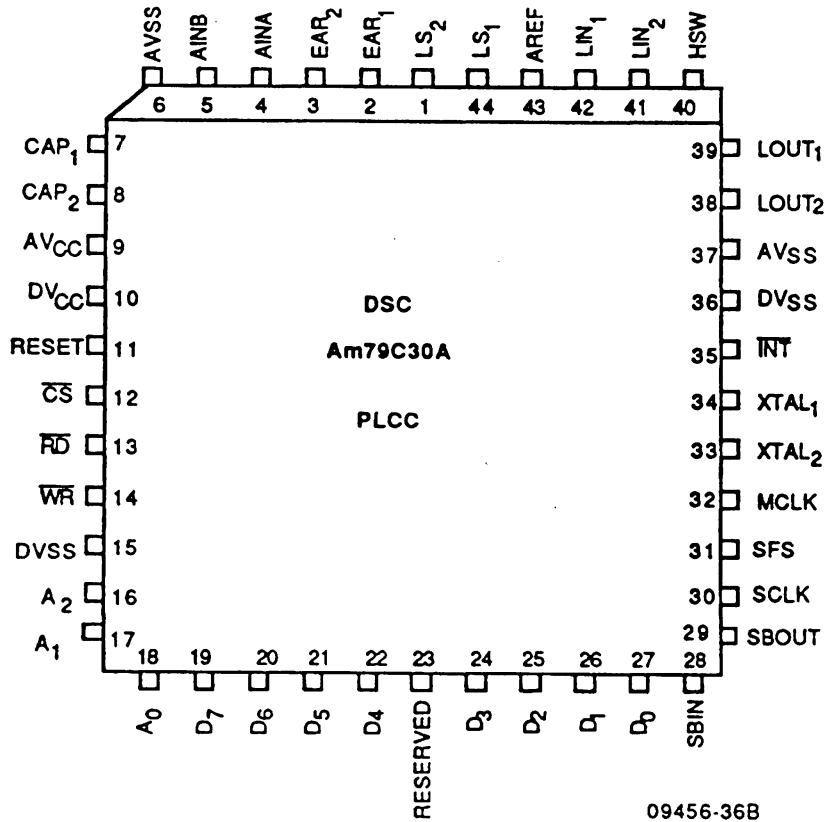
CONNECTION DIAGRAMS
Top View

40 Pin Dual-In-Line Package



09456-35B

44 Pin Plastic Leaded Chip Carrier (PLCC)



09456-36B

Figure 2. Am79C30A DSC Connection Diagrams

PIN DESCRIPTION

All signal levels are TTL compatible unless otherwise stated.

Line Interface Unit (LIU)

HSW

Hook-Switch (Input)

The HSW signal indicates if the hookswitch is on or off hook. This signal may be generated with a mechanical switch wired to ground with a pull-up resistor to V_{CC} . Any change in the HSW state causes an interrupt.

LIN1, LIN2

Subscriber Line Input (Differential Inputs)

The LIN1 and LIN2 inputs interface to the subscriber ('S' reference point) via an isolation transformer. LIN2 is the positive input, LIN1 is the negative input. These pins are not TTL compatible.

LOUT1, LOUT2

Subscriber Line Output (Differential Outputs)

The LOUT1 and LOUT2 line driver output signals interface to the subscriber line at the 'S' reference point via an isolation transformer and resistors. LOUT2 is the positive 'S' interface driver (that is, sources current during a high mark) and LOUT1 is the negative 'S' interface driver (that is, sources current during low mark). For multipoint applications, all TE's must maintain the same polarity on the 'S' interface. These pins are not TTL compatible.

Multiplexer (MUX)

SBIN

Serial Channel (Input)

The data rate on SBIN is 192 kbps. SBIN consists of three 64 kbps serial channels. Data bytes are received MSB first.

SBOUT

Serial Channel (Output)

The data rate on SBOUT is 192 kbps. SBOUT consists of three 64 kbps serial channels. Data bytes are transmitted MSB first.

SCLK

Serial Clock (Output)

SCLK is a 192 kbps synchronization clock which defines the position of the serial bits in the SBOUT and SBIN channels. Data at the SBIN input must be valid on the rising edge of SCLK. The data on the SBOUT pin changes on the falling edge of SCLK. SCLK powers up tri-stated, and is enabled when a MUX connection is programmed.

SFS

Serial Channel Frame Sync. (Output)

SFS is an 8 kHz signal which identifies the beginning of each frame by a low to high transition. The 192 kbps data stream on SBIN and SBOUT is referenced to SFS. SFS

powers up tri-stated, and is enabled when a MUX connection is programmed.

Main Audio Processor (MAP)

All MAP pins are analog, and hence not TTL compatible.

AINA, AINB

Analog (Inputs)

These analog inputs allow for two separate analog (audio) inputs to the transmit path of the codec/ filter. Input signals on either of these pins must be referenced to AREF.

AREF

Analog Reference (Output)

This is a nominal 2.4 V reference voltage output for biasing the analog inputs. Note that AREF is only available when the MAP is active.

CAP1, CAP2

Capacitor/Resistor (CAP1, Input; CAP2, Output)

An external resistor and capacitor are connected in series between these pins. These components are needed for the integrator in the Analog to Digital Converter (ADC).

EAR1, EAR2

Earpiece Interface (Differential Outputs)

EAR1 and EAR2 are the outputs from the receive path of the filter codec. These differential outputs can directly drive 600 ohms.

LS1, LS2

Loudspeaker Interface (Differential Outputs)

LS1 and LS2 are push-pull outputs which can directly drive a 50 ohm loudspeaker.

Microprocessor Interface (MPI)

A2-A0

Address Line (Inputs)

A2, A1, and A0 signals select source and destination registers for read and write operations on the data bus.

\overline{CS}

Chip Select (Input)

\overline{CS} must be low to read or write to the Am79C30A/32A. Data transfer occurs over the bidirectional data lines (D7-D0).

D7-D0

Data Bus (Bidirectional with High Impedance State)

The eight bidirectional data bus lines are used to exchange information with the microprocessor. D0 is the least significant bit (LSB) and D7 is the most significant bit (MSB). A high on the data bus line corresponds

logic '1' and low corresponds to a logic '0'. These lines act as inputs when both \overline{WR} and \overline{CS} are active and as outputs when both \overline{RD} and \overline{CS} are active. When \overline{CS} is inactive or both \overline{RD} and \overline{WR} are inactive, the D0-D7 pins are in a high impedance state.

\overline{INT}

Interrupt (Output)

An active low output on the \overline{INT} pin informs the external microprocessor that the Am79C30A/32A needs interrupt service. \overline{INT} is updated once every 125 μ s. The \overline{INT} pin remains active until the Interrupt Register (IR) is read or the Am79C30A/32A is reset.

RESET

Reset (Input)

Reset is an active high signal which causes the Am79C30A/32A to immediately terminate its present activity and initialize to the reset condition. When reset returns low, the Am79C30A/32A enters the idle mode.

\overline{RD}

Read (Input)

The active low read signal is conditioned by \overline{CS} and indicates that internal information is to be transferred onto the data bus. A number of internal registers are user accessible. The contents of the accessed register are transferred onto the data bus after the high to low transition of the \overline{RD} input.

\overline{WR}

Write (Input)

The active low write signal is conditioned by \overline{CS} and indicates that external information on the data bus is to be transferred to an internal register. The contents of the data bus are loaded on the low to high transition of the \overline{WR} input.

Oscillator (OSC)

MCLK

Master Clock (Output)

The MCLK output is available for use as the system clock for the microprocessor. It is derived from the 12.288 MHz crystal via a programmable divider in the Am79C30A/32A which provides the following MCLK output frequencies: 12.288, 6.144, 4.096, and 3.072 MHz.

XTAL1, XTAL2

External Crystal (Output/Input)

XTAL1 and XTAL2 are connected to an external parallel resonant crystal for the on-chip oscillator. XTAL2 can also be connected to an external source instead of a crystal, in which case XTAL1 should be left disconnected. The frequency must be 12.288 MHz, \pm 80 ppm.

Power Supply Pins

PLCC Packages

AVcc +5V analog power supply \pm 5% (PLCC only)

AVss Analog ground (PLCC only)

DVss Digital ground (PLCC only)

DVcc +5V digital power supply, \pm 5% (PLCC only)

DIP Packages

Vcc +5V power supply, \pm 5% (DIP only)

Vss Ground (DIP only)

Note: For best performance, decoupling capacitors should be installed between Vcc and Vss as close to the chip as possible. Do not use separate supplies for analog and digital power and ground connections.

OPERATIONAL DESCRIPTION

In order to specify the functions of the Am79C30A/32A, the device has been divided into blocks as shown in the block diagram. Each of the blocks listed below is defined separately in the Functional Description.

LIU Line Interface Unit
 MUX Multiplexer
 MAP Main Audio Processor (Am79C30A only)
 DLC Data Link Controller

MPI Microprocessor Interface
 OSC Oscillator and Timing Circuitry

User Accessible Registers/Buffers

The microprocessor interface is used to program and control the operation of the Am79C30A/32A. The following registers/buffers are user accessible in each of the blocks listed above and are described in the respective Functional Description Sections.

Registers	No.	Mnemonic
Line Interface Unit (LIU)		
LIU Status Register	1	LSR
LIU Priority Register	1	LPR
LIU Mode Registers	2	LMR
Multiframe Register	1	MF
Multiframe S-Bit/Status Buffer	1	MFSB
Multiframe Q-Bit Buffer	1	MFOB
Multiplexer (MUX)		
MUX Control Registers	4	MCR
Main Audio Processor (MAP) (Am79C30A only)		
X Filter Coefficient Registers	16	X
R Filter Coefficient Registers	16	R
GX Gain Coefficient Registers	2	GX
GR Gain Coefficient Registers	2	GR
GER Gain Coefficient Registers	2	GER
Sidetone Gain Coefficient Registers	2	STGR
Frequency Tone Generator Registers	2	FTGR
Amplitude Tone Generator Registers	2	ATGR
MAP Mode Registers	2	MMR
Data Link Controller (DLC)		
First Received Byte Address Registers	4	FRAR
Second Received Byte Address Registers	4	SRAR
Transmit Address Register (16 bit)	1	TAR
D-channel Receive Byte Limit Register (16 bit)	1	DRLR
D-channel Receive Byte Count Register (16 bit) (2 byte FIFO)	1	DRCR
D-channel Transmit Byte Count Register (16 bit)	1	DTCR
Random Number Generator Registers	2	RNGR
D-channel Mode Registers	4	DMR
D-channel Status Registers	2	DSR
Address Status Register (2 byte FIFO)	1	ASR
D-channel Error Register (2 byte FIFO)	1	DER
Microprocessor Interface (MPI)		
Initialization Register	1	INIT
Command Register	1	CR
Interrupt Register	1	IR
Data Register	1	DR
D-channel Transmit Buffer (8 byte FIFO)	1	DCTB
D-channel Receive Buffer (8 byte FIFO)	1	DCRB
Bb Transmit Buffer	1	BBTB
Bb Receive Buffer	1	BBRB
Bc Transmit Buffer	1	BCTB
Bc Receive Buffer	1	BCRB

Note: See the Microprocessor Interface section for register addressing.

Initialization

The initialization procedure is controlled via the Initialization Register (INIT) which is accessed by the microprocessor as defined in the Microprocessor Interface

section. This Initialization Register (INIT) has the following format:

Initialization Register (INIT), Read/Write

Bit #								Control	Function
7	6	5	4	3	2	1	0		
X	X	X	X	X	X	0	0	Power Mode Selection	Idle Mode (default)
X	X	X	X	X	X	0	1		Active Mode (voice & data)
X	X	X	X	X	X	1	0		Active Mode (Data only)
X	X	X	X	X	X	1	1		Reserved
X	X	X	X	X	0	X	X	Interrupt Selection	Enable INT pin (default)
X	X	X	X	X	1	X	X		Disable INT pin
X	X	0	0	0	X	X	X	Clock Divider Selection	Divide by 2 (default)
X	X	0	0	1	X	X	X		Divide by 1
X	X	0	1	0	X	X	X		Divide by 4
X	X	0	1	1	X	X	X		Divide by 2
X	X	1	0	0	X	X	X		Divide by 3
X	X	1	0	1	X	X	X		Divide by 2
X	X	1	1	0	X	X	X		Divide by 2
X	X	1	1	1	X	X	X		Divide by 2
X	1	X	X	X	X	X	X	Abort Selection	Receive abort
X	0	X	X	X	X	X	X		No Receive abort (default)
1	X	X	X	X	X	X	X		Transmit abort
0	X	X	X	X	X	X	X		No Transmit abort (default)

Reset

The Am79C30A/32A can be reset by driving the reset pin high. When power is first supplied to the Am79C30A/32A, a reset must be asserted. This

initializes the Am79C30A/32A to its default values as defined in the subsequent sections. After reset, the Am79C30A/32A enters the idle mode.

Idle Mode Operation

To conserve power the Am79C30A/32A can be placed in the idle mode. This can be done by either asserting the RESET signal or by clearing bits 0 and 1 in the INIT

register. When the Am79C30A/32A is in the idle mode, and there is no activity on any of the external interfaces, the state of the output pins are:

Pin Name	State following RESET	Idle Mode
D7-D0	High impedance	High impedance
MCLK	6.144 MHz	As programmed (see Initialization section)
INT	Logical '1'	Logical '1' can be driven low by internal conditions explained below
SBOUT	High impedance	High impedance
SFS	High impedance	8 kHz
SCLK	High impedance	192 kHz period
LS1, LS2,		
EAR1,	High impedance	High impedance
EAR2,		
AREF		
LOUT1,	High impedance	High impedance
LOUT2		

FUNCTIONAL DESCRIPTION

Microprocessor Interface (MPI)

The Am79C30A/32A can be connected to any general purpose 8-bit microprocessor via the MPI. The MCLK from the Am79C30A/32A can be used as the clock for the microprocessor. The MPI is an interrupt driven interface containing all the circuitry necessary for access to the internal programmable registers, status registers, coefficient RAM, and transmit/receive buffers.

MPI External Interface

The MPI has the following external connections:

Name	Direction	Function
D7-D0	Bidirectional	Data Bus
A2, A1, & A0	Inputs	Address Line
\overline{RD}	Input	Read enable
\overline{WR}	Input	Write enable
\overline{CS}	Input	Chip Select
\overline{RESET}	Input	Initialization
\overline{INT}	Output	Interrupt

Register Selection

\overline{CS}	\overline{RD}	\overline{WR}	A2	A1	A0	Register(s) Accessed
0	1	0	0	0	0	Command Register (CR), write only
0	0	1	0	0	0	Interrupt Register (IR), read only
0	1	0	0	0	1	Data Register (DR), write only
0	0	1	0	0	1	Data Register (DR), read only
0	0	1	0	1	0	D-channel Status Register 1 (DSR1), read only
0	0	1	0	1	1	D-channel Error Register (DER), read only (2 byte FIFO)
0	1	0	1	0	0	D-channel Transmit Buffer (DCTB), write only (8 byte FIFO)
0	0	1	1	0	0	D-channel Receive Buffer (DCRB), read only (8 byte FIFO)
0	1	0	1	0	1	Bb channel Transmit Buffer (BBTB) write only
0	0	1	1	0	1	Bb channel Receive Buffer (BBRB), read only
0	1	0	1	1	0	Bc channel Transmit Buffer (BCTB), write only
0	0	1	1	1	0	Bc channel Receive Buffer (BCRB), read only
0	0	1	1	1	1	D-channel Status Register 2 (DSR2), read only
1	X	X	X	X	X	No access (X = logical '0' or '1')

Note: The \overline{RD} and \overline{WR} signals must never both be low under normal operating conditions.

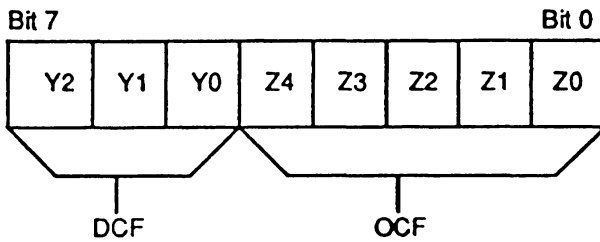
Directly Accessed Registers/Buffers

Register	Mnemonic
Command Register	CR
Interrupt Register	IR
Data Register	DR
D-channel Status Register 1	DSR1
D-channel Error Register (2 byte FIFO)	DER
D-channel Transmit Buffer (8 byte FIFO)	DCTB
D-channel Receive Buffer (8 byte FIFO)	DCRB
Bb Transmit Buffer	BBTB
Bb Receive Buffer	BBRB
Bc Transmit Buffer	BCTB
Bc Receive Buffer	BCRB
D-channel Status Register 2	DSR2

The 8-bit bidirectional data bus (D7-D0) is used to communicate with these registers. The selection of which register(s) is (are) accessed is controlled by the \overline{CS} , \overline{RD} , \overline{WR} , A2, A1, and A0 signals from the microprocessor to the Am79C30A/32A in the following manner (see Figure 4):

Command Register (CR), Write only

The Command Register (CR) is used to hold the index for the indirectly accessed registers. The CR is one byte wide and divided into two fields, the destination code field (DCF) and the operational code field (OCF):



The 'Y' bits in the DCF determine to which block the OCF is related. The 'Y' bits are defined below:

Am79C30A/32A block	Y2	Y1	Y0
RESERVED	0	0	0
INIT	0	0	1
MUX	0	1	0
MAP (Am79C30A only)	0	1	1
DLC	1	0	0
LIU	1	0	1
RESERVED	1	1	0
RESERVED	1	1	1

The 'Z' bits of the OCF contain the operation code used to address a specific register within a DCF block. For each DCF there is a different set of OCFs. Each of the OCFs and their associated data registers are defined in the following sections. When the OCF indicates a read or write operation is possible, then the appropriate access is achieved by asserting either the read or write signals to the Am79C30A/32A.

Registers within certain groups can be quickly accessed by using internal circuitry which automatically increments the indirect address (index value) contained in the OCF. The CR is first loaded with the index value, then the data bytes are transferred in sequence between the Am79C30A/32A and the microprocessor via the DR. For example, operation number 5 in the MPI-LIU definition allows operations 2 to 4 to be performed sequentially without reloading the CR. Whenever the CR register is loaded, any previous commands are automatically terminated.

In the following tables the "bytes transferred" numbers next to the OCFs are the number of bytes which are read or written to the DR after the CR has been loaded.

MPI-INIT Definition

The INIT register is used by the microprocessor to select the MCLK output frequency and the power-up/idle

states of the Am79C30A/32A. This register is accessed via the 'Z' bits in the CR as follows:

INIT OPERATION (DCF = 001)	OCF					Bytes Transferred
	Z4	Z3	Z2	Z1	Z0	
R/W INIT register	0	0	0	0	1	1

MPI-LIU Definition

The LIU contains the following registers:

LIU Registers	No.	Mnemonic
LIU Status Register	1	LSR
LIU Priority	1	LPR
LIU Mode Registers	2	LMR1, LMR2
Multiframe Register	1	MF
Multiframe S-bit Buffer	1	MFSB
Multiframe Q-bit Buffer	1	MFQB

These registers are accessed via the 'Z' bits in the CR as follows:

LIU OPERATION (DCF = 101)	OCF					Bytes Transferred
	Z4	Z3	Z2	Z1	Z0	
1. Read LSR	0	0	0	0	1	1
2. R/W LPR	0	0	0	1	0	1
3. R/W LMR1	0	0	0	1	1	1
4. R/W LMR2	0	0	1	0	0	1
5. Perform Operations 2-4	0	0	1	0	1	3
6. Read/Write MF	0	0	1	1	0	1
7. Read MFSB	0	0	1	1	1	1
8. Write MFQB	0	1	0	0	0	1

MPI-MUX Definition

The MUX contains four microprocessor read/write control registers MCR1, MCR2, MCR3, and MCR4. These registers are accessed via the 'Z' bits in the CR as follows:

MUX OPERATION (DCF = 010)	OCF					Bytes Transferred
	Z4	Z3	Z2	Z1	Z0	
1. R/W MCR1	0	0	0	0	1	1
2. R/W MCR2	0	0	0	1	0	1
3. R/W MCR3	0	0	0	1	1	1
4. R/W MCR4	0	0	1	0	0	1
5. Perform Operations 1-4	0	0	1	0	1	4

When more than one byte is being transferred due to OCF command 5, MCR1 is always accessed first, followed by MCR2, then MCR3, and finally MCR4.

Main Audio Processor (MAP) (Am79C30A Only)

This block performs the digital-to-analog (DAC) and analog-to-digital (ADC) conversions of the audio signals. The codec and filter functions are implemented using digital signal processing techniques to provide substantial flexibility and programmability. Analog interfaces are provided for a handset earpiece, a handset mouthpiece, a microphone and a loudspeaker. The MAP contains the following programmable, user accessible features:

Multi-Tone Generator

This generator can be used to generate a signal consisting of one or two tones where the frequency and amplitude of the tone is programmable. The tone(s) can be summed into the transmit (and sidetone) path for use as a DTMF tone generator, or single tones can be injected into the receive path. The tones can be used as ringing tones (to the loudspeaker output), dial tones, busy signals, ringback tones or other call progress tones.

Two Attenuation Distortion Correction Filters

There is one attenuation distortion correction filter in the transmit path and one in the receive path. These filters can be programmed to modify the frequency characteristics of the transmit or receive paths and to equalize for the characteristics of the microphones, earpiece speaker, or loudspeaker. They can also be used to add pre and/or post emphasis to make the signals match other characteristics.

Three Programmable Gain Stages

There is one gain adjustment in the transmit path and two in the receive path to provide a wide range of gain control.

Programmable Sidetone Gain

There is a built-in sidetone path which samples the transmit signal, attenuates it by a programmable amount, then sums it into the receive path.

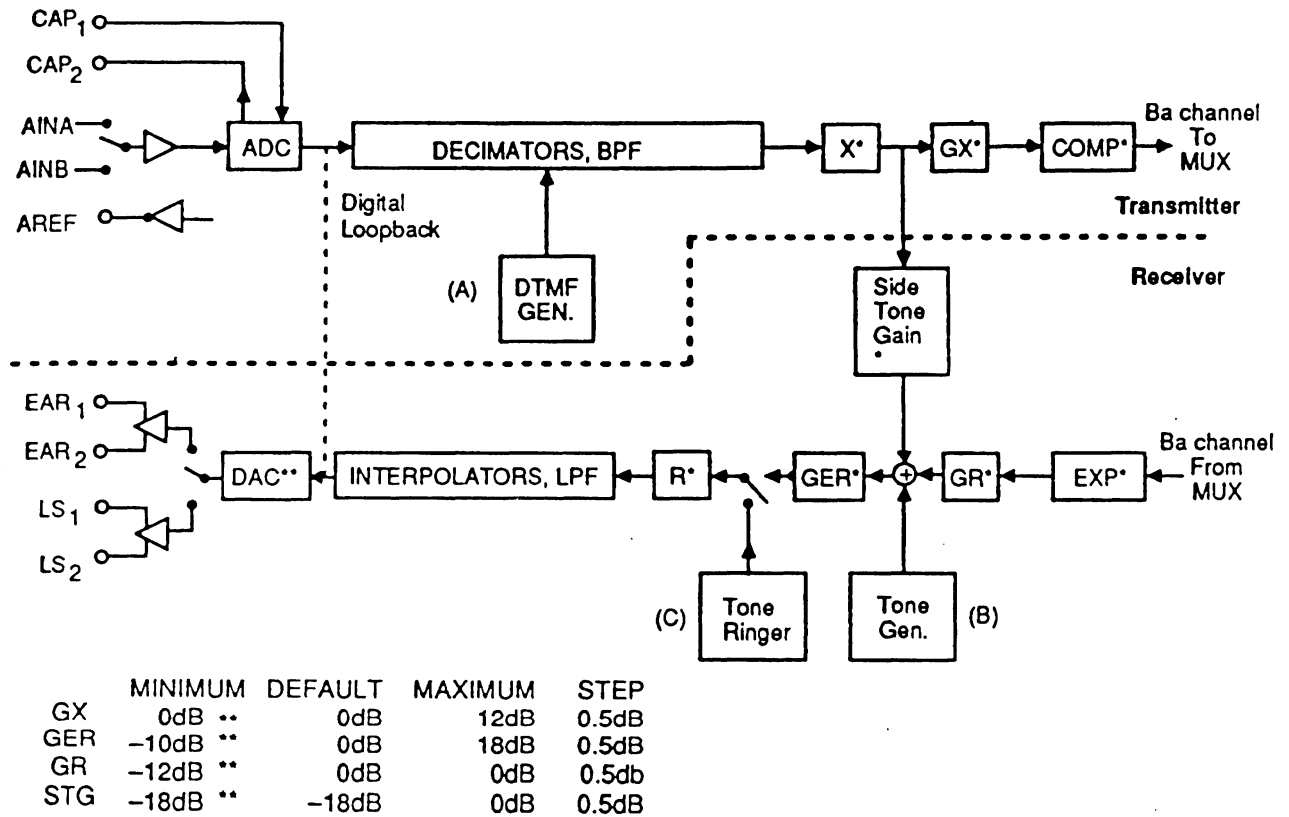
External Interface

Audio Input Port

The audio input port consists of two inputs (AINA and AINB) selectable, one at a time, by setting a bit in the MAP Mode Register 2, (MMR2 bit 0). Signals applied to these inputs must be referenced to AREF.

Earpiece and Loudspeaker Drivers

Each driver consists of push-pull amplifiers with a low impedance output. Either of these audio output ports can be selected, one at a time, via MMR2 bit 1.



*Programmable

**These registers can also be programmed for infinite attenuation to break the signal path if desired.

09456-31B

Figure 9. Main Audio Processor Block Diagram

AC Characteristics

$V_{CC} = 5\text{ V} \pm 5\%$, $V_{SS} = 0\text{ V}$; $T_A = 0^\circ\text{C} \rightarrow 70^\circ\text{C}$, $MCLK = 3.072\text{ MHz}$

MAP Analog Characteristics (Am79C30A only)

Parameter Symbol	Parameter Description	Test Conditions	Min	Max	Units
Z_{IN}	Analog input impedance AINA or AINB to AREF	$-1.25\text{ V} < V_{IN} < +1.25\text{ V}$ $f_{IN} < 4\text{ kHz}$	200		Kohm
V_{OS}	Allowable offset voltage at AINA or AINB	with respect to AREF pin	-5	+5	mv
V_{IR}	Analog input full scale reference level AINA or AINB	with respect to AREF pin	± 1.25 (nominal)		Vpeak
V_{PR}	Analog output full scale reference level (PCM code = +3 dBm0)	EAR1 to EAR2 with $R_{LOAD} > 540\text{ ohm}$ and $C_{LOAD} < 100\text{ pF}$, or LS1 to LS2 with $R_{LOAD} > 40\text{ ohm}$ and $C_{LOAD} < 100\text{ pF}$	± 2.5 (nominal)		Vpeak
Z_{LS}	Allowable Load LS1 to LS2			$R_{LOAD} > 40\text{ ohms}$ and $C_{LOAD} < 100\text{ pF}$	
Z_{EAR}	Allowable Load EAR1 to EAR2			$R_{LOAD} > 540\text{ ohms}$ and $C_{LOAD} < 100\text{ pF}$	

82072

CHMOS HIGH INTEGRATION FLOPPY DISK CONTROLLER

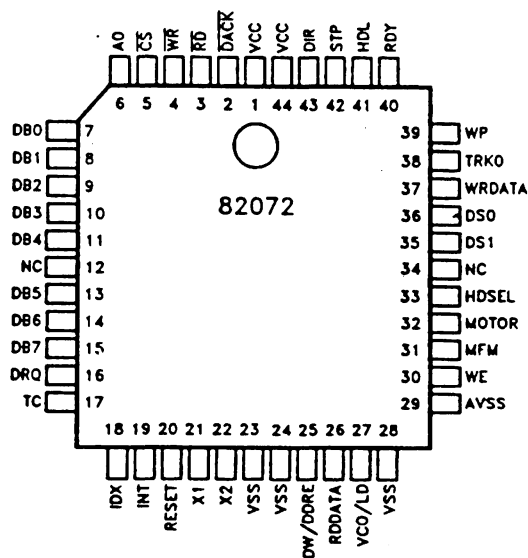
- Integrated Analog Data Separator with Software Selectable Data Rates (250K, 300K, 500K, Bit/Sec-MFM Mode)
- 16 Byte FIFO with Programmable Threshold
- High Speed Processor Interface
 - 16 MHz iAPX 386— 1 Wait State
 - 12.5 MHz iAPX 286—1 Wait State
 - 8 MHz iAPX 286— 0 Wait State
- Programmable Internal Write Precompensation Delays
- Programmable Drive Motor On/Off Delays
- Addresses Up to 256 Tracks Directly, Supports Unlimited Tracks
- Implied Seek with Read/Write Disk Commands
- Software Compatible with 8272A
- Controls 8", 5 1/4" and 3 1/2" Floppy Disk Drives
- Plastic 40 Pin DIP or 44 Pin PLCC Packages

(See Packaging Spec. Order # 231369)

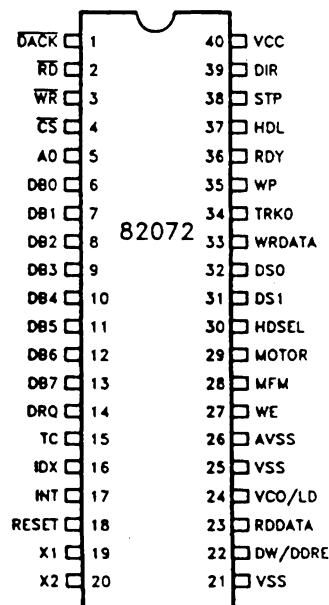
The 82072 CHMOS high integration floppy disk controller solves the many complex disk drive and microprocessor interface issues that exist today, while maintaining software compatibility with the industry standard 8272A. Features include a sophisticated on-chip analog phase lock loop with software selectable data rates, write precompensation delay, and motor on/off delays to simplify the disk drive interface. System interfacing is enhanced with the addition of a FIFO which allows a more flexible system to be designed.

The standard 82072 supports a maximum data rate of 500 Kbits per second.

The 82072 is fabricated on Intel's advanced CHMOS III technology for minimal power consumption and is available in a plastic 40 pin DIP or a plastic 44-leaded chip carrier (PLCC) package.



290122-1



290122-2

Figure 1. 82072 Pinout

Table 1. 82072 Pin Description

Symbol	DIP	PLCC	I/O	Function																																				
$\overline{\text{DACK}}$	1	2	I	DMA ACKNOWLEDGE: DMA control line that qualifies the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ inputs during DMA cycles.																																				
$\overline{\text{RD}}$	2	3	I	READ: Control signal to transfer data to the data bus from the 82072.																																				
$\overline{\text{WR}}$	3	4	I	WRITE: Control signal to transfer data into the 82072 from the data bus.																																				
$\overline{\text{CS}}$	4	5	I	CHIP SELECT: Control signal that qualifies the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ inputs.																																				
A0	5	6	I	<p>ADDRESS:</p> <table border="1"> <thead> <tr> <th>A0</th> <th>$\overline{\text{RD}}$</th> <th>$\overline{\text{WR}}$</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Illegal *</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Read Main Status Register</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Write to the Data Rate Select Register **</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>No Action</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>Illegal *</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>Read from FIFO</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>Write into FIFO</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>No Action</td> </tr> </tbody> </table> <p>* User must ensure that these inputs do not occur. ** Change from 8272A—was illegal.</p>	A0	$\overline{\text{RD}}$	$\overline{\text{WR}}$	Function	0	0	0	Illegal *	0	0	1	Read Main Status Register	0	1	0	Write to the Data Rate Select Register **	0	1	1	No Action	1	0	0	Illegal *	1	0	1	Read from FIFO	1	1	0	Write into FIFO	1	1	1	No Action
A0	$\overline{\text{RD}}$	$\overline{\text{WR}}$	Function																																					
0	0	0	Illegal *																																					
0	0	1	Read Main Status Register																																					
0	1	0	Write to the Data Rate Select Register **																																					
0	1	1	No Action																																					
1	0	0	Illegal *																																					
1	0	1	Read from FIFO																																					
1	1	0	Write into FIFO																																					
1	1	1	No Action																																					
DB0-7	6-13	7-11, 13-15	I/O	DATA BUS: Bidirectional 8-bit data bus. A0 determines whether transfer is to/from the FIFO or Main Status Register.																																				
DRQ	14	16	O	DMA REQUEST: Used to request service from a DMA controller.																																				
TC	15	17	I	TERMINAL COUNT: Control line from a DMA controller used to terminate requests for data transfers. Disk read and write commands complete the transfer to the current sector with valid CRC checking/generation.																																				
IDX	16	18	I	INDEX: Disk drive signal that indicates the beginning of a track. It is used to count retries and delay periods for internal (i.e. Motor On/Off) timers and is rising edge triggered.																																				
INT	17	19	O	Interrupt: Interrupt to host to indicate command completion or that a data transfer is required (depending upon the data transfer mode). Command completion interrupts are cleared by reading the ST0 Status Register. Data transfer interrupts are cleared when the amount of data in the FIFO reaches the full or empty level (depending on FIFO direction) or a TC is issued.																																				
RESET	18	20	I	<p>RESET: Places the 82072 in a known idle state. All disk outputs are set to a low level. All registers, except those set by the SPECIFY command, are cleared.</p> <p>From the trailing edge of Reset, there is a maximum delay of 8 microseconds until the Main Status register is valid.</p> <p>Following reset, the 82072 defaults to polling enabled.</p> <p>The default values for the new features are: internal data separator enabled, write precompensation value is 125 ns, MOTOR on delay is 0.0 sec., MOTOR off delay is 5.2 sec., data rate is dependent on DDRE setting, FIFO disabled.</p>																																				

Table 1. 82072 Pin Description (Continued)

Symbol	DIP	PLCC	I/O	Function
X1	19	21	I	CRYSTAL 1: External connection for a fundamental mode parallel resonant 24 MHz crystal for the internal oscillator. May be driven with a MOS level clock instead of a crystal. Refer to the D.C. Specifications.
X2	20	22	I	CRYSTAL 2: If an external clock is supplied on X1, this input must be left unconnected (floating).
V _{SS}	21, 25	23, 24, 28		LOGIC GROUND.
DW/DDRE	22	25	I	DATA WINDOW/DEFAULT DATA RATE ENABLED: Clock from the external PLL logic used to sample the Read Data input. When the internal PLL is used, this input pin is used to define the data rate and write precompensation values after RESET. DDRE tied high will cause the data rate and precompensation bits of the DSR to be reinitialized to the default values of 250 Kbps and 125 ns delay when a hardware/software reset is issued. DDRE tied low will cause the current data rate and precompensation values in the DSR to be retained when a hardware reset is issued. When a software reset is issued, the DSR will contain those values written into the register. DDRE tied low should be used in applications where data rate and precompensation information needs to be retained regardless of chip reset.
RDDATA	23	26	I	READ DATA: Serial FM or MFM encoded data from the disk drive.
VCO/LD	24	27	O	READ DATA GATE: This active high output enables an external PLL to synchronize to Read Data input from the disk drive. LOW DENSITY: This active high output is used by quad density disk drives to modify Read/Write head and data channel characteristics. This signal is activated when internal PLL is enabled and a data transfer rate of 250 or 300 Kbps is chosen.
AVSS	26	29		Analog ground for the Data Separator. It is recommended that care be taken to keep AVSS as noise free as possible. A separate connection to the ground plane is suggested.
WE	27	30	O	WRITE ENABLE: Disk drive control signal that enables the head to write onto the disk.
MFM	28	31	O	MFM MODE: When an external PLL is used, this output selects between single and double density (FM and MFM) modes. 1 = MFM, 0 = FM mode.
MOTOR	29	32	O	MOTOR ENABLE: Output used to activate the drive motor on the selected drive. Delays are programmable. With one output, this pin must be qualified with the drive select logic to provide motor enables for each drive.
HDSEL	30	33	O	HEAD SELECT: Signal used to select one of two sides on the disk. A 0 = side 0, a 1 = side 1.
DS1, 0	31, 32	35, 36	O	DRIVE SELECT: These outputs select one of four disk drives. DS0, DS1 = 0, 0 will select drive 0.
WRDATA	33	37	O	WRITE DATA: FM or MFM encoded serial data to the disk drive. No external precompensation is required.
TRK0	34	38	I	TRACK 0: Control line from the disk drive that indicates the head is on physical track 0 (outermost track).
WP	35	39	I	WRITE PROTECT: Input from the disk drive that indicates if the disk is physically write protected.

Table 1. 82072 Pin Description (Continued)

Symbol	DIP	PLCC	I/O	Function
RDY	36	40	I	READY: Input from the disk drive that indicates whether the drive is ready for an operation.
HDL	37	41	O	HEAD LOAD: This output loads the head onto the disk drive if required. Typically used by 8" drives.
STP	38	42	O	STEP: Output used to supply step pulses to the disk drive.
DIR	39	43	O	DIRECTION: This output, in conjunction with STP, causes the drive to move the head outward if a "0", and inward if a "1".
VCC	40	1, 44		Logic DC power supply.

NOTE:

1. Pins 12, 34 of the 44 pin PLCC package are not connected.

INTRODUCTION

The 82072 has integrated all of the complex circuitry required to interface microprocessor systems with disk drives that comply with the IBM System 34 Double Density (MFM) format or the IBM 3740 single density format (FM). The 82072 is a superset of the

8272A. Control over the new features was accomplished by adding extra registers and commands to the 82072. The 82072 will function like the 8272A Floppy Disk Controller (FDC) after being reset, with the added features being set to 8272A compatible default values. When accessing the disk drives, the 82072 is programmed the same as the 8272A.

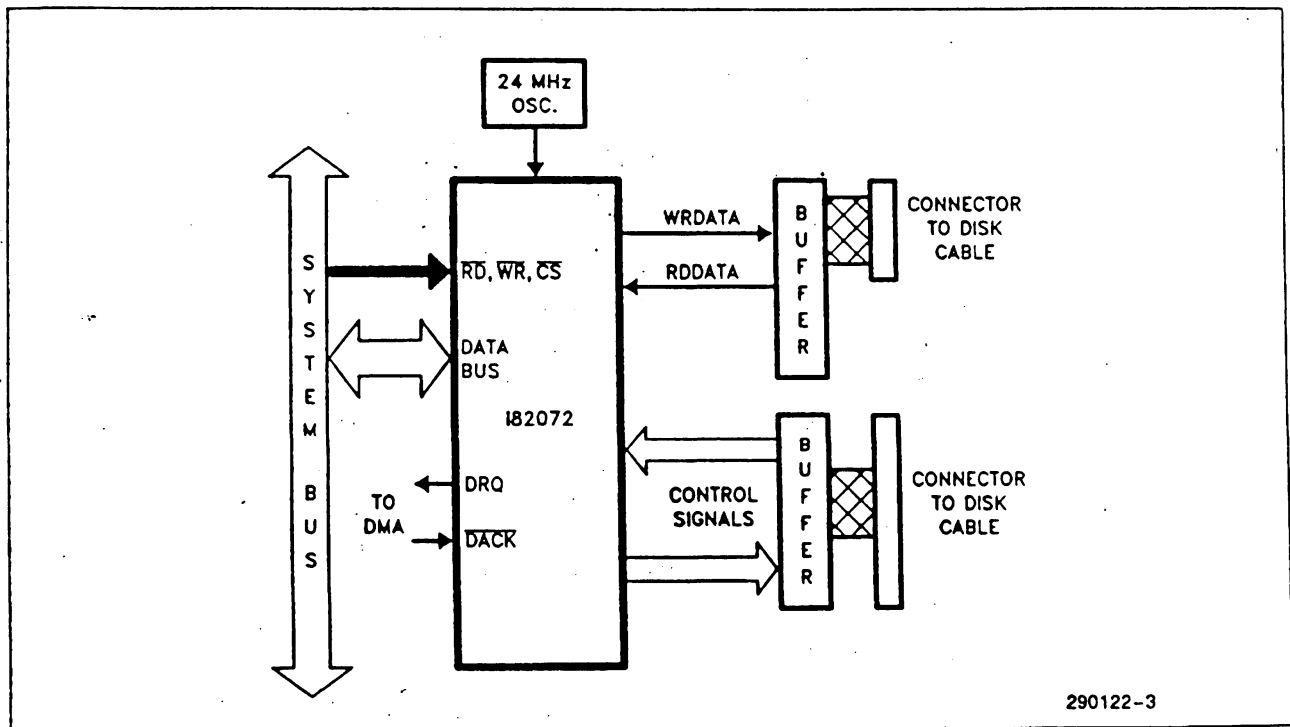


Figure 2. 82072 Typical System Block Diagram

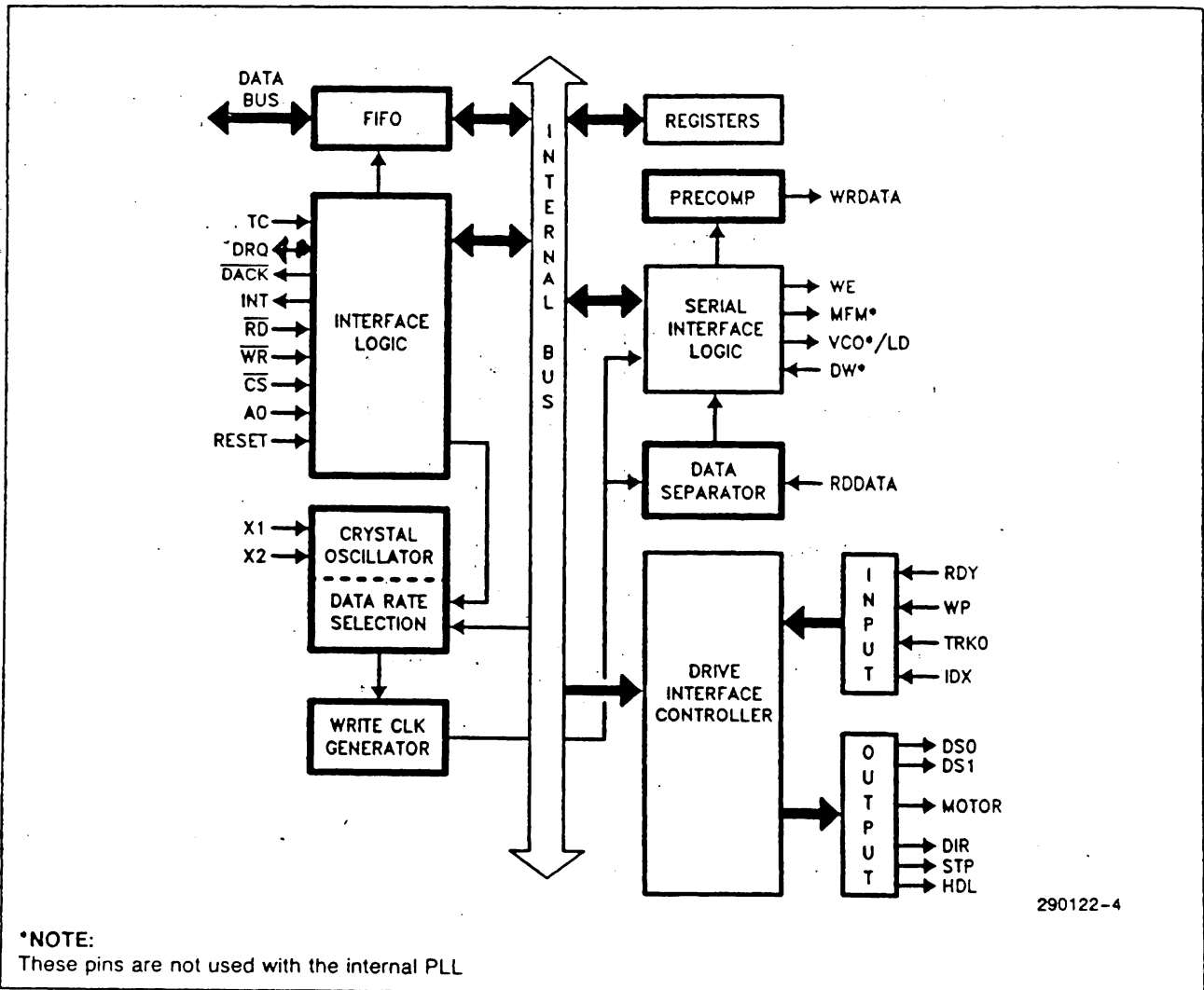
The microprocessor interface was enhanced by adding a 16 byte FIFO to reduce the timing constraints that most floppy disk controllers impose upon a system. The point at which the 82072 generates a request for a data transfer is selectable within the 16 byte range of the FIFO. The interface was further enhanced to support today's faster microprocessors (i.e., 8 MHz 80286, 10 MHz 80186) without incurring wait states. A powerdown mode has been added for low power or portable applications. With one command, the 82072 resets itself and then disconnects the power from the internal oscillator. Reset will reconnect the clock and once the 82072 is reprogrammed (if necessary), it will be ready to read and write disks again.

All of the control logic of the disk interface has been integrated into the 82072. Flexibility is maintained by allowing the user to select read and write data rates

(without any external hardware); write precompensation delays; motor on/off delay; and the track to start the precompensation on. The typical design will need the 82072, a crystal and high current drivers for the signals that interface to the disk drive. The new features (when used) need only to be chosen once after reset (although they may be modified at any time). From then on, the user programs the 82072 for disk accesses in the same manner as the 8272A.

ARCHITECTURE

Figure 3 is a block diagram of the 82072. The highlighted blocks represent areas that are either completely new or highly enhanced. All new features were adopted with the requirement of being software compatible with the 8272A.



*NOTE:
These pins are not used with the internal PLL

290122-4

Figure 3. 82072 Block Diagram

28F020

2048K (256K x 8) CMOS FLASH MEMORY

- Flash Electrical Chip-Erase
 - 2 Second Typical Chip-Erase
- Quick-Pulse Programming™ Algorithm
 - 10 μ s Typical Byte-Program
 - 4 Second Chip-Program
- 10,000 Erase/Program Cycles Minimum
- 12.0V \pm 5% V_{pp}
- High-Performance Read
 - 150 ns Maximum Access Time
- CMOS Low Power Consumption
 - 10 mA Typical Active Current
 - 50 μ A Typical Standby Current
 - 0 Watts Data Retention Power
- Command Register Architecture for Microprocessor/Microcontroller Compatible Write Interface
- Noise Immunity Features
 - \pm 10% V_{CC} Tolerance
 - Maximum Latch-Up Immunity through EPI Processing
- ETOX™ II Nonvolatile Flash Technology
 - EPROM-Compatible Process Base
 - High-Volume Manufacturing Experience
- JEDEC-Standard Pinouts
 - 32-Pin Plastic Dip
 - 32-Lead PLCC
 - 32-Lead TSOP
- Integrated Program/Erase Stop Timer
(See Packaging Spec., Order #231369)

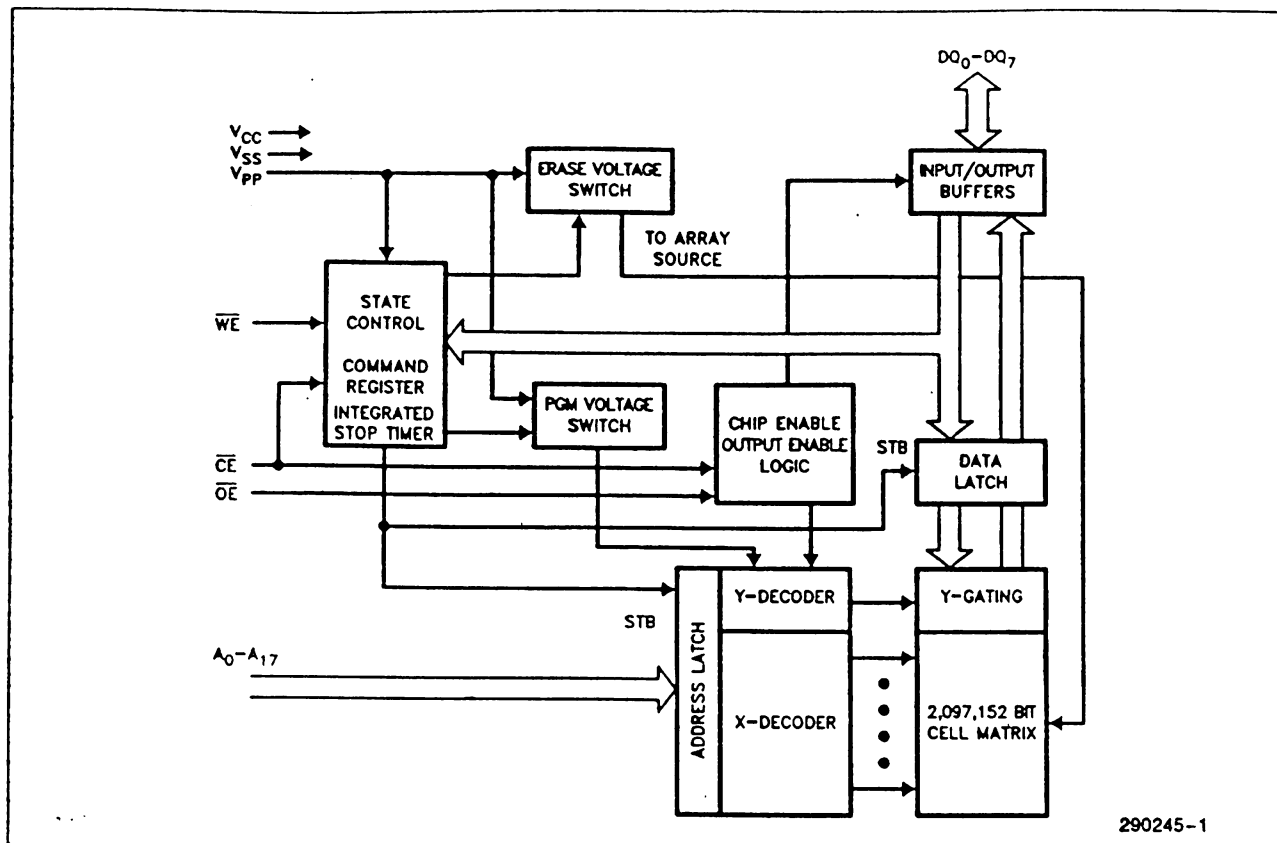
Intel's 28F020 CMOS flash memory offers the most cost-effective and reliable alternative for read/write random access nonvolatile memory. The 28F020 adds electrical chip-erasure and reprogramming to familiar EPROM technology. Memory contents can be rewritten: in a test socket; in a PROM-programmer socket; on-board during subassembly test; in-system during final test; and in-system after-sale. The 28F020 increases memory flexibility, while contributing to time- and cost-savings.

The 28F020 is a 2048-kilobit nonvolatile memory organized as 262,144 bytes of 8 bits. Intel's 28F020 is offered in 32-pin plastic DIP, 32-lead PLCC, and 32-lead TSOP packages. Pin assignments conform to JEDEC standards for byte-wide EPROMs.

Extended erase and program cycling capability is designed into Intel's ETOX™ II (EPROM Tunnel Oxide) process technology. Advanced oxide processing, an optimized tunneling structure, and lower electric field combine to extend reliable cycling beyond that of traditional EEPROMs. With the 12.0V V_{pp} supply, the 28F020 performs a minimum of 10,000 erase and program cycles well within the time limits of the Quick-Pulse Programming™ and Quick-Erase™ algorithms.

Intel's 28F020 employs advanced CMOS circuitry for systems requiring high-performance access speeds, low power consumption, and immunity to noise. Its 150 nanosecond access time provides no-WAIT-state performance for a wide range of microprocessors and microcontrollers. Maximum standby current of 100 μ A translates into power savings when the device is deselected. Finally, the highest degree of latch-up protection is achieved through Intel's unique EPI processing. Prevention of latch-up is provided for stresses up to 100 mA on address and data pins, from $-1V$ to $V_{CC} + 1V$.

With Intel's ETOX II process base, the 28F020 levers years of EPROM experience to yield the highest levels of quality, reliability, and cost-effectiveness.

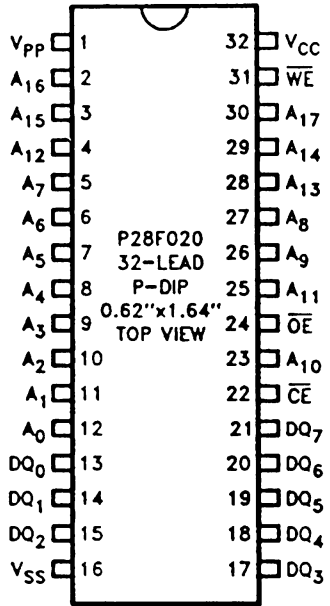


290245-1

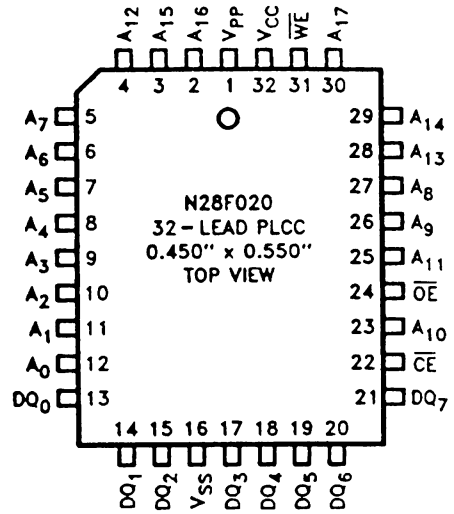
Figure 1. 28F020 Block Diagram

Table 1. Pin Description

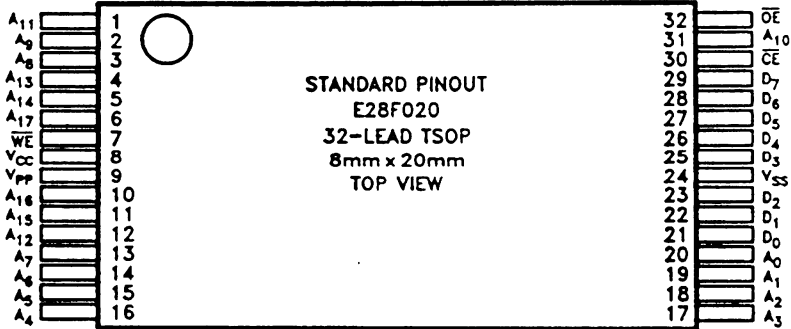
Symbol	Type	Name and Function
A ₀ -A ₁₇	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ -DQ ₇	INPUT/OUTPUT	DATA INPUT/OUTPUT: Inputs data during memory write cycles; outputs data during memory read cycles. The data pins are active high and float to tri-state OFF when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
\overline{CE}	INPUT	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. \overline{CE} is active low; \overline{CE} high deselects the memory device and reduces power consumption to standby levels.
\overline{OE}	INPUT	OUTPUT ENABLE: Gates the devices output through the data buffers during a read cycle. \overline{OE} is active low.
\overline{WE}	INPUT	WRITE ENABLE: Controls writes to the control register and the array. Write enable is active low. Addresses are latched on the falling edge and data is latched on the rising edge of the \overline{WE} pulse. Note: With $V_{PP} \leq 6.5V$, memory contents cannot be altered.
V _{PP}		ERASE/PROGRAM POWER SUPPLY for writing the command register, erasing the entire array, or programming bytes in the array.
V _{CC}		DEVICE POWER SUPPLY (5V \pm 10%)
V _{SS}		GROUND



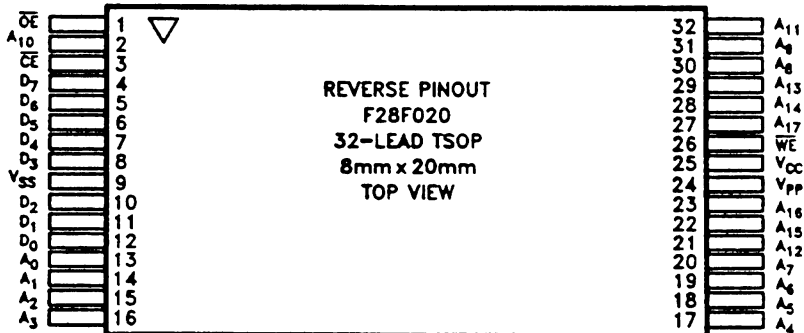
290245-2



290245-3



290245-4



290245-5

Figure 2. 28F020 Pin Configurations

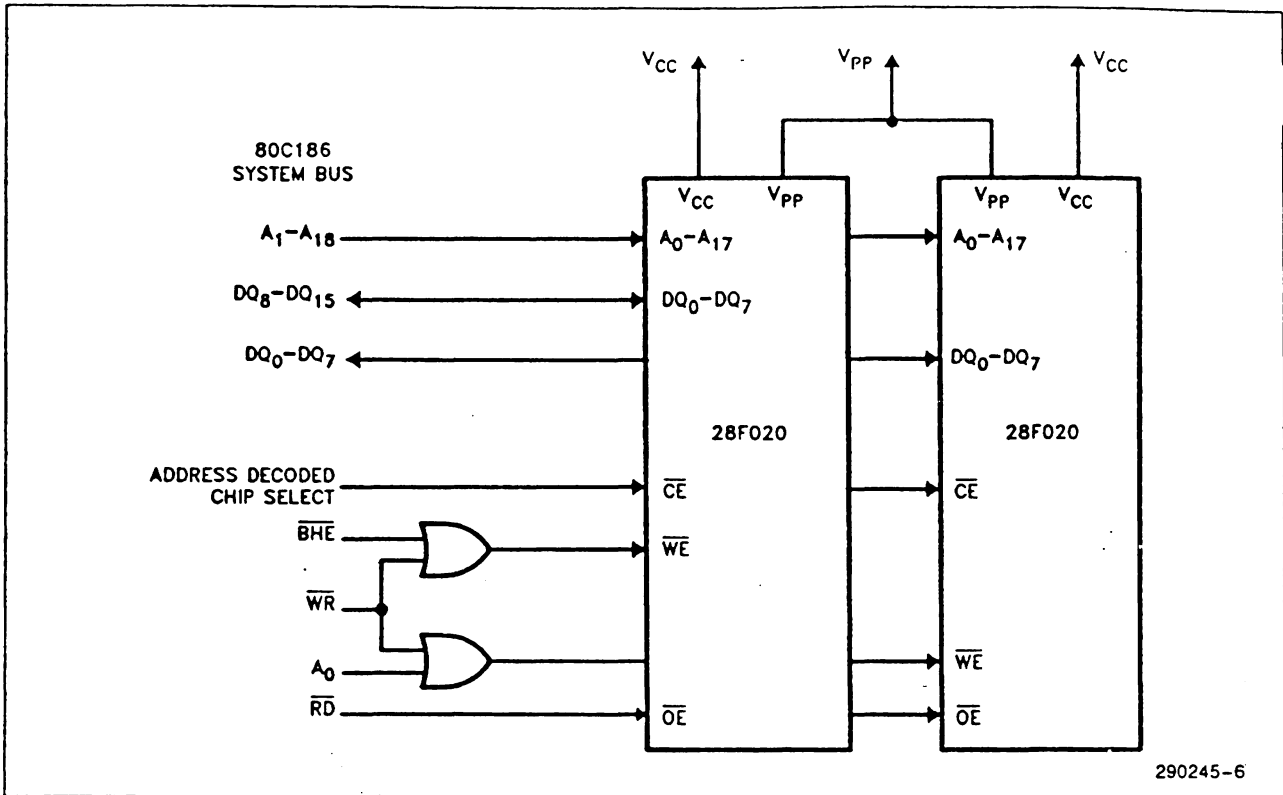


Figure 3. 28F020 in a 80C186 System

PRINCIPLES OF OPERATION

Flash-memory augments EPROM functionality with in-circuit electrical erasure and reprogramming. The 28F020 introduces a command register to manage this new functionality. The command register allows for: 100% TTL-level control inputs; fixed power supplies during erasure and programming; and maximum EPROM compatibility.

In the absence of high voltage on the V_{PP} pin, the 28F020 is a read-only memory. Manipulation of the external memory-control pins yields the standard EPROM read, standby, output disable, and intelligent Identifier™ operations.

The same EPROM read, standby, and output disable operations are available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} enables erasure and programming of the device. All functions associated with altering memory contents—intelligent Identifier, erase, erase verify, program, and program verify—are accessed via the command register.

Commands are written to the register using standard microprocessor write timings. Register contents serve as input to an internal state-machine which controls the erase and programming circuitry. Write cycles also internally latch addresses and data needed for programming or erase operations. With the appropriate command written to the register,

standard microprocessor read timings output array data, access the intelligent Identifier codes, or output data for erase and program verification.

Integrated Stop Timer

Successive command write cycles define the durations of program and erase operations; specifically, the program or erase time durations are normally terminated by associated program or erase verify commands. An integrated stop timer provides simplified timing control over these operations; thus eliminating the need for maximum program/erase timing specifications. Programming and erase pulse durations are minimums only. When the stop timer terminates a program or erase operation, the device enters an inactive state and remains inactive until receiving the appropriate verify or reset command.

Write Protection

The command register is only active when V_{PP} is at high voltage. Depending upon the application, the system designer may choose to make the V_{PP} power supply switchable—available only when memory updates are desired. When V_{PP} = V_{PP(L)}, the contents of the register default to the read command, making the 28F020 a read-only memory. In this mode, the memory contents cannot be altered.

Table 2. 28F020 Bus Operations

		Pins	$V_{PP}(1)$	A_0	A_9	\overline{CE}	\overline{OE}	\overline{WE}	DQ_0-DQ_7
Operation									
READ-ONLY	Read		V_{PPL}	A_0	A_9	V_{IL}	V_{IL}	V_{IH}	Data Out
	Output Disable		V_{PPL}	X	X	V_{IL}	V_{IH}	V_{IH}	Tri-State
	Standby		V_{PPL}	X	X	V_{IH}	X	X	Tri-State
	intelligent Identifier™ (Mfr)(2)		V_{PPL}	V_{IL}	$V_{ID}(3)$	V_{IL}	V_{IL}	V_{IH}	Data = 89H
	intelligent Identifier™ (Device)(2)		V_{PPL}	V_{IH}	$V_{ID}(3)$	V_{IL}	V_{IL}	V_{IH}	Data = BDH
READ/WRITE	Read		V_{PPH}	A_0	A_9	V_{IL}	V_{IL}	V_{IH}	Data Out(4)
	Output Disable		V_{PPH}	X	X	V_{IL}	V_{IH}	V_{IH}	Tri-State
	Standby(5)		V_{PPH}	X	X	V_{IH}	X	X	Tri-State
	Write		V_{PPH}	A_0	A_9	V_{IL}	V_{IH}	V_{IL}	Data In(6)

NOTES:

1. Refer to DC Characteristics. When $V_{PP} = V_{PPL}$ memory contents can be read but not written or erased.
2. Manufacturer and device codes may also be accessed via a command register write sequence. Refer to Table 3. All other addresses low.
3. V_{ID} is the intelligent Identifier high voltage. Refer to DC Characteristics.
4. Read operations with $V_{PP} = V_{PPH}$ may access array data or the intelligent Identifier™ codes.
5. With V_{PP} at high voltage, the standby current equals $I_{CC} + I_{pp}$ (standby).
6. Refer to Table 3 for valid Data-In during a write operation.
7. X can be V_{IL} or V_{IH} .

Or, the system designer may choose to "hardwire" V_{PP} , making the high voltage supply constantly available. In this case, all Command Register functions are inhibited whenever V_{CC} is below the write lockout voltage V_{LKO} . (See Power Up/Down Protection.) The 28F020 is designed to accommodate either design practice, and to encourage optimization of the processor-memory interface.

BUS OPERATIONS**Read**

The 28F020 has two control functions, both of which must be logically active, to obtain data at the outputs. Chip-Enable (\overline{CE}) is the power control and should be used for device selection. Output-Enable (\overline{OE}) is the output control and should be used to gate data from the output pins, independent of device selection. Refer to AC read timing waveforms.

When V_{PP} is high (V_{PPH}), the read operation can be used to access array data, to output the intelligent Identifier™ codes, and to access data for program/erase verification. When V_{PP} is low (V_{PPL}), the read operation can **only** access the array data.

Output Disable

With Output-Enable at a logic-high level (V_{IH}), output from the device is disabled. Output pins are placed in a high-impedance state.

Standby

With Chip-Enable at a logic-high level, the standby operation disables most of the 28F020's circuitry and substantially reduces device power consumption. The outputs are placed in a high-impedance state, independent of the Output-Enable signal. If the 28F020 is deselected during erasure, programming, or program/erase verification, the device draws active current until the operation is terminated.

intelligent Identifier™ Operation

The intelligent Identifier operation outputs the manufacturer code (89H) and device code (BDH). Programming equipment automatically matches the device with its proper erase and programming algorithms.

With Chip-Enable and Output-Enable at a logic low level, raising A9 to high voltage V_{ID} (see DC Characteristics) activates the operation. Data read from locations 0000H and 0001H represent the manufacturer's code and the device code, respectively.

The manufacturer- and device-codes can also be read via the command register, for instances where the 28F020 is erased and reprogrammed in the target system. Following a write of 90H to the command register, a read from address location 0000H outputs the manufacturer code (89H). A read from address 0001H outputs the device code (BDH).

Write

Device erasure and programming are accomplished via the command register, when high voltage is applied to the V_{PP} pin. The contents of the register serve as input to the internal state-machine. The state-machine outputs dictate the function of the device.

The command register itself does not occupy an addressable memory location. The register is a latch

used to store the command, along with address and data information needed to execute the command.

The command register is written by bringing Write-Enable to a logic-low level (V_{IL}), while Chip-Enable is low. Addresses are latched on the falling edge of Write-Enable, while data is latched on the rising edge of the Write-Enable pulse. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the Erase/Programming Waveforms for specific timing parameters.

COMMAND DEFINITIONS

When low voltage is applied to the V_{PP} pin, the contents of the command register default to 00H, enabling read-only operations.

Placing high voltage on the V_{PP} pin enables read/write operations. Device operations are selected by writing specific data patterns into the command register. Table 3 defines these 28F020 register commands.

Table 3. Command Definitions

Command	Bus Cycles Req'd	First Bus Cycle			Second Bus Cycle		
		Operation(1)	Address(2)	Data(3)	Operation(1)	Address(2)	Data(3)
Read Memory	1	Write	X	00H			
Read intelligent Identifier™ Codes(4)	3	Write	X	90H	Read	(4)	(4)
Set-up Erase/Erase(5)	2	Write	X	20H	Write	X	20H
Erase Verify(5)	2	Write	EA	A0H	Read	X	EVD
Set-up Program/Program(6)	2	Write	X	40H	Write	PA	PD
Program Verify(6)	2	Write	X	C0H	Read	X	PVD
Reset(7)	2	Write	X	FFH	Write	X	FFH

NOTES:

- Bus operations are defined in Table 2.
- IA = Identifier address: 00H for manufacturer code, 01H for device code.
EA = Address of memory location to be read during erase verify.
PA = Address of memory location to be programmed.
Addresses are latched on the falling edge of the Write-Enable pulse.
- ID = Data read from location IA during device identification (Mfr = 89H, Device = BDH).
EVD = Data read from location EA during erase verify.
PD = Data to be programmed at location PA. Data is latched on the rising edge of Write-Enable.
PVD = Data read from location PA during program verify. PA is latched on the Program command.
- Following the Read intelligent ID command, two read operations access manufacturer and device codes.
- Figure 6 illustrates the Quick-Erase™ Algorithm.
- Figure 5 illustrates the Quick-Pulse Programming™ Algorithm.
- The second bus cycle must be followed by the desired command register write.

Buffer Control

ma_aclk	O - BT1	Master Address Clock
ma_aen_	O - BT2	Master Address Clock
sl_aclk	O - BT1	Master Address Clock
sl_aen_	O - BT4	Master Address Clock
sbtP1clk	O - BT4	S-bus to P1 latch
sbtP1enX_	O - BT2	S-bus to P1 Data enable.
sbtP1enL_	O - BT2	S-bus to P1 Data enable.
sbtP1enM_	O - BT2	S-bus to P1 Data enable.
P1tsbck	O - BT4	P1 to S-bus latch.
P1tsbenX_	O - BT2	P1 to S-bus Data Enable.
P1tsbenL_	O - BT2	P1 to S-bus Data Enable.
P1tsbenM_	O - BT2	P1 to S-bus Data Enable.
pl_sel(1:0)	O - BT8	Pipeline Buffer Select.
b_asen_	O - BT1	Address Strobe enable.

VME

P1_BR(3:0)_	I - SCHMITTN	VMEbus Request
VME_BR_	O - BT1	VMEbus Request
P1_BGIN(3:0)	I - SCHMITTN	VMEbus Bus Grant
VME_BGOUT(3:0)	O - BT1	VMEbus Bus Grant
VME_BBSY_	O - BT1	VMEbus Bus Busy
B_BBSY_	I - SCHMITTN	VMEbus Bus Busy
P1_AS_	I - SCHMITT	VMEbus Address Strobe
VME_AS_	O - BT1	VMEbus Address Strobe
P1_DS(1:0)_	I - SCHMITTN	VMEbus Data Strobe
VME_DS(1:0)_	O - BT1	VMEbus Data Strobe
P1_WR_	I - SCHMITTN	VMEbus Write
P1_LW_	I - SCHMITTN	VMEbus Long Word
VME_LW_	O - BT1	VMEbus Long Word
P1_IRQ(7:1)_	I - SCHMITTN	VME interrupts.
P1_IACK_	I - SCHMITTN	VMEbus Interrupt Acknowledge
VME_IACK_	O - BT1	VMEbus Interrupt Acknowledge
P1_IACKIN_	I - SCHMITTN	VMEbus Interrupt Acknowledge
VME_IACKOUT	O - BT1	VMEbus Interrupt Acknowledge
P1_DTACK_	I - SCHMITTN	VMEbus Data Acknowledge
VME_DTACK_	O - BT1	VMEbus Data Acknowledge
P1_BERR_	I - SCHMITTN	VMEbus Bus Error
VME_BERR_	O - BT1	VMEbus Bus Error
P1_AM53(5:3)	I - SCHMITT	VMEbus Address Modifier 5-3
P1_AM10(1:0)	I - SCHMITT	VMEbus Address Modifier 1-0
VME_AM(5:4)	O - BT1	VMEbus Address Modifier 5-4
P1_SYSR_	I - SCHMITTN	VMEbus System Reset
VME_SYSR_	O - BT1	VMEbus System Reset
P1_AMEG(23:20)	I - SCHMITT	VMEbus Address bit 23-20
P1_A16K(15:14)	I - SCHMITT	VMEbus Address bit 15-14
P1_A16B(3:1)	I - SCHMITT	VMEbus Address bit 3-1
VME_A(31:29)	O - BT1	VMEbus Address bit 31-29

VME_SYSCLK

O - BT1

VMEBUS Clock

Test

od_
para

I - TLCHT

Output disable

O - B1

Parametric nand tree output

Signals

136

Device Type:

LIA5530

Package Type:

PFP144 (IO:136 VDD:4 VSS:8)

Functional Description

The S4-VME interfaces the Sbus to the VMEbus. When acting as a Sbus slave, i.e. VMEbus Master, VME is a physical device in type 2 or 3 space. During Sbus DMA Master cycles, i.e. VME Slave cycles, it will appear as an Sbus slot device. Refer to the Sbus specification for further explanation.

This chip contains the most of the functions required to implement a full VME Interface. In addition, buffers to drive VME and to interface Sbus to VME, plus some address decoding are needed. The following figure shows how the VME interface will hook up to the rest of the CPU.

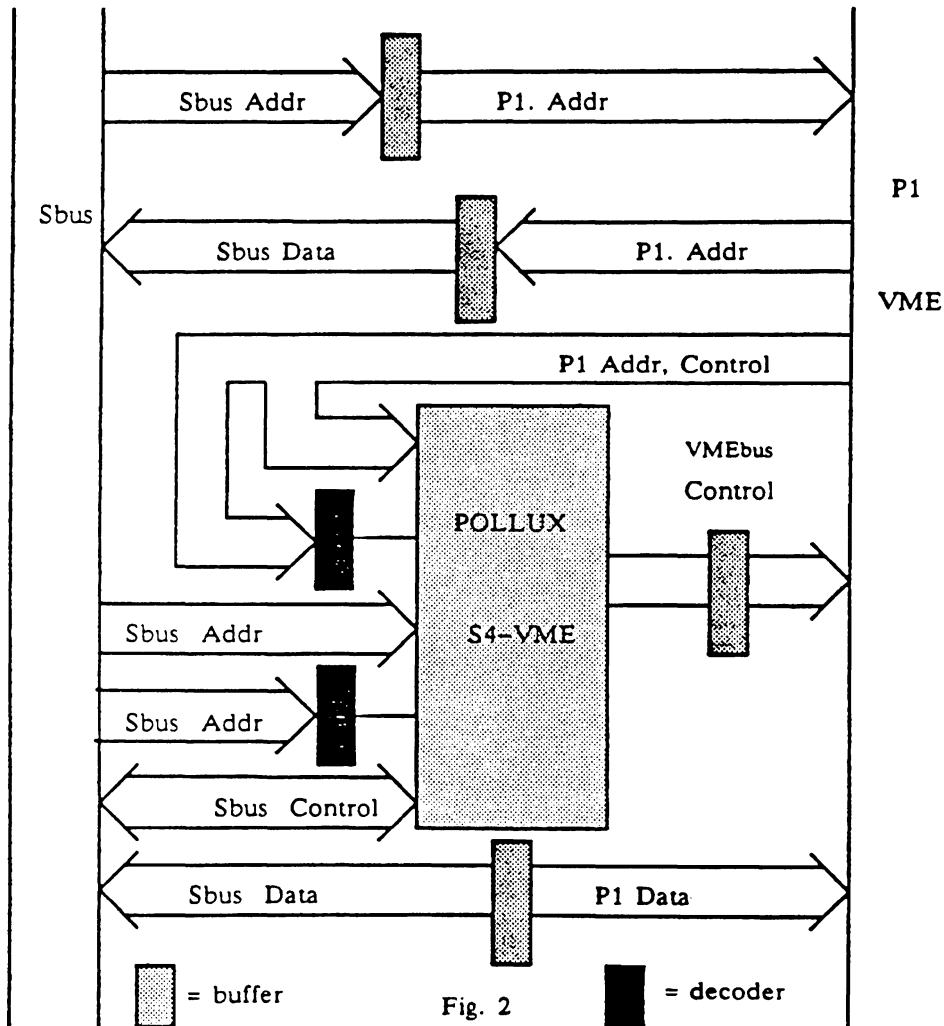


Fig. 2

Registers

The registers can be divided into the following groups:

1) Registers set on system initialization, replacing board jumpers:

Interrupt Enable Register:	Interrupts handled
	Arbitration mode

2) Registers used for multi- or co-processing:

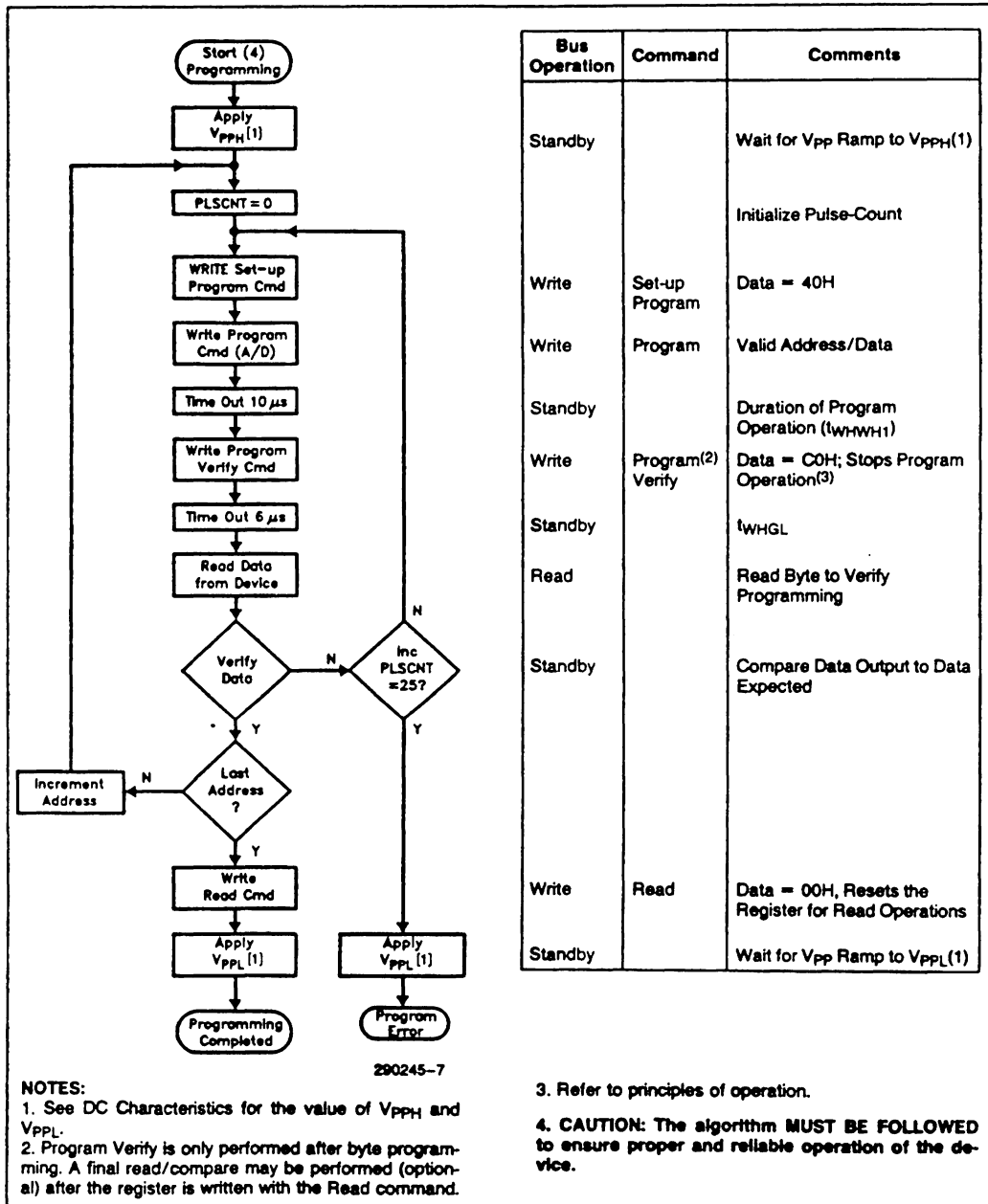
Mail Box Interrupt Register
Bus Locker Register
Slave Map Register (also replacing jumper)

3) Others

A32 VME Address mapping.	Move 512 MByte window
	enable diagnostic loop back.
Slave Map register bit 0:	enable Block Mode transfers

Following is a list of the VME Interface Registers:

Type	Name	Physical Base	Size
1	VME Bus Locker	0xEFE00000	byte
1	VME IACK cycle	0xEFE00001	byte, A[3:1], A0=1
1	Mail Box register	0xEFE00010	byte
1	VME Interrupt Enable Register	0xEFE00014	byte
1	A32MAP Register	0xEFE00018	byte
1	Slave Map Register	0xEFE0001C	byte



Bus Operation	Command	Comments
Standby		Wait for Vpp Ramp to VppH(1)
		Initialize Pulse-Count
Write	Set-up Program	Data = 40H
Write	Program	Valid Address/Data
Standby		Duration of Program Operation (t _{WHWH1})
Write	Program ⁽²⁾ Verify	Data = C0H; Stops Program Operation ⁽³⁾
Standby		t _{WHGL}
Read		Read Byte to Verify Programming
Standby		Compare Data Output to Data Expected
Write	Read	Data = 00H, Resets the Register for Read Operations
Standby		Wait for Vpp Ramp to VpPL(1)

NOTES:

1. See DC Characteristics for the value of VppH and VpPL.
2. Program Verify is only performed after byte programming. A final read/compare may be performed (optional) after the register is written with the Read command.

3. Refer to principles of operation.

4. CAUTION: The algorithm MUST BE FOLLOWED to ensure proper and reliable operation of the device.

Figure 5. 28F020 Quick-Pulse-Programming™ Algorithm

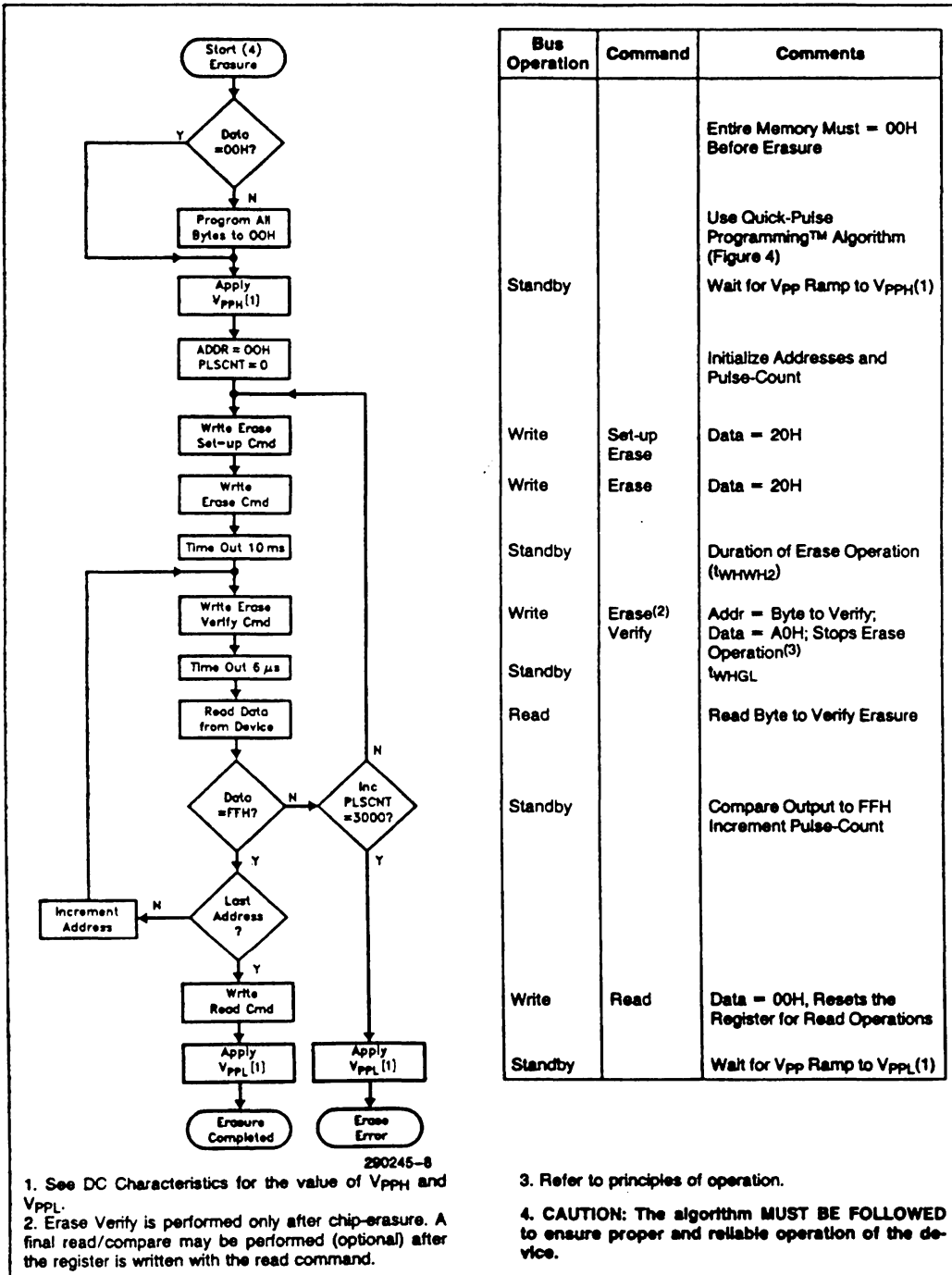


Figure 6. 28F020 Quick-Erase™ Algorithm



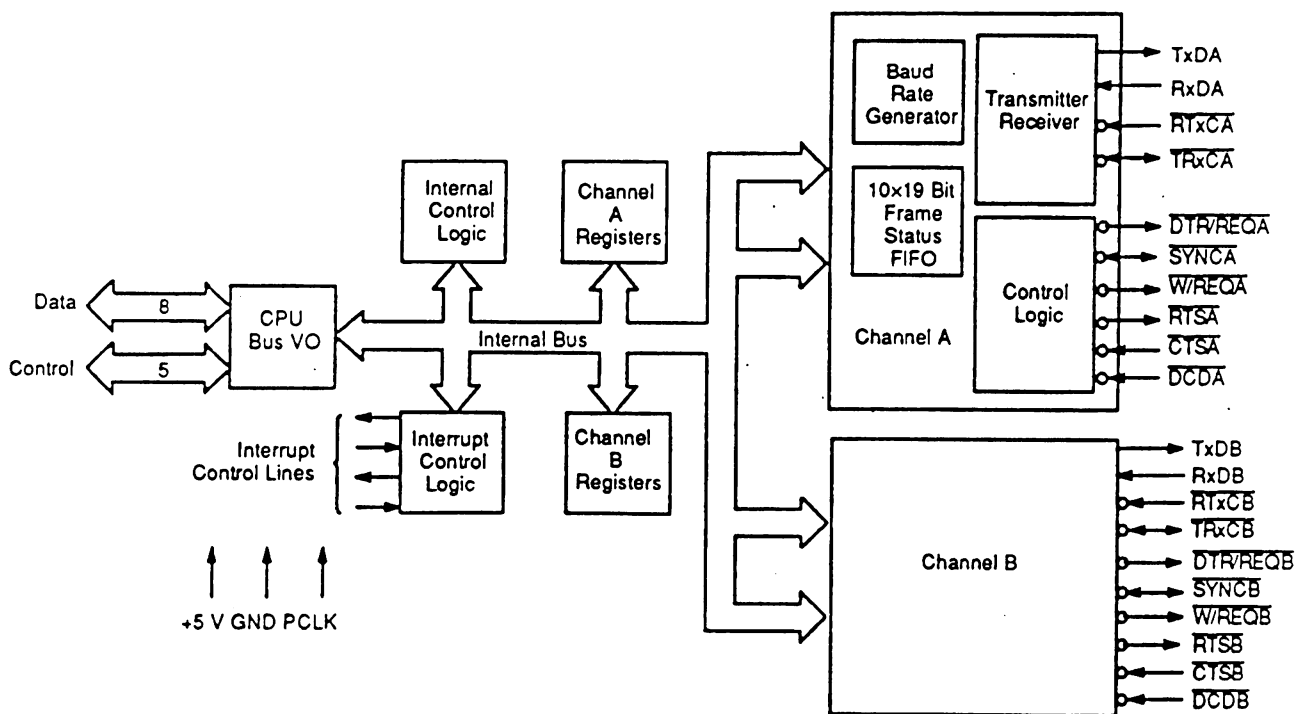
Z85C30

Enhanced Serial Communications Controller

DISTINCTIVE CHARACTERISTICS

- **Fastest data rate of any Z8530**
 - 8.192 MHz / 2.048 Mb/s
 - 10 MHz / 2.5 Mb/s
 - 12.5 MHz / 3 Mb/s
 - 16.384 MHz / 4.096 Mb/s
 - 20 MHz / 5 Mb/s (prelim)
- **Low-power CMOS technology**
- **Pin and function compatible with other NMOS and CMOS Z8530s**
- **Easily Interfaced with most CPUs**
Compatible with non-multiplexed bus
- **Many enhancements over NMOS Z8530H**
 - Allows 85C30 to be used more effectively in high-speed applications
 - Improves interface capabilities
- **Two independent full-duplex serial channels**
- **Asynchronous mode features**
 - Programmable stop bits, clock factor, character length and parity
 - Break detection/generation
 - Error detection for framing, overrun, and parity
- **Synchronous mode features**
 - Supports IBM BISYNC, SDLC, SDLC Loop, HDLC, and ADCCP Protocols
 - Programmable CRC generators and checkers
 - SDLC/HDLC support includes frame control, zero insertion and deletion, abort, and residue handling

BLOCK DIAGRAM



10216A-001A

BD008260

DISTINCTIVE CHARACTERISTICS (continued)

- Enhanced SCC functions support high-speed frame reception using DMA
 - 14-bit byte counter
 - 10 × 19 SDLC/HDLC Frame Status FIFO
 - Independent Control on both channels
 - Enhanced operation does not allow special receive conditions to lock the 3-byte DATA FIFO when the 10 × 19 FIFO is enabled
- Local Loopback and Auto Echo modes
- Internal or external character synchronization
- 2-Mb/s FM encoding transmit and receive capability using internal DPLL for 16.384-MHz product
- Internal synchronization between RxC to PCLK and TxC to PCLK
 - This allows the user to eliminate external synchronization hardware required by the NMOS device when transmitting or receiving data at the maximum rate of 1/4 PCLK frequency.

GENERAL DESCRIPTION

AMD's Z85C30 is an enhanced pin-compatible version of the popular Z8530/Z85C30 Serial Communications Controller. The Enhanced Serial Communications Controller (ESCC) is a high-speed, low-power, multi-protocol communications peripheral designed for use with 8- and 16-bit microprocessors. It has two independent, full-duplex channels and functions as a serial-to-parallel, parallel-to-serial converter/controller. AMD's proprietary enhancements make the Z85C30 easier to interface and more effective in high-speed applications due to a reduction in software burden and the elimination of the need for some external glue logic.

The Z85C30 is easy to use due to a variety of sophisticated internal functions, including on-chip baud rate generators, digital phase-locked loops, and crystal oscillators, which dramatically reduce the need for external logic. The device can generate and check CRC codes in any SYNC mode, and can be programmed to check data integrity in various modes. The ESCC also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

This versatile device supports virtually any serial data transfer application such as networks, modems, cassettes, and tape drivers. The ESCC is designed for non-multiplexed buses and is easily interfaced with most CPUs, such as 80188, 80186, 80286, 8080, Z80, 6800, 68000 and MULTIBUS.

Enhancements that allow the Z85C30 to be used more effectively in high-speed applications include:

- a 10 × 19 bit SDLC/HDLC frame status FIFO array
- a 14-bit SDLC/HDLC frame byte counter

- automatic SDLC/HDLC opening frame flag transmission
- TxD pin forced High in SDLC NRZI mode after closing flag
- automatic SDLC/HDLC Tx underrun/EOM flag reset
- automatic SDLC/HDLC Tx CRC generator reset/preset
- \overline{RTS} synchronization to closing SDLC/HDLC flag
- $\overline{DTR}/\overline{REQ}$ deactivation delay significantly reduced
- external PCLK to \overline{RxC} or \overline{TxC} synchronization requirement eliminated for PCLK divide-by-four operation

Other enhancements to improve the Z85C30 interface capabilities include:

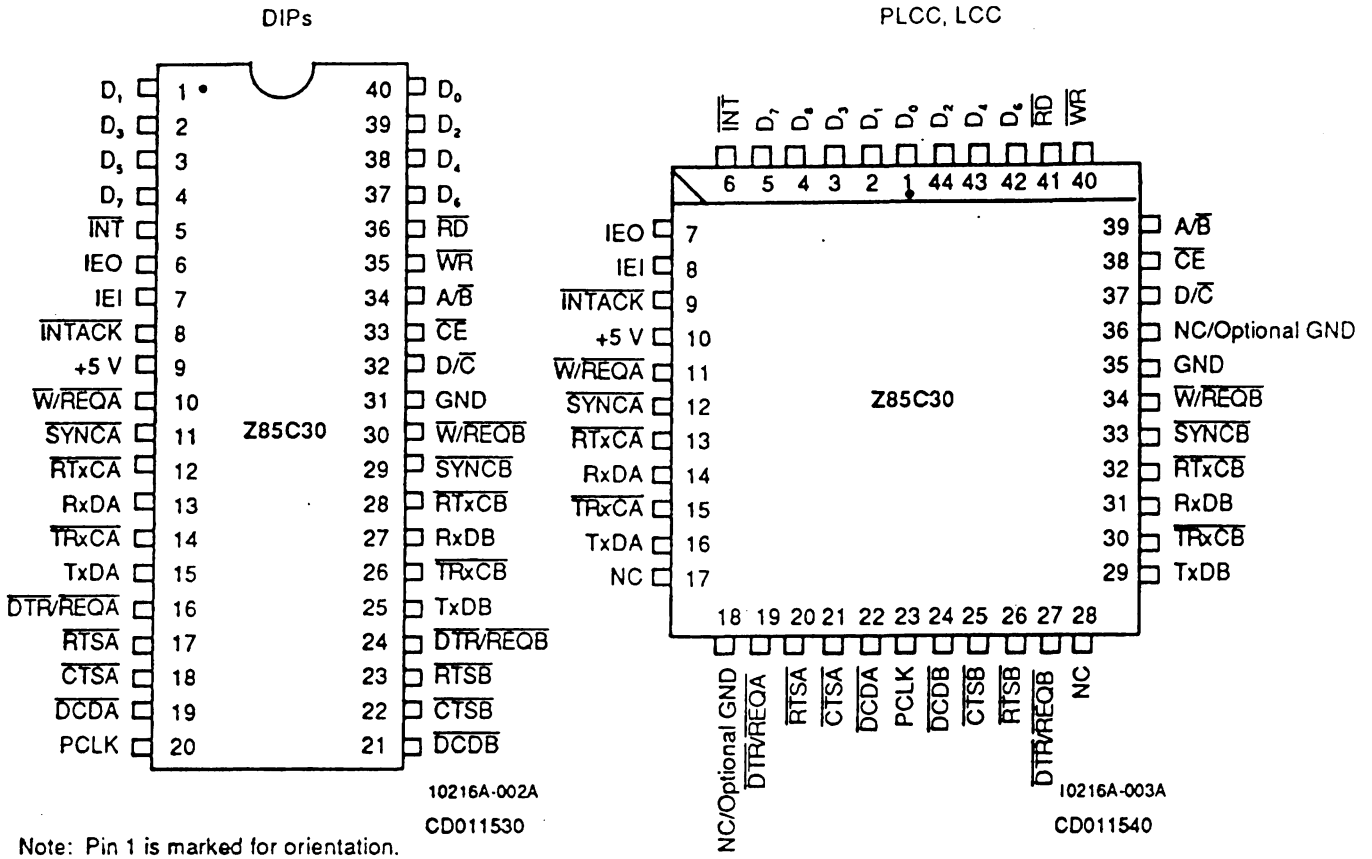
- write data valid setup time to falling edge of \overline{WR} requirement eliminated
- reduced \overline{INT} response time
- reduced access recovery time (t_{ac}) to 3 PCLK best case (3 1/2 PCLK worst case)
- improved \overline{Wait} timing
- Write Registers WR3, WR4, WR5, and WR10 made readable
- lower priority interrupt masking without \overline{INTACK}
- complete SDLC/HDLC CRC character reception

RELATED AMD PRODUCTS

Part No.	Description	Part No.	Description
Am7960	Coded Data Transceiver	Am9517A	DMA Controller
80186	Highly Integrated 16-Bit Microprocessor	5380, 53C80	SCSI Bus Controller
80286, 80C286	High-Performance 16-Bit Microprocessor	80188	Highly Integrated 8-Bit Microprocessor

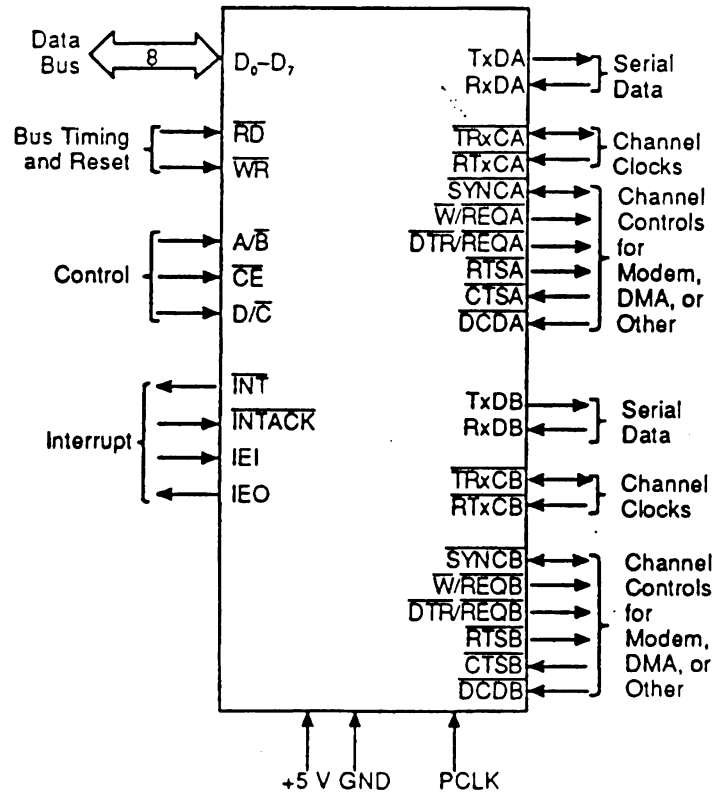
CONNECTION DIAGRAMS

Top View



2

LOGIC SYMBOL



Z85C30

2-287

PIN DESCRIPTION

Bus Timing and Reset

\overline{RD}

Read (Input; Active Low)

This signal indicates a Read operation and, when the SCC is selected, enables the SCC's bus drivers. During the Interrupt Acknowledge cycle, this signal gates the interrupt vector onto the bus if the SCC is the highest priority device requesting an interrupt.

\overline{WR}

Write (Input; Active Low)

When the SCC is selected, this signal indicates a Write operation. The coincidence of \overline{RD} and \overline{WR} is interpreted as a reset.

Channel Clocks

\overline{RTxCA} , \overline{RTxCB}

Receive/Transmit Clocks (Inputs; Active Low)

These pins can be programmed in several different modes of operation. In each channel, \overline{RTxC} may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock of the digital phase-locked loop. These pins can also be programmed for use with the respective \overline{SYNC} pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

\overline{TRxCA} , \overline{TRxCB}

Transmit/Receive Clocks (Inputs/Outputs; Active Low)

These pins can be programmed in several different modes of operation. \overline{TRxC} may supply the receive clock or the transmit clock in the input mode or supply the output of the digital phase-locked loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

Channel Controls for Modem, DMA, or Other

\overline{CTSA} , \overline{CTSB}

Clear to Send (Inputs; Active Low)

If these pins are programmed as Auto Enables, a Low on these inputs enables their respective transmitters. If not programmed as Auto Enables, they may be used as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow rise-time inputs. The SCC detects pulses on these inputs and may interrupt the CPU on both logic level transitions.

\overline{DCDA} , \overline{DCDB}

Data Carrier Detect (Inputs; Active Low)

These pins function as receiver enables if they are programmed as Auto Enables; otherwise, they may be used as general-purpose input pins. Both are Schmitt-trigger buffered to accommodate slow rise-time signals. The SCC detects pulses on these pins and may interrupt the CPU on both logic level transitions.

$\overline{DTR/REQA}$

Data Terminal Ready/Request (Outputs; Active Low)

These outputs follow the inverted state programmed into the DTR bit in WR5. They can also be used as general-purpose outputs or as Request Lines for a DMA controller.

\overline{RTSA} , \overline{RTSB}

Request to Send (Outputs; Active Low)

When the Request to Send (RTS) bit in Write Register 5 is set, the \overline{RTS} signal goes Low. When the RTS bit is reset in the asynchronous mode and Auto Enable is on, the signal goes High after the transmitter is empty. In SYNC mode, or in asynchronous mode with Auto Enable off, the \overline{RTS} pins strictly follow the inverted state of the RTS bit. Both pins can be used as general-purpose outputs.

In SDLC mode, the AUTO RTS RESET enhancement described later in this document brings \overline{RTS} High after the last 0 of the closing flag leaves the TxD pin.

\overline{SYNCA} , \overline{SYNCB}

Synchronization (Inputs/Outputs; Active Low)

These pins can act either as inputs, outputs, or part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to \overline{CTS} and \overline{DCD} . In this mode, transitions on these lines affect the state of the Sync/Hunt status bits in Read Register 0 but have no other function.

In External Synchronization mode with the crystal oscillator not selected, these lines also act as inputs. In this mode, \overline{SYNC} must be driven Low two receive clock cycles after the last bit in the SYNC character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of \overline{SYNC} .

In the Internal Synchronization mode (Monosync and Bisync) with the crystal oscillator not selected, these pins act as outputs and are active only during the part of the receive clock cycle in which SYNC characters are recognized. The SYNC condition is not latched, so these outputs are active each time a SYNC pattern is recognized (regardless of character boundaries). In SDLC mode, these pins act as outputs and are valid on receipt of a flag.

$\overline{W/REQA}$, $\overline{W/REQB}$

Wait/Request (Outputs; Open drain when programmed for a Wait function, driven High or Low when programmed for a Request function)

These dual-purpose outputs may be programmed as Request lines for a DMA controller or as Wait lines to synchronize the CPU to the SCC data rate. The reset state is Wait.

Control

A/\bar{B}

Channel A/Channel B Select (Input)

This signal selects the channel in which the Read or Write operation occurs.

\bar{CE}

Chip Enable (Input; Active Low)

This signal selects the SCC for a Read or Write operation.

D/\bar{C}

Data/Control Select (Input)

This signal defines the type of information transferred to or from the SCC. A High means data is transferred; a Low indicates a command is transferred.

Data Bus

D_7-D_0

Data Bus (Input/Output; Three State)

These lines carry data and commands to and from the SCC.

Interrupt

IEI

Interrupt Enable In (Input; Active High)

IEI is used with IEO to form an interrupt daisy chain when there is more than one interrupt-driven device. A High IEI indicates that no other higher priority device has an interrupt under service or is requesting an interrupt.

IEO

Interrupt Enable Out (Output; Active High)

IEO is High only if IEI is High and the CPU is not servicing an SCC interrupt or the SCC is not requesting an interrupt (interrupt acknowledge cycle only). IEO is connected to the next lower priority device's IEI input and thus inhibits interrupts from lower priority devices.

\bar{INT}

Interrupt Request (Output; Active Low, Open Drain)

This signal is activated when the SCC requests an interrupt.

\bar{INTACK}

Interrupt Acknowledge (Input; Active Low)

This signal indicates an active interrupt acknowledge cycle. During this cycle, the SCC interrupt daisy chain settles. When \bar{RD} becomes active, the SCC places an interrupt vector on the data bus (if IEI is High). \bar{INTACK} is latched by the rising edge of PCLK.

Serial Data

RxDa, RxDB

Receive Data (Inputs; Active High)

These input signals receive serial data at standard TTL levels.

TxDa, TxDB

Transmit Data (Outputs; Active High)

These output signals transmit serial data at standard TTL levels.

Miscellaneous

GND

Ground

PCLK

Clock (Input)

This is the master SCC clock used to synchronize internal signals. PCLK is not required to have any phase relationship with the master system clock. PCLK is a TTL-level signal. Maximum transmit rate is 1/4 PCLK.

V_{CC}

+ 5 V Power Supply

ARCHITECTURE

The ESCC internal structure includes two full-duplex channels, two 10×19 bit SDLC/HDLC frame status FIFOs, two baud rate generators, internal control and interrupt logic, and a bus interface to a non-multiplexed bus. Associated with each channel are a number of Read and Write registers for mode control and status information, as well as logic necessary to interface with modems or other external devices (see Logic Symbol).

The logic for both channels provides formats, synchronization, and validation for data transferred to and from the channel interface. The modem control inputs are monitored by the control logic under program control. All of the modem control signals are general-purpose in nature and can optionally be used for functions other than modem control.

The register set for each channel includes ten control (Write) registers, two SYNC character (Write) registers, and four status (Read) registers. In addition, each baud rate generator has two (Read/Write) registers for holding the time constant that determines the baud rate. Finally, associated with the interrupt logic is a Write register for the interrupt vector accessible through either channel, a Write-only Master Interrupt Control register, and three

Read registers: one containing the vector with status information (Channel B only), one containing the vector without status (A only), and one containing the interrupt pending bits (A only).

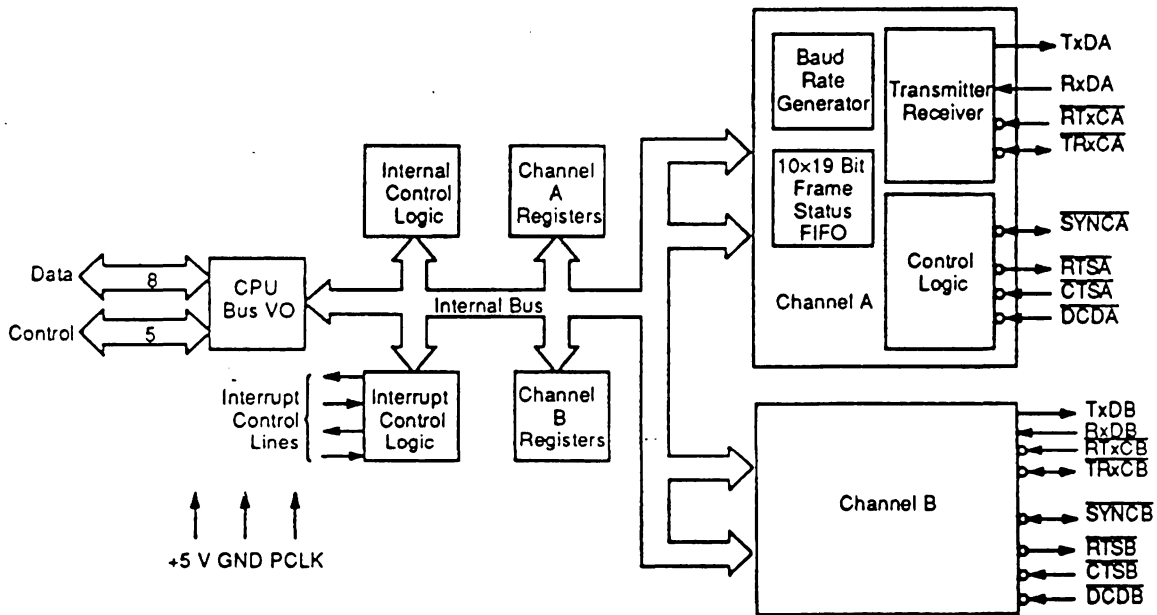
The registers for each channel are designated as follows:

WR0–WR15—Write Registers 0 through 15. An additional Write register, WR7 Prime (WR7'), is available for enabling or disabling additional SDLC/HDLC enhancements if bit D₀ of WR15 is set.

RR0–RR3, RR10, RR12, RR13, RR15—Read Registers 0 through 3, 10, 12, 13, and 15.

If bit D₂ of WR15 is set, then two additional Read registers, RR6 and RR7, are available. These registers are used with the 10×19 bit Frame Status FIFO.

Table 1 lists the functions assigned to each Read and Write register. The ESCC contains only one WR2 and WR9, but they can be accessed by either channel. All other registers are paired (one for each channel).



10216A-001A

BD008260

Figure 1. Block Diagram of ESCC Architecture

→ Data Path

The transmit and receive data path illustrated in Figure 2 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement, in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service an interrupt at the beginning of a block of high-speed data. Incoming data are routed through one of several paths (data or CRC) depending on the selected mode (the character length in asynchronous modes also determines the data path).

The transmitter has an 8-bit transmit data buffer register loaded from the internal data bus and a 20-bit transmit shift register that can be loaded either from the sync-character registers or from the transmit data register. Depending on the operational mode, outgoing data are routed through one of four main paths before they are transmitted from the Transmit Data output (TxD).

Table 1. Read and Write Register Functions

Read Register Functions		Write Register Functions	
RR0	Transmit/Receive buffer status and External status	WR0	Command Register, Register Pointers CRC initialize, initialization commands for the various modes, shift right/shift left command
RR1	Special Receive Condition status (also 10 × 19 bit FIFO Frame Reception Status if WR15 bit D ₂ is set)	WR1	Interrupt conditions and data transfer mode definition
RR2	Modified interrupt vector (Channel B only) Unmodified interrupt vector (Channel A only)	WR2	Interrupt vector (accessed through either channel)
RR3	Interrupt Pending bits (Channel A only)	WR3	Receive parameters and control
RR6	LSB Byte Count (14-bit counter) (if WR15 bit D ₂ set)	WR4	Transmit/Receive miscellaneous parameters and modes
RR7	MSB Byte Count (14-bit counter) and 10 × 19 bit FIFO Status (if WR15 bit D ₂ is set)	WR5	Transmit parameters and controls
RR8	Receive buffer	WR6	Sync character or SDLC address field
RR10	Miscellaneous XMTR, RCVR status	WR7	Sync character or SDLC flag
RR12	Lower byte of baud rate generator time constant	WR7'	SDLC/HDLC enhancements (if bit D ₀ of WR15 is set)
RR13	Upper byte of baud rate generator time constant	WR8	Transmit buffer
RR15	External/Status interrupt information	WR9	Master interrupt control and reset (accessed through either channel)
		WR10	Miscellaneous transmitter/receiver control bits, data encoding
		WR11	Clock mode control, Rx and Tx clock source
		WR12	Lower byte of baud rate generator time constant
		WR13	Upper byte of baud rate generator time constant
		WR14	Miscellaneous control bits, DPLL control
		WR15	External/Status interrupt control

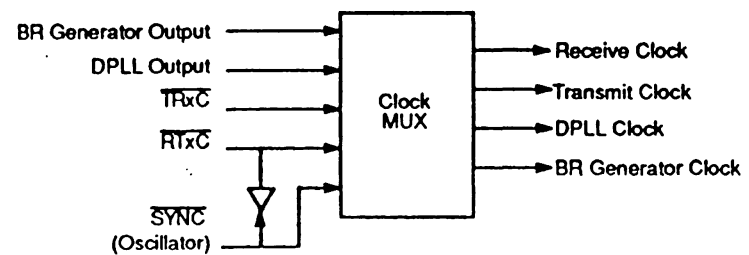
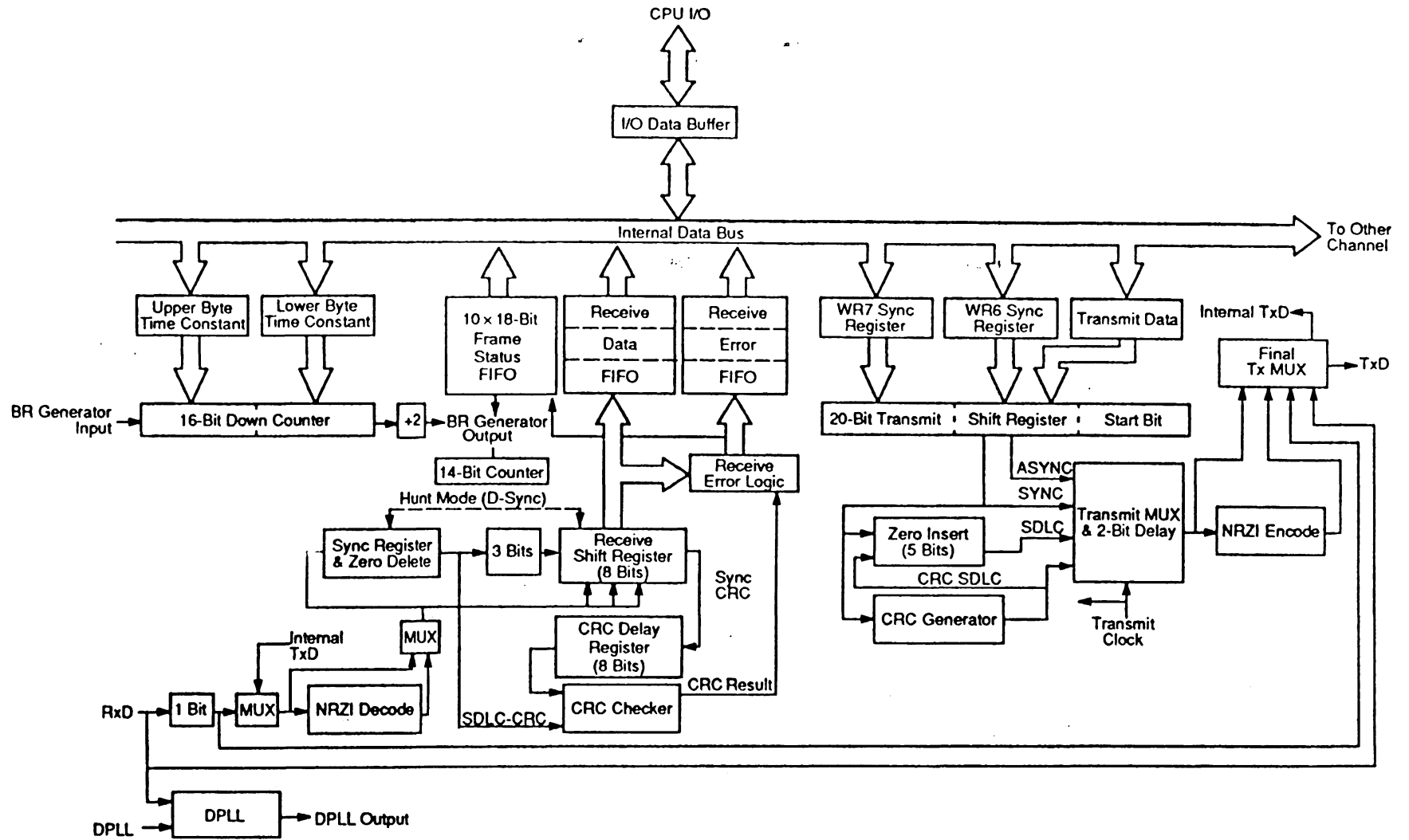


Figure 2. Data Path



Linear Products

DESCRIPTION

The NE5170 is an octal line driver which is designed for digital communications with data rates up to 100kb/s. This device meets all the requirements of EIA standards RS-232C/RS-423A and CCITT recommendations V.10/X.26. Three programmable features: (1) output slew rate, (2) output voltage level, and (3) 3-State control (high-impedance) are provided so that output characteristics may be modified to meet the requirements of specific applications.

FEATURES

- Meets EIA RS-232C/423A and CCITT V.10/X.26
- Simple slew rate programming with a single external resistor
- 0.1 to 10V/ μ s slew rate range
- High/Low programmable voltage output modes
- TTL compatible inputs

APPLICATIONS

- High-speed modems
- High-speed parallel communications
- Computer I/O ports
- Logic level translation

FUNCTION TABLE

ENABLE	LOGIC INPUT	OUTPUT VOLTAGE (V)		
		RS-423A ¹	RS-232C	
			Low Output Mode ¹	High Output Mode ²
L	L	5 to 6V	5 to 6V	$\geq 9V$
L	H	-5 to -6V	-5 to -6V	$\leq -9V$
H	X	Hi-Z	Hi-Z	Hi-Z

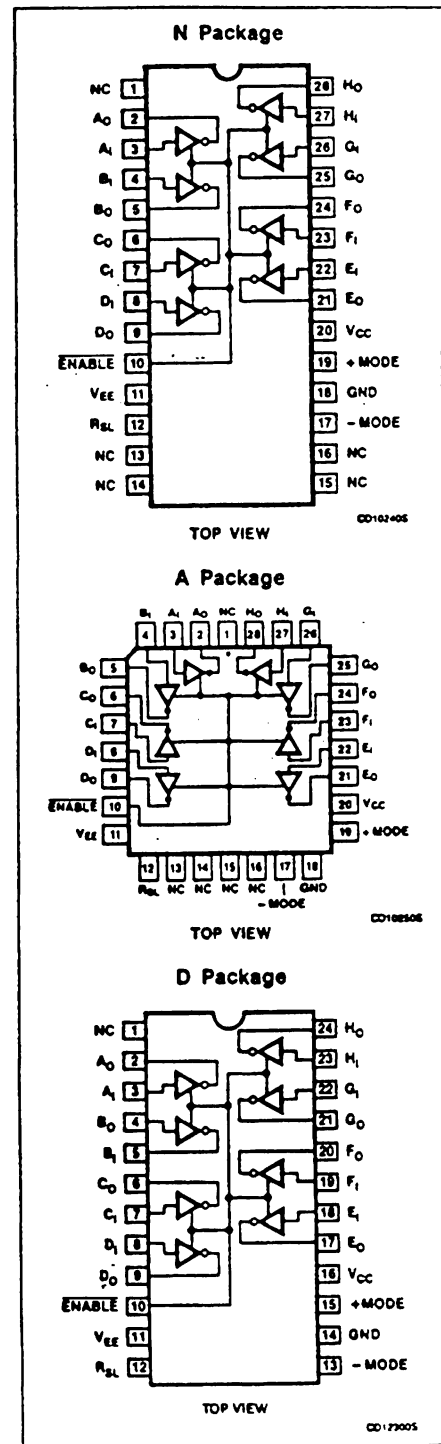
NOTES:

1. $V_{CC} = +10V$ and $V_{EE} = -10V$; $R_L = 3k\Omega$
2. $V_{CC} = +12V$ and $V_{EE} = -12V$; $R_L = 3k\Omega$

ORDERING CODE

DESCRIPTION	TEMPERATURE RANGE	ORDER CODE
28-Pin Plastic DIP	0 to +70°C	NE5170N
28-Pin PLCC	0 to +70°C	NE5170A
24-Pin SO package	0 to +70°C	NE5170D

PIN CONFIGURATIONS



Octal Line Driver

NE5170

ABSOLUTE MAXIMUM RATINGS

SYMBOL	PARAMETER	RATING	UNIT
V _{CC}	Supply voltage and + MODE	15	V
V _{EE}	Supply voltage and - MODE	-15	V
I _{OUT}	Output current ¹	± 150	mA
V _{IN}	Input voltage (ENABLE, Data)	-1.5 to +7	V
V _{OUT}	Output voltage ²	± 15	V
	Minimum slew resistor ³	1	kΩ
P _D	Power dissipation	1200	mW

DC ELECTRICAL CHARACTERISTICS V_{CC} = 10V ± 10%; V_{EE} = -10V ± 10%; ± MODES = 0V; R_{SL} = 2kΩ, 0°C ≤ T_A ≤ 70°C, unless otherwise specified.

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS		UNIT
			Min	Max	
V _{OH}	Output High voltage	V _{IN} = 0.8V R _L = 3kΩ ⁴	5	6	V
		R _L = 450Ω ⁴	4.5	6	
		R _L = 3kΩ ⁵ , C _L = 2500pF	V _{CC} - 3		
V _{OL}	Output Low voltage	V _{IN} = 2.0V R _L = 3kΩ ⁴	-6	-5	V
		R _L = 450Ω ⁴	-6	-4.5	
		R _L = 3kΩ ⁵ , C _L = 2500pF		V _{EE} + 3	
V _{OU}	Output unbalance voltage	V _{CC} = V _{EE} , R _L = 450Ω ⁴		0.4	V
I _{CEX}	Output leakage current	V _O = 6V, ENABLE = 2V or V _{CC} = V _{EE} = 0V	-100	100	μA
V _{IH}	Input High voltage		2.0		V
V _{IL}	Input Low voltage			0.8	V
I _{IL}	Logic "0" input current	V _{IN} = 0.4V	-400	0	μA
I _{IH}	Logic "1" input current	V _{IN} = 2.4V	0	40	μA
I _{OS}	Output short circuit current ¹	V _O = 0V	-150	150	mA
V _{CL}	Input clamp voltage	I _{IN} = -15mA	-1.5		V
I _{CC}	Supply current	No Load		35	mA
I _{EE}		No Load	-45		mA

NOTES:

- 1 Maximum current per driver. Do not exceed maximum power dissipation if more than one output is on.
- 2 High-impedance mode.
- 3 Minimum value of the resistor used to set the slew rate.
- 4 V_{OH}, V_{OL} at R_L = 450Ω will be ≥ 90% of V_{OH}, V_{OL} at R_L = ∞.
- 5 High Output Mode; +MODE pin = V_{CC}; -MODE pin = V_{EE}; 9V < V_{CC} < 13V; -9V > V_{EE} > -13V.

Octal Line Driver

NE5170

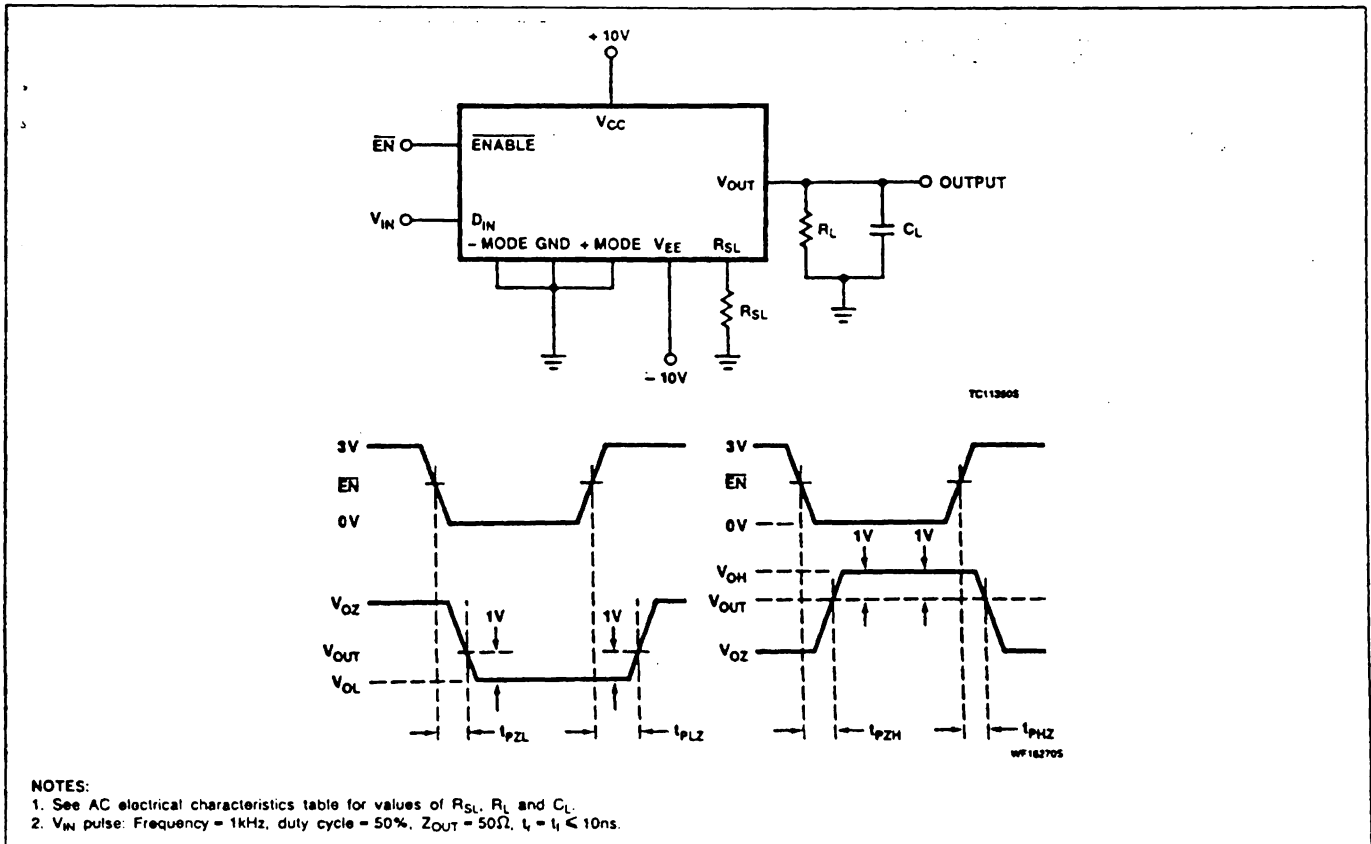
AC ELECTRICAL CHARACTERISTICS $V_{CC} = +10V$; $V_{EE} = -10V$; Mode = GND, $0^{\circ}C \leq T_A \leq 70^{\circ}C$

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS		UNIT
			Min	Max	
t_{PHZ}	Propagation delay output high to high-impedance	$R_L = 450, C_L = 50pF$ or $R_L = 3k, C_L = 2500pF$		5	μs
t_{PLZ}	Propagation delay output low to high-impedance	$R_L = 450, C_L = 50pF$ or $R_L = 3k, C_L = 2500pF$		5	μs
t_{PZH}	Propagation delay high-impedance to high output	$R_{SL} = 200k$ $R_L = 450, C_L = 50pF$ or $R_L = 3k, C_L = 2500pF$		150	μs
t_{PZL}	Propagation delay high-impedance to low output	$R_{SL} = 200k$ $R_L = 450, C_L = 50pF$ or $R_L = 3k, C_L = 2500pF$		150	μs
SR	Output slew rate ¹	$R_{SL} = 2k$	8	12	V/ μs
		$R_{SL} = 20k$	0.8	1.2	
		$R_{SL} = 200k$	0.06	0.14	

NOTE:

SR: Load condition. (A) For $R_{SL} < 4k\Omega$ use $R_L = 450\Omega$; $C_L = 50pF$; (B) for $R_{SL} > 4k\Omega$ use either $R_L = 450\Omega$, $C_L = 50pF$ or $R_L = 3k\Omega$, $C_L = 2500pF$.

AC PARAMETER TEST CIRCUIT AND WAVEFORMS



NE5180/NE5181

Octal Differential Line Receivers

Preliminary Specification

Linear Products

DESCRIPTION

The NE5180 and NE5181 are octal line receivers designed to interface data terminal equipment with data communications equipment. These devices meet the requirements of EIA standards RS-232C, RS-423A, RS-422A, and CCITT V.10, V.11, V.28, X.26 and X.27. The NE5180 is intended for use where the data transmission rate is up to 200 kb/s. The NE5181 covers the entire range of data rates up to 10 Mb/s. The difference in data rates for the two devices results from the input filtering of the NE5180. These devices also provide a failsafe feature which protects against certain input fault conditions.

FEATURES

- Meets EIA RS-232C/423A/422A and CCITT V.10, V.11, V.28
- Single +5V supply — TTL compatible outputs
- Differential inputs withstand $\pm 25V$
- Failsafe feature
- Input noise filter (NE5180 only)
- Internal hysteresis
- Available in SMD PLCC

APPLICATIONS

- High-speed modems
- High-speed parallel communications
- Computer I/O ports
- Logic level translation

FUNCTION TABLE

INPUT	FAILSAFE INPUT	LOGIC OUTPUT
$V_{ID} > 200mV^1$	X	H
$V_{ID} < -200mV^1$	X	L
Both inputs open or grounded	0V	L
	V _{CC}	H

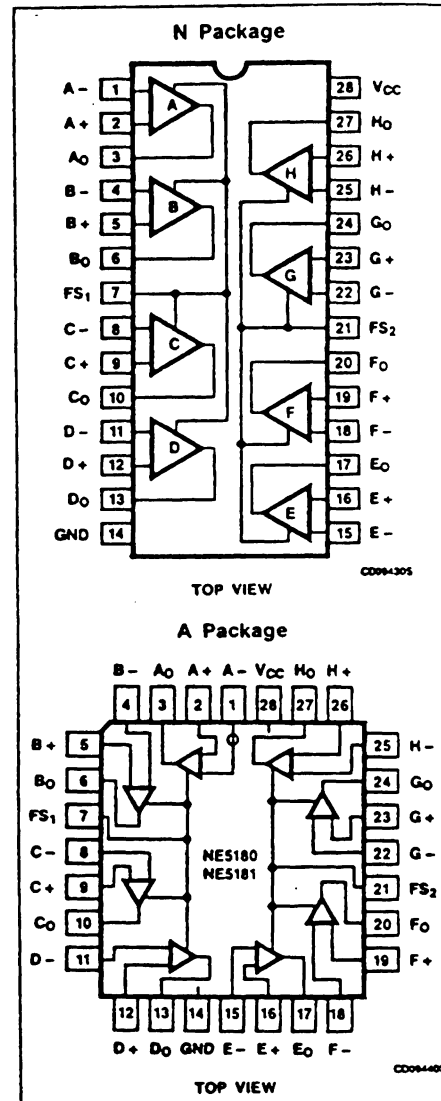
NOTE:

1. V_{ID} is defined as the non-inverting terminal input voltage minus the inverting terminal input voltage.

ORDERING INFORMATION

DESCRIPTION	TEMPERATURE RANGE	ORDER CODE
28-Pin Plastic DIP	0 to +70°C	NE5180N
28-Pin Plastic DIP	0 to +70°C	NE5181N
28-Pin PLCC	0 to +70°C	NE5180A
28-Pin PLCC	0 to +70°C	NE5181A

PIN CONFIGURATIONS

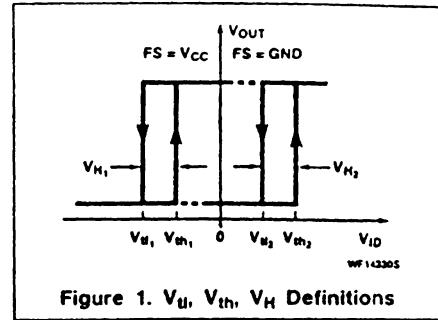


Octal Differential Line Receivers

NE5180/NE5181

ABSOLUTE MAXIMUM RATINGS $T_A = +25^\circ\text{C}$

SYMBOL	PARAMETER	RATING	UNIT
P_D	Power dissipation	800	mW
V_{CC}	Supply voltage	7	V
V_{CM}	Common-mode range	± 15	V
V_{ID}	Differential input voltage	± 25	V
I_{SINK}	Output sink current	50	mA
V_{FS}	Failsafe voltage	-0.3 to V_{CC}	V
t_{OS}	Output short-circuit time	1	sec

DC ELECTRICAL CHARACTERISTICS $V_{CC} = +5V \pm 5\%$, $0^\circ\text{C} < T_A < +70^\circ\text{C}$, input common-mode range $\pm 7V$

SYMBOL	PARAMETER	TEST CONDITIONS	NE5180		NE5181		UNIT	
			Min	Max	Min	Max		
R_{IN}	DC input resistance	$3V \leq V_{IN} \leq 25V$	3	7	3	7	$k\Omega$	
V_{OFS}	Failsafe output voltage	Inputs open or shorted to GND	$0 < I_{OUT} \leq 8\text{mA}$, $V_{failsafe} = 0V$		0.45		0.45	V
			$0 > I_{OUT} \geq -400\mu\text{A}$, $V_{failsafe} = V_{CC}$	2.7		2.7		
V_{TH}	Differential input high ⁴ threshold	$V_{OUT} \geq 2.7V$, $I_{OUT} = -440\mu\text{A}$	$R_S = 0^1$		0.2		0.2	V
			$R_S = 500^1$		0.4		0.4	
V_{tl}	Differential input low ⁴ threshold	$V_{OUT} \leq 0.45V$, $I_{OUT} = 8\text{mA}$	$R_S = 0^1$	-0.2		-0.2		V
			$R_S = 500^1$	-0.4		-0.4		
V_H	Hysteresis ⁴	$FS = 0V$ or V_{CC} (See Figure 1)	50	140	50	140	mV	
V_{IOC}	Open-circuit input voltage			2		2	V	
C_I	Input capacitance			30		30	pF	
V_{OH}	High level output voltage	$V_{ID} = 1V$, $I_{OUT} = -440\mu\text{A}$	2.7		2.7		V	
V_{OL}	Low level output voltage	$V_{ID} = -1V$	$I_{OUT} = 4\text{mA}^2$		0.4		0.4	V
			$I_{OUT} = 8\text{mA}^2$		0.45		0.45	
I_{OS}	Short-circuit output current	$V_{ID} = 1V$, Note 3	20	100	20	100	mA	
I_{CC}	Supply current	$4.75V \leq V_{CC} \leq 5.25V$, $V_{ID} = -1V$; $FS = 0V$		100		100	mA	
I_{IN}	Input current	Other inputs grounded	$V_{IN} = +10V$		3.25		3.25	mA
			$V_{IN} = -10V$	-3.25		-3.25		

NOTES

- R_S is a resistor in series with each input.
- Measured after 100ms warm-up (at 0°C).
- Only 1 output may be shorted at a time and then only for a maximum of 1 second.
- See Figure 1 for threshold and hysteresis definitions.

AC ELECTRICAL CHARACTERISTICS $V_{CC} = +5V \pm 5\%$, $0^\circ\text{C} < T_A < +70^\circ\text{C}$

SYMBOL	PARAMETER	TEST CONDITIONS	NE5180		NE5181		UNIT
			Min	Max	Min	Max	
t_{PLH}	Propagation delay — low to high	$C_L = 50\text{pF}$, $V_{ID} = \pm 1V$		500		100	ns
t_{PHL}	Propagation delay — high to low	$C_L = 50\text{pF}$, $V_{ID} = \pm 1V$		500		100	ns
f_a	Acceptable input frequency	Unused input grounded, $V_{ID} = \pm 200\text{mV}^1$		0.1		5.0	MHz
f_r	Rejectable input frequency	Unused input grounded, $V_{ID} = \pm 500\text{mV}$	5.5		NA		MHz

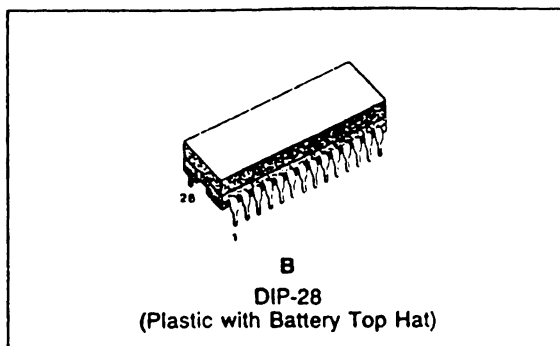
NOTE:

- $V_{ID} = \pm 1V$ for NE5181.

**8K X 8 ZEROPOWER
 TIMEKEEPER RAM**

ADVANCED DATA

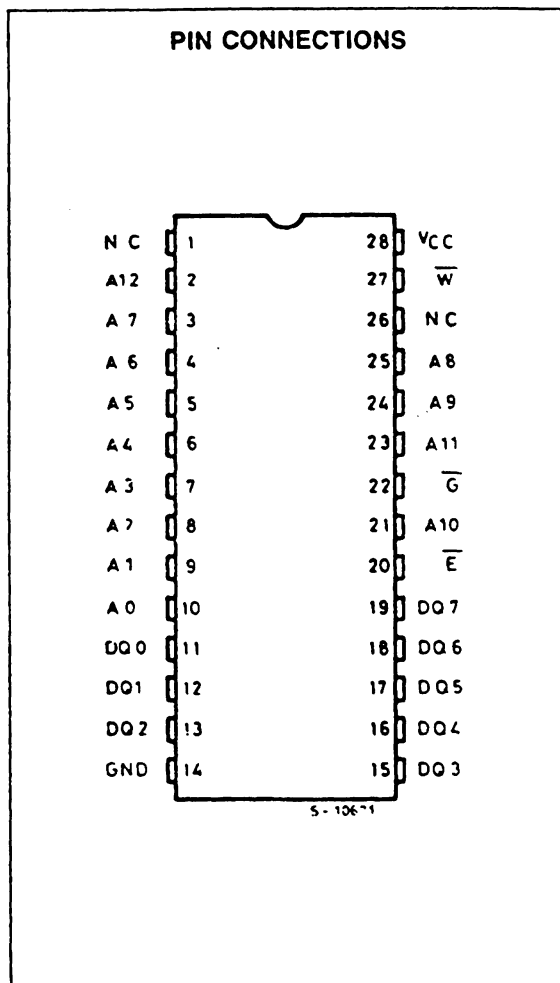
- INTEGRATED ULTRA LOW POWER SRAM, REAL TIME CLOCK, CRISTAL, POWER-FAIL CONTROL CIRCUIT AND BATTERY
- BYTEWIDE™ RAM-LIKE CLOCK ACCESS
- BCD CODED YEAR, MONTH, DAY DATE, HOURS, MINUTES AND SECONDS
- SOFTWARE CONTROLLED CLOCK CALIBRATION FOR HIGH ACCURACY APPLI-CATIONS
- PREDICTED WORST CASE BATTERY STORAGE LIFE OF 11 YEARS @ 70°C
- PIN AND FUNCTION COMPATIBLE WITH JEDEC STANDARD 8K X8 SRAMs
- AUTOMATIC POWER-FAIL CHIP DESELECT/WRITE PROTECTION



Part Number	Access Time	R/W Cycle Time
MK48T08-10	100 ns	100 ns
MK48T08-12	120 ns	120 ns
MK48T08-15	150 ns	150 ns
MK48T08-20	200 ns	200 ns

PIN NAMES

A0-A12	ADDRESS INPUTS
\bar{E}	CHIP ENABLE
GND	Ground
NC	NO CONNECTION
V _{CC}	+ 5 VOLTS
\bar{W}	WRITE ENABLE
G	OUTPUT ENABLE
DQ0-DQ7	DATA IN/DATA OUT



TRUTH TABLE MK48T08

V _{CC}	\bar{E}	\bar{G}	\bar{W}	MODE	DQ	POWER
<V _{CC} (max)	V _{IH}	X	X	Deselect	High-Z	Standby
	V _{IL}	X	V _{IL}	Write	D _{IN}	Active
V _{CC} (min)	V _{IL}	V _{IL}	V _{IH}	Read	D _{OUT}	Active
	V _{IL}	V _{IH}	V _{IH}	Read	High-Z	Active
<V _{PFD} (min) >V _{SO}	X	X	X	Deselect	High-Z	CMOS Standby
≤V _{SO}	X	X	X	Deselect	High-Z	Battery Back-up

DESCRIPTION

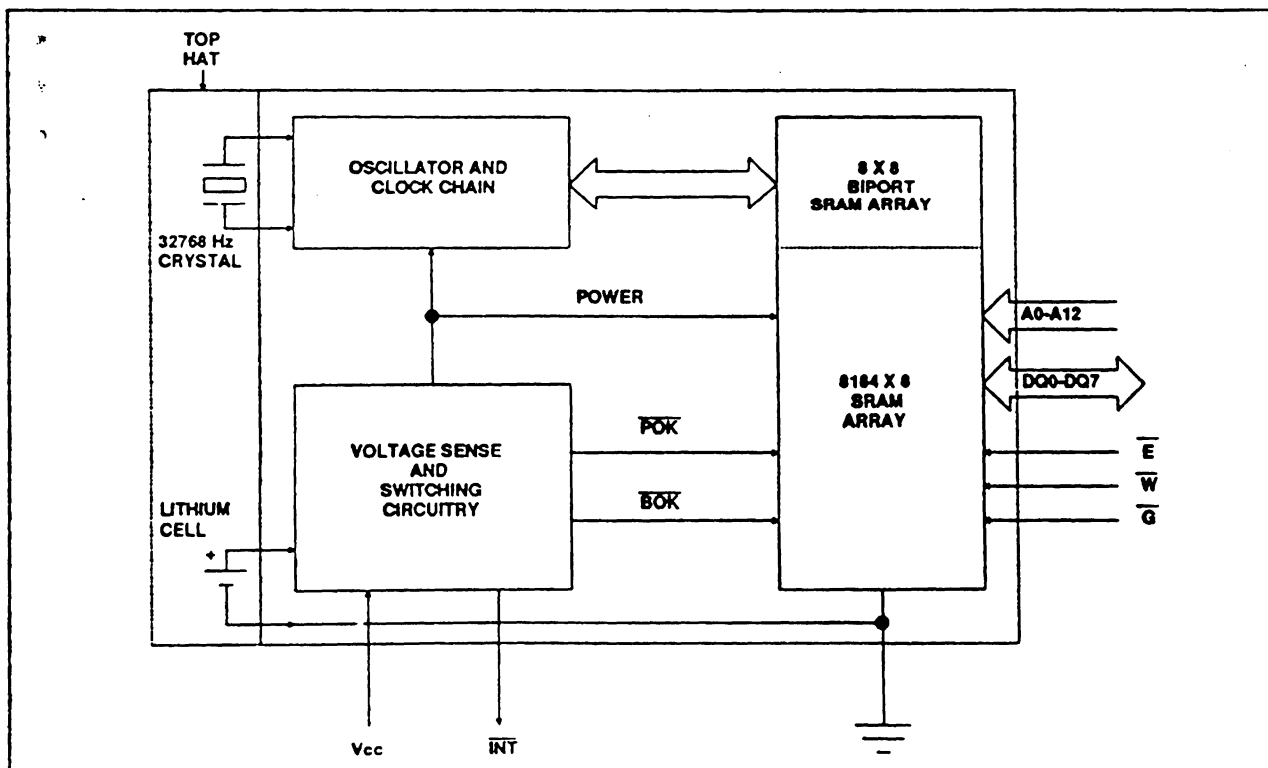
The MK48T08 combines an 8K × 8 full CMOS SRAM, a BYTEWIDE accessible real time clock, a crystal and a long life lithium carbon monofluoride battery, all in a single plastic DIP package. The MK48T08 is a non-volatile pin and function equivalent to any JEDEC standard 8K × 8 SRAM.

It also easily fits into many EPROM and EEPROM sockets, providing the non-volatility of the PROMs without any requirement for special write timing, or limitations on the number of writes that can be performed.

Access to the clock is as simple as conventional BYTEWIDE RAM access because the RAM and the clock are combined on the same die. As figure 1 indicates, the TIMEKEEPER registers are located in the upper eight locations of the RAM. The registers contain, beginning at the top; year, month, date, day, hour, minutes, and seconds data in 24 hour BCD format. Corrections for 28, 29 (Leap Year), 30 and 31 day months are made automatically. The eighth location is a Control register. These registers are not the actual clock counters; they are BiPORT read/write Static RAM memory locations. The MK48T08 includes a clock control circuit that, once every second, dumps the counters into the BiPORT RAM.

Because the Clock Registers are constructed using BiPORT memory cells, access to the rest of the RAM proceeds unhindered by updates to the TIMEKEEPER registers, even if the TIMEKEEPER registers are being updated at the very moment another location in the memory array is accessed. The MK48T08 also has its own Power-fall Detect circuit. The circuit deselects the device when ever V_{CC} is out of range, providing a high degree of data security in the midst of unpredictable system operations brought on by low V_{CC}.

FIGURE 1. BLOCK DIAGRAM



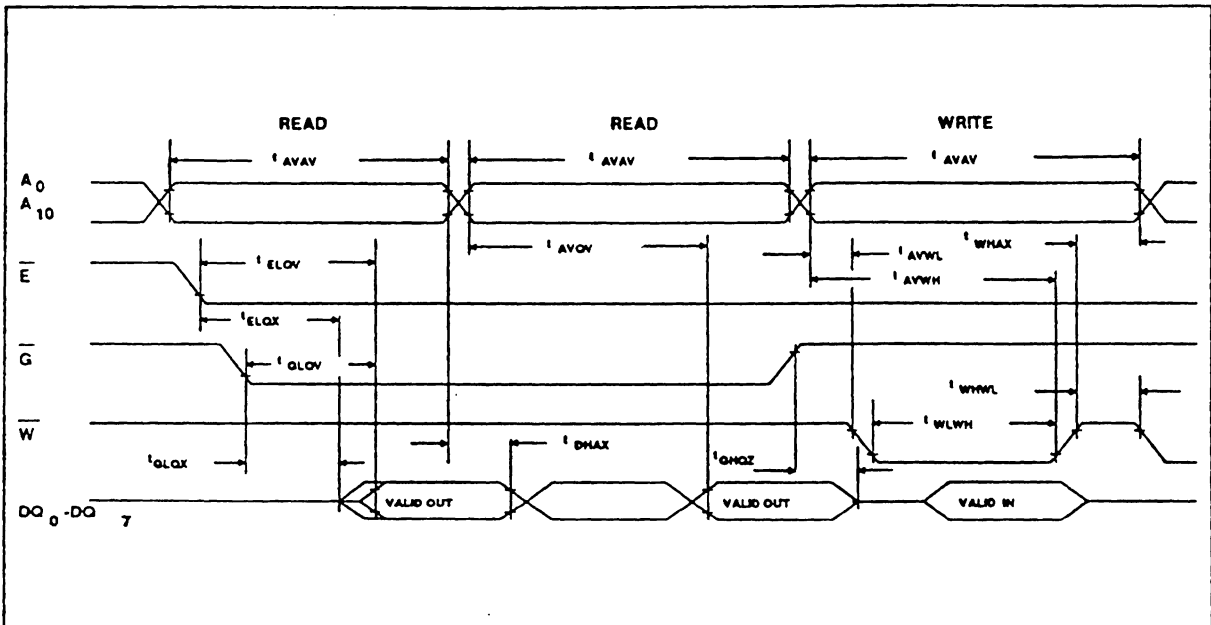
READ MODE

The MK48T08 is the Read Mode whenever \overline{W} (Write Enable) is high and \overline{E} (Chip Enable) is low. The device architecture allows ripple-through access to any of the 8192 address locations in the static storage array. Valid data will be available at the Data I/O pins within t_{AA} after the last address input signal is stable, providing that the \overline{E} and \overline{G} access times are satisfied.

If \overline{E} or \overline{G} access times are not yet met, valid data

will be available at the latter of Chip Enable Access Time (t_{CEA}) or at Output Enable Access Time (t_{OEA}). The state of the eight three-state Data I/O signals is controlled by \overline{E} and \overline{G} . If the Outputs are activated before t_{AA} , the data lines will be driven to an indeterminate state until t_{AA} . If the Address inputs are changed while \overline{E} and \overline{G} remain low, output data will remain valid for Output Data Hold Time (t_{OH}) but will go indeterminate until the next Address Access.

FIGURE 2. READ CYCLE TIMING



AC ELECTRICAL CHARACTERISTICS (READ CYCLE)
 (0°C ≤ TA ≤ +70°C, VCC = 5.0 V ± 10%/ -5%)

ALT. SYM.	STD. SYM.	PARAMETER	MK48T08-10		MK48T08-12		MK48T08-15		MK48T08-20		UNITS	NOTE
			MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX		
t _{RC}	t _{AVAV}	Read Cycle Time	100		120		150		200		ns	
t _{AA}	t _{AVQV}	Address Access Time		100		120		150		200	ns	3
t _{CEA}	t _{ELQV}	Chip Enable Access Time		100		120		150		200	ns	3
t _{CEZ}	t _{EHQZ}	Chip Enable Data Off Time		50		60		75		100	ns	
t _{OEA}	t _{GLQV}	Output Enable Access Time		50		60		75		100	ns	3
t _{OEZ}	t _{GHQZ}	Output Enable Data Off Time		40		50		60		80	ns	
t _{OEL}	t _{GLQX}	Output Enable to Q Low-Z	5		5		5		5		ns	
t _{CEL}	t _{ELQX}	Chip Enable to Q Low-Z	10		10		10		10		ns	
t _{OH}	t _{DHAX}	Output Hold from Address	5		5		5		5		ns	

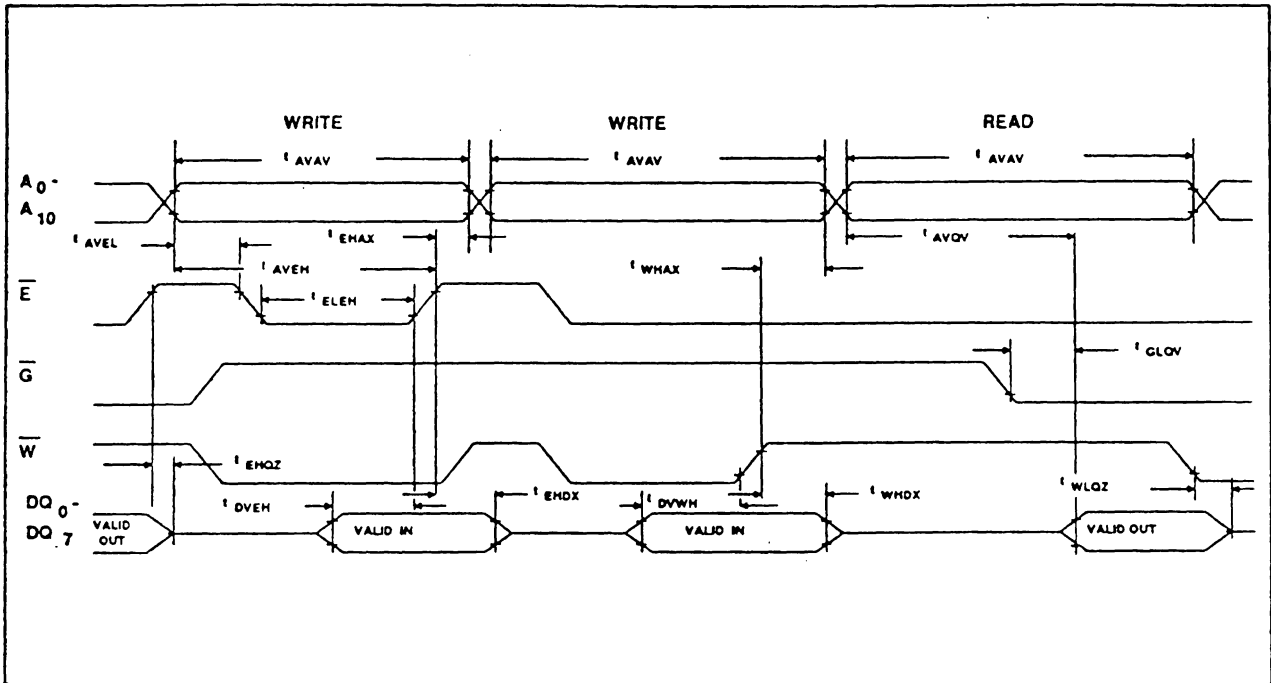
WRITE MODE

The MK48T08 is in the Write Mode whenever \overline{W} and \overline{E} control lines are low. The start of a write is referenced to the latter occurring falling edge of \overline{W} or \overline{E} . A write is terminated by the earlier rising edge of \overline{W} or \overline{E} . The addresses must be held valid throughout the cycle. \overline{E} or \overline{W} must return high for minimum of t_{WR} prior to the initiation of another

read or write cycle. Data-in must be valid t_{DS} prior to the end of write and remain valid for t_{DH} afterward.

Because \overline{G} is a Don't Care in the Write Mode and a low on \overline{W} will return the outputs to High-Z, \overline{G} can be tied low and two-wire RAM control can be implemented. A low on \overline{W} will disable the outputs t_{WEZ} after \overline{W} falls. Take care to avoid bus contention when operating with two-wire control.

FIGURE 3. WRITE CYCLE TIMING



AC ELECTRICAL CHARACTERISTICS (WRITE CYCLE)

(0°C ≤ TA ≤ +70°C, VCC = 5.0 V + 10%/ -5%)

ALT. SYM.	STD. SYM.	PARAMETER	MK48T08-10		MK48T08-12		MK48T08-15		MK48T08-20		UNITS	NOTE
			MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX		
t _{WC}	t _{AVAV}	Write Cycle Time	100		120		150		200		ns	
t _{AS}	t _{AVWL}	Address Setup Time \overline{W} Low	0		0		0		0		ns	
t _{AS}	t _{AVEL}	Address Setup Time \overline{E} Low	0		0		0		0		ns	
t _{CEW}	t _{ELEH}	Chip Enable to End of Write	80		100		130		180		ns	
t _{AW}	t _{AVWH}	Add. Valid to End of Write	80		100		130		180		ns	
t _{AW}	t _{AVEH}	Add. Valid to End Write	80		100		130		180		ns	
t _{WEW}	t _{WLWH}	Write Pulse Width	50		70		100		150		ns	
t _{CEZ}	t _{EHOZ}	\overline{E} Data Off Time		50		60		75		100	ns	
t _{WEZ}	t _{WLQZ}	\overline{W} Data Off Time		50		60		57		100	ns	

AC ELECTRICAL CHARACTERISTICS (WRITE CYCLE) (Continued)
 ($0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$, $V_{CC} = 5.0\text{ V} \pm 10\%/-5\%$)

ALT. SYM.	STD. SYM.	PARAMETER	MK48T08-10		MK48T08-12		MK48T08-15		MK48T08-20		UNITS	NOTE
			MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX		
t_{WR}	t_{WHAX}	\bar{W} High to Address Change	10		10		10		10		ηS	
t_{WR}	t_{EHAX}	\bar{E} High to Address Change	10		10		10		10		ηS	
t_{WR}	t_{WHWL}	\bar{W} High to \bar{W} Low next Cycle	10		10		10		10		ηS	
t_{DS}	t_{DVWH}	Data Setup Time to \bar{W} High	50		60		70		80		ηS	
t_{DS}	t_{DVEH}	Data Setup Time to \bar{E} High	50		60		70		80		ηS	
t_{DH}	t_{WHDX}	Data Hold Time \bar{W} High	5		5		5		5		ηS	
t_{DH}	t_{EHDX}	Data Hold Time \bar{E} High	5		5		5		5		ηS	

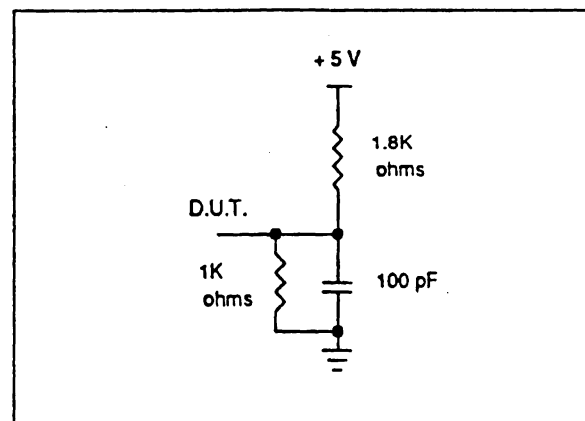
AC TEST CONDITIONS

Input Levels: 0.6V to 2.4V

Transition Times: 5 ns

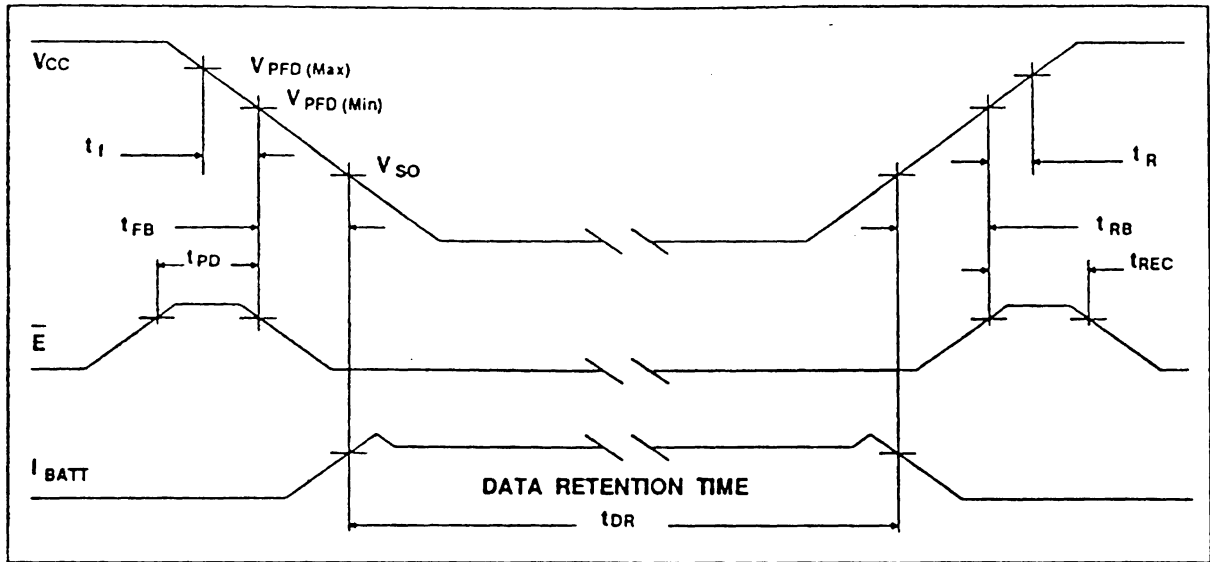
Input and Output Timing

Reference Levels: 0.8V or 2.2V

FIGURE 4. OUTPUT LOAD DIAGRAM

CAPACITANCE

SYMBOL	PARAMETER	MAX	UNITS	NOTES
C_1	Capacitance on all pins (except DQ)	7.0	pF	
C_{DQ}	Capacitance on DQ pins	10.0	pF	

FIGURE 5. POWER-UP / POWER-DOWN CONDITIONS



AC ELECTRICAL CHARACTERISTICS (POWER-UP/DOWN TIMING)
 (0°C ≤ TA ≤ +70°C)

SYMBOL	PARAMETER	MIN	MAX	UNITS	NOTES
t _{PD}	\bar{E} or \bar{W} at V _{IH} before Power Down	0		μs	
t _f	V _{PF(D) (Max)} to V _{PF(D) (Min)} V _{CC} Fall Time	300		μs	
t _{FB}	V _{PF(D) (Min)} to V _{SO} V _{CC} Fall Time	10		μs	
t _{RB}	V _{SO} to V _{PF(D) (Min)} V _{CC} Rise Time	1		μs	
t _R	V _{PF(D) (Min)} to V _{PF(D) (Max)} V _{CC} rise Time	0		μs	
t _{REC}	\bar{E} or \bar{W} at V _{IH} after Power Up	2		ms	

DC ELECTRICAL CHARACTERISTICS (POWER-UP/DOWN TRIP POINTS)
 (0°C ≤ TA ≤ +70°C)

SYMBOL	PARAMETER	MIN	TYP	MAX	UNITS	NOTES
V _{PF(D)}	Power-fail Deselect Voltage	4.5	4.6	4.75	V	
V _{SO}	Battery Back-up Switchover Voltage		3.0		V	
t _{DR}	Expected Data Retention Time (Oscillator On)	5			YEARS	

CAUTION
 Negative undershoots below -0.3 volts are not allowed on any pin while in the Battery Back-up mode.

CLOCK OPERATIONS

Reading the Clock

Updates to the TIMEKEEPER registers should be halted before clock data is read to prevent reading of data in transition. Because the BiPORT TIMEKEEPER cells in the RAM array are only data registers, and not the actual counter, updating the registers can be halted without disturbing the clock itself.

Updating is halted when a "1" is written into the "Read" bit, the seventh most significant bit in the Control Register. As long as a "1" remains in that position, updating is halted. After a Halt is issued, the registers reflect the count, that is day, date, and time that were current at the moment the Halt command was issued.

All of the TIMEKEEPER register are updated simultaneously. A Halt will not interrupt an update in progress. Updating is within a second after the bit is reset a "0".

Setting the Clock

The eight bit of the Control register is the "Write" bit. Setting the Write bit to a "1", like the Read bit, halts updates to the TIMEKEEPER registers. The user can then load them with the correct day, date and time data in 24 Hour BCD format. Resetting the Write bit to a "0" then transfers those values to the actual TIMEKEEPER counters and allows normal operation to resume.

Stopping and Starting the Oscillator

The oscillator may be stopped at any time. If the device is going to spend a significant amount of time on the shelf, the oscillator can be turned off to minimize current drain from the battery. The "Stop" bit is the MSB for the Seconds Register. Setting it to a "1" stops the oscillator.

FIGURE 6. THE MK48T08 REGISTER MAP

ADDRESS	DATA								FUNCTION
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
1FFF	—	—	—	—	—	—	—	—	YEAR 00-99
1FFE	X	X	X	—	—	—	—	—	MONTH 01-12
1FFD	X	X	—	—	—	—	—	—	DATE 01-31
1FFC	X	FT	X	X	X	—	—	—	DAY 01-07
1FFB	X	X	—	—	—	—	—	—	HOUR 00-23
1FFA	X	—	—	—	—	—	—	—	MINUTES 00-59
1FF9	ST	—	—	—	—	—	—	—	SECONDS 00-59
1FF8	W	R	S	—	—	—	—	—	CONTROL

ST = STOP BIT
W = WRITE BIT

R = READ BIT
S = SIGNBIT

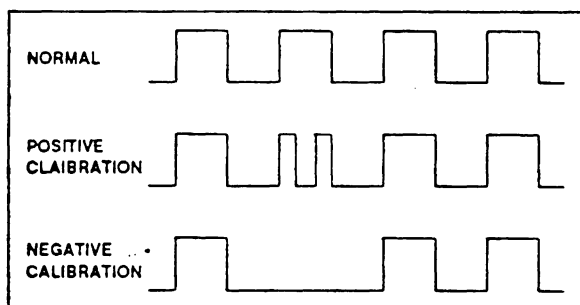
FT = FREQUENCY TEST
X = UNUSED

Calibrating the Clock

The MK48T08 is driven by a quartz controlled oscillator with a nominal frequency of 32768 Hz. The crystal is mounted in the tophat along with the battery. A typical MK48T08 is accurate within ± 1 minute per month at 25°C without calibration. The devices are tested not to exceed 35 PPM (parts per million) oscillator frequency error at 25°C, which comes to about ± 1.53 minutes per month. Of course the oscillation rate of any crystal changes with temperature. Figure 6 shows the frequency error that can be expected at various temperatures.

Most clock chips compensate for crystal frequency and temperature shift error with cumbersome trim capacitors. The MK48T08 design, however, employs periodic counter correction. The calibration circuit adds or subtracts counts from the oscillator divider circuit at the divide by 128 stage, as shown in figure 7. The number of times pulses are blanked (subtracted, negative calibration) or split (added, positive calibration) depends upon the value loaded into the five bit Calibration byte found in the Control Register. Adding counts speeds the clock up, subtracting counts slows the clock down.

FIGURE 7. ADJUSTING THE DIVIDE BY 128



The Calibration byte occupies the five lower order bits in the Control register. This byte can be set to represent any value between 0 and 31 in binary form. The sixth bit is a sign bit; "1" indicates positive calibration, "0" indicates negative calibration. Calibration occurs within a 64 minute cycle. The first 62 minutes in the cycle may, once per minute, have one second either shortened or lengthened by 256 oscillator cycles, that is one tick of the divide by 128 stage of the clock chain. If a binary "1" is loaded into the register, only the first 4

minutes in the 64 minute cycle will be modified; if a binary 6 is loaded, the first 24 will be affected and so on.

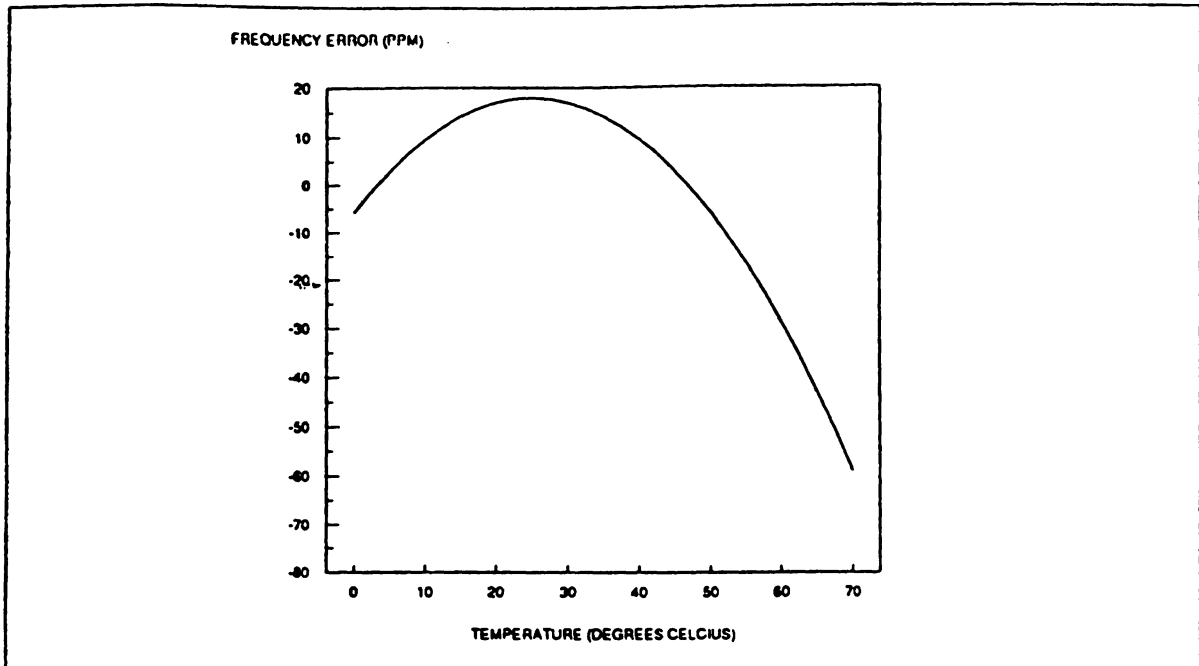
Therefore, each calibration step has the effect of adding or subtracting 512 oscillator cycles for every 125,829, 120 actual oscillator cycles, that is 4.068 PPM of adjustment per calibration step in the user 126.14 PPM calibration range. Assuming that the oscillator is in fact running at exactly 32768 Hz, each of the 31 increments in the Calibration byte would represent 10.7 seconds per month.

Two methods are available for ascertaining how much calibration a given MK48T08 may require. The first involves simply setting the clock, letting it run for a month and comparing it to a known accurate reference (like WWV broadcasts). While that may seem crude, it allows the designer to give the end user the ability to calibrate his clock as his environment may require, even after the final product is packaged in a non-user serviceable enclosure. All the designer has to do is provide a simple utility that accessed the Calibration byte. The utility could even be menu driven and made foolproof.

The second approach is better suited to a manufacturing environment, and involves the use of some test equipment. When the Frequency Test (FT.) bit, the seventh-most significant bit in the day Register, is set to a "1", and the oscillator is running at 32768 Hz, the LSB (DQ0) of the Seconds Register will toggle at a 512 Hz. Any deviation from 512 Hz indicates the degree and direction of oscillator frequency shift at the test temperature. For example, a reading of 512.01024 Hz would indicate a + 20 PPM oscillator frequency error, requiring a -5 (000101) to be loaded into the Calibration Byte for correction. Note that setting or changing the Calibration Byte does not affect the Frequency Test output frequency. The device must be selected and addresses must stable at Address 1FF9 when reading the 512 Hz on DQ0.

The FT. bit must be set using the same method used to set the clock, using the Write bit. The LSB of the Seconds Register is monitored by holding the MK48T08 in an extended read of the Seconds Register, without having the Read bit set. The FT. bit MUST be reset to a "0" for normal clock operations to resume.

FIGURE 8. FREQUENCY ERROR WITHOUT CALIBRATION



DATA RETENTION MODE

With V_{CC} applied, the MK48T08 operates as a conventional BYTEWIDE static RAM. Should the supply voltage decay, the RAM will automatically power-fail deselect, write protecting itself when V_{CC} falls within the $V_{PFD(max)}$, $V_{PFD(min)}$ window. The MK48T08 has a $V_{PFD(max)}$ - $V_{PFD(min)}$ window of 4.75 volts to 4.5 volts, allowing users constrained to a 10% power supply specification to use the device.

Note: A mid-write cycle power failure may corrupt data at the currently addressed location, but does not jeopardize the rest of the RAM's content. At voltages below $V_{PFD(min)}$, the user can be assured the memory will be in a write protected state, provided the V_{CC} fall time does not exceed t_f . The MK48T08 may respond to transient noise spikes that reach into the deselect window if this should occur during the time the device is sampling V_{CC} . Therefore decoupling of the power supply lines is recommended.

The power switching circuit connects external V_{CC} to the RAM and disconnects the battery when V_{CC} rises above V_{SO} . Normal RAM operation can resume t_{REC} after V_{CC} exceeds $V_{PFD(max)}$. Caution should be taken to keep \bar{E} or \bar{W} high as V_{CC} rises past $V_{PFD(min)}$ as some systems may perform

inadvertent write cycles after V_{CC} rises but before normal system operation begins.

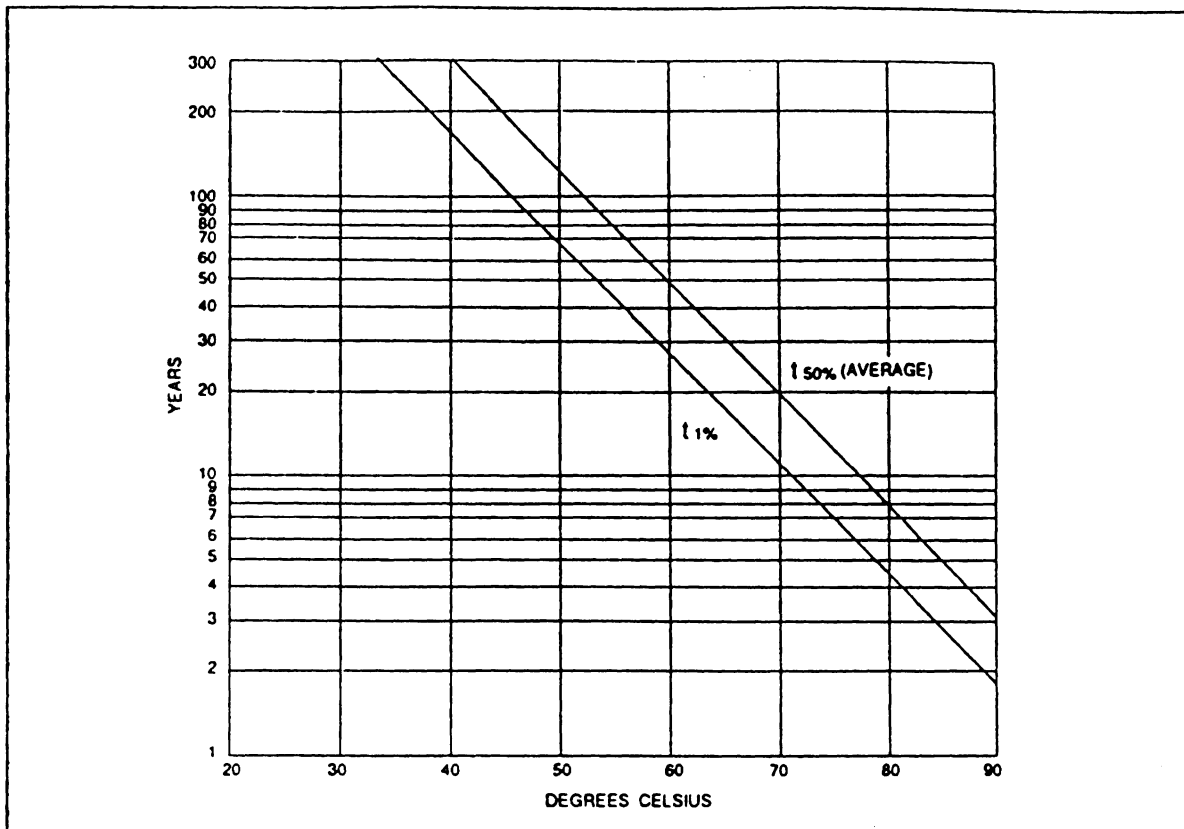
PREDICTING BACK-UP SYSTEM LIFE

The useful life of the battery in the MK48T08 is expected to ultimately come to an end for one of two reasons; either because it has been discharged while providing current to an external load; or because the effects of aging render the cell useless before it can actually be discharged. Fortunately, these two effects are virtually unrelated, allowing discharge, or Capacity Consumption and the effects of aging, or Storage Life to be treated as two independent but simultaneous mechanisms, the earlier of which defines Back-up System life.

The current drain that is responsible for Capacity Consumption can be reduced either by applying V_{CC} or turning off the oscillator. With the oscillator off, only the leakage currents required to maintain data in the RAM are flowing. With V_{CC} on, the battery is disconnected from the RAM. Because the leakage currents of the MK48T08 are so low, then can be neglected in practical Storage Life calculations.

Therefore, to extend the life of components that are just sitting on the shelf (not in system use) the oscillator should be turned off.

FIGURE 9. MK48T08 PREDICTED BATTERY STORAGE LIFE VS. TEMP.



Predicting Storage Life

Figure 9 illustrates how temperature affects Storage Life of the MK48T08 battery. As long as V_{CC} is applied or the oscillator is turned off, the life of the battery is controlled by temperature and is virtually unaffected by leakage currents drawn by the MK48T08.

Storage Life predictions presented in Figure 9 are extrapolated from temperature accelerated life-test data collected in over 100 million device hours of continuing bare cell and encapsulated cell battery testing by SGS-THOMSON. Obviously, temperature accelerated testing cannot identify non-temperature dependent failure mechanisms. However, in view of the fact that no random cell failures have been recorded in any of SGS-THOMSON's on going battery testing since it began in 1982, we believe the chance of such failure mechanisms surfacing is extremely small.

For the purpose of the testing, a cell failure is defined as the inability of a cell stabilized at 25°C to produce a 2.0 volt closed-circuit voltage across a 250K load resistance.

A Special Note: The summary presented in Figure 9 represents a conservative analysis of the data presently available. While SGS-THOMSON is most likely in possession of the largest collection of battery life data of this kind in the world, the results presented should not be considered absolute or final; they can be expected to change as yet more data becomes available. We believe that future read points of life test presently under way and improvements in the battery technology itself will result in a continuing improvement of these figures.

Two end of life curves are presented in Figure 9. They are labeled "Average" (t_{50%}) and (t_{1%}). These terms relate to the probability that a given number of failure will have accumulated by a particular point in time. If, for example, expected life at 70°C is at issue, Figure indicates that a particular MK48T08 has a 1% chance of having a battery failure 11 years into its life and a 50% chance of failure at the 20 year mark. Conversely, given a sample of device, 1% of them can be expected to experience battery failure within 11 years; 50% of them can be expected fail within 20 years.

The $t_{1\%}$ figure represents the practical onset of wear out, and is therefore suitable for use in what would normally be thought of as a worst-case analysis. The $t_{50\%}$ figure represents "normal" or "average" life. It is, therefore, accurate to say that the average device will last " $t_{50\%}$ ".

Battery life is defined as beginning at the date of manufacture. Each MK48T08 is marked with a four digit manufacturing date code in the form YYWW (example: 8625 = 1986, week 25).

Calculating Predicted "Storage Life of the Battery"

As Figure 9 indicates, the predicted Storage Life on the battery in the MK48T08 is a function of temperature.

Because the ambient temperature profile is dependent upon application controlled variable, only the user can estimate predicted Storage Life in a given design. As long as ambient temperature is held reasonably constant, expected Storage Life can be read directly from Figure 9. If the MK48T08 spends an appreciable amount of time at a variety of temperatures, the following equation should be used to estimate Storage Life.

$$\text{Predicted Storage Life} = 1 + \left\{ \left[\frac{TA_1}{TT} \right] \div SL_1 \right\} + \left\{ \left[\frac{TA_2}{TT} \right] \div SL_2 \right\} + \dots + \left\{ \left[\frac{TA_N}{TT} \right] \div SL_N \right\}$$

Where TA_1, TA_2, TA_N , = Time at Ambient Temperature 1, 2, ect

$$TT = \text{Total Time} = TA_1 + TA_2 + \dots + TA_N$$

SL_1, SL_2, SL_N = Predicted Storage Life at Temp. 1, Temp. 2, ect. (See Figure 9)

Example Predicted Storage Life Calculation

A cash register/terminal operates in an environment where the MK48T08 is exposed to temperatures of

30°C (86°F) or less 4672 hrs./yr.; temperatures greater than 25°C, but less than 40°C (104°F), for 3650 hrs./yr.; and temperatures greater than 40°C, but less than 70°C (158°F), for the remaining 438 hrs./yr.

Reading Predicted $t_{1\%}$ values from Figure 10; $SL_1 = 456$ yrs., $SL_2 = 175$ yrs. $SL_3 = 11.4$ yrs.

Total Time (TT) = 8760 hrs./yr. $TA_1 = 4672$ hrs./yr. $TA_2 = 3650$ hrs./yr. $TA_3 = 438$ hrs./yr.

$$\text{Predicted Typical Storage Life} \geq 1 \div \left\{ \left[\frac{4672}{8760} \right] \div 456 \right\} + \left\{ \left[\frac{3650}{8760} \right] \div 175 \right\} + \left\{ \left[\frac{438}{8760} \right] \div 11.4 \right\}$$

$$\text{Predicted Typical Storage Life} \geq 126 \text{ years}$$

ABSOLUTE MAXIMUM RATINGS*

Voltage On Any Pin Relative to GND _____ -0.3 V to +7.0V
 Ambient Operating (V_{CC} On) Temperature (T_A) _____ 0°C to +70°C
 Ambient Storage (V_{CC} Off, Oscillator Off) Temperature _____ -20°C to +70°C
 Total Device Power Dissipation _____ 1 Watt
 Output Current Per Pin _____ 20 mA

* Stresses greater than those under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

RECOMMENDED DC OPERATING CONDITIONS(0°C ≤ T_A ≤ +70°C)

SYMBOL	PARAMETER	MIN	MAX	UNITS	NOTES
V_{CC}	Supply voltage	4.75	5.5	V	
GND	Supply Voltage	0	0	V	
V_{IH}	Logic "1" Voltage All Inputs	2.2	$V_{CC} + 0.3$	V	
V_{IL}	Logic "0" Voltage All Inputs	-0.3	0.8	V	

DC ELECTRICAL CHARACTERISTICS(0°C ≤ T_A ≤ +70°C) ($V_{CC}(\text{Max}) \leq V_{CC} \leq V_{CC}(\text{Min})$)

SYMBOL	PARAMETER	MIN	MAX	UNITS	NOTES
I_{CC1}	Average V_{CC} Power Supply Current		80	mA	
I_{CC2}	TTL Standby Current ($\bar{E} = V_{IH}$)		5	mA	
I_{CC3}	CMOS Standby Current ($\bar{E} = V_{CC} - 0.2V$)		3	mA	
I_{IL}	Input Leakage Current (Any Input)	-1	+1	μA	
I_{OL}	Output Leakage Current	-5	+5	μA	
V_{OH}	Output Logic "1" Voltage ($I_{OUT} = -1.0$ mA)	2.4		V	
V_{OL}	Output Logic "0" Voltage ($I_{OUT} = 2.1$ mA)		0.4	V	

Pin Description

Symbol	Type I/O/BD - cell	Description
IU Interface		
iu_clk	I - DRVT8	Integer Unit Clock. Main system clock.
ctl1	Input- TLCHT	Atomic load-store indication during device cycles.
SBus Interface		
sb_br_	O - BT4	S-bus Bus request.
sb_bg_	I - TLCHN	S-bus Grant.
sb_as_	I - TLCHN	S-bus address strobe
sb_siz(2:0)	BD - BD4TU	S-bus Size.
sb_ack8_	BD - BD4TU	S-bus 8-bit Acknowledge.
sb_ack32_	BD - BD4TU	S-bus 32-bit Acknowledge.
sb_err_	BD - BD4TU	S-bus Error. Failed transfer.
sb_merr_	I - TLCHT	S-bus Late Error. Failed transfer.
sb_rd	BD - BD4TU	S-bus read, High for Read cycles.
sb_a(4:0)	I - TLCHT	S-bus Address bits 4-0.
sb_d(31:24)	BD - BD4TU	S-bus data.
sb_reset_	I - TLCHN	S-bus reset.
b_irq(7:1)_	O - BT1OD	S-bus interrupts = VME interrupts
mb_irq_	O - BT1OD	Interrupt Request for mail box interrupt.
MMU		
io_sel_	I - TLCHN	Non-type 0 space strobe.
type(1:0)	I - TLCHT	MMU type bits.
B_A31_29L	I - TLCHN	Decoding of P1_A(31:29)
B_A28_26L	I - TLCHN	Decoding of P1_A(28:26)
B_A25_24L	I - TLCHN	Decoding of P1_A(25:24)
pa(28:20)	I - TLCHT	Physical address bits.
pa19_16H_	I - TLCHT	Decoding of pa(19:16)= 0xF.
Misc		
pur_	I - TLCHT	Power-up reset
gen_por_	O - BT1	Generate Power on reset.
ureson_	I - TLCHT	User reset on
uresoff_	I - TLCHT	User rest off
sysclkxin	I - OSCIM	Xtal in, VME system clock
sysclkxout	O - OSCIM	Xtal out VME system clock
slot1_	I - TLCHN	Slot Indicator.

Master Interface:

A full 32 bit Master Interface is provided. Complete mapping of 512 MByte is possible all the time, but this 512 MByte window can be moved through all 4 GByte through a VME Mapping register.

A32, A24, A16 Address Modes, D32, D16, D8(E0) Data Sizes are supported.

On Master cycles, RMW, i.e. atomic load-store cycles are implemented according to the VME spec. The VME data strobe will be asserted two times as the VME address strobe is asserted. This differs from other Sun boards, where instead the ownership of VME is kept while a read and a write takes place. The problem with this is that if the access is to a board with several devices competing for an onboard bus, there is no way this board can recognize the access as a RMW. For example.: a disk board with multiple on board devices would not be able to prevent a shared resource to be stolen in between what looks like a standard Read and Write even though the VMEbus is hold.

Block mode transfers are not supported. They are only supported by the Slave interface.

Unaligned transfers (UAT) are not supported. Accesses of this kind, i.e. 16-bit accesses to base addr. + 1 or addr. + 3, or 32 bit accesses to base addr. + 1, base addr. + 2 or base addr. + 3 will not be recognized, resulting in a Bus error time out.

A32 Map Register

Type	Device Addr	Device	Physical Space
1	0xEFE00018	A32 Map Reg.	1 byte

31	30	29	28	27	26	25	24
VME Master Addr	0	0	0	0	0	0	LoopB

This register can re-map VME Address bits A[31:29] to enable full 4 Gbyte mapping. Only 8-bit accesses should be used, 16- or 32-bit accesses will not be acknowledged, resulting in a bus error time out.

Initialization: The register will be cleared to 0's on resets, thus mapping the 512 MByte MMU mapping to 0-512 MByte and disabling loopback mode.

Bit:

31:29	0x0 map VME Master range	0x00000000-0x1FFFFFFF	0-512 MByte
	0x1 - " - "	0x20000000-0x3FFFFFFF	512-1 GByte
	0x2 - " - "	0x40000000-0x5FFFFFFF	1-1.5 "
	0x3 - " - "	0x60000000-0x7FFFFFFF	1.5-2 "
	0x4 - " - "	0x80000000-0x9FFFFFFF	2-2.5 "
	0x5 - " - "	0xA0000000-0xBFFFFFFF	2.5-3 "
	0x6 - " - "	0xC0000000-0xDFFFFFFF	3-3.5 "
	0x7 - " - "	0xE0000000-0xFFFFFFFF	3.5-4 "
28:25	Not used, always read back as 0		
24	This bit set will enable VME loopback mode, used for diagnostics.		

Slave Interface:

A32, A24 address mode.

1 MByte of memory is accessible, normally the lowest MByte on VME, but this can be changed to be in the 1-16 MByte range to support multi CPU board implementations. The address range will be incremented in 1 MByte steps.

VME defined Read-Modify-Write cycles is guaranteed to be atomic with respect to other VME Masters. Atomicity between VMEbus Slave accesses and CPU/Ethernet accesses to main memory, however, is not guaranteed. To guarantee this, the VME Bus Locker should be used, see the section about the Bus Locker register.

Unaligned accesses to the Slave Interface range will return a Bus Error back to the external VME Master.

The address modifiers on VME are used to define Address mode (A32/A24/A16), privilege, and data/program/block-transfer access. The Slave decoder and Mail Box Interrupter will monitor all address modifier bits with the exception of AM2, the access privilege bit. All slave accesses as a DVMA device will be set to supervisor mode, which is in accordance with the Sun-4 Architecture.

The 1 MByte VME address space selected will always be mapped to the highest megabyte in the Virtual Address Space, again in accordance with the Sun-4 Architecture.

Single Transfers/Block Mode Transfers

The DMA enable bit in the System Enable Register can disable SDVMA and therefore all Slave cycles. If this bit is set, single slave cycle can take place.

Block Mode transfers are enabled if the corresponding bit in the Slave Map Register is set. Only blocks of 4 transfers of aligned 32-bit words are supported. Longer blocks will result in a Bus Error from the fifth transfer and on. Accesses with other data sizes will get a Bus Error returned.

Slave Map Register

Type	Device Addr	Device	Reg.	Physical Space
1	0xEFE0001C	SMAP		1 byte

31	30	29	28	27	26	25	24
BLM	0	0	0	VME Slave Address base			

The address space for System DVMA accesses is 1 MByte, normally the lowest Megabyte in accordance with the Sun-4 Architecture, but bits 28:25 can be used to re-map the SDVMA address space. Slave Block Mode transfers can also be enabled by setting bit 31. This lowest 1 MByte on VME is mapped to the highest 1 MByte within the Virtual Address space. During register accesses, only 8-bit accesses should be used. 16- or 32-bit accesses will not be acknowledged, resulting in a bus error time out.

Initialization: The register will be reset to all 0's on resets.
Thus, the mapping will default to 0-1 MByte and block mode transfers will be disabled.

Bit:

31: This bit set will enable Block Mode Transfers.

30:28 Not used, read back as 0

27:24	0x0 map SDVMA space to	0x00000000-0x000FFFFFF	0-1 MByte
	0x1 - " - "	0x00100000-0x001FFFFFF	1-2 "
	0x2 - " - "	0x00200000-0x002FFFFFF	2-3 "
	0x3 - " - "	0x00300000-0x003FFFFFF	3-4 "
	0x4 - " - "	0x00400000-0x004FFFFFF	4-5 "
	0x5 - " - "	0x00500000-0x005FFFFFF	5-6 "
	0x6 - " - "	0x00600000-0x006FFFFFF	6-7 "
	0x7 - " - "	0x00700000-0x007FFFFFF	7-8 "
	0x8 - " - "	0x00800000-0x008FFFFFF	8-9 "
	0x9 - " - "	0x00900000-0x009FFFFFF	9-10 "
	0xA - " - "	0x00A00000-0x00AFFFFFF	10-11 "
	0xB - " - "	0x00B00000-0x00BFFFFFF	11-12 "
	0xC - " - "	0x00C00000-0x00CFFFFFF	12-13 "
	0xD - " - "	0x00D00000-0x00DFFFFFF	13-14 "
	0xE - " - "	0x00E00000-0x00EFFFFFF	14-15 "
	0xF - " - "	0x00F00000-0x00FFFFFF	15-16 "

Mail Box

A Mail Box interrupt function is provided. This mail box interrupt will detect accesses to the A16 address space to a location programmed into the mail box register. No real memory is provided at the location, but the mail box will respond with a VME DTACK, acknowledging the access and generate an on board interrupt if the Enable bit is set. VME Address bits A(15:14, 3:1) will be checked and compared to the contents of the bits in the Mail Box Register. This corresponds to programming the location to one of the first 16-bit words at one of the four 16 KByte blocks of the address range. Any VME A16 address space, 8-bit or 16-bit read or write to the location programmed into the Mail Box Register will generate an onboard interrupt on level 8 if enabled. 32-bit wide access are not allowed and will not be acknowledged, resulting in a Time out Bus Error for the other accessing VME Master.

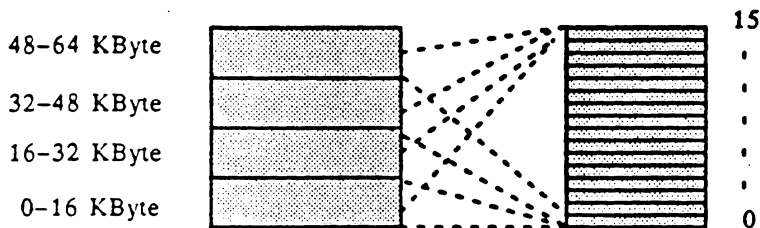
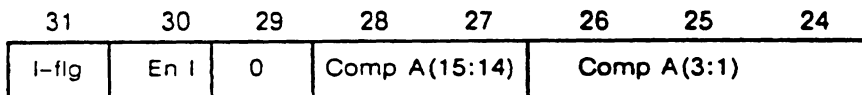


Fig. 6

Mail Box Register

Type	Device Addr	Device	Physical Space
1	0xEFE00010	Mail Box Reg.	1 byte



This register will set the VME address that will be monitored and will, if enabled, interrupt the IU via an on-board interrupt. Only 8-bit accesses to the register should be used, 16- or 32-bit accesses will not be acknowledged, resulting in a bus error time out. For mail box cycles, the VME bus will only be monitored for A16, D8 or D16 accesses.

Initialization: All bits will be initialized to 0's.

Interrupt: level 1.

Bit: Note

- 31 This bit set indicates that a mailbox interrupt is pending. A read of this register will reset the interrupt on the trailing edge of the read pulse. A write to this bit is ignored.
- 30 This bit set will enable the mail box interrupt.
- 29 This bit is ignored, read back as 0
- 28 To compare with VME Address bit 15
- 27 - " - " 14
- 26 - " - " 3
- 25 - " - " 2
- 24 - " - " 1

VME Bus Locker

A Bus locker function is provided to enable the CPU to do an atomic Read-Modify-Write (RMW) to its onboard memory without interrupting an incoming RMW from the VME Slave interface. This function is only to be used when the board is running together with one or more external CPU boards (Multiprocessing). In these cases, message passing is done through shared memory locations that need to be accessed in a defined way.

VME Bus Locker Register

In a Non-Multiprocessing application, this register should be left alone. When used, bit 24 of the register should ALWAYS be SET for each update, to guarantee a consistent bus arbitration behavior.

Type	Device Addr	Device	Physical Space
1	0xEFE00000	Bus Locker. Reg.	1 byte

31	30	29	28	27	26	25	24
Ownd	0	0	0	0	0	Req Bus	Use locker

Initialization: All bits will be initialized to 0's.
Thus, the VME bus locker will be inactive and not affecting the board's bus arbitration.

Bit:

- 31 This bit indicates that the VME Bus is owned. No external VME board can access memory. Bit is read only, a write to this bit will be ignored.
- 30 This bit is ignored, read back as 0.
- 29 - " - " - " - " -
- 28 - " - " - " - " -
- 27 - " - " - " - " -
- 26 - " - " - " - " -
- 25 This bit initiates a VME Bus request. Once owned, it will be held.
- 24 This bit set will enable the Bus request function. It should only be set for multiprocessing applications. If set, it should never be changed (unless system is restarted).

In multiprocessing applications, update register at system initialization time by writing 0x01 to the register. The procedure for accessing an onboard memory location is:

- 1 Disable VME Interrupts to prevent deadlock. (VME Interrupts need to acquire VME)
2. Request VME bus locking by writing 0x3 to the register.
3. Read register and check bit 31 to until VME bus is owned and locked.
4. Do RMW (atomic Load-Store) to onboard memory.
5. Unlock VME by writing 0x1 to register.
6. Enable VME Interrupts

Thus bit 24 is kept set.

Note ! Note that if VME is not unlocked in time, other accesses might time out causing a system crash. Always unlock VME ASAP after the RMW is completed !

Interrupt Handler

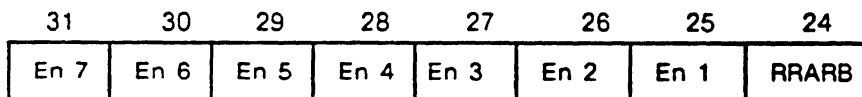
The interrupt handler can selectively support all interrupt levels. These are enabled through the VME Interrupt/Bus Arbiter register. Note that this register is a replacement for a jumper and should normally be fixed depending on the VME Interrupt handler configuration. Run-time enabling of the interrupts are normally done with the Interrupt Enable register, see the System Specification.

Single level arbitration is normally used. If some VME boards do not get enough access to the bus, round robin select can be used. In that case the arbiter will give equal priority to all four levels. See the VMEbus spec.

Interrupt Enable/ Bus Arbiter Mode Register

The Interrupt Enable Register can selectively enable all VME interrupt levels.

Type	Device Addr	Device	Physical Space
1	0xEFE00014	VME Int. Reg.	1 byte



This Register enables VME interrupts to be fed to the Interrupt logic in the S-4 MMU chip. This register replaces 8 jumpers on the board. Only 8-bit accesses should be used, 16- or 32-bit accesses will not be acknowledged, resulting in a bus error time out.

Initialization: bits 31:25 will be set and bit 24 reset.
Thus, VME interrupts will be enabled and the arbiter mode will be SGL, single level arbitration.

Bit:

31	This bit set will enable VME Interrupt 7/Sun-4 level 13 to Int.logic
30	- " - " - " - " - " - " - 6/ - " - 11 -
29	- " - " - " - " - " - " - 5/ - " - 9 -
28	- " - " - " - " - " - " - 4/ - " - 8 -
27	- " - " - " - " - " - " - 3/ - " - 5 -
26	- " - " - " - " - " - " - 2/ - " - 3 -
25	- " - " - " - " - " - " - 1/ - " - 2 -

24 This bit set will set the bus arbiter into a RRS mode (Round Robin Select).
This bit reset will set the bus arbiter into a SGL level bus arbiter (arbitration level 3 only).
Note that enabling of the bus arbiter function is done via a jumper (Slot1 jumper).

Note! This register replaces onboard jumpers. Changes to this register should only be done at system initialization time not to be changed later (unless system is restarted).

VME IACK Cycles

Interrupting devices out on the VMEbus require an interrupt acknowledge cycle to obtain an interrupt vector. A device space operation is used by the interrupt handler to run the acknowledge cycle and obtain the vector. The operation is Device space byte read with A[3-1] set to the VMEbus interrupt level being acknowledged. This location should only be accessed with 8-bit reads, 16- or 32-bit accesses will not be acknowledged, resulting in a bus error time out.

Note that an access to these seven addresses are not to an on chip register, but instead to execute a VME Interrupt Acknowledge cycle, acquiring the Interrupt Vector from the interrupting device.

Type	Device Addr	Device	Physical Space
1	0xEFE00000	VME Int.Vector	A[3:1]=VME Int. level, A0=1, Read, only 8-bit accesses should be used.

31 30 29 28 27 26 25 24

8-bit Interrupt Vector from interrupting VME Device Level A(3:1)

This translates into the following interrupt responses:

Acknowledging VME interrupt	Device Address
1	0xEFE00003
2	0xEFE00005
3	0xEFE00007
4	0xEFE00009
5	0xEFE0000B
6	0xEFE0000D
7	0xEFE0000F

Initialization: Not applicable.

IACK Daisy Chain Driver

Active if Board is located in slot one, indicated by the Slot1 jumper.

Bus Requester

The Bus requester is an ROR requester. It releases the bus when other VME Masters/Requesters requests the bus.

Bus Arbiter

Two types of arbiters are provided, a Single Level Arbiter or a Round Robin Select Arbiter. They are selected through the VME Interrupt/Bus Arbiter register. Note that the arbiters will only be enabled if the board resides in slot 1. (Slot1 jumper)

In arbitration mode, locked arbitration will be detected and released by the arbiter after 3.3 ms. The Bus arbiter will then drive BBSY active for the minimum period required by the VMEbus spec. This is to prevent requesters that in the mean time have decided to withdraw their requests to be able to do so. Note that a withdrawal of a VME Request normally is in violation of the VME Specification. (Rule 3.11)

Bus Time Out Period

There are two time out parameters in a VME cycle in addition to the On Board Bus arbitration timer: Rerun Time Out and Abort.

Rerun time out: If the time from the Master cycle starts until the VME slave responds with a DTACK exceeds 1.6 us, the cycle will be rerun by the IU so high-priority devices can get on the Sbus. In the mean time, the Master cycle and all output signals will be frozen, ignoring any DTACK coming in. When the cycle is rerun it will be reconnected again.

Abort: Two stages of the access exists. If the bus is not owned, it has to be acquired. Then the actual VMEbus cycle can start. A Master cycle will time out after 300 us. That means, the response time from another VME slave including the time for acquiring the bus must be less than 300 us.

If the VMEbus is not owned when an abort occurs, the bus requester will keep the VME Bus Request asserted even though the CPU cycles has been aborted. It will then assert a dummy Bus Busy (BBSY) when the Bus Grant is received. This is to comply with VME rule 3.11.

If the bus is owned when the abort takes place, a Bus Error will be generated. This Bus Error would normally be generated by an accessed VME slave. This is to comply with VME rule 2.48.

System Reset

On power up, the CPU board will generate SYSRESET for a minimum of 200 ms, if the Slot1 jumper indicates the boards resides in slot 1. Incoming VMEbus resets will always generate an on board reset.

Jumper

Only one jumper is required for the VME Interface: Slot1, thereby replacing numerous other jumpers. These other settings will be defined by software register settings.

Function	"slot 1" position	"not in slot 1"	VME
SYSTEM CLOCK	enabled	tri-state	
VME IACK Daisy-chain Driver	enabled	IACKIN drives IACKOUT	
SYSRESET	sb_reset will generate VME SYSRESET.	sb_reset will not generate VME SYSRESET	
BUS ARBITER	enabled (type determined by disabled Interrupt reg bit 0)	disabled	

Programmable settings

The following registers include settings otherwise often provided by jumpers.

Slave Map Register	Defines the memory address space of the 1MByte on board VME Slave (System DVMA).
VME Interrupt Handler Register	Defines which interrupt levels to handle. Defines what type of VME Bus arbiter (If board resides in slot 1)
A32 Map Register	Defines where to map the CPU Master Address Space of 512 MByte.

Performance:

The VME Gate Array logic will be designed to meet the Sbus and VMEbus specifications at a 20 MHz clock, correlating to a 50 ns clock cycle.

Bus arbitration for other VMEbus masters will take place faster thanks to the Early Release feature.

System DVMA can offer a higher bandwidth with the VME Block Mode transfer.

Note that these numbers are given only as a way to describe the VME performance. Since the overall bandwidth depends on the other board's response time, the application code, amount of other tasks running concurrently, other concurrent DMA activity, cache hit/miss ratio etc. these figures should NOT be expected to on a running system.

Master Cycles

Conditions: Ideal VME slave accessed (30 ns DS to DTACK response). Bus already owned. No lingering DTACK or AS.

A Master cycle will be 9 cycles, corresponding to a theoretical 9 MByte/s bandwidth for back-to-back cycles.

Slave Cycles

Conditions: Ideal VME master (0 ns DTACK to DS response). Fast Sbus grant (two clock periods).

Cycle	duration [ns]	theoretical bandwidth
Single read	560-630 ns	6.9-7.9 MByte/s
Single write	510-580 ns	6.3-7.2 MByte/s
Block read	950-1250 ns	12.7-17.7 MByte/s
Block write	920-1160 ns	13.7-17.4 MByte/s

Bus Arbitration

Single level Arbiter:

Conditions: No VME master owns the bus, Bus Locker disabled.

VME Bus Request to VME Bus Grant: 2 Cycles

Conditions: No VME master owns the bus, Bus Locker enabled

VME Bus Request to VME Bus Grant: 3 Cycles

Round Robin Arbiter:

Conditions: No VME master owns the bus, Bus Locker disabled.

VME Bus Request to VME Bus Grant: 2-5 Cycles

Conditions: No VME master owns the bus, Bus Locker enabled

VME Bus Request to VME Bus Grant: 3-6 Cycles

Timing Specification

Conditions: VCC=4.75 to 5.25 V, TRA = 0 - 70 C, Output load = 130 pF

Symbol From To (note1) Min Max unit Note

Sbus signals

t11	clk high	clk high	50	50	ns	
t12	clk high	sb_ack valid	3	33	ns	2
t13	clk high	sb_d valid	3	34	ns	
t14	clk high	sb_rd valid	3	30	ns	
t15	clk high	sb_siz valid	3	26	ns	
t16	clk high	sb_br_	3	24	ns	

IACK Daisy Chain

t17	p1_ds(1:0)_ ass.	vme_iackout ass.	43	155	ns	3, 7
t18	p1_as_ neg.	vme_iackout neg.		15	ns	3, 7

Requester/Arbiter

t19	clk high	vme_bbsy_		26	ns	
t110	clk high	vme_bgout		25	ns	
t111	clk high	vme_br		25	ns	
t112	clk high	b_asen_ ass.		29	ns	
t113	clk high	b_asen_ neg.		38	ns	
t114	vme_as_ neg.	b_asen_ neg.		12	ns	

Master

t115	clk high	vme_am		28	ns	4
t116	clk low	vme_as ass.		21	ns	
t117	clk high	vme_as neg.		25	ns	
t118	clk low	vme_ds ass.		23	ns	
t119	clk high	vme_ds neg.		20	ns	
t20	clk high	ma_aclk		24	ns	
t21	clk high	ma_aen_		19	ns	

Slave cycles

t22	clk high	sl_aclk		23	ns	
t23	clk high	sl_aen_		14	ns	
t24	clk low	SbtP1clk	6	19	ns	
t25	clk high	P1tSbclk		21	ns	
t26	clk high	P1tSben_		20	ns	

Slave single cycles

t27	clk high	vme_dtack		20	ns	
t28	clk high	vme_berr		20	ns	

t29	clk high	SbtP1en_		20	ns	
Slave Block Mode Reads						
t30	P1_DS_ ass.	vme_dtack_	37	719	ns	7
t31	P1_DS_ ass.	vme_berr_	37	719	ns	7
t32	P1en_ ass	vme_dtack_ ass.	28	103	ns	7
t33	P1en_ neg	vme_dtack_ neg.	30	108	ns	7
t34	P1_DS_ ass.	SbtP1en_ ass.	8	616	ns	7
t35	P1_DS_ neg.	SbtP1en_ neg.	6	28	ns	7
t36	P1_DS_ ass.	pl_sel valid	6	29	ns	7
Slave Block Mode Writes						
t37	P1_DS_ ass.	vme_dtack_	35	659	ns	7
t38	P1_DS_ ass.	vme_berr_	35	659	ns	7
t39	P1_DS_ ass.	P1tSbclk	7	37	ns	7
t40	clk high	P1tSben_	5	15	ns	
t41	clk high	pl_sel valid		20	ns	
misc. VME						
t42	vme_sysr_ ass.	vme_sysr_ neg.	200		ms	5
Setup time						
	sbus signals, ctl1, type(1:0)		15 ns.			6
	pa(28:20)		23 ns			
	b_axx_xxl		19 ns			

Hold time for all the above signals is 0 ns except for sb_err_ which is 1 ns.

- 1) If signal direction is not specified, both high and low delays are included.
- 2) sb_ack: sb_ack8_, sb_ack32_, sb_err_, sb_merr_.
- 3) IACK D.C. driver, slot1_ asserted
- 4) the clock following sb_as assertion
- 5) When chip is in test mode, i.e. P1_BGIN_.0 = low and slot1_ = low the vme_sysr counter and the Bus grant counter will be preset to 32 clocks from their max value.
- 6) Sbus signals: sb_as_, sb_a, sb_ack8_, sb_ack32_, sb_err_, sb_merr_sb_rd, sb_siz, sb_br_.
- 7) This timing is not related to clock

Revision History

3/7/89 First release.

9/21/89 Revision 1.0 release

- sb_merr_ changed from BD-BD4TU to I-TLCHT (Page 2)
- P1 inputs changed from SCHMITC to SCHMITT (Page 3)
- LSI device type changed from LMA9239 to LIA5530 (Page 4)
- VDD increased from 3 pins to 4 pins (Page 4)
- VSS increased from 4 pins to 8 pins (Page 4)

Section 6

PROGRAMMING & FIRMWARE MANUAL

6.0 Programming

- Address Spaces
- Memory Management
- Cache Control
- System Control and Error Handling
- Memory
- Onboard Peripherals
- Interrupts
- LED's

6.1 Address Spaces

The SPARC architecture defines several different address spaces. A user program can only access two of them: User Data and User Instruction spaces. When the processor is in supervisor mode additional address spaces are available (even beyond the obvious Supervisor Instruction and Supervisor Data) using the load and store alternate instructions. The User and Supervisor Instruction and Data spaces make up Device space (for memory and peripherals). The other ASIs make up Control space, intended for use by system functions, such as cache and MMU control.

6.1.1 Control Space

The CPU-2CE supports System Space, Segment Map, Page Map, and various cache Flush spaces.

Table: 6.1 Address Space Indicator Contents			
ASI#	Use	Address Space Type	Where Defined
0x0	Reserved	Control Space	-
0x1	Reserved	Control Space	-
0x2	System space	Control Space	System Control Space
0x3	Segment Map	Control Space	MMU Control Space
0x4	Page Map	Control Space	MMU Control Space
0x5	High Speed Segment Flush	Control Space	Main Memory Cache Control Space
0x6	High Speed Page Flush	Control Space	Main Memory Cache Control Space
0x7	High Speed Context Flush	Control Space	Main Memory Cache Control Space
0x8	User Instruction	Device Space	Device Space
0x9	Supervisor Instruction	Device Space	Device Space
0xA	User Data	Device Space	Device Space
0xB	Supervisor Data	Device Space	Device Space
0xC	Flush Cache by Segment	Control Space	Main Memory Cache Control Space
0xD	Flush Cache by Page	Control Space	Main Memory Cache Control Space
0xE	Flush Cache by Context	Control Space	Main Memory Cache Control Space
0xF	Hardware Virtual Flush	Control Space	Main Memory Cache Control Space

6.1.2 System Space (ASI = 2)

System Space can only be accessed by using load and store alternate instructions while the processor is in supervisor mode. It is used to access various system control registers as well as for direct access to cache tags and data for diagnostic purposes.

Table: 6.2 System Space (ASI=2)

Address	Use	Comment
0000 0000	ID PROM	Not available on CPU-2CE
3000 0000	Context Register	
4000 0000	System Enable Register	
6000 0000	Bus Error Register	
7000 0000	Diagnostic Register	Unused; no timeout
8000 0000	Cache Tags	Diagnostic use
9000 0000	Cache Data	Diagnostic use
F000 0000	Serial Port	MMU Bypass

ID PROM

The ID PROM is unused on the CPU-2CE. The equivalent function is provided in the NVRAM, located in Device Space.

Context Register

The Context Register is a 4-bit read/write register which controls which of 16 MMU contexts are in current use. Writes to this register have an immediate effect, subject to the normal pipelining imposed by the SPARC processor. The context register should be written and read as a byte only. The context Register is cleared on Reset.

System Enable Register

The System Enable register controls the cache and SBus as follows:

Table: 6.3 System Enable Register System Enable Register: ASI=2, Address=0x40000000)

BOOT	Zero	DMA_EN	CACHE	Zero	RESET	Zero	TLBDis
7							0

BOOT/ 0 = All Supervisor References go to EPROM (MMU bypass)
 1 = Normal operation (MMU Enabled)

Zero Always reads as zero; unused

DMA_EN 1 = SBus DMA allowed.

CACHE 1 = Enables cache. (Cache tags should be cleared before enabling)

RESET 1 = System Reset

TLBDis 1 = MicroTLB disabled. For hardware debugging only; should be zero.

The System Enable register is cleared on Reset, leaving the system in Boot state with DMA and cache disabled.

Synchronous Error Register

There are six Error Registers in control space: two address and two status registers for Synchronous and Asynchronous errors. (In addition to these registers, the parity control/status register exists in Device space.)

Table: 6.4 Synchronous Error Register

Synchronous Error Register: ASI=2, Address=0x6000 0000 WORD ACCESS ONLY:

Write 15	Zero	Zero	Zero	Zero	Zero	Zero	Zero 8
Invalid 7	Protection	Timeout	SBEError	MEMError	Zero	Zero	Watchdog 0

Table: 6.5 Synchronous Error Register	
Write	1 = Error occurred during a write cycle
Invalid	1 = Valid bit was zero in a page map entry or top 3 adr bits differ.
Protection	1 = Protection error, write to read-only page, or user access of supervisor-on ¹ page.
Timeout	1 = Non-existent device was accessed.
SBEError	1 = SBus device returned error-ack (device-dependent).
MEMError	1 = Memory (parity) error. (LErr on SBus) See parity register
Watchdog	1 = Restart due to IU error. (Trap while traps disabled.)

In certain circumstances the IU will be given a memory exception which it ignores (such as when an annulled instruction falls on an invalid page). In this case, the Invalid error bit would be set in the synchronous error register, but the IU did not take a trap. The result is that later, more than one bit may be set in the Error Register and software must determine which is correct from the address in the Synchronous Error Address Register.

Synchronous Error Address Register

Table: 6.6 Synchronous Error Register: ASI=2, Address=0x6000 0004 WORD ACCESS ONLY:

Virtual Address of last synchronous error (A31:0)

The Synchronous error registers will always hold information about the last synchronous error which occurred.

The SEVAR is cleared on power-on or programmer-initiated reset. Synchronous error registers are not cleared on a watchdog reset.

Asynchronous Error Register

The Asynchronous Error register reports buffered write errors and SBus-Controller detected DVMA Errors.:

Table: 6.7 Synchronous Error Register: ASI=2, Address=0x6000 0008 WORD ACCESS ONLY:

Size 1	Size 0	Invalid	ProtErr	Timeout	DVMAErr	Zero	Zero	SBErr	Multiple
--------	--------	---------	---------	---------	---------	------	------	-------	----------

Table: 6.8 Synchronous Error Register	
Multiple	1 = More than one Async Error since AER last cleared.
SBErr	1 = IU Writing to a Slave device resulted in an Error Ack.
DVMAErr	1 = Error during DVMA access.
Timeout	1 = SB_AS_ asserted for more than 256 clocks, or DVMA illegal adrs.
ProtErr	1 = DVMA attempted write to read-only page.
Invalid	1 = MMU Valid bit not set for DVMA access.
Size[1:0]	Copy of IU Size bits for failing writeback operation.

Error Acknowledges during DVMA are not reported to the IU. It is the responsibility of the DVMA master to report errors to the IU.

Reading the Asynchronous Error register clears it. SB_Reset_ clears it. Any asynchronous error freezes both Async error registers until read (with the exception of the Multiple bit).

Asynchronous Error Address Register

(Word access only)

Table: 6.9 Asynchronous Error Register : ASI=2, Address=0x6000 0008 WORD ACCESS ONLY:

Size 1	Size 0	Invalid	ProtErr	Timeout	DVMAErr	Zero	Zero	SBErr	Multiple
9		7							0

Bits 31:10 are read-only zeros.

Asynchronous Error Data Registers

Data from the first asynchronous write error since the AER was last cleared is stored at ASI=2, Address 0x6000 0010 and 0x6000 0014. Word accesses only, please. These registers are cleared on reset. The data at 0x6000 0014 is only valid if the Size code is 11, indicating a doubleword store was taking place. Cleared to zeroes on power-on reset, software-reset, and watchdog reset.

Cache Tags and Data

The Cache Tags are located in ASI 2 with the most significant four bits of the virtual address 8. Tags are unique every 32 bytes for 64K bytes (in other words, there is a 64 K Cache, which uses a 32-byte linesize; there are 2K tags). Each tag is 32 bits wide and should be accessed only as a full word. Tags are formatted as follows:

Table: 6.10 Cache Tags

31	25	21	20	19	18	15	1	0
Zero	CID	W	S	V	Zero	Tag ID		Zero

Cache Data is located in ASI 2 with the most significant four bits of the virtual address = 0x9. It is significant for 64K Bytes.

Serial Port Bypass

The Serial Port at ASI = 2, Address 0xF000 0000 is intended as an MMU bypass to allow the boot PROM to give messages even though the MMU and DRAM are not functioning or verified.

Segment Map

ASI = 3 gives access to the MMU segment map. Only bits 18 through 29 of the address are significant in this address space. See the MMU section for details.

Page Map

ASI = 4 gives access to the MMU page map. Only bits 12 through 29 are significant in this address space (bits 18 through 29 are mapped through the segment map). See the MMU section for details.

Flush Cache Segment

A *write* to an address in ASI = 0xC will cause the cache line corresponding to that address to be flushed (marked invalid) if:

- Cache Tag address (29:18) matches virtual address (29:18) and
- [that page is marked supervisor-only or the cache tag context matches the current context id]

Flush Cache Page

A *write* to an address in ASI = 0xD will cause the cache line corresponding to that address to be flushed (marked invalid) if:

- Cache Tag address (29:16) matches virtual address (29:16)

and

- [that page is marked supervisor-only or the cache tag context matches the current context id]

Add Hardware Cache Flushes!!

Flush Cache Context

A *write* to an address in ASI = 0xE will cause the cache line corresponding to that address to be flushed (marked invalid) if:

The cache tag context matches the current context and the supervisor-only bit for that cache tag is *not* set. Cache context flushes will not invalidate supervisor-only pages. (All supervisor-only pages are assumed to be mapped throughout all contexts.)

6.1.3 Device Space

Device Space corresponds to memory and peripherals. In general, memory is located in Type Zero space, on board and SBus IO devices are located in Type 1 space, and VME devices are located in type 2 or 3 space, (this space indicator is part of the MMU page table entry, and could be thought of as just an extension to the physical address).

Type Zero Space

The CPU-2CE supports up to 64 MBytes of DRAM on-board, and up to 128 MBytes with the optional SRX board.

Type One Space

Type 1 Space contains on-board IO devices and SBus slots. It is distinguished from Type Zero space by the MMU Page Table Entry. By convention all addresses have the MS-Address bits are set to 1 (PA28-31.)

Table: 6.11 Type One Space

<u>Address</u>	<u>Device</u>
F000 0000	Keyboard and Mouse serial ports
F100 0000	TTYA and B serial ports
F200 0000	TOD Clock and NVRAM
F300 0000	Counter-Timer Registers
F400 0000	Memory Error Register (Parity) (On-board memory)
F400 0004	Memory Control Register (On-board memory)
F400 0008	Memory Error Register (Parity) (Off-board memory)
F400 000C	Memory Control Register (Off-board memory)
F500 0000	Interrupt Control Register
F600 0000	EPROM
F720 0000	Floppy Controller
F720 1000	ISDN/Audio chip
F740 0000	Auxiliary IO register
F800 0000	SBus Slot Zero (Onboard)*
FA00 0000	SBus Slot One*
FC00 0000	SBus Slot Two*
FE00 0000	SBus Slot Three*

* All Sbus Slots occupy 25 bits of Address Space.

Serial Ports

The serial port controller used for both the Keyboard/Mouse and the general-purpose TTY serial ports is the Zilog 85C30 Serial Communications Controller. For further information see the Zilog datasheet. The sub-addresses for the SCC's are:

F000 0000	Mouse Control Port
F000 0002	Mouse Data Port
F000 0004	Keyboard Control Port
F000 0006	Keyboard Data Port
F100 0000	TTY B Control Port
F100 0002	TTY B Data Port
F100 0004	TTY A Control Port
F100 0006	TTY A Data Port

- Interrupts occur at level 12.
- The serial ports use a 4.915 MHz clock as a base for baud rate generation.
- There is no need for software to provide timing loops between accesses to the serial ports. Hardware will guarantee the required recovery time of the SCCs.

TOD/NVRAM

The Time-of-Day clock and Non-Volatile RAM is implemented in a Thomson Mostek MK48T08 RAM with the following address map:

Note: To insure proper SunOS bootup FORCE uses only 2 Kbytes of the 8 Kbyte chip. Through an inverter in a pal, FORCE modifies the base address to be the last 2 Kbytes of the NVRAM. A12:13 are inverted on the TOD.

F200 0000	-NVRAM
F200 07F7	Reserved, do not use
F200 07F8	TOD Control
F200 07F9	Seconds (00-59)
F200 07FA	Minutes (00-59)
F200 07FB	Hour (00-23)
F200 07FC	Day (01-07)
F200 07FD	Date (00-31)
F200 07FE	Month (01-12)
F200 07FF	Year (00-99)

Counter-Timer Registers

Two 21-bit microsecond resolution counters can provide interrupts at levels 10 and 14. They are at the following addresses:

F300 0000	Counter 0
F300 0004	Limit 0 Interrupt Level 10
F300 0008	Counter 1
F300 000C	Limit 1 Interrupt Level 14.

- Interrupts are enabled or disabled in the Interrupt Control Register.
- Each counter starts at a value of "one microsecond" (0x400)
- When a counter reaches the value in its corresponding Limit register the Limit bit is set (L) and an interrupt (if enabled) is generated.
- The Limit registers initialize on reset to zero.
- Reading the Limit Register clears its L bit. Reading the Counter does not.
- Read counters as a word to guarantee that they do not increment during the read.

Memory Error Registers

The Parity Control/Status register for the bottom 64 MBytes of DRAM is at F400 0000 in Type 1 space. The Parity Control/Status register for the top 64 MBytes of DRAM is at F400 0008 in Type 1 space:

Perror	Set on any parity error
2ndError	Set when parity error occurs and PError=1
PTest	Set to invert parity for testing.
PEnable	Set to enable checking
Perr[24-0]	Set to indicate which byte failed.

The bits that indicate errors (Perror, 2ndError, Perr n) are cleared when the register is read. All bits are cleared on reset.

The Perr n bits have the same meaning as on older hardware (note that early older documentation was incorrect).

In word-parity mode, all four Perr bits will be set on any parity error.

Memory Control Register

The RAM+ control register is at 0xF4000004 in Type 1 space for the on-board DRAM and at 0xF400 000C in Type 1 space for off-board DRAM:

Table: 6.15 Type 1 Space for off Board DRAM

Unused 7	Unused	Unused	Unused	Unused	Unused	Dis Ref	Parity33 0
-------------	--------	--------	--------	--------	--------	---------	---------------

Dis Ref when set will stop refresh. This is for manufacturing test only.

Parity when set will use word-wide parity. It should remain cleared on CPU-2CE.

The Memory Control Register is cleared on Reset.

Interrupt Control Register

The Interrupt Control Register is at F500 0000:

Table: 6.16 Interrupt Control Register	
Enx	Enable Interrupt Level x.
SWIx	Force interrupt at Level x
Master Enable	Set to allow any interrupts.

Clear bit 0, the Master Enable bit, to reset level 15 interrupts.

Floppy Controller

The Intel 82072 Floppy Disk Controller is at F720 0000. It should be accessed in byte mode only. See the Intel datasheet for details of the register bits.

F720 0000 MSR (reads), DRS (writes)

F720 0001 FIFO (reads and writes)

ISDN/Audio

The AMD 79C30 ISDN/Audio chip is at F720 1000. See the AMD datasheet for details. The ISDN portions of this chip are not utilized on CPU-2CE hardware or software.

Aux I/O Register

The Auxiliary I/O register is at F740 0003 as a byte-only device. It has eight bits which may be used as inputs or outputs. Only six of these bits are used on the CPU-2CE. To use a bit as an input, it should be written as a "1" (it will read back as whatever value is driven onto it). This register is set to all 1's on reset.

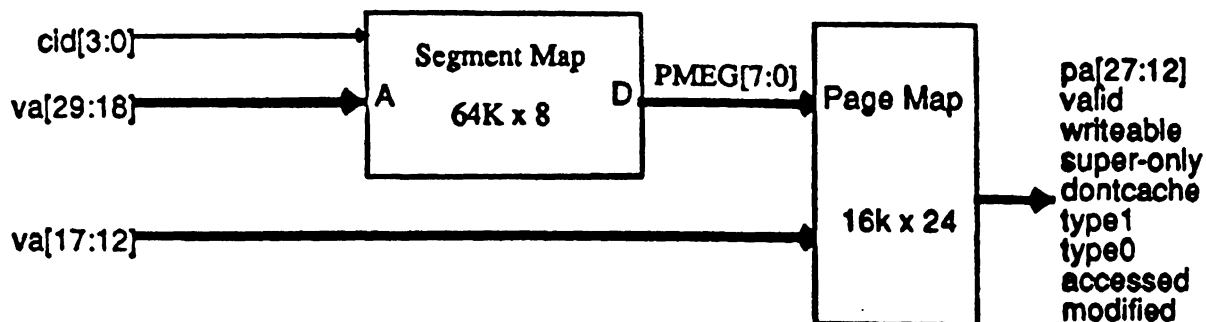
Table: 6.17 Aux I/O Register

LED	1 = Front-panel LED On
Eject	Transition from 0 to 1 to 0 ejects floppy. DriveSel must be set for eject to take place.
TC	1 = Terminal Count. Must be driven for specific amount of time, dependent upon floppy data rate. (See <i>82072 Users Manual</i> for timing constraints.)
DriveSel	1 = Select Floppy Drive.
DiskChg	(Input) Reads as one if drive is selected and no disk is present in the drive.
Density	(Input) Reads as one if floppy drive detects high-density floppy inserted. (Current floppy drives do not support this signal; software must determine density by trial and error.)

6.2 Memory Management

The CPU-2CE provides a standard two-level MMU, identical to older designs except for sizes and performance.

- 16 MMU contexts (up from 8 on older)
- 30 Virtual Address bits in
- 32 Physical Address bits out to VME (28 bits to SBus and onboard devices)
- 256 Page Map Entry Groups (up from 128 on older)
- 4K Byte Page Size
- 256K Byte Segment Size

Figure: 6.1 Illustration Memory Management

6.2.1 Address Translation

As long as the system is not in Boot state (system enable register bit 7 not zero), all virtual addresses put out by the CPU in Supervisor and User Instruction and Data spaces are translated through the MMU. The upper 12 significant virtual address lines, along with the current context number, select one of 256 Page Map Entry Groups. This value, along with the lower 6 significant virtual address lines, selects a Page Map Entry which returns a Physical Address and the permission and statistics bits for the 4KByte page.

Exception: The lower address bits of EPROM accesses are not translated. The EPROM should always be mapped into contiguous virtual memory locations on a EPROM-sized boundary.

6.2.2 Modifying the MMU

In order to store a new value into the Segment map, simply write a byte to ASI=3. Only virtual address bits [29:18] along with the current context ID [3:0] are significant for this operation. PMEG values may be read as well. *Use only byte loads and stores for segment map operations.*

In order to store a new value into the Page map, the segment map must first have been initialized for the high-order address bits to choose a Page Map Entry Group in the current context. Then a *word* may be stored to ASI=4 to set the page map value:

Changing the MMU for a virtual address which was marked cacheable requires that the cache be flushed of that address range.

6.2.3 MMU Protection Criteria

All Device Space accesses (when the system is not in Boot mode) are subject to address translation (with the exception of the EPROM, as listed above), and are subject to MMU protection checks. These checks basically validate whether a given access is allowed for the current user. Using the bits in the Page Map entry given above, access is allowed if:

- The V bit is set *and*
 - The S bit is cleared or the CPU is in Supervisor state *and*
 - The W bit is set or this is a read operation (not a store and not a ldst).

 - The X bit in the Page Map entry is set to cause this page to *not* be cached.

 - The Type bits are effectively extensions to the physical address; only values of 0 and 1 are used on the CPU-2CE.
- 0- Onboard memory 1- Onboard I/O
2- VME D16 3- VME D32
- The A bit is set if this page has been accessed. The bit is cleared by software and used in least-recently-used algorithms.

- The M bit is set if this page has been modified.

In addition, the most significant three virtual address bits must all be one or all be zero, or an error will be reported in the AER as a DVMA Timeout, when accessing Type 0 or 1 Space.

The statistics bits are not modified in the event of any protection or illegal address violations. Accesses which result in Error Ack or timeouts from the SBus will have modified their MMU entries as if the access had been successful.

All DVMA activity takes place in Context Zero as Supervisor.

6.3 Programming LED's

Table: 6.18 LED/SW3 Register (Longword access only)

Bit #	RO	WO
31	SW3 bit 3	L11
30	SW2 bit 2	L10
29	SW1 bit 1	L21
28	SW0 bit 0	L20
27	VME Sysfail* from S4VME asic	R15*
26	P1_Sysfail* (from p1)	Init Enable Sysfail and Abort SW1
25	abort	MAR Enable for 32 Bit Mode
24	N/C (undefined)	

Table is from page 17 of the schematics. L is LED

6.3.1 SPARC CPU-2CE LED Programming Example

Programming examples for CPU-2CE front panel LED's

```

#include <asm_linkage.h>
#include <cpu_addr.h>
#include <mmu.h>
#include <scc.h>
#include <map.h>
#include <misc.h>
#include <ce_misc.h> (SEE BELOW FOR EXAMPLE OF THIS FILE)

extern      map_memory();
extern      unmap_memory();

static byte shadow_reg;          /* hold current value of ledsw */

/*
 *      Set and leave LEDs continuously on.
 */
set_leds(pattern)
byte pattern;
{
    byte *ledsw = (byte*)LEDSW_LOC;
    byte temp;

    map_memory(ledsw, LEDSW_PADDR, VW1_ATTRIBUTES, PSIZE);
    shadow_reg &= LEDMASK; /* clear LED field */
    shadow_reg |= pattern;
    shadow_reg &= ~RUNMODE;          /* enable LED function */
    ledsw[0] = shadow_reg;
    unmap_memory(ledsw, PSIZE);
}

/*
 *      Set Run-Mode
 */
set_runmode()
{
    byte *ledsw = (byte*)LEDSW_LOC;
    byte temp;

    map_memory(ledsw, LEDSW_PADDR, VW1_ATTRIBUTES, PSIZE);
    shadow_reg |= RUNMODE;
    ledsw[0] = shadow_reg;
    unmap_memory(ledsw, PSIZE);
}

/*
 *      Read the rotary switch.
 */
byte read_sw()
{
    byte *ledsw = (byte*)LEDSW_LOC;
    byte temp;

    map_memory(ledsw, LEDSW_PADDR, VW1_ATTRIBUTES, PSIZE);
    temp = ~(ledsw[0] >> 4);
    unmap_memory(ledsw, PSIZE);
    return (temp & 0xf);
}

```

6.3.1.1 Example of the ce-misc.h file

```

/*
 * file: ce_misc.h
 */

/* CPU-2CE front panel LED definitions */

#define LEDSW_LOC 0x1fa00000
#define LEDSW_PADDR 0xff000000 /* Slot 3, offset +0x1000000 */

#define LEDMASK 0x0f /* LED field mask */
#define INTENABLE 0x02 /* Interrupt enable: Set/Clear IL15 */
#define RUNMODE 0x08 /* Run-mode: enable LEDS */
#define DIAGMODE 0x00 /* Diagnostic mode: disable LEDS */

```

6.4 Additional Forth Commands for LEDs/Hexswitch and (Optional Feature Flash EEPROM)

These are added to the CPU-2CE OBP:

1. LEDS/HXSW:

- a. set-diagmode (--)
Let user changes mode to DIAGNOSTIC MODE or turn off RUN MODE.
- b. set-runmode (--)
Let user changes mode to RUN MODE.
- c. .hexswitch (-- value)
Display value of on board hexswitch.
- d. leds! (value --)
Set color combination for front panel LEDs. Bit combination for LEDS is as follows (in Hex):

00	:	Red	01	:	Green	10	:	Yellow	11	:	Blank
----	---	-----	----	---	-------	----	---	--------	----	---	-------

Example:

```

Value = LED2 LED1
          00 00 --> 0x0 --> RED RED
                                     ( will generate sysfail
                                     )
          00 01 --> 0x1 --> RED GREEN
          01 01 --> 0x5 --> GREEN GREEN
          11 11 --> 0xF --> BLANK BLANK

```

- e. .leds (-- value)
Display value of front panel LEDS.
- f. p2led-on (--)
Turn on P2 LED.
- g. p2led-off (--)
Turn off P2 LED.

2. FLASH EEPROM:

- j. erase-flash (--)
Erase Flash EEPROM.
- k. burn-flash (bvadr size --)
Given the virtual address of the buffer and the size in-byte of the user program, this command will burn that program into the Flash EEPROM.

Section 7

VME

7.0 VMEbus Interface

7.1 Features of the SPARC CPU-2CE VMEbus Interface

The features of the SPARC CPU-2CE VMEbus interface are listed below:

- A32/A24/A16 Master (Sun-4 VME Device) and A24/A32 Slave DVMA device as SBus DMA device
- Single-level and round-robin arbitration with bus arbiter timer
- VME Interrupt Handler with IACK Daisy Chain Driver
- Clock driver
- System resetter
- Full 4-gigabyte VME addressing with mapping register
- Simple user-setup provided, only one jumper is required (slot one). All other options are selected through register settings, stored in EEPROM.
- A special loopback cycle is provided to enable stand-alone testing of the interface.
- Watchdog Timer for VME System Bus which issues a VMEbus error.
- Fair Requester option provides all VME Masters along the bus grant daisy chain with equal access to the VMEbus, independent of the Master's slot location.
- Programmable Mailbox Interrupt Level
- Adjustable Rerun Counter allows for longer rerun intervals during VME access
- VME Interrupt Monitor Register provides ability to monitor VME Interrupt lines in real time
- Multiprocessing support with Mail Box Interrupts, programmable slave base address ranges, true Read Modify Write (RMW) to local memory and to shared VME resource with the use of the VME Bus Locker
- VME DVMA can be disabled via the DVMA Disable Bit. This can be done without affecting the SPARC CPU-2CE's Ethernet or SCSI ports.
- Bus Locker provides the ability to perform Read-Modify-Write (RMW) cycles on the VMEbus.

All of the features listed above (except Watchdog Timer) require SunOS 4.1.1 (or newer). Each of the features is optional in that each can be enabled or disabled.

The Watchdog Timer feature works with any SPARC CPU-2CE-compatible release of SunOS and is enabled when the CPU is configured as a Slot 1 Controller.

The VME Chip includes all VME logic on a 7000 gate CMOS LMA9K 1.5 m-second Gate Array. Address and data paths are external, using 29FCT521, 29FCT52, 29821 and 29FCT821 registers. Decoding of VME address A(31:24) is done externally using AS27 gates. Decoding of Physical address PA(19:16) is done externally using an F20. All VMEbus outputs are driven through bipolar drivers using AS641, LS04 and F125.

7.2 VMEbus Basics - An Introduction

The VMEbus is an asynchronous 32 bit bus, with multi master, multi level arbitration specifying Single Cycles, Block Mode Cycles and Read-Modify-Write Cycles. Table 7.1 defines certain conceptual logical units.

Table: 7.1 VME Concept Definitions

Concept	Definition
Master	accesses Slaves.
Slave	responds with [DTACK/BERR] to Master.
BUS Requester	Requests and keeps the bus on order from a Master.
BUS ARBITERS	Grants bus to Requesters through [BG] daisy chain. Must be in slot 1.
Interrupter	Interrupts bus, responds to Interrupt handlers with [DTACK/BERR] and supplies an Interrupt Vector.
IACK DAISY CHAIN Drive	Drives [IACKIN/IACKOUT] chain when [IACK] and [DS(1:0)] are asserted. Must be in slot 1.
SYS Clock Driver	Drives [SYSCLK]. Must be in slot 1.
Sysfail is generated by writing red red to the LED port. See the LED programming example.	Asserts Sysfail
An on card Voltage Supervisor asserts SYSRESET for a minimum of 200 ms after +5V DC has reached 4.5 Volts, if the slot 1 jumper is inserted. The ACFAILL is not driven or monitored. Incoming SYSRESET generates an on-card reset.	Asserts [SYSRESET].

7.3 VME Performance

The timing for the VMEbus interface is as follows:

8 Bit Transfers	1.6 MB per second
16 Bit Transfers	3.2 MB per second
32 Bit Transfers	6.4 MB per second

The VME Gate Array logic meets the Sbus and VMEbus specifications at a 20 MHz clock, correlating to a 50 ns clock cycle.

Bus arbitration for other VMEbus masters takes place faster by using the Early Release feature.

NOTE: These numbers are given only as a way to describe the SPARC CPU-2CE CPU performance. The overall bandwidth depends on the other card's response time, the application code, amount of other tasks running concurrently, other concurrent DMA activity, cache hit/miss ratio, etc. These figures should NOT be expected in a real world system.

7.4 VME Addresses

Two Address Modifier bits (AM[5:4]) define three address spaces.

Table 7.2 shows how these address spaces are defined.

Table: 7.2 VME Addresses		
Concept	Definition	Address Modifier Bits 5:4
A16	16 address bits are needed. Can address 64 KB.	10
A24	24 address bits are needed. Can address 16 MB.	11
A32	32 address bits are needed. Can address 4 GB.	00

Three Address modifier bits (AM[2:0]) indicate privilege and type of access: Supervisor, Non-Privileged, Program Data or Block Transfer access. Data transfers can be D32/D16/D8 (quad-byte, double-byte or single-byte size). Unaligned transfer means that a quad-byte or a double-byte transfer takes place on a non-even boundary. Unaligned transfers are not supported by the SPARC CPU-2CE card.

Table 7.3 lists the supported VME address spaces.

Table: 7.3 VME Address Spaces

Addresses	Comments
A16	Master accesses are possible.
A24	Master accesses are possible.
A32	Master accesses are possible.
D16	Master accesses are possible.
D32	Master accesses are possible.
A16	Mailbox cycles are monitored and can generate the interrupt.
D8 (only)	Mailbox cycles are monitored and can generate the interrupt.
A24	Slave cycles are supported.
A32	Slave cycles are supported.
D8	Slave cycles are supported.
D16	Slave cycles are supported.
D32	Slave cycles are supported.

7.5 VME Implementation

The VME Interface adheres to the VMEbus Specification Rev C.1. A later version of the specification, released by ANSI, the ANSI/IEEE 1014-1987, has changed some optional features to mandatory. The new rules are: 2.61-68 and 4.49. The SPARC CPU-2CE VME Interface follows all these new rules except for 2.67 requiring ALL D32 slaves to include Unaligned Transfer capability. The following VMEbus features are implemented:

- A 300 micro-second VMEbus timer. Master cycles time out after 300 micro-second generating a Bus Error on VMEbus. This is with onboard master setup in slot 1.
- Single-level VMEbus arbitration (SGL) and round-robin arbitration (RRS).
- Read-Modify-Write accesses to local and VME memory are supported via the bus locker.
- Unaligned transfers are not implemented nor supported. Unaligned accesses get a Bus error returned. Unaligned transfers are 32-bit accesses not to address (1:0)=00 and 16-bit accesses not to address (0)=0. See the ANSI/IEEE change described above.

- Address only cycles are not decoded, an active data strobe is required for the start of any DVMA access.
- The VME Interface allow Address pipe-lining during Slave accesses, but do not utilize the function on VME Master cycles.
- Early release of bus is possible. Other masters can arbitrate getting bus mastership while VME Master access is finishing (Address Strobe still asserted). The VME Interface is also prepared for other cards early release of the bus, by awaiting the negation of the Address Strobe.
- The VME Interface requests VME mastership on bus level 3 as an ROR (Release On Request).
- Bus Clear is not monitored.
- An Interrupt handler that can serve any of the seven interrupts is available. A selection of which levels to serve is done by programming a SW register. The Interrupt handler is a D08(0), handling only 8-bit interrupt vectors.
- No Interrupter function is provided. Signaling between CPU cards in multiprocessor configurations is done through the Mail Box Interrupt.
- An IACK Daisy Chain function is provided when the card is in slot one.
- A System Clock Driver is active, if the card is in slot one.
- No Serial Clock Driver is implemented.
- An on card Voltage Supervisor asserts SYSRESET for a minimum of 200 ms after +5V DC has reached 4.5 Volts, if the slot 1 jumper is inserted. The ACFAILL is not driven or monitored. Incoming SYSRESET generates an on-card reset.
- Sysfail is generated on the bus by writing red red to LEDs.

7.6 Major VME Register Groups

This section lists the major groups of VME registers. The registers can be divided into the following groups:

1. Registers set on system initialization, replacing card jumpers:

Interrupt Enable Register.

Interrupts handled.

Arbitration mode.

2. Registers used for multi-processing or co-processing:

Interrupt Monitor Register

Mailbox Interrupt Register

Rerun/Mailbox Level Register

Slave Map Register (also replacing jumper).

3. Others:

A32 VME Address mapping.

Move 512 MB window.

Enable diagnostic loop back.

Slave Map register bit0.

Enable Block Mode transfers.

Table 7.4 lists the VME registers, their physical base, and size.

Table: 7.4 VME Registers

Name	Type	Physical Base	Size
VME Bus Locker	1	0xEFE00000	byte
VME IACK cycle	1	0xEFE00001	byte A[3:1], A0=1
Mail Box Register	1	0xEFE00010	byte
VME Interrupt Enable Register	1	0xEFE00014	byte
A32MAP Register	1	0xEFE00018	byte
Slave Map Register	1	0xEFE0001C	byte
Rerun/Mailbox Level Register	1	0xEFE00008	byte
Interrupt Monitor Register	1	0xEFE00004	byte

7.6.1 Accesses To Byte Registers

A byte access to a register involves a data transaction on the eight most significant data lines (D[31:24]). The information on the rest of the data lines (D[23:0]) is ignored. This applies to all byte register accesses listed in this chapter.

7.7 Master Interface

In compatibility mode a full 32-bit Master Interface is provided. Complete mapping of 512 MB is possible all the time, but this 512 MB window can be moved through all 4 GB through a VME Mapping register.

In enhanced mode the additional MMU address bits are supplied to the VMEbus allowing complete access to any addresses on the VMEbus. No window limitations exist.

A32, A24, A16 Address Modes, D32, D16, D8(E0) Data Sizes are supported.

Master Cycles (RMW) Atomic load-store

On Master cycles atomic load-store cycles are implemented according to the VME spec. The VME data strobe asserts two times as the VME address strobe is held asserted. This differs from other Sun cards, where instead the ownership of VME is kept while a read and a write takes place. The problem with this is that if the access is to a card with several devices competing for an on-card bus, there is no way this card can recognize the access as a RMW. For example: a disk card with multiple on card devices would not be able to prevent a shared resource to be stolen in between what looks like a standard Read and Write even though the VMEbus is locked.

Block Mode Transfers

Block mode transfers are not supported. (They are only supported by the Slave interface.)

Unaligned Transfers

Unaligned transfers (UAT) are not supported. The following access types are not supported and result in a bus error timeout:

- 16-bit access to base address plus one or three
- 24-bit access
- 32-bit access to base address plus one, two, or three
- 32-bit access to type two space

The CPU-2CE can directly map 4 GB with software switch enabled.

7.7.1 A32 Map Register Base Location

This register can re-map VME Address bits A[31:29] to enable accesses to any 512 MByte range of VME addresses. Only 8-bit accesses should be used; 16- or 32-bit accesses are not acknowledged, resulting in a bus error time out. This register is only needed when operating the CPU-2CE in compatibility mode.

7.7.2 A32 Map Register Initialization

The register is cleared to zeroes on reset, thus mapping the 512 MB MMU mapping to 0-512 MB and disabling loopback mode.

Table 7.5 provides a breakdown of the A32 Map register. Table 7.6 provides a breakdown of the A32 Map register bits. Table 7.7 defines the functions of various bit values.

Table: 7.5 A32 Map Register

Type	Device Address	Device	Physical Space
1	0xEFE00018	A32 Map Register	1 byte

Table: 7.6 A32 Map Register Address Bits

31	30	29	28	27	26	25	24
VME Master Address			0	0	0	0	LoopB

Table: 7.7 A32 Map Register Bit Definitions

Bit	Range	Value	Range Start	Range End
31:29	0x0 Maps VME Master Range to:	0x00000000-0x1 FFFFFFF	0 MB	512 MB
31:29	0x1 Maps VME Master Range to:	0x20000000-0x3 FFFFFFF	512 MB	1 MB
31:29	0x2 Maps VME Master Range to:	0x40000000-0x5 FFFFFFF	1 GB	1.5 GB
31:29	0x3 Maps VME Master Range to:	0x60000000-0x7 FFFFFFF	1.5 GB	2 GB
31:29	0x4 Maps VME Master Range to:	0x80000000-0x9 FFFFFFF	2 GB	2.5 GB
31:29	0x5 Maps VME Master Range to:	0xA0000000-0xB FFFFFFF	2.5 GB	3 GB
31:29	0x6 Maps VME Master Range to:	0xC0000000-0xD FFFFFFF	3 GB	3.5 GB
31:29	0x7 Maps VME Master Range to:	0xE0000000-0xF FFFFFFF	3.5 GB	4 GB
28:25	These bits are not used, and always read back as 0.			
24	This bit enables the VME loopback mode (LoopB), used for diagnostics only.			

7.7.3 VME Registers Programming Example

The following code fragment shows how to handle a VME interrupt. The system interrupt handler, when it determines that an interrupt originates on an IU level which corresponds to a VME interrupt, calls one of the entry points defined below (`_vmelevel1-7`). We then attempt to obtain the actual VME vector by reading the VME IACK register appropriate to our VME interrupt level. If the vector obtained is a valid VME vector, we use the vector as an index into the table "vme_vector." "Vme_vector" is a table of pointers to interrupt handlers for various VME devices.

```

/*
 * Handle the 7 levels of VME interrupts:
 *   Read the VME IACK register appropriate to
 *   the VME level. This will:
 *       (1) Perform an interrupt acknowledge cycle
 *       (2) Get the VME vector of the interrupting
 *           device.
 */
.globl _vmelevel1, _vmelevel2, _vmelevel3
.globl _vmelevel4, _vmelevel5, _vmelevel6
.globl _vmelevel7

_vmelevel1:
    set    VME_IACK1, %g1        ! %g1 = address of IACK register
    b     _vme_read_vector      ! go read the VME vector
    nop

_vmelevel2:
    set    VME_IACK2, %g1        ! %g1 = address of IACK register
    b     _vme_read_vector      ! go read the VME vector
    nop

_vmelevel3:
    set    VME_IACK3, %g1        ! %g1 = address of IACK register
    b     _vme_read_vector      ! go read the VME vector
    nop

_vmelevel4:
    set    VME_IACK4, %g1        ! %g1 = address of IACK register
    b     _vme_read_vector      ! go read the VME vector
    nop

_vmelevel5:
    set    VME_IACK5, %g1        ! %g1 = address of IACK register
    b     _vme_read_vector      ! go read the VME vector
    nop

_vmelevel7:
    set    VME_IACK7, %g1        ! %g1 = address of IACK register
    b     _vme_read_vector      ! go read the VME vector
    nop

/*
 * This entry is called directly from interrupt.
 */
_vmelevel6:
    set    VME_IACK6, %g1        ! %g1 = address of IACK register
    b     _vme_read_vector      ! go read the VME vector
    nop

/*
 * VME interrupt. Do vectoring.
 * The VME vector is read off the VME bus. If this fails (data fault),
 * _trap will check the PC of the fault and jump to _spurious.
 *
 * The vmelevel# code sets up %g1 to contain the address of the
 * appropriate VME IACK register before it branches here.
 */
.global      _vme_read_vector

#define      VEC_MAX          255
#define      VEC_MIN          64

_vme_read_vector:
    ldub    [%g1], %l6           ! read vector #, acknowledge int.
    cmp     %l6, VEC_MAX         ! check vector limits
    bg     spurious_vme
    subcc   %l6, VEC_MIN, %g3     ! normalize vector
    bl     spurious_vme

```

```

sll    %g3, 2, %g5          ! scale for interrupt counting
sll    %g3, 3, %g3          ! scale vector
set    intrcnt, %g4        ! per-device interrupt counts table
ld     [%g5 + %g4], %g1    ! interrupt count
set    vme_vector, %g2    ! table of interrupt vectors
inc    %g1                 ! count interrupt
st     %g1, [%g5 + %g4]    ! and store result
ld     [%g2 + %g3], %g1    ! get handler address
add    %g2, 4, %g2        ! generate address of arg ptr
ld     [%g2 + %g3], %o0    ! delay slot, get arg ptr
call   %g1                 ! call handler
ld     [%o0], %o0         ! read arg ptr

        b,a    int_rtt          ! restore previous stack pointer
/* end_vme_read_vector */
/*
 * Vme vectors are compatible with the sun3 family in which
 * there were possible valid vectors from 64 to 255 inclusive.
 * This requires 192 vectors, each vector is two words long
 * the first word being the interrupt routine address and the
 * second word is the arg.
 *
 * Vectors 0xC8-0xFF (200-255) are reserved for customer use.
 */
/*
 * The vme vectoring uses the following table of routines
 * and arguments. The values for the vectors in the following
 * table are loaded by autoconf at boot time.
 */
#define ERRV    .word _spurious, 0

        .seg    "data"
        .align 4

        .global    vme_vector
_vme_vector:
        | vector numbers
ERRV; ERRV; ERRV; ERRV ! 0x40 - 0x43  sc0 | sc?
ERRV; ERRV; ERRV; ERRV ! 0x44 - 0x47  xdc0 | xdc1 | xdc2 | xdc3
ERRV; ERRV; ERRV; ERRV ! 0x48 - 0x4B  xyc0 | xyc1 | xyc?
ERRV; ERRV; ERRV; ERRV ! 0x4C - 0x4F  future disk controllers
ERRV; ERRV; ERRV; ERRV ! 0x50 - 0x53  future disk controllers
ERRV; ERRV; ERRV; ERRV ! 0x54 - 0x57  future disk controllers
ERRV; ERRV; ERRV; ERRV ! 0x58 - 0x5B  future disk controllers
ERRV; ERRV; ERRV; ERRV ! 0x5C - 0x5F  future disk controllers
ERRV; ERRV; ERRV; ERRV ! 0x60 - 0x63  tm0 | tml | tm?
ERRV; ERRV; ERRV; ERRV ! 0x64 - 0x67  xtc0 | xtc1 | xtc?
ERRV; ERRV; ERRV; ERRV ! 0x68 - 0x6B  future tape controllers
ERRV; ERRV; ERRV; ERRV ! 0x6C - 0x6F  future tape controllers
ERRV; ERRV; ERRV; ERRV ! 0x70 - 0x73  ec?
ERRV; ERRV; ERRV; ERRV ! 0x74 - 0x77  ie0 | ie1 | ie?
ERRV; ERRV; ERRV; ERRV ! 0x78 - 0x7B  future ethernet devices
ERRV; ERRV; ERRV; ERRV ! 0x7C - 0x7F  future ethernet devices
ERRV; ERRV; ERRV; ERRV ! 0x80 - 0x83  vpc0 | vpc1 | vpc?
ERRV; ERRV; ERRV; ERRV ! 0x84 - 0x87  vp?
ERRV; ERRV; ERRV; ERRV ! 0x88 - 0x8B  mt10 | mt11 | mt12 | mt13
ERRV; ERRV; ERRV; ERRV ! 0x8C - 0x8F  SunLink SCP (System DCP-8804)
ERRV; ERRV; ERRV; ERRV ! 0x90 - 0x93  Sun-3 zs0 (8 even vectors)
ERRV; ERRV; ERRV; ERRV ! 0x94 - 0x97  Sun-3 zs1 (8 odd vectors)
ERRV; ERRV; ERRV; ERRV ! 0x98 - 0x9B  Sun-3 zs0 (8 even vectors)
ERRV; ERRV; ERRV; ERRV ! 0x9C - 0x9F  Sun-3 zs1 (8 odd vectors)
ERRV; ERRV; ERRV; ERRV ! 0xA0 - 0xA3  future serial
ERRV; ERRV; ERRV; ERRV ! 0xA4 - 0xA7  pc0 | pc1 | pc2 | pc3
ERRV; ERRV; ERRV; ERRV ! 0xA8 - 0xAB  cg2 | future frame buffers
ERRV; ERRV; ERRV; ERRV ! 0xAC - 0xAF  gpl | future graphics processors
ERRV; ERRV; ERRV; ERRV ! 0xB0 - 0xB3  sky0 | ?
ERRV; ERRV; ERRV; ERRV ! 0xB4 - 0xB7  SunLink / channel attach
ERRV; ERRV; ERRV; ERRV ! 0xB8 - 0xBB  (token bus) tbi0 | tbi1 | ?
ERRV; ERRV; ERRV; ERRV ! 0xBC - 0xBF  Reserved for Sun
ERRV; ERRV; ERRV; ERRV ! 0xC0 - 0xC3  Reserved for Sun
ERRV; ERRV; ERRV; ERRV ! 0xC4 - 0xC7  Reserved for Sun
ERRV; ERRV; ERRV; ERRV ! 0xC8 - 0xCB  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xCC - 0xCF  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xD0 - 0xD3  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xD4 - 0xD7  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xD8 - 0xDB  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xDC - 0xDF  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xE0 - 0xE3  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xE4 - 0xE7  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xE8 - 0xEB  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xEC - 0xEF  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xF0 - 0xF3  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xF4 - 0xF7  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xF8 - 0xFB  Reserved for User
ERRV; ERRV; ERRV; ERRV ! 0xFC - 0xFF  Reserved for User

```


7.8 Slave Interface

The slave interfaces operate in A32 and A24 address mode.

1 MB of memory is accessible, normally the lowest MB on VME, but this can be changed to be in the 1-16MB range to support multi-CPU card implementations. The address range is adjustable in 1MB steps.

Unaligned accesses to the Slave Interface range returns a Bus Error back to the external VME Master.

The address modifiers on VME are used to define Address mode (A32/A24/A16), privilege, and data/program/block-transfer access. The Slave decoder and Mail Box Interrupter monitors all address modifier bits with the exception of AM2, the access privilege bit.

All slave accesses as a DVMA device are set to supervisor mode, which is in accordance with the Sun-4 Architecture.

The 1 MB VME address space selected is always mapped onboard to the highest megabyte in the Virtual Address Space, again in accordance with the Sun-4 Architecture.

7.8.1 Slave Transfer Control

A bit in the cache System Enable Register can disable SDVMA and therefore all Slave cycles. If this bit is set, slave cycle can take place.

7.8.2 Slave Map Register

The address space for System DVMA accesses is 1 MB, normally the lowest Megabyte in accordance with the Sun-4 Architecture, but bits 27:24 can be used to re-map the SDVMA address space. The 1 MB address space on VME is mapped to the highest 1 MB within the Virtual Address space. During register accesses, only 8-bit accesses should be used. 16- or 32-bit accesses are not acknowledged, resulting in a bus error time out.

If a slave access occurs to a page within 1 MByte address space that does not have physical memory mapped to it then the VME BERR signal will be asserted. Therefore to properly utilize the slave feature, the programmer must insure that the slave map register is properly set to the correct 1 MByte VME address range, and that the physical memory pages are mapped into the DVMA space where Slave accesses are expected.

NOTE: Enabling Slave accesses from VME gives any VMEbus Master access to the entire DVMA virtual address range. It is possible that erroneous accesses to this space will affect the operation of the SCSI and/or ethernet controllers since their buffers are kept within the DVMA space.

7.8.2.1 Slave Map Register Initialization

The register is reset to all zeroes on resets. Thus, the mapping defaults to 0-1MB and block mode transfers are disabled.

Tables 7.8 and 7.9 provide more information about the Slave Map Register and bit assignments. Table 7.10 defines the bit configurations.

Table: 7.8 Slave Map Register Address

Type	Device Address	Device	Physical Space
1	0xEFE0001C	Slave Map Register	1 byte

Table: 7.9 Slave Map Register Address Bits

31	30	29	28	27	26	25	24
0	DVMA	0	0	VME Slave Address base			

NOTE: The VME-4 gate array uses bit 30 of the Slave Map Register for DVMA. Bit 30, when set, disables any access to the local DVMA by other VME Masters and disables mailbox interrupts.

7.8.3 CPU-2CE Slave Memory Example

This example requires a CPU-2CE as a slave board and another VMEbus master that can generate an extended address modifier code. To setup a slave window on the CPU-2CE type the following lines of code.

```
30 fff00000 100000 map-segments <cr>
0 obmem fff00000 100000 map-pages <cr>
0 map-slave <cr>
```

The description of the above two lines is as follows:

```
<Segment Number> <Virtual Address> <Length> <Map Segments>
<Physical Address> <obmem> <Virtual Address> <Length> <Map Pages>
```

Once these lines are typed in, it is possible to use the slave memory from a master at location 0 using an extended address modifier code.

To verify that the CPU-2CE received the correct data type the following on the slave.

```
fff00000 100 dump <cr>
```

Table: 7.10 Slave Master Register Bit Definitions

Bit	Description	VMEbus Address	Range Start	Range End
31	This bit is not used and must be 0.			
30	This bit when set disables VME slave accesses			
29:28	This bit is not used and is read back as 0.			
27:24	0x0 Maps System DVMA Space to:	0x00000000-0x000FFFFFF	0 MB	1 MB
27:24	0x1 Maps System DVMA Space to:	0x00100000-0x001FFFFFF	1 MB	2 MB
27:24	0x2 Maps System DVMA Space to:	0x00200000-0x002FFFFFF	2 MB	3 MB
27:24	0x3 Maps System DVMA Space to:	0x00300000-0x003FFFFFF	3 MB	4 MB
27:24	0x4 Maps System DVMA Space to:	0x00400000-0x004FFFFFF	4 MB	5 MB
27:24	0x5 Maps System DVMA Space to:	0x00500000-0x005FFFFFF	5 MB	6 MB
27:24	0x6 Maps System DVMA Space to:	0x00600000-0x006FFFFFF	6 MB	7 MB
27:24	0x7 Maps System DVMA Space to:	0x00700000-0x007FFFFFF	7 MB	8 MB
27:24	0x8 Maps System DVMA Space to:	0x00800000-0x008FFFFFF	8 MB	9 MB
27:24	0x9 Maps System DVMA Space to:	0x00900000-0x009FFFFFF	9 MB	10 MB
27:24	0xA Maps System DVMA Space to:	0x00A00000-0x00AFFFFFF	10 MB	11 MB
27:24	0xB Maps System DVMA Space to:	0x00B00000-0x00BFFFFFF	11 MB	12 MB
27:24	0xC Maps System DVMA Space to:	0x00C00000-0x00CFFFFFF	12 MB	13 MB
27:24	0xD Maps System DVMA Space to:	0x00D00000-0x00DFFFFFF	13 MB	14 MB
27:24	0xE Maps System DVMA Space to:	0x00E00000-0x00EFFFFFF	14 MB	15 MB
27:24	0xF Maps System DVMA Space to:	0x00F00000-0x00FFFFFF	15 MB	16 MB

7.9 Mail Box

A Mail Box interrupt function is provided to allow other devices to interrupt the SPARC CPU-2CE. This mail box detects accesses to the A16 address space to a location programmed in the mail box register.

No real memory is provided at this location, but the mail box responds with a VME DTACK, acknowledging the access and generate an on card interrupt Level 13 if the Enable bit is set. VME Address bits A(15:14, 3:1) are in the Mail Box Register. This corresponds to programming the location

to one of the eight 16-bit words in one of the four 16K blocks of the address range. Any VME A16 address space, 8-bit or 16-bit read or write to the location programmed into the Mail Box Register generates an on-card interrupt on level 13 if enabled. 32-bit wide accesses are not allowed and will not be acknowledged, resulting in a Bus Error Time out to the accessing VME Master.

Since the address mapping of the location monitor compares only the address bits A15, A14 and A3, A2, A1 (Bits A13 up to A4 are NOT compared at all), the location monitor address is mirrored 1023 times in the selected 16K range. The user is advised to use addresses with address bits A13 to A4 set to zero.

7.9.1 Mail Box Register Base Location

This register is used to program the VME address to be monitored and, if enabled, mailbox interrupts the IU via an on-card interrupt. Only 8-bit accesses to the register should be used, 16- or 32-bit accesses are not acknowledged, resulting in a bus error time out. For mail box cycles, the VME bus is only monitored for A16, D8 or D16 accesses.

7.9.1.1 Mail Box Register Initialization

All bits are initialized to zeroes.

Tables 7.11, 7.12, and 7.13 provide more information about the meaning of the Mail Box Register and its bits.

Table: 7.11 Mail Box Register Address

Type	Device Address Space	Device	Physical Space
1	0xEFEE0010	Mail Box Register	1 byte

Table: 7.12 Mail Box Register Address Bits

31	30	29	28	27	26	25	24
I-fig	En I	0	Comp Address (15:14)	Comp Address (3:1)			

7.9.1.2 Mail Box Register Interrupt Level

SPARC CPU-2CEs have programmable interrupt levels of 2, 3, 5, 8, 9, 11, and 13.

Table: 7.13 Mail Box Register Bit Definitions

Bit	Description
31	This bit set indicates that a mailbox interrupt is pending. A read of this register resets the interrupt on the trailing edge of the read pulse. A write to this bit is ignored.
30	This bit set enables the mail box interrupt.
29	This bit is ignored, and is read back as 0.
28	Compared with VME Address Bit 15.
27	Compared with VME Address Bit 14.
26	Compared with VME Address Bit 3.
25	Compared with VME Address Bit 2.
24	Compared with VME Address Bit 1.

7.10 Mailbox Interrupt Level - Rerun Length Register

This register sets the interrupt level that a mailbox interrupts on. A default Interrupt level uses pin MB_IRQ (level 13 on SBus), but any of the VME interrupt lines, pins B_IRQ n , could be used instead by setting the appropriate level. Only 8-bit accesses to the register should be used. Accesses of 16 or 32 bits are not acknowledged, resulting in a bus error time out.

By default, the Mailbox Interrupt level is set to level 13. On SPARC CPU-2CEs the Mailbox Interrupt register (bits 29:31) is used to set the interrupt level associated to the Mailbox interrupt. Then the user can program which Processor Interrupt will be generated when a Mailbox Interrupt is initiated.

31	30	29	28	27	26	25	24
MB 2	MB 1	MB 0	Rer 4	Rer 3	Rer 2	Rer 1	Rer 0

Long VME Master cycles (more than 16 clocks) are normally interrupted and frozen. They will be run at a later time, letting other SBus masters access the bus. By increasing the value of the rerun counter, longer cycles are possible, increasing the chance of getting a VME slave Acknowledgement avoiding the delay of a rerun cycle.

On SPARC CPU-2CEs the VME gate array can be tuned to increase the VME system throughput using the Mailbox Interrupt level Rerun Register bits [28:24]. By increasing the rerun interval, you can reduce the number of reruns the IU has to perform before an acknowledge is generated by a VME slave, increasing VME system throughput.

By default, the rerun interval is set to 16 clocks. For example, it takes 18 clocks to get an acknowledge from a slave. If the rerun register is set to 3 (rerun interval is 19 clocks), the IU can finish a VME transfer with just one cycle. However, if the default setting is used (16 clocks), it may take at least 20 to 21 clock cycles (requiring two rerun cycles) to finish the transaction.

Initialization: All bits are initialized to zero.

"Bits	Note"
Bits 31:29 = 0	Mail box interrupt on pin MB_IRQ_ or _int13
Bits 31:29 = 1	Mail box interrupt on pin B_IRQ1_ or _int2
Bits 31:29 = 2	Mail box interrupt on pin B_IRQ2_ or _int3
Bits 31:29 = 3	Mail box interrupt on pin B_IRQ3_ or _int5
Bits 31:29 = 4	Mail box interrupt on pin B_IRQ4_ or _int8
Bits 31:29 = 5	Mail box interrupt on pin B_IRQ5_ or _int9
Bits 31:29 = 6	Mail box interrupt on pin B_IRQ6_ or _int11
Bits 31:29 = 7	Mail box interrupt on pin B_IRQ7_ or _int13
Bits 28:24 = 0	Default rerun, cycle 16 states
Bits 28:24 = 1	cycle 17 states
Bits 28:24 = 2	cycle 18 states
Bits 28:24 = 3	cycle 19 states
Bits 28:24 = 4	cycle 20 states
	~
	~
	~
Bits 28:24 = 31	cycle 47 states

7.11 Bus Locker

A bus lock function provides a way to enable the CPU to do an atomic Read-Modify-Write (RMW) to its on-board memory without being interrupted by an incoming RMW from another VME Master. This function is only used when the card is running together with one or more external CPU cards (multiprocessing). In these cases, message passing is done through shared memory locations that need to be accessed in a defined way.

The bus locker is used to lock the VMEbus before a VME RMW cycle is executed to prevent a deadlock situation between two VME Masters. A deadlock situation can occur when two VME Masters want to perform RMW cycles at the same time.

7.11.1 VME Bus Locker Register

In a non-multiprocessing application, this register should be left alone. When used, bit 24 of the register should ALWAYS be SET for each update, to guarantee a consistent bus arbitration behavior.

7.11.2 Initialization

All bits are initialized to 0's. Thus, the VME bus locker is inactive and does not affect the card's bus arbitration.

Table: 7.14 Bus Locker Register Bit Definitions

Bit	Description
31	This bit indicates the VMEbus is owned. No external VME card can access memory. Bit is read only, a write to this bit is ignored.
30	This bit is ignored, and is read back as 0.
29	This bit is ignored, and is read back as 0.
28	This bit is ignored, and is read back as 0.
27	This bit is ignored, and is read back as 0.
26	This bit is ignored, and is read back as 0.
25	This bit initiates a VMEbus request. Once owned, it is held.
24	This bit enables the Bus request function. It should only be set for multiprocessing applications. If set, it should never be changed (unless system is restarted).

In multiprocessing applications, update the register at the system initialization time by writing 0x01 to the register. The procedure for accessing an on-card memory location is:

1. Disable all interrupts. (VME IACK cycles need to acquire control of VME.)
2. Request VMEbus locking by writing 0x3 to the register.
3. Read register and check bit 31 to until VMEbus is owned and locked.
4. Do Read-Modify-Write (RMW) (atomic Load-Store) to on-card memory.
5. Unlock VME by writing 0x1 to register.
6. Enable all interrupts.

NOTE: If the VME is not unlocked in time, other accesses might time out causing a system crash. Always unlock VME as soon as possible after the RMW is completed!

NOTE: VME interrupts must be disabled during the bus locking procedure to prevent potential deadlock, since VME IACK cycles requires access to VME. Interrupts could otherwise cause a deadlock situation if it happens while the bus is locked.

7.12 Interrupt Handler

The interrupt handler can selectively support all interrupt levels. These are enabled through the VME Interrupt/Bus Arbiter register. Note that this register is a replacement for a jumper and should normally be fixed depending on the VME Interrupt handler configuration. Run-time enabling of the interrupts are normally done with the Interrupt Enable register, see the System Specification.

7.12.1 Interrupt Enable/Bus Arbiter Mode Register

This register enables VME interrupts to be fed to the Interrupt logic in the S-4 MMU chip. Only 8-bit accesses should be used, 16- or 32-bit accesses are not acknowledged, resulting in a bus error time out. The Interrupt Enable Register can selectively enable all VME interrupt levels.

7.12.1.1 Interrupt Enable Register Initialization

At initialization of the Interrupt Enable register, bits 31:25 are set and bit 24 is reset, enabling VMEbus interrupts and setting arbiter mode to single level arbitration (SGL). Tables 7-15, 7-16, and 7-17 provide more detailed information about the Interrupt Enable register.

NOTE: This register replaces on-card jumpers. Changes to this register should only be done at system initialization time, and should not be changed at a later time unless the system is restarted.

Table: 7.15 Interrupt Enable Register Address

Type	Device Address	Device	Physical Space
1	0xEFE00014	VME Interrupt Enable Register	1 byte

Table: 7.16 Interrupt Enable Register Bit Definitions

Bit	Description
31	This bit enables VME Interrupt 7/Sun-4 level 13 to Internal Logic.
30	This bit enables VME Interrupt 6/Sun-4 level 11 to Internal Logic.
29	This bit enables VME Interrupt 5/Sun-4 level 9 to Internal Logic.
28	This bit enables VME Interrupt 4/Sun-4 level 8 to Internal Logic.
27	This bit enables VME Interrupt 3/Sun-4 level 5 to Internal Logic.
26	This bit enables VME Interrupt 2/Sun-4 level 3 to Internal Logic.
25	This bit enables VME Interrupt 1/Sun-4 level 2 to Internal Logic.
24	This bit enables FAIR Arbitration.

NOTE: If VME interrupt level 7 or 6 is needed with VME there is a conflict of the floppy and audio ports interrupts. Either the audio or floppy driver must be commented out of the `GENERIC.VME` and `GENERIC_SMALL.VME` to disable one. Remember to copy the files before modifying them. Search the `GENERIC.VME` and the `GENERIC_SMALL.VME` files in the `.conf` directory for `audioamd`. Use the `"#"` pound symbol at the head of the line to comment out either the audio or `fd` line. Then rebuild the kernel. See the `readme` file in the `.conf` directory for further details.

NOTE: Enabling the bus arbiter function is done via a jumper (Slot 1 jumper).

7.13 Bus Requester

The Bus requester is an ROR requester. It releases the bus when other VME Masters/Requesters requests the bus. The SPARC CPU-2CE can only request on Bus Request level 3.

7.13.1 Bus Arbiter

A Single Level Arbiter (level 3) and a FAIR Arbiter are provided. Note that the arbiter should be enabled only if the card resides in slot 1. (Slot1 jumper.)

In arbitration mode, locked arbitration is detected and released by the arbiter after 3.3 ms. The Bus arbiter then drives `BBSY` active for the minimum period required by the VMEbus spec. This is to prevent requesters that in the mean time have decided to withdraw their requests to be able to do so. Note that a withdrawal of a VME Request normally is in violation of the VME Specification. (Rule 3.11).

7.14 Bus Time Out Period

There are two time out parameters in a VME cycle in addition to the On Board Bus arbitration timer: Rerun Time Out and Abort.

7.14.1 Rerun Time Out

If the time from the onboard Master cycle starts until the VME slave responds with a `DTACK` exceeds 1.6 us, the cycle is rerun by the IU so high-priority devices can get on the Sbus. In the mean time, the Master cycle and all output signals is frozen, ignoring any `DTACK` coming in. When the cycle is rerun, it is reconnected again.

7.14.2 Abort

With the onboard master setup in slot 1, two stages of the access exists. If the bus is not owned, it has to be acquired. Then the actual VMEbus cycle can start. A onboard or offboard Master cycle times out after 300 us. That means, the response time from another VME slave including the time for acquiring the bus must be less than 300 us.

If the VMEbus is not owned when an abort occurs, the bus requester keeps the VMEbus Request asserted even though the CPU cycles has been aborted. It then asserts a dummy Bus Busy (`BBSY`) when the Bus Grant is received. This is to comply with VME rule 3.11.

If the bus is owned when the abort takes place, a Bus Error is generated. This Bus Error would normally be generated by an accessed VME slave. This is to comply with VME rule 2.48.

7.15 VMEbus Watchdog Timer

The SPARC CPU-2CE has a VMEbus watchdog timer to monitor the operation of the VMEbus. This function prevents the VMEbus from being locked when a Master accesses a nonexisting VME slave and when the master doesn't have its own bus time-out circuit. A VMEbus error occurs if a DTACK is not generated within 300 microseconds (fixed) after a Data Strobe is asserted on the VMEbus. This is only active if Slot 1 function is enabled.

7.16 System Reset and the Reset Switch

On power up, the CPU card generates SYSRESET for a minimum of 200 ms, if the Slot1 jumper indicates the card resides in slot 1. Incoming VMEbus resets always generate an on-card reset.

7.16.1 Sources for a System Reset

The sources of a system reset are listed below:

- Power-on Reset.
- Watchdog Reset.
- Software Reset.

NOTE: When the SPARC CPU-2CE is reset, the "daisy-chained" VME signals are not passed on for the duration of the reset cycle, regardless of the SPARC CPU -2CE's configuration (Slot 1 or non-Slot1 device).

"The Power-On Reset"

On power-up, the SPARC CPU-2CE Board generates a sysreset on the VMEbus for a minimum of 200 ms if it is configured as the slot 1 (VMEbus system controller) CPU. All SPARC CPU-2CEs on the VMEbus are reset in a system reset.

"The Reset Switch"

The Reset is located on the front panel of the card, labeled Reset. The switch rests in a null position. When it is activated, a reset procedure is activated that is performed by the VME Gate Array. If the board is configured as the slot 1 CPU, a VMEbus reset will be generated to reset all other non-slot 1 SPARC CPU-2CEs. For additional information, see the chapter VMEbus Interface in this manual.

"The Watchdog Reset"

The watchdog error is the result of a double bus error as detected by the SBus Controller on a SPARC CPU-2CE. If this occurs, an on-board reset is generated, and if the SPARC CPU-2CE is configured as the slot 1 controller, a VMEbus reset will be generated to reset all other non-slot 1 devices on the VMEbus.

"The Software Reset"

As the name implies, this reset is generated by a SPARC CPU-2CE through a software trap which generates an on-board reset. If the board is configured as the slot 1 CPU, a VMEbus sysrest signal will be generated to reset all other non-slot 1 SPARC CPU-2CEs.

7.17 Jumper

A jumper is used to set the VME Interface for slot 1. The default sets the board as a VME slot 1 card. If jumpered, the SPARC CPU-2CE operates as the VME system arbiter and slot 1 card.

Table 7.17 lists the function of the various jumper settings.

Table: 7.17 Slot 1 Jumper & Functions

Function	Slot 1 Position	Not in Slot 1
VME System Clock	Enabled.	Tri-state.
VME IACK Daisy-chain Driver	Enabled.	IACKIN drives IACKOUT.
SYSRESET	Generates VME SYSRESET.	Does not generate VME SYSRESET.

7.18 Programmable Register Settings

Table: 7.18 Programmable Register Settings

Register	Setting Definition
VME Interrupt Handler Register	Defines which interrupt levels to handle. Defines what type of VME Bus arbiter (only if the card resides in slot 1).
A32 Map Register	Defines where to map the CPU Master Address Space of 512 MB.

7.19 VME Interrupt Monitor Register

SPARC CPU-2CE features a VMEbus Interrupt Monitor Register which provides a quick way to determine the source of the interrupt on the IU. The Monitor register is a direct status of the VME interrupt lines. Only byte reads to this register are permitted. The VMEbus Interrupt status is contained in bits 31:25. Other operations to this register result in a bus error time-out.

Tables 7-19 and 7-20 list more detailed information about the Interrupt Monitor register.

Table: 7.19 VME Interrupt Monitor Register Address

Type	Device Address	Device	Physical Space
1	0xEFE00004	VME Interrupt Monitor	1 byte

Initialization: Bit 24 is reset (zero). Bits 31:25 are read-only.

Table: 7.20 Interrupt Monitor Register Address Definitions

31	30	29	28	27	26	25	24
IRQ 7	IRQ 6	IRQ 5	IRQ 4	IRQ 3	IRQ 2	IRQ 1	FAIR

NOTE: Bit 24 set to a one sets the requestor to *Fair Mode*. A bit set among bits 31:25 indicates a pending interrupt.

7.19.1 Fair Mode Requester

The SPARC CPU-2CE includes an optional fair mode requester bit (Bit 24 of the Interrupt Monitor register) to allow Bus Fairness operations when the CPU is used in a multiprocessor environment. This allows VMEbus masters in the system to have equal access to the VMEbus.

With the Fair bit set, the VME interface controller asserts a VME request only when no other bus request is pending. Resetting bit 24 of the Monitor register disables this feature.

7.20 VME IACK Cycles

When the SPARC CPU-2CE is set up to respond to vectored VME interrupt requests, it drives the IACK line to acquire the interrupt vector from interrupting devices through software routines. A device space operation is used by the Interrupt handler to generate an Interrupt Acknowledge cycle. The operation is done only by doing a Device Space byte read at location 0xEFE0000X with Address bits 3 to 1 set to the VMEbus interrupt level being acknowledged. A 16- or 32-bit access is not acknowledged and results in a bus error time-out. By doing this, an acknowledge process is started by first acquiring the bus, then by driving the IACK signal at the same time as with other control lines such as AS*, DS*, A1-3, etc.

NOTE: An access to these seven addresses are not to an on chip register, but instead to execute a VME Interrupt Acknowledge cycle, acquiring the Interrupt Vector from the interrupting device.

Tables 7-21, 7-22, and 7-23 provide more detailed information about the VME mail box register, IACK cycles registers, and responses.

Table: 7.21 Mail Box Register Address

Type	Device Address	Device	Physical Space
1	0xEFE0000	VME Int. Vector	1 A[3:1]=VME Int. level. A0=1, Read, only 8-bit access should be used.

Table: 7.22 VME IACK Cycles Register Address Bits

31	30	29	28	27	26	25	24
8-bit Interrupt Vector from interrupting VME Device Level A(3:1)							

This translates into the following interrupt responses:

Table: 7.23 VME IACK Cycles Interrupt Responses

Acknowledging VME Interrupt	Device Address
1	0xEFE00003
2	0xEFE00005
3	0xEFE00007
4	0xEFE00009
5	0xEFE0000B
6	0xEFE0000D
7	0xEFE0000F

7.20.1 Daisy Chain IACK Driver

Since the SPARC CPU-2CE does not have the capability to interrupt the VMEbus through the VME interrupt request lines, it does not provide interrupt vectors at all. Thus, it drives the daisy chain IACKOUT when an IACKIN is active. When set up as a slot 1 card, it drives the IACKOUT as soon as the non-daisy chain IACK signal is active.

7.20.2 Master Cycles

Ideal VME slave accessed (30 ns DS to DTACK response). Bus already owned. No lingering DTACK or AS. A Master cycle is 9 cycles, corresponding to a theoretical 9 MB/s bandwidth for back-to-back cycles.

7.20.3 Slave Cycles

Ideal VME master (0 ns DTACK to DS response). Fast Sbus grant (two clock periods).

Table 7.24 provides more information about the duration of slave cycles and the theoretical bandwidth.

Table: 7.24 Slave Cycles Duration and Theoretical Bandwidth

Cycle	Duration (ns)	Theoretical Bandwidth
Single Read	560-630 ns	6.9 to 7.9 MB
Single Write	510-580 ns	6.3 to 7.2 MB

7.21 Bus Arbitration

Table 7.25 lists the characteristics for single-level and Fair arbitration.

Table: 7.25 Bus Arbitration

Arbiter	Conditions
Single Level Arbiter	No VME master owns the bus, Bus Locker disabled. VME Bus Request to VME Bus Grant: 2 Cycles.
	No VME master owns the bus, Bus Locker enabled. VME Bus Request to VME Bus Grant: 3 Cycles.
Round-Robin Arbiter	No VME master owns the bus, Bus Locker disabled. VME Bus Request to VME Bus Grant: 2-5 Cycles.
	No VME master owns the bus, Bus Locker enabled. VME Bus Request to VME Bus Grant: 3-6 Cycles.

7.22 Interrupts

Table 7-26 defines the interrupt sources and their levels:

Table: 7.26 VME Interrupt Levels and Sources

Level	Source	Additional Source
15	Abort	
15	SYSFAIL	
15	Memory: Parity Error	ECC Uncorrectable Error
14		Counter/Timer 1
13	VME Level 7 VME Mailbox interrupt	Audio
12		Serial Ports
11	VME Level 6	Floppy Disk
10		Counter/Timer 0
9	VME Level 5	SBUS 7
8	VME Level 4	SBUS 6 (Video)
7		SBUS 5
6		SW IRQ6
5	VME Level 3	Ethernet, SBUS 4
4		SCSI SW IRQ4
3	VME Level 2	SBUS 3
2	VME Level 1	SBUS 2
1		SW IRQ1 SBUS 1

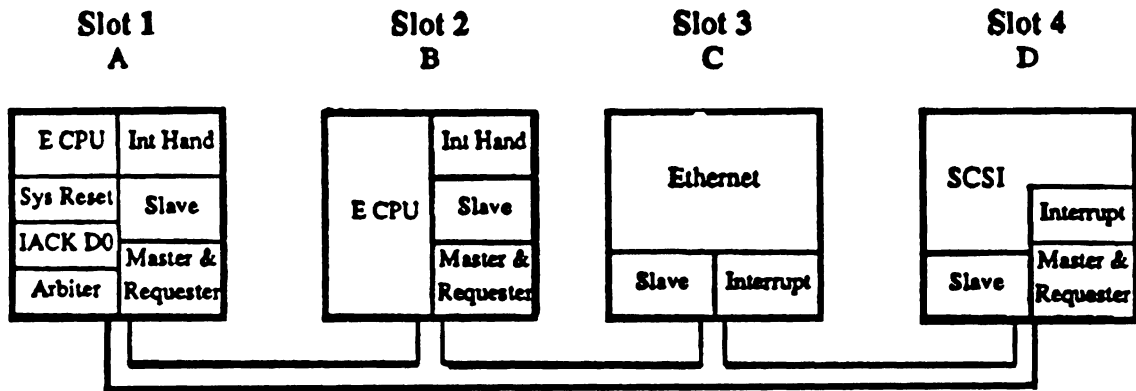
All type 1 devices defined in the architecture, including the serial ports, use autovectorred interrupts.

The interrupt register in device space enables all interrupts, causes software interrupts, and enables specified interrupts. It is described in the chapter *Device Space*. The memory error registers in system space, provide information about any memory errors that occur. They are described in the chapter *System Space*.

7.23 Example of a VME System

Figure 7-1 is a diagram of a sample VME system with two SPARC CPU-2CE cards, a third-party Ethernet card, and a third-party SCSI card. The text below describes how this system configuration should be defined and implemented. This example is generic and is provided only as a guide for understanding the SPARC CPU-2CE VMEbus.

Figure: 7.1 Sample VME System Diagram



Board A in slot 1 is a SPARC CPU-2CE with the slot 1 functions enabled (Arbiter, IACK Daisy Chain driver, SYS Clock Driver). It is an Interrupt Handler for level 1-3. Board B is also a SPARC CPU-2CE with the slot 1 functions disabled. It is an Interrupt Handler for level 4-7. Board C is an Ethernet card with a Slave and an Interrupter. Board D is a SCSI card with DMA capabilities. It has a Master, a Slave and an Interrupter.

Board A needs to start a disk transfer by writing to command registers in card D. But first he has to acquire the bus. His Master orders the on card Requester to get the bus. The Requester asserts [BR]. If no one owns the bus, indicated by an inactive [BBSY], the arbiter grants the bus with [BG]. Board A's Requester receives this, assert [BBSY] and allow his Master to start a bus cycle.

If another card's Requester was requesting the bus at the same time with a [BR], card A would still get it since the bus grant from the arbiter in slot 1 is passed from card to card via a Daisy Chain [BGIN,BGOUT] and card A is the first card in the Chain.

Board A's Master asserts addresses, data and then strobe [AS] and [DS(1:0)].

Next, card D's Slave detects that the cycle is aimed at him and respond with a [DTACK] once the data has been captured (the write to the command register). The SCSI card starts the disk access and when he has filled his buffer with data, activate his Master. The Master requests the bus via it's on-card Requester.

The SPARC CPU-2CE requester is of Release On Request type, so he keeps the bus with [BBSY] until someone else asks for it. When Board D's Requester requests the bus, Board A's Requester drops [BBSY]. Now the Arbiter can grant the bus to card D with a [BG].

Board D acquires the bus asserting [BBSY] and the Master can start a DMA transfer to card A's on card memory via A's Slave. the Slave responds with a DTACK for every cycle. Once the whole transfer is complete, card D asserts interrupt 3 via his interrupter signaling he's finished.

Board A's interrupt handler is activated and does an Interrupt Acknowledge cycle. This cycle is almost identical to a Master read, but instead of reading data, the interrupt vector is read. [IACK] is asserted to signal an IACK cycles, and address bits 1-3 indicates what level the IACK cycle is responding to.

[IACK] from any interrupt handler is passed to slot1, where the back plane directly connects it to the start of the [IACKIN/IACKOUT] daisy-chain. This is done to enable interrupt handlers to be spread out on different cards if needed. The IACK Daisy Chain Driver in slot 1 drives the chain. Board B is not an interrupter and therefore always passes [IACKIN] to [IACKOUT].

Board C passes the incoming [IACKIN] to its [IACKOUT], unless he is interrupting on the same level at the same time. If so, he does not pass the [IACKIN] signal, but instead respond to the IACK cycle with a DTACK and his vector, different from card D's. In the mean time, card D would not see any [IACKIN] and therefore just keep his interrupt asserted until card C passes it on.

Board D's interrupter responds to an IACK cycle with the corresponding bits set to the level he is interrupting on. Board A jumps to the disk card interrupt routine, with the understanding that the disk transfer has finished and now is ready and stored in the DVMA buffer. The transfer is complete.

7.24 VMEbus Device Driver

7.24.1 VMEbus Device Driver System Calls

The VMEbus Device Driver supports the "usual" system calls: `open()`, `close()`, `read()`, `write()`, `ioctl()` and `mmap()`. Consult your SunOS documentation for detailed information regarding the general use of these functions.

When making calls to these functions (in reference to the memory/devices managed by the VMEbus Device Driver) the following points are of interest:

- `open()` verifies that the minor device number, of the file you are opening, is one of the following:

VME16D16	VME16D32
VME24D16	VME24D32
VME32D16	VME32D32

These constants are defined in the header file `vme.h`.

- `read()` and `write()` will not attempt to read past the end of the valid space defined by the minor number of the "file" you are attempting to read. In other words, attempts to `read()` or `write()` past the address `0xffff` when using a file handle opened in A16 space will not be attempted by the VMEbus Device Driver. Likewise, `read()` and `write()` will not read past the address `0xfffff` in A24 space.

- The following operations are defined for an `ioctl()` call:

<code>VME_SET_REG</code>	Write to a VME interface specific register.
<code>VME_GET_REG</code>	Read from a VME interface specific register.
<code>VME_RDSWITCH</code>	Read the current state of the HEX switch.
<code>VME_WRLEDS</code>	Write to the front-panel LED's.
<code>VME_ENA_DIS_VME32</code>	Enable full 32-bit address access or place the VME interface into CPU-1E compatibility mode.
<code>VME_GET_PHYS</code>	Translate a DVMA address into its corresponding VME physical address.

The labels described above are defined in `vme.h` supplied with the VMEbus Device Driver. A sample application, called `v_rw.c`, that uses each of these `ioctl()` calls is also supplied with the VMEbus Device Driver. Please refer to these files for detailed information.

- The S4-VME interface device has limitations when used to access the A32 space which must be taken into account when assigning addresses to VME peripheral devices on the VMEbus. These limitations are associated with hardware and cannot be fixed by software (either the VMEbus Device Driver or the application software).

The VMEbus interface chip on the CPU-2CE directly decodes only 29 address bits resulting in what can be described as decoding a series of eight 512 megabyte "windows" (there is additional hardware to assist in distinguishing which "window" is being used). The VMEbus interface chip always interprets the upper 16 megabytes of each of these "windows" as A24 and A16 space. Thus, one cannot map the following VMEbus addresses into A32 space:

Table 7.27 A16 - Only Access

Start Address	End Address
0x1FFF.0000	0x1FFF.FFFF
0x3FFF.0000	0x3FFF.FFFF
0x5FFF.0000	0x5FFF.FFFF
0x7FFF.0000	0x7FFF.FFFF
0x9FFF.0000	0x9FFF.FFFF
0xBFFF.0000	0xBFFF.FFFF
0xDFFF.0000	0xDFFF.FFFF
0xFFFF.0000	0xFFFF.FFFF

Table 7.28 A24 - Access

Start Address	End Address
0x1F00.0000	0x1FFE.FFFF
0x3F00.0000	0x3FFE.FFFF
0x5F00.0000	0x5FFE.FFFF
0x7F00.0000	0x7FFE.FFFF
0x9F00.0000	0x9FFE.FFFF
0xBF00.0000	0xBFFE.FFFF
0xDF00.0000	0xDFFE.FFFF
0xFF00.0000	0xFFFE.FFFF

7.24.2 VMEbus Address Modifiers

Program/Data/Block Transfer Address Modifiers

The CPU-2CE will generate only the Data Address Modifier Code. The Program and Block Transfer Address Modifier Codes are not supported.

Supervisory/Non-Privileged Address Modifiers

The CPU-2CE can generate either the Supervisory or the Non-Privileged Address Modifier Codes. This is controlled by the Integer Unit's (IU) Processor Stat Register (PSR).

Typically a user's application will operate at a "non-privileged" level resulting in the assertion of the User Address Modifier Code when the user's application access VMEbus addresses that have been allocated by the `mmap()` system call. If the user's application is a driver that operates at the same privilege level as the kernel the Supervisor Address Modifier Code will be asserted when that driver accesses VMEbus addresses.

Accesses to the VMEbus made within the kernel operate at a "privileged" level and will result in assertion of the Supervisor Address Modifier Code. This is the case when one uses the `read()` and `write()` system calls to access a VMEbus device. If the user uses the `mmap` call to create a pointer the VME AM code issued will reflect the current privilege level of the calling program.

Extended/Standard/Short Address Modifiers

The CPU-2CE may generate the Extended, Standard or Short Address Modifier Codes with the following restrictions:

Table: 7.29 Address Modifier Codes

Address Modifier	Value	Description
Extended (A32)	0x09, 0x0D	No Unaligned transfers are allowed
Standard (A24)	0x39, 0x3D	No Unaligned transfers are allowed
Short (A16)	0x29, 0x2D	No Unaligned transfers are allowed

7.24.3 VMEbus Device Driver Limitations

Floppy and Audio Interrupt Conflicts

VMEbus IRQ6 and the standard SunOS floppy disk device driver cannot be used simultaneously. Likewise, VMEbus IRQ7 and the standard SunOS audio device driver cannot be used simultaneously.

The floppy and audio device drivers will not share their respective interrupts with other device drivers because they use Programmed I/O to complete their tasks, rather than DMA. Programmed I/O is relatively slow, therefore, these device drivers reserve their corresponding interrupt levels for exclusive use. Thus, if either of these devices is configured as part of the kernel their respective VMEbus IRQ cannot be used.

Potential for SunOS Patches to Conflict

Because several of the files supplied as a part of the VMEbus Device Driver are modifications of standard SunOS files there is a potential for SunOS patches to conflict with the VMEbus Device Driver. In general, it is best to consult FORCE COMPUTERS' Technical Support to verify if a patch can be made to the SunOS kernel without interfering with the VMEbus Device Driver.

If FORCE COMPUTERS has incorporated a specific SunOS patch there will be additional files supplied with the VMEbus Device Driver that should be used instead of the SunOS patch files. Using the FORCE COMPUTERS supplied files will insure that the patch is installed in a way that is compatible with the VMEbus Device Driver. The standard files supplied with the VMEbus Device Driver are based on the original, unpatched SunOS sources and do not incorporate any SunOS patches that have occurred since the specific release on which the VMEbus Device Driver is based.

GENERIC.VME versus GENERIC_SMALL.VME

Two kernel configurations are supplied with the VMEbus Device Driver. The **GENERIC.VME** configuration assumes all devices will be configured (including the floppy and audio device drivers) except for the **ONC/VME Device Driver** (see Section 8).

When installing or configuring the VMEbus device driver using the **GENERIC.VME** configuration file you must **ALWAYS** ensure the file `/etc/hostname.vm0` is created and contains the name of the VME node. This is similar to adding a second ethernet card.

7.25 VME Programming Examples

Two code examples are presented below, one written in **FORTH**, and the other written in **C**. These examples provide an indication to you on how to program for the VMEbus.

Additional reference material required for a complete definition of the **FORTH** and **C** words used in the following code examples: *SBUS Developer's Kit*, Sun Part No. 825-1219-xx, (Sun SBUS Information).

7.25.1 FORTH Programming Examples

Map the VMEbus to a Virtual Address

The first **FORTH** example illustrates how to gain access to a known range of physical addresses on the VMEbus by mapping the desired VME address-range to a virtual address-range allocated from the Virtual Address Pool.

The routine `memmap` maps a range of physical addresses within the specified "space" to a range of virtual addresses allocated from the Virtual Pool and returns the base virtual address which was allocated.

The "space" parameter can refer to On-Board Memory, On-Board I/O, or any combination of VME-Bus Data-widths and Address-widths from among the following: D16 or D32 , and A16, A24, or A32

```

\ Initialization and constants.
hex          \ Set the numeric base for subsequent interpretation

100.0000    constant  my-vme-phys-addr  \ Base-address of the desired VME range
2.0000      constant  my-size           \ Size of the desired VME address-range

0 value my-virt          \ Place-holder for the virtual address

\ Now map the address-range:

my-vme-phys-addr        \ First parameter for memmap is the base-address
                        \ (physical address) of the desired address-range

vmed32a32              \ Second parameter for memmap is the "space". For
                        \ this example, we are choosing 32-bit-wide data
                        \ and 32-bit-wide addressing. The other valid
                        \ options for this parameter, in this example,
                        \ could be:          vmed32a24  vmed32a16
                        \                   vmed16a32  vmed16a24  vmed16a16

my-size                \ Third parameter for memmap is the size of the
                        \ desired address-range.

memmap                \ The allocated virtual address is now on the stack.

is my-virt            \ Save it in the place-holder.

\ You can now read from and write to any address in your specified
\ range on the VMEbus, simply by reading from or writing to the
\ corresponding virtual address.

\ When you are done with the range, unmap the VMEbus address-range
\ and return the virtual address-range to the Allocation Pool:

my-virt my-size free-virtual

```

Map Local Memory to the VMEbus

The next FORTH example illustrates how to map a range of your local memory to the VME slave port, allowing other VMEbus masters to obtain read/write access to your local memory in a DMA mode. This operation is normally undertaken by a device-driver under the /VME device-node, and, in fact requires the services of certain words which are methods within that node, so this example will also illustrate various ways to arrange the required set-up for different situations.

```

\ Common part of the set-up: Setting this CPU2CE's slave-port. There is a
\ configuration-variable called vme-slavemap whose value is 0 to
\ (decimal) 15 (or 0f hex) which identifies the slave-port number of
\ the CPU2CE. If there is more than one CPU2CE, or other device capable
\ of acting as a VME-slave, on the VME-bus, then each one must have a
\ different and unique value for its slave-port number. If your board
\ already has its vme-slavemap configuration-variable set to the proper
\ value, then you will not need to repeat this sequence. We will assume
\ that we are setting this CPU2CE's slave-port number to 2.

setenv vme-slavemap 2          \ Set the configuration-variable
                              \ to the proper value.

vme-slavemap vme-slavemap!    \ Load the slave-port register with the
                              \ value in the configuration-variable.
                              \ Normally, this is done at startup time.

```

```

\ Set-up example 1: A device-driver under the /VME device-node, whose
\ base-address is at 0c0.0000 and which operates with 16-bit data-width
\ and 24-bit address-width
\
0 0 " vmed16a24,0c0.0000" " /vme" begin-package

\ The above begins the definition of a new device-driver package
\ under the /VME device-node. It would normally precede the
\ instructions which lead to the interpretation of the FCode
\ for the device-driver in question. It supplies the essential
\ information about the address and space at which the device
\ is configured. The interpretation of the device-driver FCode
\ would then be followed by the end-package command; however,
\ those considerations are beyond the scope of this example.

\ We will skip over such preliminaries as supplying the device's
\ name or specifying the structure of its control registers.

\ Since the present device is an offspring of the /VME node, it
\ does not automatically have access to the required /VME
\ methods. For convenience, we will define them here, as calls
\ to the methods of the same name in the parent's node -- i.e.,
\ in the /VME node:

: map-in ( phys space size -- virt )          " map-in"      $call-parent ;
: map-out ( virt size -- )                    " map-out"     $call-parent ;
: dma-alloc ( n -- vaddr )                     " dma-alloc"    $call-parent ;
: dma-free ( vaddr n -- )                      " dma-free"     $call-parent ;
: dma-map-in ( vaddr n cache? -- devaddr )    " dma-map-in"   $call-parent ;
: dma-map-out ( vaddr devaddr n -- )          " dma-map-out"  $call-parent ;

my-address          \ This will be the second value given in the second
                    \ string parameter to begin-package prior to
                    \ probing; in this example 0c0.0000

constant dev-address \ Make it a constant, to be consistent with
                    \ the second set-up example

my-space            \ This will be the first value given in the second
                    \ string parameter to begin-package prior to
                    \ probing; in this example, vmed16a24

constant dev-space  \ Make it a constant, to be consistent with
                    \ the second set-up example

\ Set-up example 2: The operator is doing some on-the-fly experimentation
\ and merely wants to have the /VME device-node's methods available:

cd /vme            \ Make the /VME device-node the top of the vocabulary
                  \ context, and also the current vocabulary where
                  \ definitions will be placed. The latter is not
                  \ desirable, because words defined here will become
                  \ methods of the /VME device-node, and OpenBoot
                  \ does not permit you to forget a device's methods.
                  \ Since you are experimenting, you will likely want
                  \ to forget and re-define your definitions as your
                  \ experimentation progresses.

also forth definitions \ This will leave the /VME device-node in
                      \ your vocabulary search-order, but put
                      \ the forth vocabulary back as the top
                      \ of the context and as the current
                      \ vocabulary for definitions.

\ When you are finished, you can restore the normal search-
\ order with the command-sequence:
\ only forth also definitions

```

\ Again, we skip over preliminaries such as specifying the structure
 \ of the device's control registers.

0c0.0000 constant dev-address
 vmed16a24 constant dev-space

\ The body of the example follows:

0 value dev-reg-base \ Place-holder for the allocated virtual base-address
 \ of the device's control registers structure.
 0 value my-DMA-addr \ Place-holder for the virtual address we will use to
 \ access our slave port memory.
 0 value dev-DMA-addr \ Place-holder for the address that the device, acting
 \ as Bus-Master, will use to access our slave
 \ port memory in DMA mode.

\ Map-in the device's control registers

dev-address dev-space \ First two parameters for map-in
 reg-size \ Hypothetical constant specifying the size of
 \ the device's control registers structure.
 map-in \ The allocated virtual address is now on the stack.
 is dev-reg-base \ Save it in the place-holder.

\ Now we will allocate and map some memory for our slave port,
 \ "on the fly".

dev-DMA-size \ Hypothetical constant specifying the size of our slave
 \ port memory.
 dma-alloc \ Allocate the memory; virtual address is now on the stack.
 is my-DMA-addr \ Save it in the place-holder.

my-DMA-addr \ First parameter for dma-map-in is the virtual
 \ address which was allocated by dma-alloc
 dev-DMA-size \ Second parameter for dma-map-in is the size of
 \ the range to map to the VME slave port
 false \ The third parameter for dma-map-in specifies
 \ whether caching should be enabled, if available
 dma-map-in \ The address that the device, acting as Bus-Master,
 \ will use to access our slave port memory,
 \ is now on the stack
 is dev-DMA-addr \ Save it in the place-holder.

\ Load the address that the device, acting as Bus-Master, will use
 \ to access our slave port memory, into its appropriate control
 \ register.

dev-DMA-addr \ Address the device should use...
 dev-reg-base \ Base-address of the device's control registers
 >dev-addr \ Hypothetical word to adjust the base address to
 \ the offset for the Device-Address Register.
 ! \ Store the address in the Device-Address Register


```
\ The device can now read from and write to the VME-bus address held in
\ dev-DMA-addr (and copied to its Device-Address Register) to access
\ the CPU2CE's slave port memory. The CPU2CE can access the same memory
\ by using the virtual address held in my-DMA-addr. When you are done
\ with this device's access to the VME slave port, unmap things and
\ return the virtual address-range to the Allocation Pool:
```

```
my-DMA-addr dev-DMA-addr dev-DMA-size dma-map-out
```

```
my-DMA-addr dev-DMA-size dma-free
```

```
\ Map-out the device's control registers, now that you're done
```

```
dev-reg-base reg-size map-out
0 is dev-reg-base
```

Map an SBUS Address to a Virtual Address

The third FORTH example illustrates how to gain access to a known range of physical addresses within the SBUS (actually, an known offset from the start of the SBUS) by mapping the desired address-range to a virtual address-range allocated from the Virtual Address Pool.

```
0010.0000 constant my-SBUS-offset \ Base-address of the desired SBUS range
0000.4000 constant my-SBUS-size \ Size of the desired SBUS address-range
```

```
0 value my-virt \ Place-holder for the virtual address
```

```
my-SBUS-offset sbus my-SBUS-size mmap is my-virt
```

```
\ mmap can also be used with the following non-VME spaces:
\ obmem obio
```

```
\ You can now read from and write to any address in your specified
\ range on the SBUS, simply by reading from or writing to the
\ corresponding virtual address.
```

```
\ When you are done with the range, unmap the SBUS address-range
\ and return the virtual address-range to the Allocation Pool:
```

```
my-virt my-SBUS-size free-virtual
```

7.25.2 C Programming Example

The C example below shows how to open a VME space and obtain a virtual pointer so that the VME space can be read/written to.

In the Unix environment several options are available to the programmer. The VME-bus supports 16, 24, and 32 bit addressing, along with 8, 16, and 32 bit data transfers. In /dev several special files are provided that allow for the different configurations. For example, vme24d32 provides a 24 bit address with 32 bit data transfers. These special files can be opened and seeks performed. (See MEM in section 4S of the manual pages) However it is usually more efficient to use pointers. A pointer to an opened special VME file can be obtained using mmap(). (See MMAP in section 2 of the manual pages)

```
#include <stdio.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/mman.h>
```

```

#include <vme.h>                /* important VME driver definitions */

main(argn, argv)
int argn ;
char **argv ;
{

    int fd, i, len, off ;
    off_t addr ;
    caddr_t ptr ;
    unsigned short *p ;
    unsigned short pat ;

    if (argn < 3) {
        printf("usage: %s addr len [pat]\n", argv[0]) ;
        return ;
    }
    pat = 0xaaaa ;                /* assign a default pattern to be used */

    /*
    * Get "addr" from the argument list. Convert that address to a 32k aligned
    * base address plus an offset ("addr" and "off").
    */
    addr = (int)strtol(argv[1], (char **)NULL, 0) ; /* get addr from argv */
    off = (addr & 0x7fff) >> 1 ;                /* extract offset */
    addr &= 0xffff8000 ;                        /* align base address */

    len = (int)strtol(argv[2], (char **)NULL, 0) ; /* get length of trnsfr */

    if (argn == 4)                        /* get data pattern */
        pat = (short)strtol(argv[3], (char **)NULL, 0) ;

    printf("addr = 0x%x\n", addr) ;                /* display the values */
    printf("off = 0x%x\n", off*sizeof(pat)) ;    /* we are going to use */
    printf("len = %d\n", len) ;
    printf("pat = 0x%x\n", pat) ;

    /*
    * Now open VME space as address 24 data 16 (VME24D16).
    */
    if((fd = open("/dev/vme24d16", O_RDWR)) == -1) {
        printf("error opening vme bus\n") ;
        return ;
    }

    /*
    * Use mmap to obtain a virtual pointer to VME space. Then, convert the
    * pointer to an unsigned short for 16 bit reads and writes.
    */
    if((ptr = mmap(0, len, (PROT_READ|PROT_WRITE),
        MAP_SHARED, fd, addr)) == (caddr_t)-1) {
        printf("mmap failed\n") ;
        return ;
    }
    p = (unsigned short *)ptr ;

    /*
    * Write VME memory with the specified data pattern. Then, read the data
    * back and compare to the specified data pattern.
    */
    for (i=0 ; i<len/2 ; i++)                /* write "pat" to the VME space */
        *(p+i+off) = pat ;

    for (i=0 ; i<len/2 ; i++) {                /* read VME space, look for "pat" */
        if (*(p+i+off) != pat) {
            printf("fail location %x read %x exp %x\n", p+i+off,
                *(p+i+off), pat) ;
            break ;                            /* break out of read loop */
        }
    }
}

```


Section 8

ONC/VME

The VMEbus provides facilities for data sharing between multiple processors on the same backplane, but custom programming is usually required to make use for those facilities. None of the standard commands for inter-processor communication (e.g. *rlogin*) are of any help without a separate network connection.

ONC/VME corrects that problem by defining a way in which two or more machines on the same VME backplane can use the backplane as if it were an IP network interface. All of the standard TCP/IP networking facilities, including NFS, can then be used by any of the machines on that backplane without an Ethernet or other secondary interconnection between hosts. Any facilities specific to a particular sort of network (for instance, the UNIX *etherfind* utility) will not operate on an ONC/VME interface.

ONC/VME allows you to connect multiple processors residing in the same cardcage without having to use the usual network equipment such as thick-wire Ethernet or a multiplexing device.

ONC/VME uses the VME backplane as the network device. The processors share memory across the bus. A host leaves an internet protocol (IP) packet in a known location in a processor's memory, then signals the destination host that a packet is waiting.

The processors must be able to *export* a section of local memory to an arbitrary VME address that is megabyte-aligned. Efficiency of the protocol is maximized if the processors are able to send and receive VME mailbox interrupts.

8.1 Capabilities

ONC/VME supports most TCP/IP applications. These include:

- remote programs such as *rsh*, *rcp*, *rlogin*
- *telnet*
- *ftp*
- *spray*
- *nfs*

8.2 Using ONC/VME

Once the configuration has been completed and the ONC backplane driver (*vm0*) is active, you can use this device as you would an Ethernet interface. Section 8.3 describes the steps required to set up and boot one or more client SPARC CPU-2CEs over the VMEbus backplane. Section 8.4 describes the steps required for two or more SPARC CPU-2CEs to communicate over the VMEbus.

8.3 Booting Over the VMEbus Backplane

This section describes the steps required to set up and boot one or more client SPARC CPU-2CE's over the VMEbus backplane. It also describes the required configurations for hardware, software, and the PROM/NVRAM.

CAUTION If `yplib` is desired, add the server and client CPU's *hostname* to the Master YP Map or disable `yplib` by entering:

```
kill -9 <PID>
```

You can find out the <PID> by entering:

```
ps -axj | grep YP
```

In the latter case, `yplib` must be disabled because the server and the client CPUs are not known over the NIS Master database and when `add_client (8)` is run, the ethernet address of the client(s) will not be used correctly.

8.3.1 Installation Example

This installation example lists the variables that must be setup prior to *server and client* ONC/VME operation.

- Each SPARC CPU-2CE must be assigned an Internet address for *vm0* and a corresponding *hostname*. This is a different address than that of *ethernet le0*.
- Each SPARC CPU-2CE must be assigned a unique slave map address space in the range of 0-15 (*vme-slavemap*).

Table 8.1 lists the names, values, and variables for the server CPU. Table 8.2 lists the names, values, and variables for the client CPU.

Table: 8.1 Example Server Configuration

Hostname:	server-vm
Internet Address:	199.9.9.1
Server vme-slavemap:	0

Table: 8.2 Example Client Configuration

Hostname:	client-vm
Internet Address:	199.9.9.2
Client vme-slavemap:	1
Client Ethernet Address:	8:0:50:1:0:3

8.3.2 Hardware Configuration

Install two or more SPARC CPU-2CE cards into a VMEbus card cage. Make sure no CPU is plugged into a slot reserved for the P2 extended bus of another CPU (i.e. VSB).

8.3.3 Software Configuration

The SPARC CPU-2CE designated as the *server* must have server software installed when SunOS/Solaris software was loaded.

Power cycle the system and put each SPARC CPU-2CE at the `ok` prompt.

8.3.4 PROM/NVRAM Configuration

The CPU-2CE must contain version 2.3.6 or higher ROM to work properly with ONC\VME. To determine your ROM's version number do the following:

- enter the `forthmon` monitor
- type `.version` at the `ok` prompt

Several variables in the PROM/NVRAM must be set up in order to boot over the VME backplane. The (server CPU) must set *vme-slavemap*.

Each *client* CPU must set the following NVRAM variables:

- *boot-device*
- *vm-server-slavemap*
- *vm-server-addr*
- *vm-ip-addr*

The sections that follow list the PROM/NVRAM variables and show examples of how to set them. See section 2 for more detailed information about these parameters.

8.3.4.1 vme-slavemap

Each SPARC CPU-2CE must be assigned a unique *slave map address space* (*vme-slavemap*) in the range of 0-15.

Set the server's NVRAM *vme-slavemap* variable to zero, as shown below:

```
OK  setenv vme-slavemap 0
```

Set the client's NVRAM *vme-slavemap* variable to a value between 1 and 15. The example below shows how to set the example client's *vme-slavemap*:

```
OK  setenv vme-slavemap 1
```

8.3.4.2 boot-device

Set the client's NVRAM *boot-device* variable to *vmnet*.

```
OK  setenv boot-device vmnet
```

8.3.4.3 vm-server-slavemap

Set the client's NVRAM *vm-server-slavemap* variable to zero, as shown below:

```
OK  setenv vm-server-slavemap 0
```

8.3.4.4 vm-server-addr

Set the client's NVRAM *vm-server-addr* variable to the Internet address of the client's server.

The Internet address is made up of four hexadecimal numbers expressed in decimal form. For example, the Internet address 199.9.9.1 converts to the hexadecimal value *0xc7090901*.

```
OK  setenv vm-server-addr 0xc7090901
```

8.3.4.5 vm-ip-addr

Set the client's NVRAM *vm-ip-addr* variable to the Internet address of the client. This entry also requires a hexadecimal form.

```
OK  setenv vm-ip-addr 0xc7090902
```

8.3.5 Initializing the System

Reboot the system. The clients will be in a loop requesting a `rarp` address, is a normal condition prior to installation completion.

8.3.6 Files To Edit on the Server

When the server is booted and running `vmunix`, edit or create the following files. You must become `root` to make these changes. The following changes assume you have installed the VME/ONC driver onto your system according to the insulation instructions provided with the driver.

8.3.6.1 `/sys/sun4c/conf/GENERIC.VME`

Edit the `/sys/sun4c/conf/GENERIC.VME` file and uncomment the entry for the ONC/VME driver, as follows:

```
device          vm0
```

Uncommenting this line in the `GENERIC.VME` file causes the ONC/VME driver to be included in the kernel. The kernel must be reconfigured for this change to have effect

8.3.6.2 `/etc/hosts`

Edit the `/etc/hosts` file and add the Internet address. This should be the same address set into NVRAM `vm-server-addr` (0xc7090901 in our example below).

```
120.40.35.3      server loghost
199.9.9.1       server-vm      <-- add this line
199.9.9.2       client-vm     <-- add this line
```

Use the banner word at the prompt to determine your ethernet address.

8.3.6.3 `/etc/ethers`

Edit the `/etc/ethers` file and add an entry for the client's ethernet address (8:0:50:1:0:3 in our example below).

```
8:0:50:1:0:3    client-vm      <-- add this line
```


8.3.6.4 ONC Driver Interrupt Conflict with the Audio and Floppy Drivers

The ONC/VME driver uses the VME Mailbox interrupt as part of the communication protocol. The VME Mailbox interrupt level, by default, is set to level 13 and may conflict with the Audio driver.

The ONC/VME driver is not automatically configured in the `GENERIC.VME` or `GENERIC_SMALL.VME` files. If you wish to use the ONC/VME driver you must uncommment it from the `GENERIC.VME` or `GENERIC_SMALL.VME` files. Then you must to run the config utility using either `GENERIC.VME` or `GENERIC_SMALL.VME` to rebuild the kernel.

If the ONC/VME driver is configured to include the Floppy or the Audio device drivers, then the Open Boot `vme-rerun` variable needs to be set. The VME Mailbox interrupt must be set at an IPL, Interrupt Process Level, level other than level 11 or 13. Interrupt levels 11 and 13 are reserved exclusively for the Floppy and Audio drivers.

For example, to set the VME/ONC driver to use IPL level 9 type the following in Open Boot.

```
setenv vme-rerun 0xa0
```

8.3.7 Adding a Client

The next step requires you to run `add_client` and answer the questions. Refer to the Networking and Communications Administration section for more detailed information. Enter the following:

```
> /usr/etc/install/add_client -i
```

Once the clients have been added, edit the following files on the server.

8.3.7.1 /etc/hostname.vm0

Edit or create the file `/etc/hostname.vm0`.

```
server-vm          <-- add this line
```

8.3.7.2 /etc/bootparams

Edit the file `/etc/bootparams` and substitute the server mount points with the correct server name (the one used in `hostname.vm0`). In each line of this file, change `server` to `server-vm`.

```
client-vm    root=server:/export/root/client-vm    <-- Change this line
client-vm    root=server-vm:/export/root/client-vm  <-- To this
.
.
.
```

8.3.7.3 /export/root/client-vm/etc/fstab

Edit the file `/export/root/client-vm/etc/fstab` and substitute *server-vm* for each occurrence of *server*.

```
server:/export/root/client-vm / nfs rw 0 0    <-- Change this line
server-vm:/export/root/client-vm / nfs rw 0 0  <-- To this
      .
      .
      .
```

8.3.7.4 /export/root/client-vm/etc/hosts

Create the file `hosts` in the directory `/export/root/client-vm/etc` and add the Internet addresses of the server and client(s).

```
199.9.9.2      client-vm      <-- Add this line
199.9.9.1      server-vm     <-- Add this line
```

8.3.7.5 /export/root/client-vm/etc/hostname.vm0

Create the file `hostname.vm0` in the directory `/export/root/client-vm/etc` and add the line shown in the example below:

```
client-vm      <-- Add this line
```

Delete the file `/etc/hostname.le0`.

```
cp /vmunix /export/root/client-vm/vmunix
```

8.3.8 System Start-Up

After performing all of the configuration and setup tasks, it is time to start up the system.

- fasthalt the server
- When the server displays the OK prompt, enter `reset`, and reboot the system.

The server should boot, and when it starts the network daemons, the slave CPUs should boot over the VME backplane.

8.4 Activating ONC/VME

This section describes the steps required to activate ONC/VME for communicating between two or more SPARC CPU-2CEs sharing the same VMEbus backplane.

Refer to Section 7, for detailed information on configuring multiple CPUs in the same VMEbus backplane.

8.4.1 Setting Up a Private Network

This section shows the steps necessary to set up a private network that allows two SPARC CPU-2CEs to communicate using ONC/VME. For our example, the *host:1* CPU shall be called *ravel* and the *host:2* CPU shall be called *pink*.

Each SPARC CPU-2CE must be assigned a unique *slave map address space (vme-slavemap)* in the range of 0-15. Set the NVRAM variable *vme-slavemap* for each CPU. Set *ravel*'s *vme-slavemap* to zero and *pink*'s to one. At the OK prompt of *ravel (host:1)*, enter the following:

```
OK setenv vme-slavemap 0
```

At the OK prompt of *pink (host:2)*, enter the following:

```
OK setenv vme-slavemap 1
```

1. If *ypbind* is desired, add each host to the Master YP map or disable *ypbind* by entering

```
kill -9 <PID>
```

You can find out the <PID>, by entering:

```
ps -axj |grep yp
```

In the latter case, *ypbind* must be disabled because the private network is not known over the NIS master database, only locally.

2. Edit the */etc/hosts* files on *ravel* and *pink* to include the following:

```
199.9.9.1          ravel_vmip      <-- unique hostname
199.9.9.2          pink_vmip       <-- for each CPU
```

3. Edit or create the */etc/hostname.vm0* file on *ravel* to have the following line:

```
ravel_vmip
```

And on *pink* edit or create the */etc/hostname.vm0* file to have the following line:

```
pink_vmip
```

4. Activate *vm0*, *VME-IP service*, on *ravel* and *pink* by entering the following:

```
ifconfig vm0 ravel-vmip          execute this line on a ravel
ifconfig vm0 pink-vmip           execute this line on a pink
```

Verify that *vm0* is active by entering `ifconfig vm0` on each of the SPARC CPU-2CEs.

Now that *vm0* is active, functions such as *spray*, *rlogin*, *ftp*, *rsh*, and *rcp* can be performed as if on an ethernet network interface.

8.5 Theory of Operation

The heart of the ONC/VME protocol is a region of memory, exported by a processor to a well-known address in VME space. Other hosts may examine and modify this memory region, which is in a predefined format. Packets are transmitted by copying them into the source host's ONC/VME region, then notifying the destination host that a packet is ready. The destination host copies the packet into its own local memory (not its ONC/VME region) and acknowledges receipt of the packet.

Whenever a host modifies another host's ONC/VME region, it sends a VME mailbox interrupt to that host to signal that the host should examine its region and act on the changes.

The ONC/VME protocol is defined for IP (Internet Protocol) packets only. No other protocols (ARP, XNS, LocalTalk, etc.) may transmit packets over an ONC/VME network interface.

An ONC/VME network, from a high-level point of view, looks like any other network interface, with the exception of the protocol restriction above. Broadcast packets are transmitted by sending them individually to all other active hosts on the backplane.

8.6 Memory Usage

The following subsections detail the layout and addressing of the data structures defined by the ONC/VME protocol.

8.6.1 Addressing Scheme

ONC/VME memory regions are accessible in VME 32-bit data, 24-bit address space. Each host is assumed to occupy 1 megabyte in this space. The first three-quarters of that megabyte are unused by ONC/VME, and can be used by individual systems for their own purposes. The ONC/VME memory region starts at the last quarter-megabyte (i.e., at offset \$C0000) of each megabyte in VME A24D32 space. Since A24D32 space contains 16 megabytes, a maximum of 16 hosts are allowed in an ONC/VME network.

Each host has a host number, unrelated to its IP address. A host number determines which megabyte contains that host's ONC/VME region. The host number also determines the host's VME mailbox address. VME mailboxes occupy VME A16D16 space. The first 16 bytes of that space contain the first 8 mailboxes, at 2-byte intervals starting at location 0. The other 8 mailboxes are at location \$8000 in the A16D16 space, also at 2-byte intervals. Hosts numbered 0 through 7 use the first set of mailboxes (multiplying their host numbers by 2 to get the address in A16D16 space,) while hosts numbered 8 through 16 use the second set of addresses, multiplying the lower three bits of their host number by 2 and adding to \$8000 to obtain the mailbox address.

8.6.2 ONC/VME Region Layout

Table 8.3 shows the layout of the ONC/VME memory region (also called VMIP space, for historical reasons). Brief descriptions of the fields follow.

Table: 8.3 ONC/VME Region Layout

Offset	Size	Contents
0	2	Magic number (\$3D26)
2	2	Reserved
4	4	Packet-ready flag from host 0
8	4	Packet-ready flag from host 1
...		...
64	4	Packet-ready flag from host 15
68	4	State of host 0
72	4	State of host 1
...		...
128	4	State of host 15
132	4	IP address of host 0
136	4	IP address of host 1
...		...
192	4	IP address of host 15
196	4	Valid-host flag for host 0
200	4	Valid-host flag for host 1
...		...
256	4	Valid-host flag for host 15
260	4	Length of packet from host 0
264	4	Length of packet from host 1
...		...
320	4	Length of packet from host 15
324	1152	Contents of packet for host 0
1476	1152	Contents of packet for host 1
...		...
17604	1152	Contents of packet for host 15

8.6.2.1 Magic Number

The magic number must be placed at the beginning of a host's ONC/VME memory region, so that other hosts can probe to determine whether or not ONC/VME is active. The magic number also serves to identify the revision of the ONC/VME protocol. \$3D26 is the magic number for version 1 of the protocol.

8.6.2.2 Packet-Ready Flags

When a source host is ready to send a packet, it sets a packet-ready flag in the destination host's region. A packet-ready flag is set if it's nonzero.

8.6.2.3 Host States

A local host's notion of a remote host's state can have one of three values. State 0 means that the local host has sent a packet to the remote, but the remote hasn't yet processed the packet. State 1 is set by the remote when it's done copying the packet from the local host's region. State 2 indicates that the local host knows that the remote has finished processing the most recent packet, and is ready to receive another. The host state is in network byte order.

8.6.2.4 IP Addresses

ONC/VME uses standard four-byte IP addresses.

8.6.2.5 Valid-Host Flags

The valid-host flags are provided as shortcuts for source hosts. They allow a host to quickly determine which of its peers are running the ONC/VME protocol.

8.6.2.6 Packet Lengths

When a source host has determined the length of the packet it wishes to transmit, it sets the destination host's packet-length field to the size, including headers, of the packet. The packet-length fields are in network byte order.

8.6.2.7 Packet Contents

There is one packet buffer for each of the 16 hosts allowed by the protocol. The packet data, including headers, are copied into one of these buffers for transmission.

8.7 Protocol Operation

The following sections describe the operation of the ONC/VME protocol.

8.7.1 Initialization

ONC/VME initialization occurs in three stages.

8.7.1.1 Initializing the Hardware

First, any hardware-specific initialization, such as memory allocation and mapping, is performed. This usually includes allocating memory for the ONC/VME region and performing whatever MMU-specific functions are necessary to export the region to the appropriate location in VME space. Mailboxes should also be allocated, if necessary.

8.7.1.2 Initializing the Local Region

Once memory has been allocated for the local ONC/VME memory region, the region must be initialized. The packet-ready, state, valid-host, and length fields are zeroed out. The IP address of the local host is placed in the appropriate place in the region (*appropriate* meaning the IP address field whose number equals the local host number, host number 5 would set IP address number 5.) The host sets its own valid-host flag, in the same way.

At this point, the region is ready to be accessed by other hosts, so the magic number is set. The magic number should *not* be set until the IP address field is initialized.

8.7.1.3 Probing for Remote Hosts

For each of the other 15 possible hosts, the local host tries to read the start of the remote's ONC/VME region. If the magic number is present, the local host copies the remote host's IP address to the appropriate slot in its own region, and sets the remote host's valid-host flag in the local region. The local host sets its notion of the remote's state (i.e., the "state" entry in the local host's region whose number corresponds to the remote's host number) to two. The local host also places its own IP address into its slot in the remote host's region, and sets its valid-host flag in the remote region.

8.7.2 Sending a Packet

When a host has a packet to send, it looks through its list of valid hosts. For each host marked as valid, the local host checks the remote host's IP address against the packet's destination address. If no addresses match, the local host should indicate to the process sending the packet that the destination host doesn't exist.

Assuming a match is found, the following steps are taken. Assume that "X" is the host number of the remote host.

1. If the remote's state (according to its state entry in the local region) is zero, queue the packet up for later transmission or discard it.
2. Set the remote's state to zero.

3. Copy the packet, including IP headers, into the local packet buffer corresponding to the remote's host number. If there is non-IP header material at the beginning of the packet, it is not copied; only IP packets are placed in the packet buffers.
4. Set the local length entry corresponding to the remote's host number to the size of the packet.
5. Set the packet-ready flag corresponding to the *local* host number in the remote's ONC/VME region.

Send a VME mailbox interrupt to the remote host. The contents of the mailbox are ignored.

At this point, the first stage of packet transmission is finished. It is usual to exit the network driver, though the protocol allows for synchronous operation if desired.

Assuming the protocol is running asynchronously, when a mailbox interrupt is received by the local host, it does the following:

1. Check its local packet-ready flags. If any are set, a remote host has a packet for the local host. See "Receiving a Packet," below.
2. Check the states in its local region. If a state is 1, set it to 2, and send the next queued packet, if any, for the appropriate remote host.

8.7.2.1 Broadcast Packets

If a packet's destination address indicates that it should be broadcast to the entire ONC/VME network, simply send the packet separately to each host marked as valid in the local region.

8.7.3 Receiving a Packet

Packet receipt is initiated by the mailbox interrupt. When a mailbox interrupt is received, the local host does the following:

1. Check its local packet-ready flags. If none are set, the mailbox interrupt is an acknowledgement of packet receipt; see the previous section.
2. For each packet-ready flag, copy the packet from the local host's packet buffer in the remote ONC/VME region. The length of the packet is, of course, determined by reading the local host's packet length entry from the remote's memory.
3. Set the local host's state *in the remote region* to 1, meaning that the local host has received the packet and will no longer examine the remote's region.
4. Send a mailbox interrupt to the remote.
5. Process the packet appropriately for the local operating system.

Note that with the exception of setting the packet-ready flag, the sending side does not access the receiving side's ONC/VME memory region. The receiver performs almost all the VME memory accesses.

It is convenient, though not required, to set the remote's valid-host flag in the local region whenever a packet is received, in case the remote was marked invalid temporarily (see "Error Handling" and "Shutting Down" below.)

8.7.3.1 Shutting Down

To remove a host from a ONC/VME network, set its valid-host entry in all remote hosts' regions to zero. Set the local magic number to an invalid value.

8.7.3.2 Error Handling

If a host is removed from the network, remote systems will not be able to access its memory properly. This will usually result in a data access exception such as a timeout. The local operating system should handle such exceptions, noting when they occur during the processing of an ONC/VME packet. When an exception occurs during ONC/VME processing, set the valid-host flag of the offending host to zero, so that no further packets can be sent to that host.