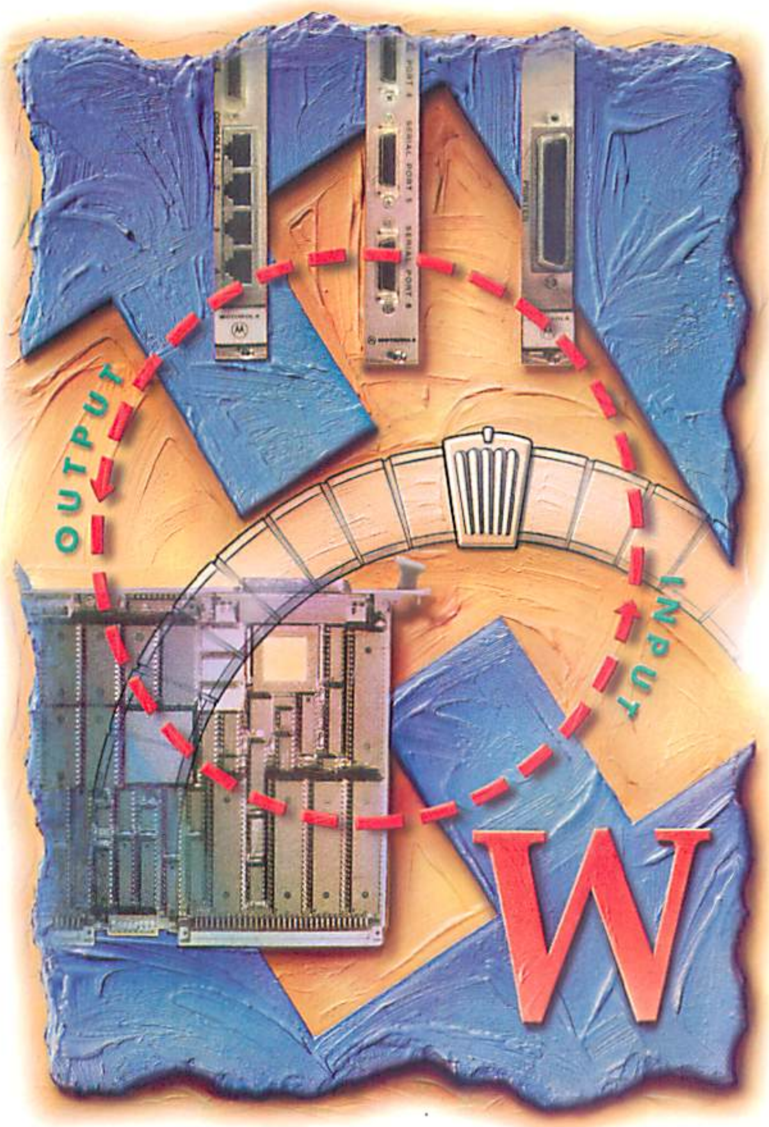
MOTOROLA

MVME133/ MVME133A-20 MPU VMEmodule User's Manual



HARDWARE

PREFACE

This manual provides general information, hardware preparation, installation instructions, operating instructions, and functional description for the MVME133/MVME133A MPU VMEmodule.

This manual is intended for anyone who wants to design OEM systems, supply additional capability to an existing system, or, in a lab environment, for experimental purposes.

A basic knowledge of computers and digital logic is assumed.

To use this manual, you may need to be familiar with the publications listed in the *Related Documentation* section in Chapter 1 of this manual.

MOTOROLA

MVME133ABUG/L1

JUNE 1987

MVME133ABUG

CUSTOMER LETTER

Both the MVME133 series monoboard microcomputers and the MVME133A series have debugging packages, MVME133BUG and MVME133ABUG, respectively. These debugging packages are very similar, but not identical.

Table 1 lists exactly which debugging package works with which module, and which doesn't.

TABLE 1. MVME133 and MVME133A Debugging Packages

DEBUGGING PACKAGE	OPERATION WITH MVME133 SERIES MODULES	OPERATION WITH MVME133A MODULES
MVME133Bug (EPROM SET)	Recommended.	NOT RECOMMENDED. IT IS NOT DESIGNED TO WORK WITH THE 133A MODULE. It may not come up at all. If it does, it attempts to access the Real-Time Clock (RTC) at $00FBxxxx. This address, however, falls into the VMEbus address range on the MVME133A. Therefore, it generates a Long Bus Error.
M68V2XSBG133 (source diskettes)	Recommended.	NOT RECOMMENDED. See above.
MVME133ABug (EPROM set)	NOT RECOMMENDED. It always addresses the RTC with $FFFBxxxx ([A25=1]). Therefore, if J11 pins 1-2 are jumpered, the MVME133 generates a VMEbus interrupt on Level 3 every time the RTC is accessed.	Recommended.
M68V2XSBG133A (source diskettes, when available)	NOT RECOMMENDED. See above.	Recommended.

Important Information

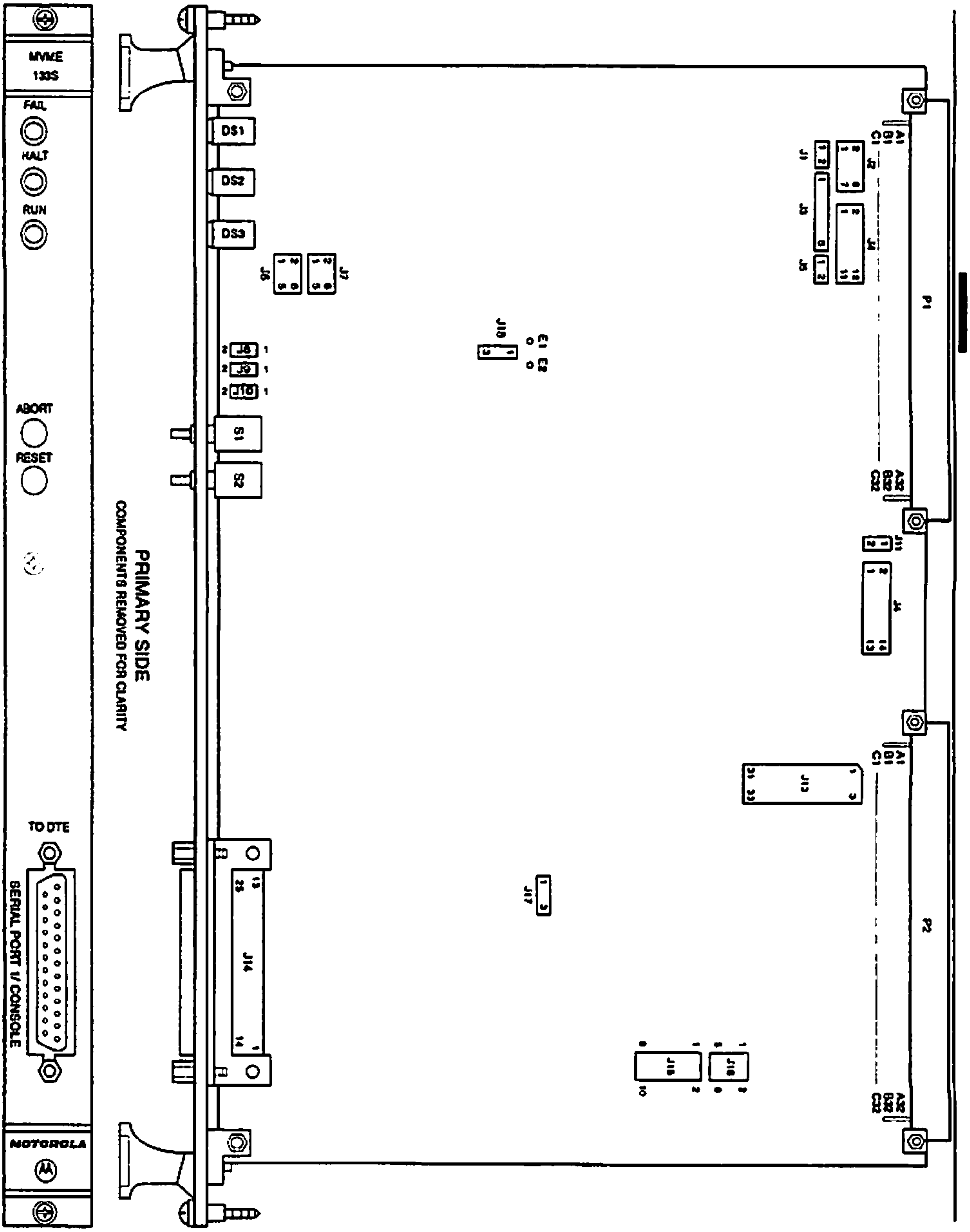


Figure 2-1. MVME133 Headers and Connectors

Table 2-1. Factory Installed Jumper Configuration (cont'd)

Header		Jumper Configuration	Function
Serial port B configuration select	J13	1-2, 4-5, 7-8, 10-11, 13-14, 16-17, 19-20, 22-23, 25-26, 28-29, 31-32	Port B as DCE (to terminal)
Software-readable header for module status register	J15	1-2, 3-4, 5-6, 7-8, 9-10	Module Status Register (MSR) bits 0 through 4 all =0
Serial ports RTXCx source select	J16	1-3, 2-4	RTXCA and RTXCB inputs from onboard 1.23 MHz signal
VMEbus data width select	J17	1-2	VMEbus is 32-bit data when [A24] is low, 16-bit data when [A24] is high
VMEbus addresssize select	J18	2-3	VMEbus contains both 24-bit and 32-bit address devices; onboard DRAM responds only to 24-bit address accesses
Cache disable		E1 and E2 not connected (MVME133A-20)	MC68020 on-chip cache memory not disabled

NOTE: J14 is the front-panel RS-232C connector.

MM58274 Real-Time Clock information is contained in:

Title: Data Sheet and Application Note 365
Source: National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, CA 95051

Support Information

The MVME133S Intelligent Communications Controller Support Information Manual, part number SIMVME133S, contains the connector interconnect signal information, parts list, and schematics for the MVME133S/MVME133S-001/MVME133SA-020. Refer to the Related Documentation table in this chapter for ordering information.

Manual Terminology

Throughout this manual, a convention has been maintained whereby data and address parameters are preceded by a character which specifies the numeric format as follows:

$	dollar	specifies a hexadecimal number
%	percent	specifies a binary number
&	ampersand	specifies a decimal number

Unless otherwise specified, all address references are in hexadecimal throughout this manual.

An asterisk (*) following the signal name for signals which are level significant denotes that the signal is true or valid when the signal is low.

An asterisk (*) following the signal name for signals which are edge significant denotes that the actions initiated by that signal occur on high to low transition.

In this manual, *assertion* and *negation* are used to specify forcing a signal to a particular state. In particular, *assertion* and *assert* refer to a signal that is active or true; *negation* and *negate* indicate a signal that is inactive or false. These terms are used independently of the voltage level (high or low) that they represent.

Related Documentation

The following publications may provide additional helpful information. If not shipped with this product, they may be purchased by contacting your local Motorola sales office.

Document Title	Motorola Publication Number
MVME133S Support Information (refer to the Support Information section in this chapter)	SIMVME133S/Dx
MVME133 Debugging Package User's Manual	MVME133BUG/Dx
MVME133A Debugging Package User's Manual	MVME133ABUG/Dx
MC68020, MC68EC020 Microprocessors User's Manual	M68020UM/AD
MC68881/2 Floating-Point Coprocessor User's Manual	MC68881UM/AD
MC68901 Multi-Function Peripheral Data Sheet	MC68901/D

NOTE: Although not shown in the above list, each Motorola Computer Group manual publication number is suffixed with characters that represent the revision level of the document, such as /D2 (the second revision of a manual); a supplement bears the same number as the manual but has a suffix such as /D2A1 (the first supplement to the manual).

The following publications are available from the sources indicated.

VMEbus Specification information is contained in:

Title: Versatile Backplane Bus: VMEbus
ANSI/IEEE Std 1014-1987

Source: The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, New York 10017

Z85C30 Serial Communications Controller information is contained in:

Title: SCC User's Guide

Source: Zilog, Inc.
210 East Hacienda Avenue
Campbell, CA 95008-6600

- Four 28-pin JEDEC sockets for ROMs/PROMs/EPROMs/EEPROMs (in two banks, each 16 bits wide) (total 256KB maximum)
- Three 8-bit programmable timers for tick and watchdog functions
- Time-of-day clock/calendar (real-time clock) (MM58274A)
- Front panel asynchronous DB25 serial debug RS-232C port (on MC68901 MFP)
- Dual multiprotocol (synchronous/asynchronous) serial ports (Z85C30):
  - RS-485/RS-422A (port A)
  - RS-232C (port B)
- VMEbus system controller functions with level 3 arbiter
- Single level bus requester (level is jumper selectable)
- VMEbus interrupter
- VMEbus interrupt handler for all seven levels
- Front panel RUN, HALT, and FAIL status LEDs
- Front panel RESET and ABORT switches
- Remote reset through edge connector P2
- Five-position software-readable header, part of Module Status Register (MSR).

Specifications

The MVME133 specifications are provided in Table 1-1.

CHAPTER 1
GENERAL INFORMATION

Introduction

This manual provides general information, preparation for use and installation instructions, operating instructions, and functional description for the MVME133S, MVME133S-001, and MVME133SA-020 MPU VMEmodules.

The module is referred to as the MVME133 throughout this manual except in cases where individual references are required.

Model Designations

The MVME133 is available in three versions, as listed in Table 1-1.

Table 1-1. Model Designations

Model Number	Description
MVME133S	12.5 MHz MC68020 with 12.5 MHz MC68881
MVME133S-001	16.67 MHz MC68020 with 16.67 MHz MC68881
MVME133SA-020	20 MHz MC68020 with 20 MHz MC68881

Features

The features of the MVME133 include:

- Double-high/single-wide VME board
- Address 24/Data 32 (A24/D32) VMEbus master (A24/D16 compatible) (MVME133S, MVME133S-001)
- Address 32/Data 32 (A32/D32) VMEbus master (A32/D16, A24/D32, A24/D16 compatible) (MVME133SA-020)
- MC68020 Microprocessor with 32-bit address and data
- 1MB of shared local dynamic RAM, 32 bits wide, accessible from VMEbus

NOTICE

While reasonable efforts have been made to assure the accuracy of this document, Motorola, Inc. assumes no liability resulting from any omissions in this document, or from the use of the information obtained therein. Motorola reserves the right to revise this document and to make changes from time to time in the content hereof without obligation of Motorola to notify any person of such revision or changes.

RESTRICTED RIGHTS LEGEND

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by Motorola, Inc.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Motorola, Inc.
Computer Group
2900 South Diablo Way
Tempe, Arizona 85282

MOTOROLA

MVME133/D2A2
November 1993

Supplement to
MVME133/MVME133A-20
MPU VMEmodule
User's Manual
(MVME133/D2)

This supplement provides changes in support of the MVME133S VMEmodule family. The MVME133 series modules have been replaced by the MVME133S series modules, as follows:

- The MVME133 is replaced by the MVME133S.
- The MVME133-1 is replaced by the MVME133S-001.
- The MVME133A is replaced by the MVME133SA-020.

All information in the manual applies to the MVME133S series modules, except that where any information in the manual applies specifically to a particular module, it now applies to the equivalent new module as listed above. References to the Z8530 serial communications controller apply to the Z85C30, which replaces it.

This supplement replaces a previous supplement, MVME133/D2A1. The attached pages are replacements or additions for the corresponding pages in the manual. Place this page behind the title page of the manual as a record of this change. Please remove and replace pages according to the following table:

Replace Old	With New
1-1/1-2	1-1/1-2
1-7/1-8	1-7/1-8
2-3/2-4	2-3/2-4
2-5/2-6	2-5/2-6

- A vertical bar (|) in the margin of a revised page indicates a text change or addition.
- The supplement number is shown at the bottom of each revised page.

Just in Time

70414-08 PRINTED IN USA 5-96 IRONWOOD LITHO 961177 500

MOTOROLA

VME133SA/LT1
March 1995

Dear Valued Customer:

The MC68881 Floating Point Co-Processor used on the MVME133S series has been discontinued by the manufacturer. New versions of the MVME133S series will use the MC68882 Floating Point Co-Processor. The following table contains the old and new marketing numbers.

Old Marketing Number	Coprocessor Type	New Marketing Number	Coprocessor Type
MVME133S	MC68881	MVME133SA	MC68882
MVME133S-001	MC68881	MVME133S-001A	MC68882
MVME133SA-020	MC68881	MVME133SA-020A	MC68882

Some software changes may be required to support the new coprocessor. The following information will identify the type of changes required when migrating from an MC68881 to an MC68882.

Floating-point exceptions are enabled through the FPCR Exception Enable byte. The following is a list of floating point exceptions:

BSUN	(branch -or-set-on-unordered)
SNAN	(signaling Nan)
OPERR	(operand error)
DZ	(divide-by-zero)
OVFL	(overflow)
UNFL	(underflow)
INEX2	(inexact on binary format)
INEX1	(inexact on packed BCD)

Each of these exceptions has an entry in the vector table. If these floating point exception handlers are written specifically with the MC68881 in mind, then chances are that some work is needed to migrate to the MC68882.

Important Information

For systems that enable any of the above mentioned floating point exceptions, modification of existing MC68881 floating point exception handlers is needed. Outlined below are the main things to watch out for:

1. The MC68882 requires that the first floating point instruction be an FSAVE instruction. Once this state frame is generated, bit 27 of the BIU Flag Word in this saved frame must be set. This modified frame must then be placed back into the coprocessor via the FRESTORE instruction. Though the MC68881 does not require this code modification, it is not affected adversely by this modification.
2. The MC68882 state frames are 32 bytes larger than their equivalent MC68881 counterparts. For systems that need precise control of the system stack, this difference needs to be considered.
3. Since the MC68882 state frame is slightly larger, offsets to items within the state frame is different relative to the top of the frame. A way around this is by accessing items in the state frame relative to the bottom of the frame. This way, software may be written to work for either an MC68882 or the MC68881.

Motorola Debugger Products

This change affects the MVME133S series debugger products. The following table lists the correct firmware revision for the associated hardware revision.

Old Assembly Number	Correct Debugger Revision	New Assembly Number	Correct Debugger Revision
01-W3818B01E	MVME133BUG	01-W3818B01F	MVME133BUG12
01-W3818B02C	MVME133BUG	01-W3818B02D	MVME133BUG12
01-W3818B03E	MVME133ABUG	01-W3818B03F	MVME133ABUG12
Note: The new debugger revisions are backwards compatible with older hardware.			

Motorola Real-Time Operating Systems

There are no work-arounds required if you are using Motorola's VMEexec real-time operating system. It correctly handles any differences between the MC68881 and MC68882.

Third-Party Real-Time Operating Systems

If you encounter problems with an operating system not supplied by Motorola, you should contact the supplier directly for additional information.

VERSAdos Real-Time Operating System

If you are using the VERSAdos real-time operating system and you have problems related to this change, contact Linden Technologies at 602-954-8602 or by FAX at 602-954-8013.

MC68882 Technical Information

Additional technical information regarding Motorola MC68881 and MC68882 devices can be found in the *MC68881/882 Floating-Point Coprocessor User's Manual,* part number MC68881UM/AD. It can be obtained from the Motorola Literature Distribution Center. The phone number is 602-994-6561 or 800-441-2447 (U.S. only) and the FAX number is 602-994-6430.

MOTOROLA

CEMARKA/LT1
December 1995

CE

European Notice: Board products with the CE marking comply with the EMC Directive (89/336/EEC). Compliance with this directive implies conformity to the following European Norms:

EN55022 (CISPR 22)	Radio Frequency Interference
EN50082-1 (IEC801-2, IEC801-3, IEEC801-4)	Electromagnetic Immunity

The product also fulfills EN60950 (product safety) which is essentially the requirement for the Low Voltage Directive (73/23/EEC).

This board product was tested in a representative system to show compliance with the above mentioned requirements. A proper installation in a CE-marked system will maintain the required EMC/safety performance.

Important Information

MOTOROLA

MVME133/D2A2
November 1993

Supplement to
MVME133/MVME133A-20
MPU VMEmodule
User's Manual
(MVME133/D2)

This supplement provides changes in support of the MVME133S VMEmodule family. The MVME133 series modules have been replaced by the MVME133S series modules, as follows:

- The MVME133 is replaced by the MVME133S.
- The MVME133-1 is replaced by the MVME133S-001.
- The MVME133A is replaced by the MVME133SA-020.

All information in the manual applies to the MVME133S series modules, except that where any information in the manual applies specifically to a particular module, it now applies to the equivalent new module as listed above. References to the Z8530 serial communications controller apply to the Z85C30, which replaces it.

This supplement replaces a previous supplement, MVME133/D2A1. The attached pages are replacements or additions for the corresponding pages in the manual. Place this page behind the title page of the manual as a record of this change. Please remove and replace pages according to the following table:

Replace Old	With New
1-1/1-2	1-1/1-2
1-7/1-8	1-7/1-8
2-3/2-4	2-3/2-4
2-5/2-6	2-5/2-6

- A vertical bar (|) in the margin of a revised page indicates a text change or addition.
- The supplement number is shown at the bottom of each revised page.

(P) MVME133/D2A2

Just in Time

70414-08 PRINTED IN USA 5-96 IRONWOOD LITHO 961177 500

NOTICE

While reasonable efforts have been made to assure the accuracy of this document, Motorola, Inc. assumes no liability resulting from any omissions in this document, or from the use of the information obtained therein. Motorola reserves the right to revise this document and to make changes from time to time in the content hereof without obligation of Motorola to notify any person of such revision or changes.

RESTRICTED RIGHTS LEGEND

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by Motorola, Inc.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Motorola, Inc.
Computer Group
2900 South Diablo Way
Tempe, Arizona 85282

CONTENTS

APPENDICES

INDEX

FIGURES

TABLES

CHAPTER 1
GENERAL INFORMATION

Introduction

This manual provides general information, preparation for use and installation instructions, operating instructions, and functional description for the MVME133S, MVME133S-001, and MVME133SA-020 MPU VMEmodules.

The module is referred to as the MVME133 throughout this manual except in cases where individual references are required.

Model Designations

The MVME133 is available in three versions, as listed in Table 1-1.

Table 1-1. Model Designations

Model Number	Description
MVME133S	12.5 MHz MC68020 with 12.5 MHz MC68881
MVME133S-001	16.67 MHz MC68020 with 16.67 MHz MC68881
MVME133SA-020	20 MHz MC68020 with 20 MHz MC68881

Features

The features of the MVME133 include:

- Double-high/single-wide VME board
- Address 24/Data 32 (A24/D32) VMEbus master (A24/D16 compatible) (MVME133S, MVME133S-001)
- Address 32/Data 32 (A32/D32) VMEbus master (A32/D16, A24/D32, A24/D16 compatible) (MVME133SA-020)
- MC68020 Microprocessor with 32-bit address and data
- 1MB of shared local dynamic RAM, 32 bits wide, accessible from VMEbus

- Four 28-pin JEDEC sockets for ROMs/PROMs/EPROMs/EEPROMs (in two banks, each 16 bits wide) (total 256KB maximum)
- Three 8-bit programmable timers for tick and watchdog functions
- Time-of-day clock/calendar (real-time clock) (MM58274A)
- Front panel asynchronous DB25 serial debug RS-232C port (on MC68901 MFP)
- Dual multiprotocol (synchronous/asynchronous) serial ports (Z85C30):
  - RS-485/RS-422A (port A)
  - RS-232C (port B)
- VMEbus system controller functions with level 3 arbiter
- Single level bus requester (level is jumper selectable)
- VMEbus interrupter
- VMEbus interrupt handler for all seven levels
- Front panel RUN, HALT, and FAIL status LEDs
- Front panel RESET and ABORT switches
- Remote reset through edge connector P2
- Five-position software-readable header, part of Module Status Register (MSR).

Specifications

The MVME133 specifications are provided in Table 1-1.

Table 1-2. Module Specifications

Characteristics	Specifications
Power requirements (with full set of ROMs/PROMs/EPROMs/EEPROMs)	+5 Vdc @ 4 A (typ), 5 A (max) (MVME133) +5 Vdc @ 4.5 A (typ), 6 A (max) (MVME133-1) +5 Vdc @ 5 A (typ), 7 A (max) (MVME133A-20) +12 Vdc @ 100 mA (typ), 250 mA (max) -12 Vdc @ 100 mA (typ), 250 mA (max)
Microprocessor	MC68020
Coprocessor	MC68881
Clock signal to MPU and FPC	12.5 MHz (MVME133) 16.75 MHz (MVME133-1) 20 MHz (MVME133A-20)
Addressing	
Total range (onboard and offboard)	16 Mb (A24) for the MVME133 series, 4 Gb for the MVME133A-20
ROM/PROM/EPROM/EEPROM	256Kb maximum: four sockets (two banks of two each, 16 bits wide) for 2K x 8, 8K x 8, 16K x 8, 32K x 8, or 64K x 8 devices
Dynamic RAM	1Mb (32 bits wide)
Time-of-Day Clock/Calendar	0.1 second resolution
Timers (on MC68901 MFP)	4 total (3 available to user)
Debug port (not available)	8-bit
Watchdog	8-bit
Tick	8-bit
Spare	8-bit

Table 1-2. Module Specifications (cont'd)

Characteristics	Specifications
Bus configuration	Data Transfer Bus (DTB) master or slave, with 24-bit address (A24) and 32-bit data (D32) (MVME133) Data Transfer Bus (DTB) master or slave, with 32-bit or 24-bit address (A32 or A24) and 32-bit or 16-bit data (D32 or D16) (MVME133A-20)
Interrupt handler	Any or all onboard interrupts plus up to seven VMEbus interrupts
Interrupter	Level 3 with status ID of $FF
Bus arbitration	When MVME133 is system controller, it arbitrates bus requests/grants on level 3 only
Reset	May be accomplished by: • SYSRESET* • Power up • RESET switch • Watchdog timer timeout • Remote reset • MC68020 RESET instruction
Serial I/O ports	Port A multiprotocol RS-485/422A through P2 Port B multiprotocol RS-232C through P2 Asynchronous RS-232C debug serial port DCE through front panel J14
Operating Temperature	0° to 55° C at point of entry of forced air (approximately 150 LFM)

Table 1-2. Module Specifications (cont'd)

Characteristics	Specifications
Storage Temperature	-40° to 85° C
Relative humidity	5% to 90% (non-condensing)
Physical characteristics (excluding front panel)	
Height	9.187 inches (233.35 mm)
Width	6.299 inches (160.0 mm)
Thickness	0.063 inches (1.6 mm)
Connectors	
VMEbus	DIN No. 41612C96 male (P1, P2)
RS-232C	DB-25 female (J14)

Cooling Requirements

The MVME133 is specified, designed, and tested to operate reliably with an incoming air temperature range from 0° C to 55° C with forced air cooling at a velocity typically achievable by using a 71 CFM axial fan. Temperature qualification is performed in a standard Motorola VMEsystem chassis. Twenty-five watt load boards are inserted in two card slots, one on each side, adjacent to the board under test, to simulate a high power density system configuration. An assembly of three axial fans, rated at 71 CFM per fan, is placed directly under the VME card cage. The incoming air temperature is measured between the fan assembly and the card cage, where the incoming airstream first encounters the module under test. Test software is executed as the module is subjected to ambient temperature variations. Case temperatures of critical, high power density integrated circuits are monitored to ensure component vendors specifications are not exceeded.

While the exact amount of airflow required for cooling depends on the ambient air temperature and the type, number, and location of boards and other heat sources, adequate cooling can usually be achieved with 10 CFM and 150 LFM flowing over the module. Less airflow is required to cool the module in environments having lower maximum ambients. Under more favorable thermal conditions, it may be possible to operate the module reliably at higher than 55 degrees C with increased airflow. It is important to note that there are several factors, in addition to the rated CFM of the air mover, which determine the actual volume and speed of air flowing over a module.

FCC Compliance

This MVME133 was tested in an FCC-compliant chassis, and meets the requirements for Class A equipment. FCC compliance was achieved under the following conditions:

a. Shielded cables on all external I/O ports.

b. Cable shields connected to earth ground via metal shell connectors bonded to a conductive module front panel.

c. Conductive chassis rails connected to earth ground. This provides the path for connecting shields to earth ground.

d. Front panel screws properly tightened.

For minimum RF emissions, it is essential that the above conditions be implemented; failure to do so could compromise the FCC compliance of the equipment containing the module.

General Description

The MVME133 is a double-high VMEmodule. It takes one slot in a VME system. The module has large onboard DRAM (1Mb), ROM/PROM/EPROM/EEPROM capability (up to 256Kb), serial ports including debug port, floating point coprocessor, tick timer, watchdog timer, time-of-day clock/calendar, and VMEbus interface with system controller functions.

The MVME133 is a single-board MPU module intended to be used in a single processor system, but not stand-alone. It is an excellent choice for applications requiring real-time operation such as industrial automation and robotics.

The optionally available MVME133BUG or MVME133ABUG (as appropriate) debug monitor firmware package offers 39 debug, up/downline load, and disk bootstrap load commands, as well as a full set of onboard diagnostics and a one-line assembler/disassembler.

The MVME133 contains no parallel ports. To use a parallel device, such as a printer, with the MVME133, you must add a module such as the MVME050 System Controller Module to your system.

Related Documentation

The following publications may provide additional helpful information. If not shipped with this product, they may be purchased by contacting your local Motorola sales office.

Document Title	Motorola Publication Number
MVME133S Support Information (refer to the Support Information section in this chapter)	SIMVME133S/D*x*
MVME133 Debugging Package User's Manual	MVME133BUG/D*x*
MVME133A Debugging Package User's Manual	MVME133ABUG/D*x*
MC68020, MC68EC020 Microprocessors User's Manual	M68020UM/AD
MC68881/2 Floating-Point Coprocessor User's Manual	MC68881UM/AD
MC68901 Multi-Function Peripheral Data Sheet	MC68901/D

NOTE: Although not shown in the above list, each Motorola Computer Group manual publication number is suffixed with characters that represent the revision level of the document, such as /D2 (the second revision of a manual); a supplement bears the same number as the manual but has a suffix such as /D2A1 (the first supplement to the manr:al).

The following publications are available from the sources indicated.

VMEbus Specification information is contained in:

Title: Versatile Backplane Bus: VMEbus
ANSI/IEEE Std 1014-1987

Source: The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, New York 10017

Z85C30 Serial Communications Controller information is contained in:

Title: SCC User's Guide

Source: Zilog, Inc.
210 East Hacienda Avenue
Campbell, CA 95008-6600

MM58274 Real-Time Clock information is contained in:

Title: Data Sheet and Application Note 365
Source: National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, CA 95051

Support Information

The MVME133S Intelligent Communications Controller Support Information Manual, part number SIMVME133S, contains the connector interconnect signal information, parts list, and schematics for the MVME133S/MVME133S-001/MVME133SA-020. Refer to the Related Documentation table in this chapter for ordering information.

Manual Terminology

Throughout this manual, a convention has been maintained whereby data and address parameters are preceded by a character which specifies the numeric format as follows:

$	dollar	specifies a hexadecimal number
%	percent	specifies a binary number
&	ampersand	specifies a decimal number

Unless otherwise specified, all address references are in hexadecimal throughout this manual.

An asterisk (*) following the signal name for signals which are level significant denotes that the signal is true or valid when the signal is low.

An asterisk (*) following the signal name for signals which are edge significant denotes that the actions initiated by that signal occur on high to low transition.

In this manual, *assertion* and *negation* are used to specify forcing a signal to a particular state. In particular, *assertion* and *assert* refer to a signal that is active or true; *negation* and *negate* indicate a signal that is inactive or false. These terms are used independently of the voltage level (high or low) that they represent.

CHAPTER 2
HARDWARE PREPARATION AND INSTALLATION

Introduction

This chapter provides the unpacking, hardware preparation, and installation instructions for the MVME133 modules.

Unpacking Instructions

NOTE

> If the carton is damaged upon receipt, request carrier's agent be present during unpacking/ inspection of equipment.

Unpack equipment from shipping carton. Refer to packing list and verify that all items are present. Save packing material for storing and reshipping of the module.

Hardware Preparation

To select the desired configuration and ensure proper operation of the MVME133, certain changes may be made before installation. These changes are made through jumper or wire-wrap arrangements on the headers. The location of the headers and connectors is illustrated in Figure 2-1. The module has been factory tested and is shipped with factory-installed jumper configuration that is also shown in Figure 2-1 and Table 2-1. The module is operational with the factory-installed jumpers and will operate with the optional Debug Monitor (MVME133BUG or MVME133BUGA) The module is configured to provide the system functions required for a VMEbus system.

Table 2-1. Factory Installed Jumper Configuration

Header		Jumper Configuration	Function
System controller enable select	J1	1-2	MVME133 is system controller
Onboard RAM offset address select	J2	1-2, 3-4 (MVME133/-1) 1-2, 3-4, 5-6, 7-8 (MVME133A-20)	RAM offset address is $00000000 as seen from VMEbus
VMEbus requester level select	J3	5-6	
	J4	1-2, 5-6, 7-8, 9-11, 10-12	level 3 requested
RMW cycle type select	J5	1-2	MVME133 requests VMEbus mastership on all RMW cycles
ROM/PROM/ EPROM /EEPROM size select	J6	1-3, 4-6 J7: 1-3, 4-6	Banks 1 & 2 each set for two 64K x 8 ROMs/PROMs/EPROMs
Global timeout select	J8	1-2	Enabled if the MVME133 is the system controller
RESET switch select	J9	1-2	Enabled
ABORT switch select	J10	1-2	Enabled
VMEbus interrupter select	J11	1-2	Enabled. DO NOT CHANGE THIS CONFIGURATION.
VMEbus interrupt handler select	J12	1-2, 3-4, 5-6, 7-8, 9-10, 11-12, 13-14	IRQ1* through IRQ7* all enabled

Table 2-1. Factory Installed Jumper Configuration (cont'd)

Header		Jumper Configuration	Function
Serial port B configuration select	J13	1-2, 4-5, 7-8, 10-11, 13-14, 16-17, 19-20, 22-23, 25-26, 28-29, 31-32	Port B as DCE (to terminal)
Software-readable header for module status register	J15	1-2, 3-4, 5-6, 7-8, 9-10	Module Status Register (MSR) bits 0 through 4 all =0
Serial ports RTXC*x* source select	J16	1-3, 2-4	RTXCA and RTXCB inputs from onboard 1.23 MHz signal
VMEbus data width select	J17	1-2	VMEbus is 32-bit data when [A24] is low, 16-bit data when [A24] is high
VMEbus addresssize select	J18	2-3	VMEbus contains both 24-bit and 32-bit address devices; onboard DRAM responds only to 24-bit address accesses
Cache disable		E1 and E2 not connected (MVME133A-20)	MC68020 on-chip cache memory not disabled

NOTE: J14 is the front-panel RS-232C connector.

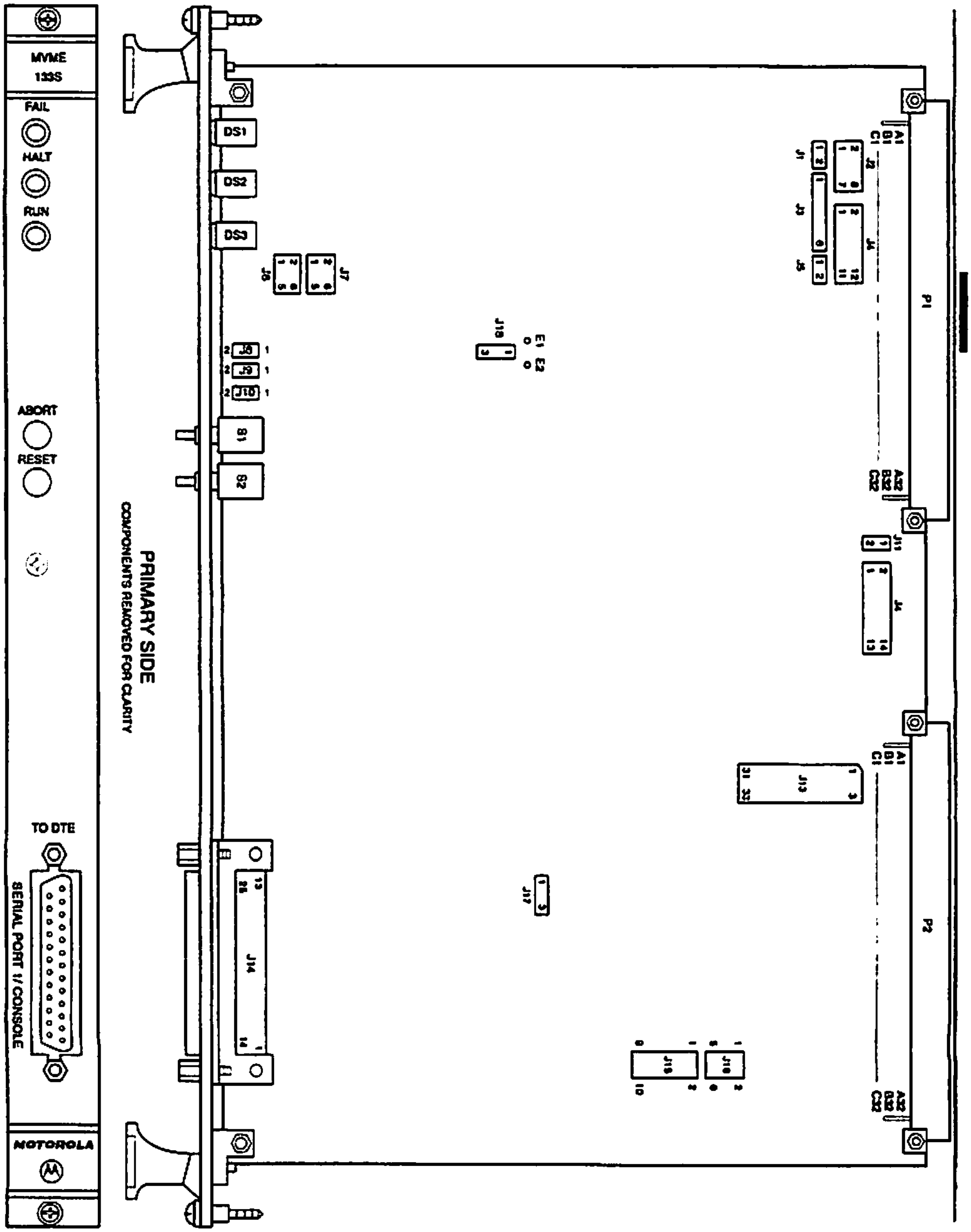
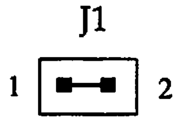
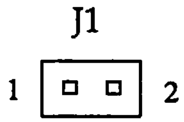


Figure 2-1. MVME133 Headers and Connectors

System Controller Enable Header (J1)



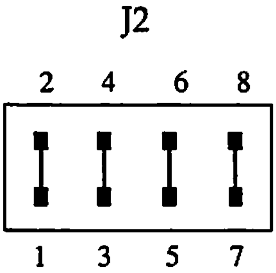
System Controller
(Factory Configuration)



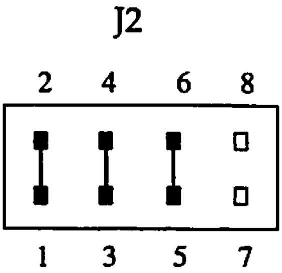
Not System Controller

Onboard RAM Offset Select Header (J2)

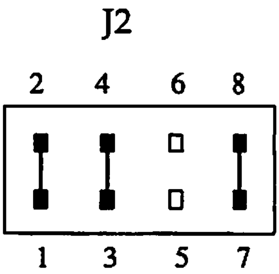
J2 controls the offset address of the RAM as seen by the VMEbus. The RAM is shared. To the onboard logic (e.g., a monitor), the RAM address is fixed at $00000000-$000FFFFF. Refer to the memory map information in Chapter 3 and the shared RAM information in Chapter 4 for more details.



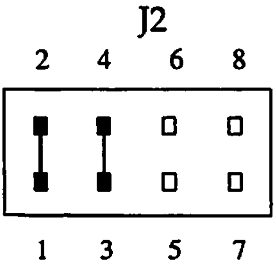
OFFSET $00000000



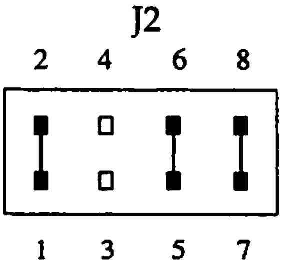
OFFSET $00100000



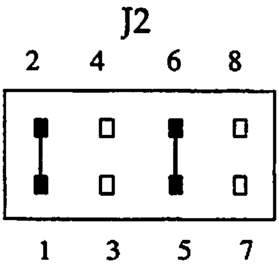
OFFSET $00200000



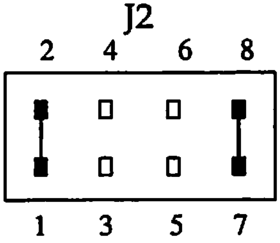
OFFSET $00300000



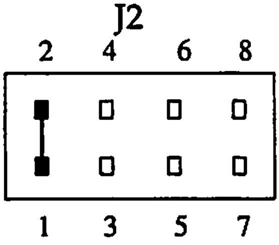
OFFSET $00400000



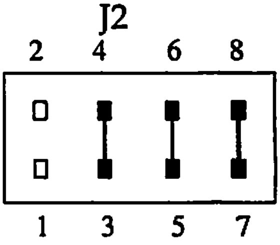
OFFSET $00500000



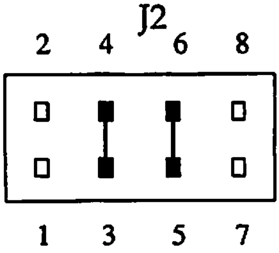
OFFSET $00600000



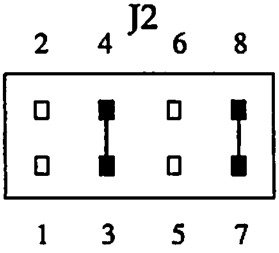
OFFSET $00700000



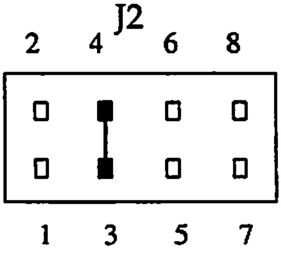
OFFSET $00800000



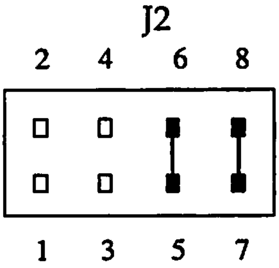
OFFSET $00900000



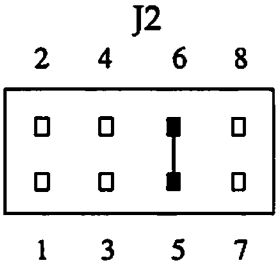
OFFSET $00A00000



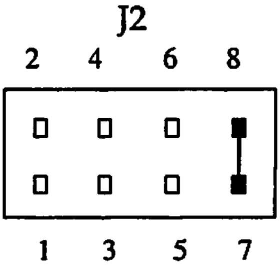
OFFSET $00B00000



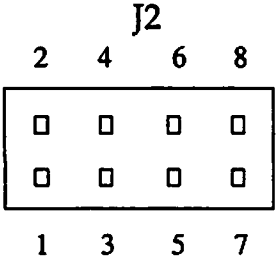
OFFSET $00C00000



OFFSET $00D00000

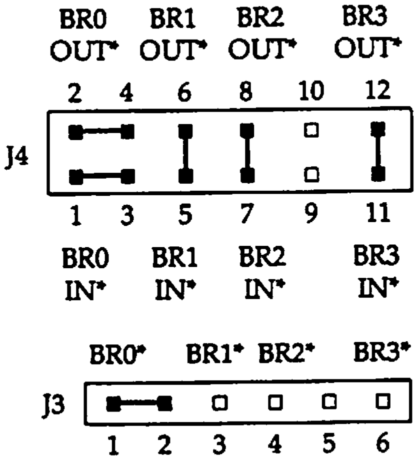


OFFSET $00E00000

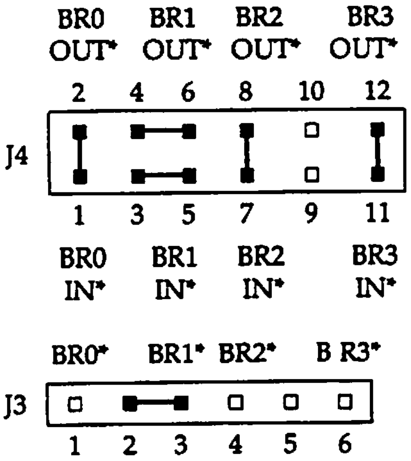


OFFSET $00F00000

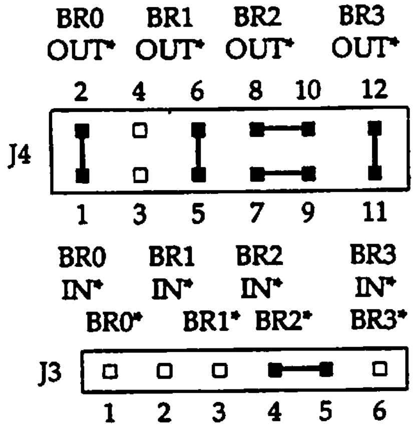
VMEbus Requester Level Select Headers (J3, J4)



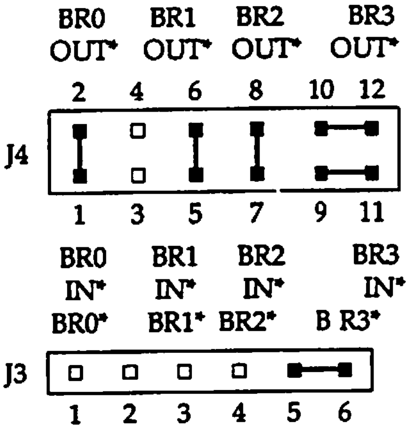
LEVEL 0



LEVEL 1

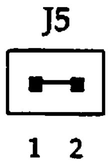


LEVEL 2

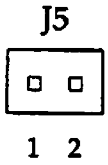


LEVEL 3
(Factory Configuration)

RMW Cycle Type Select Header (J5)

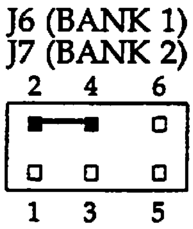


MVME133 requires its VMEbus requester to obtain VMEbus mastership for all RMW cycles in order to maintain the integrity of these cycles. (Factory configuration)

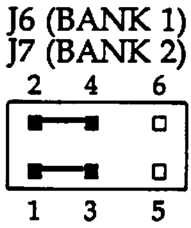


MVME133 does not require its VMEbus requester to obtain VMEbus mastership for RMW cycles to its onboard RAM. Software must never generate RMW cycles to VMEbus, and other VMEbus masters must never perform multiple address RMW cycles to the MVME133 onboard shared RAM.

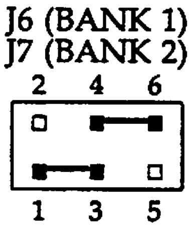
ROM/PROM/EPROM/EEPROM Size Select Headers (J6, J7)



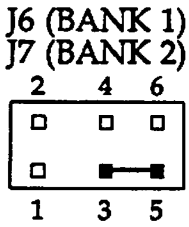
8K x 8 or 16K x 8
ROM/PROM/EPROM



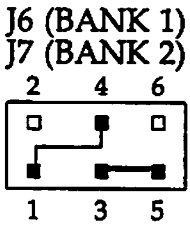
32K x 8
ROM/PROM/EPROM



64K x 8
ROM/PROM/EPROM
(Factory Configuration)

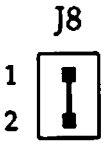


2K x 8 or 8K x 8
EEPROM

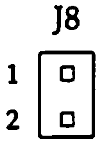


32K x 8
EEPROM

Global Timeout Select Header (J8)

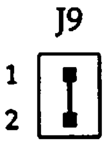


Global time out enabled. If configured as system controller (refer to the *System Controller Enable Select Header* section in this chapter) MVME133 activates BERR* if DS0* and /or DS1* are low for more than 50 to 57 microseconds. (factory configuration)

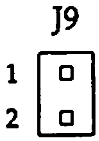


Global timeout disabled: This may cause a system problem. Refer to the *System Considerations* section in this chapter.

RESET Switch Select Header (J9)

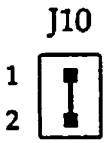


RESET Switch Enabled
(Factory Configuration)

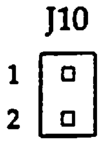


RESET Switch Disabled

ABORT Switch Select Header (J10)

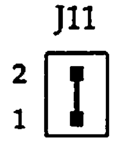


ABORT Switch Enabled
(Factory Configuration)



ABORT Switch Disabled

VMEbus Interrupter Select Header (J11)

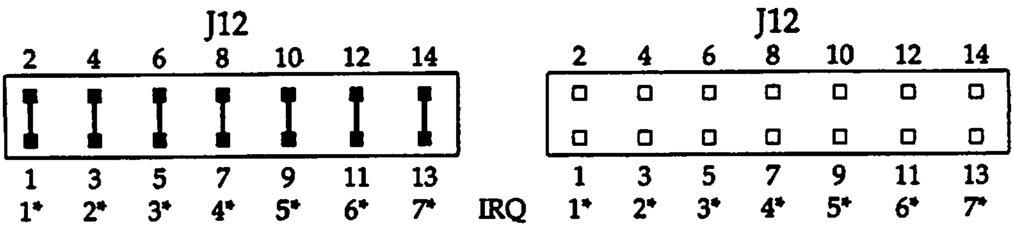


Enabled
(Factory Configuration)

CAUTION

Do not change this configuration or the MVME133 will not operate properly.

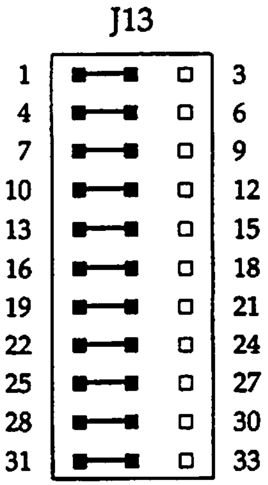
VMEbus Interrupt Handler Select Header (J12)



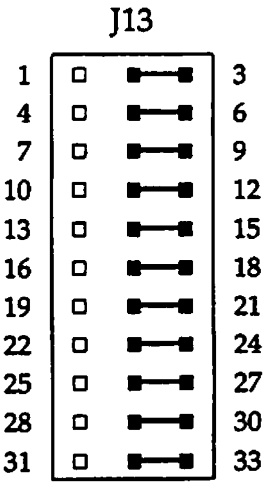
All Seven Enabled
(Factory Configuration)

All Seven Disabled

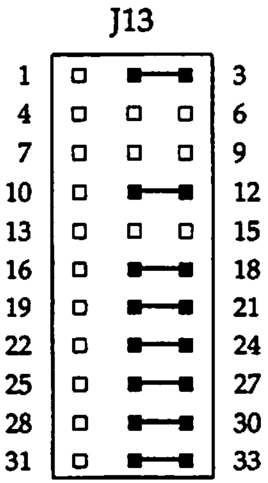
Serial Port B Configuration Select Header (J13)



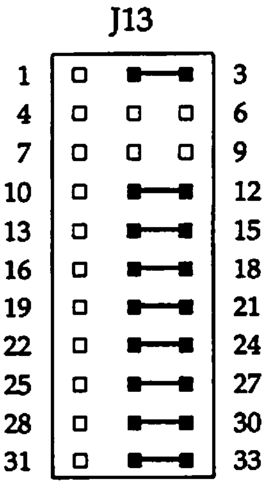
Port B as DCE (to terminal) (Factory Configuration)



Port B as DTE (to modem). RTXC is input to TRXCB pin of Z8530 and is echoed on TTXC.

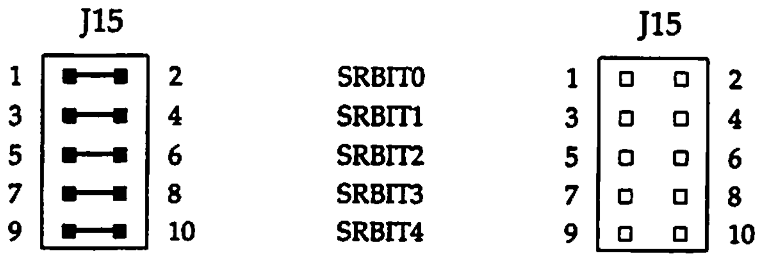


Port B as DTE (to modem). RTXC is input to TRXCB pin of Z8530. TTXC is not connected.



Port B as DTE (to modem). TTXC is output from TRXCB pin of Z8530. RTXC is not connected.

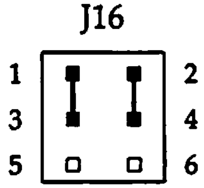
Software-Readable Header for Module Status Register (MSR) (J15)



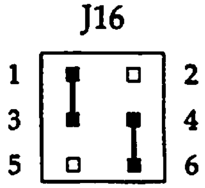
SRBIT0 through SRBIT4 = 0
(Factory Configuration)

SRBIT0 through SRBIT4 = 1

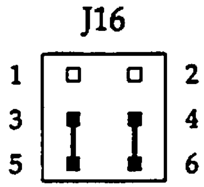
Serial Ports RTXCx Source Select Header (J16)



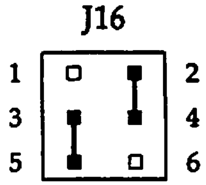
RTXCA input from onboard 1.23 MHz.
RTXCB input from onboard 1.23 MHz.
(Factory Configuration)



RTXCA input from onboard 1.23 MHz.
RTXCB input from TTXC if port B is configured to go to a terminal, or from RXC if port B is configured to go to a modem.



RTXCA input from TT if port A is configured as a slave, or from RT if port A is configured as a master. RTXCB input from onboard 1.23 MHz.

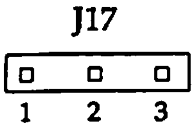


RTXCA input from TT if port A is configured as a slave, or from RT if port A is configured as a master. RTXCB input from TTXC if port B is configured to go to a terminal, or from RXC if port B is configured to go to a modem.

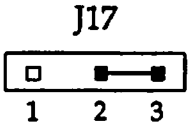
NOTE

Refer to the *System Considerations* section in this chapter for possible installation of terminators for port A signals

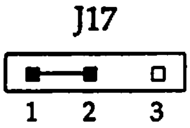
VMEbus Data Width Select Header (J17)



VMEbus is 16-bit data.



MVEbus is 32-bit data.

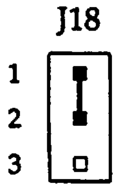


VMEbus is 32-bit data when [A24] is low. VMEbus is 16-bit data when [A24] is high. (Factory Configuration)

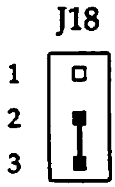
NOTE

Refer to the *Data Bus Structure* and *Accessing the VMEbus* sections in Chapter 4 for an explanation of the MVME133 data bus.

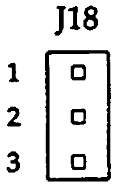
VMEbus Address Size Select Header (J18)



VMEbus is 32-bit address. Onboard DRAM responds only to 32-bit addresses



VMEbus contains both 24-bit and 32-bit address devices. Onboard DRAM responds only to 24-bit address accesses. (Factory Configuration)



VMEbus contains both 24-bit and 32-bit address devices. Onboard DRAM responds only to 32-bit accesses.

NOTE

Refer to the *VMEbus Address Size* section in Chapter 4 for an explanation of the MVME133 address bus and VMEbus memory map.

Cache Disable Test Points (E1, E2)

You may hardware disable the MC68020 onchip cache memory by wire-wrapping test point pins E1 and E2 together. (E1 and E2 are next to J18.) This connects a ground to the CDIS* pin of the MC68020, preventing cache hit. The factory configuration is with E1 and E2 not connected, leaving the cache function under software control.

Installation Instructions

The following paragraphs discuss installation of the MVME133 into a VME chassis, connection of an RS-232C terminal and cable, and system considerations. Ensure that the desired ROM/PROM/EPROM/EEPROM devices (such as those for the MVME133BUG debug monitor in sockets XU31 and XU46) are installed and configured, and that all other headers are configured for desired operation.

Module Installation

When the VME133 is ready for installation, proceed as follows:

a. Turn all equipment power OFF and disconnect power cable from ac power source.

> **CAUTION**
>
> **Connecting modules while power is applied may result in damage to components on the module.**

> **WARNING**
>
> **DANGEROUS VOLTAGES, CAPABLE OF CAUSING DEATH, ARE PRESENT IN THIS EQUIPMENT. USE EXTREME CAUTION WHEN HANDLING, TESTING, AND ADJUSTING.**

b. Remove chassis cover as instructed in the equipment user's manual

c. Remove the filler panel(s from the appropriate card slot(s) at the front of the chassis. Do not install in card slot 1 unless the module is configured for system controller.

d. Carefully slide the MVME133 module into the card slot. Be sure the module is seated properly into connectors on the backplane. Fasten the module in the chassis with the screws provided, making good contact with the transverse mounting rails to minimize RFI emissions.

e. Connect the required cables to the MVME133 at the P2 backplane connector where the module is installed, to mate with peripherals at the RS-232C and/or RS-485 serial ports. These cables are not provided with the MVME133 and must be made or provided by the user. Use shielded cables for all connections to peripherals to minimize radiation. Connect the peripherals to the cables. Install any other required VMEmodules in the system. Refer to the *Multiprotocol Serial Ports* section in Chapter 4.

f. At the MVME133 slot in the backplane, remove the IACKIN*/IACKOUT* jumper and the BGIN*/BGOUT* jumper of the used priority level. Install BGIN*/BGOUT* jumpers on the unused levels.

g. Replace cover (if removed)

h. Turn the equipment power ON.

Terminal Connection

The RS-232C port on the front panel is configured for DCE (to terminal) operation. A 25-pin RS-232C cable may be connected to the front panel female connector J14, with the other end connected to a compatible terminal. This cable is not provided with the MVME133 module, and must be made or provided by the user. Use shielded cables for all connections to peripherals to minimize radiation. J14 has a metal shell and jack posts that are electrically connected to the MVME133 front panel. If the MVME133 front panel is electrically connected to the chassis ground, the shell and jack posts are connected to chassis ground. Refer to Figures 4-9 through 4-12 and Appendix B for detailed information on the signals supported.

NOTE

You may change J14 to a "to modem" configuration by providing a "null-modem" cable that switches certain signals.

System Considerations

The MVME133 may be used by itself or with other VMEbus controllers. The MVME133 is not intended to be used as an Intelligent Peripheral Controller (IPC). It is intended to be used as a VMEbus master.

(MVME133) Whether the MVME133 operates as a VMEbus master or as a VMEbus slave, it is configured for 24 bits of address and 32 bits of data (A24/D32). Many VME chassis are A24/D16 or A32/D32. The MVME133 is configured as a 16-bit data port or as a 32-bit data port depending on address alignment on a longword boundary, and the status of address bit A24. Refer to the theory in Chapter 4. There can also be system problems with bus error (BERR*) and Read-Modify-Write (RMW) cycles. These are also discussed in Chapter 4.

(MVME133) The MVME133 uses the address modifier lines in such a way that it performs short or standard addressing (AM = $2D, $29; $3E, $3D, $3A, or $39,) when it is VMEbus master, but it responds to standard addressing (AM = $3E, $3D, $3A, or $39) when it is a VMEbus slave. Refer to the VMEbus specification for a complete description of all the address modifier codes.

(MVME133A-20) Whether the MVME133A-20 operates as a VMEbus master or as a VMEbus slave, it is configured for 32 or 24 bits of address and 32 or 16 bits of data (A32 or A24/ D32 or D16). Refer to the theory in Chapter 4. There can also be system problems with bus error (BERR*) and Read-Modify-Write (RMW) cycles. These are also discussed in Chapter 4.

(MVME133A-20) The MVME133A-20 uses the address modifier lines in such a way that it performs short, standard, or extended addressing (AM = $2D, $29; $3E, $3D, $3A, $39; $0E, $0D, $0A, or $09) when it is VMEbus master, but it responds to standard or extended addressing (AM = $3E, $3D, $3A, $39; $0E, $0D, $0A, or $09) when it is a VMEbus slave. Refer to the VMEbus specification for a complete description of all the address modifier codes.

The MVME133 contains no parallel ports. You must add a module such as the MVME050 System Controller Module to your system to use a parallel device (such as a printer) with the MVME133.

A single SIP resistor package, R23, with four 120-ohm resistors, is used for proper and reliable system operations with the RS-485 port (port A). In systems where an RS-485 multi-drop cable is used to connect many RS-485 ports (e.g., many MVME133s), noise on the cable may be read as spurious data and/or cause undesired interrupts unless the cable is properly terminated. The recommended method is to terminate each of the two ends of the cable with a 120-ohm resistor. For systems using MVME133, proper termination is accomplished by the existing R23 (eight-pin resistor pack with four 120-ohm resistors) on the two MVME133 modules, one at each end of the RS-485 cable. This termination is also useful when devices connected to the MVME133 RS-485 port are OFF or not online when the RS-485 port is enabled.

If the MVME133 tries to access offboard resources in a non-existent location, and the system does not have a global bus timeout, the MVME133 waits forever for the VMEbus cycle to complete. Local bus timeout on the MVME133 does not terminate a VMEbus cycle. The system might lack this global bus timeout if the MVME133 is system controller but its onboard global bus timeout is disabled (J8 has no jumper), or if the MVME133 is not the system controller, and there is no global bus timeout elsewhere in the system.

CHAPTER 3
OPERATING INSTRUCTIONS

Introduction

This chapter provides the necessary information to use the MVME133 in a system configuration. This includes controls, indicators, and memory map details.

Controls and Indicators

All controls and indicators are located on the front panel. They are:

- ABORT switch
- RESET switch
- FAIL indicator
- RUN indicator

ABORT Switch S1

The ABORT switch is debounced and brought into the interrupt handler as a level 7 interrupt. Refer to the *Interrupt Handler* section in Chapter 4 for details.

RESET Switch S2

The RESET switch resets all onboard devices, including the MPU.

If the MVME133 is the system controller, it drives SYSRESET* low. The MVME133 also drives SYSRESET* low at power up if it is configured as the system controller. Refer to the *Reset* section in Chapter 4 for details.

FAIL (DS1), HALT (DS2), and RUN (DS3) Indicators

There are three LEDs on the MVME133:

- RUN (green) is on when [AS*] is low.
- HALT (red) is on when reset (any reset except the RESET instruction from the MPU) is true or when the [HALT*] line is low.
- FAIL (red) is on when the [BRDFAIL] line is high.

Table 3-1 describes the module status for all possible combinations of these LEDs.

Table 3-1. Front Panel LEDs and Module Status

FAIL **DS1** **Red**	**HALT** **DS2** **Red**	**RUN** **DS3** **Green**	**Status**
OFF	OFF	OFF	No power is applied to the module, or the MPU is not the current local bus master.
OFF	OFF	ON	Normal operation.
OFF	ON	OFF	MPU is halted.
OFF	ON	ON	MPU is running and VMEbus deadlocks or Real-Time Clock (RTC) read cycles are occurring. Frequency of VMEbus deadlocks or RTC reads determines intensity of HALT LED.
ON	OFF	OFF	MPU is not current local bus master. Also, [BRDFAIL] has not been cleared since reset or software has set [BRDFAIL].
ON	OFF	ON	[BRDFAIL] has not been cleared since reset or software has set [BRDFAIL].
ON	ON	OFF	MPU is halted and [BRDFAIL] has not been cleared since reset or software has set [BRDFAIL].
ON	ON	ON	MPU is running and VMEbus deadlocks or RTC read cycles are occurring. Frequency of VMEbus deadlocks or RTC reads determines intensity of HALT LED. Also [BRDFAIL] has not been cleared since reset or software has set [BRDFAIL].

MVME133 Memory Maps and map decoder

At the beginning of each MPU cycle, the map decoder determines the cycle type and which device or function is selected.

Cycle types are determined by the function code lines FC2 through FC0, which are driven by the MC68020. Table 3-2 shows the cycle types and the devices that respond.

Table 3-2. Cycle Types and Responding Devices

FC2	FC1	FC0	Cycle Type	MVME133 Devices that Respond
0	0	0	reserved	None except dynamic RAM (MVME133) None (causes local timeout) (MVME133A-20)
0	0	1	user data	All except interrupt handler and MC68881
0	1	0	User Program	All except interrupt handler and MC68881
0	1	1	reserved	None (causes local timeout)
1	0	0	reserved	None except dynamic RAM (MVME133) None (causes local timeout) (MVME133A-20)
1	0	1	supervisory data	All except interrupt handler and MC68881
1	1	0	supervisory program	All except interrupt handler and MC68881
1	1	1	MPU (IACK)	VMEbus, Z8530, MC68901, interrupt handler
1	1	1	MPU (coprocessor)	MC68881 FPC

Main Memory Map

The memory map for the MVME133 is shown in Table 3-3, and the memory map for the MVME133A-20 is shown in Table 3-4. These memory maps identify the devices that respond in:

- User Data space
- User Program space
- Supervisory Data space
- Supervisory Program space

Table 3-3. MVME133 Main Memory Map

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
XX000000 XX000007	Onboard ROM for first four memory cycles after reset. Onboard dynamic RAM thereafter.				1
XX000008 XX0FFFFF	Onboard dynamic RAM (DRAM)				1
XX100000 XXEFFFFF	VMEbus				1
XXF00000 XXF1FFFF	Onboard bank 1 ROM/PROM/EPROM/ EEPROM				1,3
XXF20000 XXF3FFFF	Onboard bank 2 ROM/PROM/EPROM/ EEPROM				1
XXF40000 XXF5FFFF	Onboard bank 1 repeats in this space.				1
XXF60000 XXF7FFFF	Onboard bank 2 repeats in this space.				1
XXF80000	MSR	MFP GPIP			1,2
XXF80002	MSR	MFP AER			1,2
XXF80004	MSR	MFP DDR			1,2
XXF80006	MSR	MFP IERA			1,2
XXF80008	MSR	MFP IERB			1,2
XXF8000A	MSR	MFP IPRA			1,2
XXF8000C	MSR	MFP IPRB			1,2
XXF8000E	MSR	MFP ISRA			1,2
XXF80010	MSR	MFP ISRB			1,2
XXF80012	MSR	MFP IMRA			1,2
XXF80014	MSR	MFP IMRB			1,2
XXF80016	MSR	MFP VR			1,2
XXF80018	MSR	MFP TACR			1,2
XXF8001A	MSR	MFP TBCR			1,2

Table 3-3. MVME133 Main Memory Map (cont'd)

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
XXF8001C	MSR	MFP TCDCR			1,2
XXF8001E	MSR	MFP TADR			1,2
XXF80020	MSR	MFP TBDR			1,2
XXF80022	MSR	MFP TCDR			1,2
XXF80024	MSR	MFP TDDR			1,2
XXF80026	MSR	MFP SCR			1,2
XXF80028	MSR	MFP UCR			1,2
XXF8002A	MSR	MFP RSR			1,2
XXF8002C	MSR	MFP TSR			1,2
XXF8002E	MSR	MFP UDR			1,2
XXF80030 XXF9FFFF	The MSR and MFP registers appear repeatedly in this space.				1
XXFA0000	SIOB RR0 SIOB WR0				1
XXFA0001	SIOB RXDATA SIOB TXDATA				1
XXFA0002	SIOA RR0 SIOA WR0				1
XXFA0003	SIOA RXDATA SIOA TXDATA				1
XXFA0004 XXFAFFFF	The above SIO registers appear repeatedly in this space.				1

Table 3-3. MVME133 Main Memory Map (cont'd)

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
XXFB0000	UUUU RTC00				1,4
XXFB0001	UUUU RTC01				1,4
XXFB0002	UUUU RTC02				1,4
XXFB0003	UUUU RTC03				1,4
XXFB0004	UUUU RTC04				1,4
XXFB0005	UUUU RTC05				1,4
XXFB0006	UUUU RTC06				1,4
XXFB0007	UUUU RTC07				1,4
XXFB0008	UUUU RTC08				1,4
XXFB0009	UUUU RTC09				1,4
XXFB000A	UUUU RTC10				1,4
XXFB000B	UUUU RTC11				1,4
XXFB000C	UUUU RTC12				1,4
XXFB000D	UUUU RTC13				1,4
XXFB000E	UUUU RTC14				1,4

Table 3-3. MVME133 Main Memory Map (cont'd)

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
XXFB000F	UUUU RTC15				1,4
XXFB0010 XXFB7FFF	The above RTC registers appear repeatedly in this space.				3
XXFB8000 XXFBFFFF	VMEbus interrupt				1
XXFC0000 XXFEFFFF	VMEbus extended address range				1
XXFF0000 XXFFFFFF	VMEbus short I/O space				1

NOTES:

1. XX denotes "don't care". However, when [A24] is zero, the VMEbus is treated as a 32-bit data port for longword aligned transfers and when [A24] is one, the VMEbus is treated as a 16-bit data port for all transfers. Refer to the *VMEbus Master Interface and Address Buffers* section and Table 4-2 in Chapter 4. Also note that the Real-Time Clock (RTC) with [A25] = 1 causes the VMEbus interrupter to activate a level 3 interrupt. Reading the RTC with [A25] = 0 has no effect on the interrupter.

2. The Module Status Register (MSR) is read only. Do not write to the MSR. Although write accesses are ignored by the MSR, they affect the MC68901 Multi-Function Peripheral (MFP). The MSR is connected to D24-D31 and the MFP is connected to D16-D23.

3. Writes to EEPROMs must always be 16-bit wide.

4. UUUU denotes four undefined bits. RTC = Real-Time Clock.

Table 3-4. MVME133A-20 Main Memory Map

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
00000000 00000007	Onboard ROM for first four memory cycles after reset. Onboard dynamic RAM thereafter.				5
00000008 000FFFFF	Onboard dynamic RAM (DRAM)				5
00100000 00EFFFFF	VMEbus standard address range (A32/A24 system) VMEbus extended address range (A32 system)				5
00F00000 FFEFFFFF	VMEbus extended address range				5
FFF00000 FFF1FFFF	Onboard bank 1 ROM/PROM/EPROM/ EEPROM				1
FFF20000 FFF3FFFF	Onboard bank 2 ROM/PROM/EPROM/ EEPROM				1
FFF40000 FFF5FFFF	Onboard bank 1 repeats in this space.				1
FFF60000 FFF7FFFF	Onboard bank 2 repeats in this space.				1
FFF80000	MSR	MFP GPIP			2
FFF80002	MSR	MFP AER			2
FFF80004	MSR	MFP DDR			2
FFF80006	MSR	MFP IERA			2
FFF80008	MSR	MFP IERB			2
FFF8000A	MSR	MFP IPRA			2
FFF8000C	MSR	MFP IPRB			2
FFF8000E	MSR	MFP ISRA			2
FFF80010	MSR	MFP ISRB			2

Table 3-4. MVME133A-20 Main Memory Map (cont'd)

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
FFF80012	MSR	MFP IMRA			2
FFF80014	MSR	MFP IMRB			2
FFF80016	MSR	MFP VR			2
FFF80018	MSR	MFP TACR			2
FFF8001A	MSR	MFP TBCR			2
FFF8001C	MSR	MFP TCDCR			2
FFF8001E	MSR	MFP TADR			2
FFF80020	MSR	MFP TBDR			2
FFF80022	MSR	MFP TCDR			2
FFF80024	MSR	MFP TDDR			2
FFF80026	MSR	MFP SCR			2
FFF80028	MSR	MFP UCR			2
FFF8002A	MSR	MFP RSR			2
FFF8002C	MSR	MFP TSR			2
FFF8002E	MSR	MFP UDR			2
FFF80030 FFF9FFFF	The MSR and MFP registers appear repeatedly in this space.				2
FFFA0000	SIOB RR0 SIOB WR0				6
FFFA0001	SIOB RXDATA SIOB TXDATA				6
FFFA0002	SIOA RR0 SIOA WR0				6
FFFA0003	SIOA RXDATA SIOA TXDATA				6

Table 3-4. MVME133A-20 Main Memory Map (cont'd)

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
FFFA0004 FFFAFFFF	The above SIO registers appear repeatedly in this space.				6
FFFB0000	UUUU RTC00				3
FFFB0001	UUUU RTC01				3
FFFB0002	UUUU RTC02				3
FFFB0003	UUUU RTC03				3
FFFB0004	UUUU RTC04				3
FFFB0005	UUUU RTC05				3
FFFB0006	UUUU RTC06				3
FFFB0007	UUUU RTC07				3
FFFB0008	UUUU RTC08				3
FFFB0009	UUUU RTC09				3
FFFB000A	UUUU RTC10				3
FFFB000B	UUUU RTC11				3
FFFB000C	UUUU RTC12				3

Table 3-4. MVME133A-20 Main Memory Map (cont'd)

Address Range	D31 - D24	D23 - D16	D15 - D08	D07 - D00	Notes
FFFB000D	UUUU RTC13				3
FFFB000E	UUUU RTC14				3
FFFB000F	UUUU RTC15				3
FFFB0010 FFFB7FFF	The above RTC registers appear repeatedly in this space.				3
FFFB8000 FFFBFFFF	VMEbus interrupt				4
FFFC0000 FFFEFFFF	VMEbus extended address range				5
FFFF0000 FFFFFFFF	VMEbus short I/O space				5

NOTES:

1. Writes to EEPROMs must always be 16-bit wide.
2. The Module Status Register (MSR) is read only. Do not write to the MSR. Although write accesses are ignored by the MSR, they affect the MC68901 Multi-Function Peripheral (MFP). The MSR is connected to D24-D31 and the MFP is connected to D16-D23.
3. The MM58274 Real-Time Clock (RTC) is only 4-bit wide but appears as an 8-bit port to the MPU. The four most significant bits are undefined and are shown in this table as UUUU.
4. RTC registers also appear repeatedly in this space. To generate a VMEbus interrupt on level 3, perform a read access to any address within this space.
5. Refer to the *VMEbus Master Interface and Address Buffers* section and Table 4-2 in Chapter 4. for discussion of VMEbus address and data width.

6. The Z8530 Serial Communications Controller is listed in this table as the SIO.

MPU Space Memory Map Assignments

Coprocessor and Interrupt Acknowledge (IACK) are the only types of MPU cycles (FC2 through FC0 = %111) supported by the MVME133.

Refer to Table 3-2. All other types of MPU space cycles generated by the MPU are ignored and cause local bus timeout on the MVME133. Among the other types of MPU space cycles which the MC68020 is capable of generating but which the MVME133 does not support are those using breakpoint acknowledge, access level control, or MOVES instructions.

Coprocessor Interface Register Map

The only coprocessor on the MVME133 is the MC68881 Floating Point Coprocessor (FPC). The map decoder selects the MC68881 whenever the MPU executes a coprocessor cycle (FC2-FC0 = %111 and [A19]-[A16] = %0010).

The recommended Coprocessor ID (Cp-ID) for the MC68881 is binary %001. However, the MVME133 selects the MC68881 regardless of the Cp-ID. The Cp-ID is bits 9 through 11 of the coprocessor word.

The MC68881 registers are addressed by the [A04]-[A01] as shown in Table 3-5.

Table 3-5. MC68881 Floating Point Coprocessor (FPC) Interface Register Map

[A04]-[A00]	**Offset**	**MC68881 Register**	
(Binary)	(Hex)	D31 D16	D15 D00
%0000X	$00	Response (Read Only)	
%0001X	$02	Control (Write Only)	
%0010X	$04	Save (Read Only)	
%0011X	$06	Restore (Read/Write)	
%0100X	$08	(Reserved)	
%0101X	$0A	Command (Write Only)	
%0110X	$0C	(Reserved)	
%0111X	$0E	Condition (Write Only)	
%100XX	$10	Operand (Read/Write)	
%1010X	$14	Register Select (Read Only)	
%1011X	$16	(Reserved)	
%110XX	$18	Instruction Address (Write Only)	
%111XX	$1C	Operand Address (Read/Write)	

NOTES: 1. Write accesses to reserved or read-only registers are ignored, but read accesses to reserved or write-only registers return all ones.

2. X means don't care.

Shared DRAM Address Memory Map.

(MVME133 only) You select the VMEbus address of the onboard shared DRAM by using jumpers on J2 and/or by reprogramming U22, PALDP. J2 selects which of the four programmable addresses in U22 is the shared memory address. Refer to the *Onboard RAM Base Address Select Header (J2)* section in Chapter 2. When U22 contains the default factory program, J2 selects the base addresses given in Table 3-6.

Table 3-6. MVME133 Shared Onboard RAM Memory Map

J2 Connections	Shared Memory Access
1-2 and 3-4	$000000 - $0FFFFF
1-2	$100000 - $1FFFFF
3-4	$200000 - $2FFFFF
none	$300000 - $3FFFFF

(MVME133) The onboard RAM responds to the VMEbus only when AM0 through AM5 indicate standard, privileged, or non-privileged data or program space, and when the MVME133 is not the VMEbus master.

(MVME133A-20) You select the VMEbus address of the onboard shared DRAM by using jumpers on J2 and/or reprogramming U82. U82 selects one 16Mb block within the 4Gb range of the MVME133A-20. The default factory program for U82 puts the base address of this 16Mb block at $00000000. J2 then selects one of 16 address spaces within this 16-Mb block for the 1 Mb of onboard shared DRAM. Refer to the *Onboard RAM Base Address Select Header* section in Chapter 2. When U82 contains the default factory program, J2 selects the offset addresses shown in Table 3-7. Refer to Appendix A for U82 program details.

Table 3-7. MVME133A-20 Shared Onboard DRAM Memory Map

J2 Connections	Shared Memory Address
1-2, 3-4, 5-6, 7-8	$00000000 - $000FFFFF
1-2, 3-4, 5-6	$00100000 - $001FFFFF
1-2, 3-4, 7-8	$00200000 - $002FFFFF
1-2, 3-4	$00300000 - $003FFFFF
1-2, 5-6, 7-8	$00400000 - $004FFFFF
1-2, 5-6	$00500000 - $005FFFFF
1-2, 7-8	$00600000 - $006FFFFF
1-2	$00700000 - $007FFFFF
3-4, 5-6, 7-8	$00800000 - $008FFFFF
3-4, 5-6	$00900000 - $009FFFFF
3-4, 7-8	$00A00000 - $00AFFFFF
3-4	$00B00000 - $00BFFFFF
5-6, 7-8	$00C00000 - $00CFFFFF
5-6	$00D00000 - $00DFFFFF
7-8	$00E00000 - $00EFFFFF
none	$00F00000 - $00FFFFFF

(MVME133A-20) You select the onboard DRAM to respond to either 24-bit or 32-bit address acesses by the VMebus. J18 defines the address size of the system (refer to the *VMEbus Address Size Select Header* section in Chapter 2). The onboard DRAM only responds to extended addresses in a 32-bit system. The onboard DRAM only responds to the VMEbus accesses when the addresses match and the address modifiers (AM0-AM5) indicate privileged or non-privileged data or program space, and when the MVME133A-20 is not the current VMEbus master.

Interrupt Acknowledge Map

The MC68020 distinguishes Interrupt Acknowledge (IACK) cycles (FC2 through FC0 = %111) from other MPU space cycles by placing the binary value %1111 on [A19] through [A16]. It also specifies the level that is being acknowledged by using [A03] through [A01]. The interrupt handler selects which device within the level is being acknowledged. Refer to the *Interrupt Handler* section in Chapter 4 for further details.

CHAPTER 4
FUNCTIONAL DESCRIPTION

Introduction

This chapter provides the block diagram level description for the MVME133. The general description provides an overview of the MVME133, followed by a detailed description of each section. Figure 4-1 shows the simplified block diagram of the MVME133.

General Description

The MVME133 is a VMEbus MPU module. The MVME133 has an MC68020 MPU, 1Mb of dynamic RAM (accessible from the VMEbus), a real-time clock, a serial debug port, two multiprotocol serial ports (one with RS-232C interface and one with RS-485 interface), three 8-bit timers, four ROM sockets, one A32/D32 VMEbus interface, one simple single-level VMEbus interrupter, one seven-level VMEbus interrupt handler, and the VMEbus system controller functions

Data Bus Structure

The data bus structure on the MVME133 is arranged to accommodate the 8-bit, 16-bit, 32-bit, and 16/32-bit ports that reside on the module. Figure 4-2 shows the data bus structure of the MVME133.

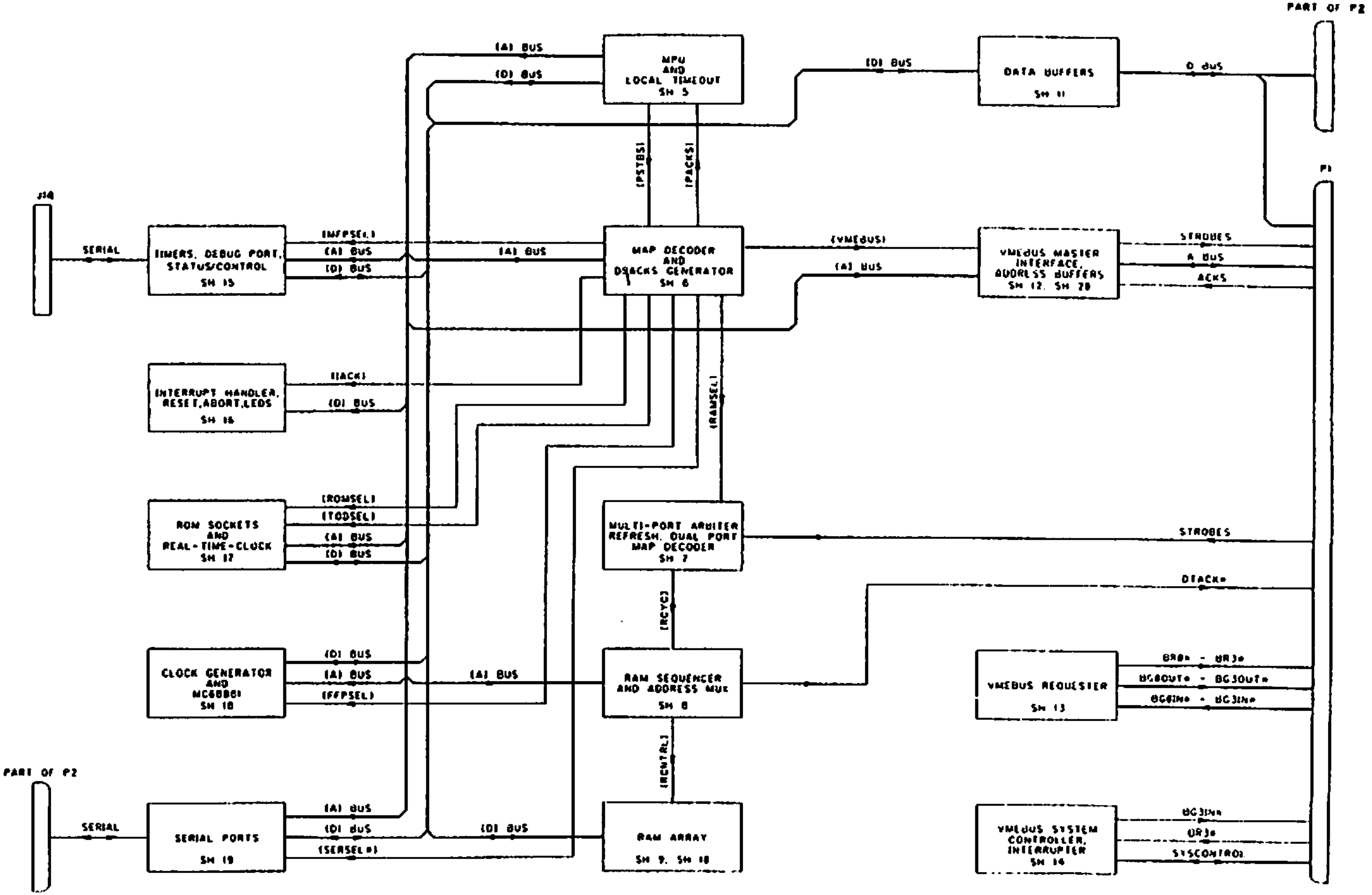

PART OF P2
P1
D BUS
DATA BUFFERS SH 11
(D) BUS
MPU AND LOCAL TIMEOUT SH 5
(A) BUS
(D) BUS
(PSTBS)
(PACKS)
STROBES
A BUS
ACKS
VMEBUS MASTER INTERFACE, ADDRESS BUFFERS SH 12, SH 28
(VMEBUS)
(A) BUS
MAP DECODER AND DSACKS GENERATOR SH 6
J14
SERIAL
TIMERS, DEBUG PORT, STATUS/CONTROL SH 15
(MFPSEL)
(A) BUS
(D) BUS
INTERRUPT HANDLER, RESET, ABORT, LEDS SH 16
(IACK)
(D) BUS
(RAMSEL)
ROM SOCKETS AND REAL-TIME-CLOCK SH 17
(ROMSEL)
(TODSEL)
(A) BUS
(D) BUS
MULTI-PORT ARBITER REFRESH, DUAL PORT MAP DECODER SH 7
STROBES
DTACK*
(RCYC)
CLOCK GENERATOR AND MC68881 SH 18
(D) BUS
(A) BUS
(FFPSEL)
RAM SEQUENCER AND ADDRESS MUX SH 8
VMEBUS REQUESTER SH 13
BR0* - BR3*
BG0OUT* - BG3OUT*
BG0IN* - BG3IN*
(RCNTRL)
PART OF P2
SERIAL
SERIAL PORTS SH 19
(A) BUS
(D) BUS
(SERSEL#)
RAM ARRAY SH 9, SH 18
VMEBUS SYSTEM CONTROLLER, INTERRUPTER SH 14
BG3IN*
BR3*
SYSCONTROL

Figure 4-1. Block Diagram

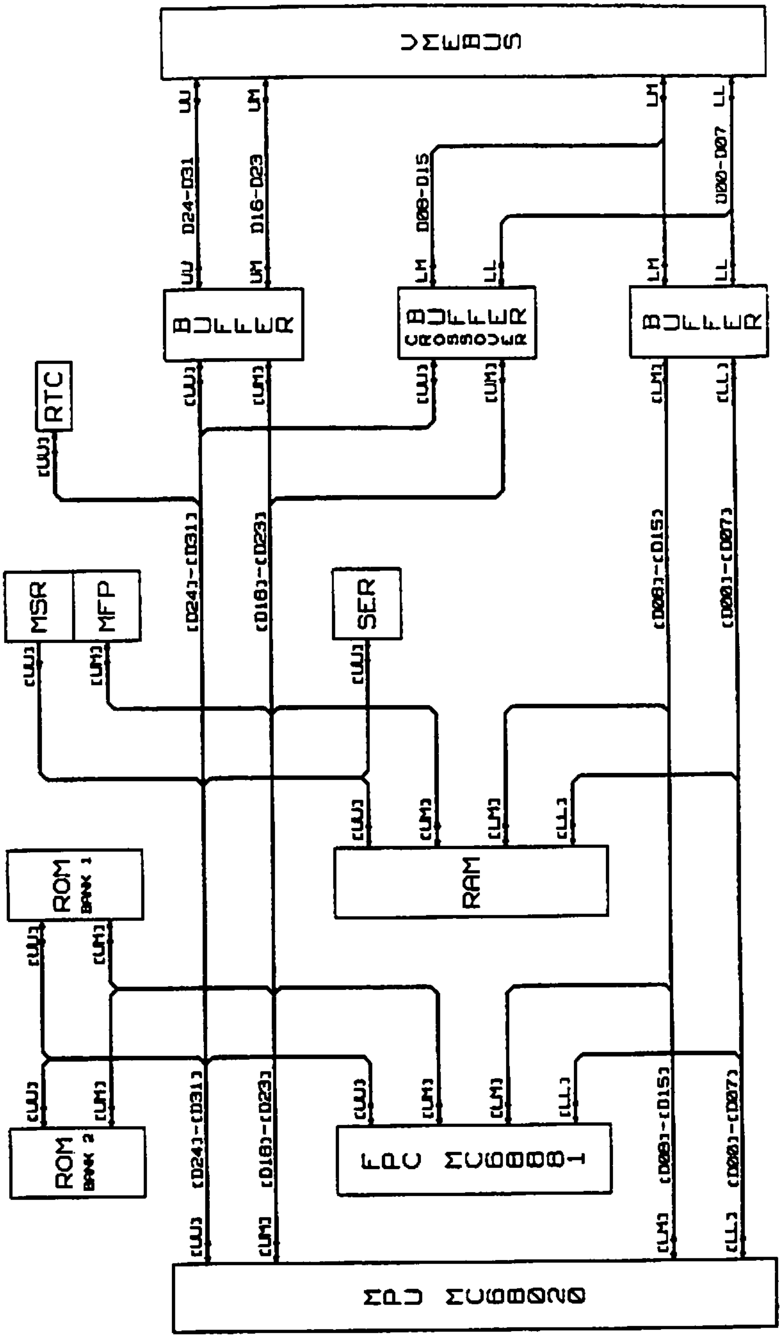


Figure 4-2. Data Bus Structure

Memory Map

The operation of the map decoder and a detailed discussion of the various memory maps in the MVME133 are given in Chapter 3. Included are descriptions of the main memory map, coprocessor interface register map, and shared memory map.

Module Timing

Table 4-1 and the following paragraphs give general characteristics of the MVME133 module timing.

Table 4-1. Timing

Type of Access	MVME133		Notes
	Read	**Write**	
Onboard MPU to MC68881	3 cycles	3 cycles	1,2
MPU to local ROM/PROM/EPROM/ EEPROM	6 cycles	6 cycles	1,8
MPU to onboard DRAM	4 cycles	4 cycles	1,3
Onboard MPU to VMEbus	4+N cycles	5+N cycles	4,5
VMEbus to onboard DRAM	500 ns	500 ns	6,7

NOTES:
1. No arbitration overhead.
2. Except for read accesses to Response or Save Coprocessor Interface Register (CIR) which take 5 MPU clock cycles.
3. Except for RMW cycles where MVME133 must obtain VMEbus mastership before any RAM cycle may be started.
4. Assume MVME133 is the current VMEbus master.
5. N = (DS0*/DS1* to DTACK* time)/(MPU clock cycle time). N should be rounded up to the nearest integer.
6. DS0*/DS1* activated to DTACK* time.
7. Typical values. Actual values may be greater or less depending on the state of the MVME133.
8. Device access times must be as specified in the *ROM/PROM/ EPROM/EEPROM* section in this chapter, and Figure 4-5.

RAM Cycle Times

MC68020 accesses to onboard RAM require four MPU clock cycles (three minimum + one wait cycle). The clock frequencies are:

- 12.5 MHz (MVME133)
- 16.67 MHz (MVME133-1)
- 20 MHz (MVME133A-20)

MC68020 RMW cycles to onboard RAM can require more than four MPU clock cycles if the MVME133 does not already have the VMEbus mastership and J5 pins 1 to 2 are connected. Refer to the *MPU, Clocks, and Local Timeout* section in this chapter for more details on local accesses.

VMEbus Access Time to Onboard RAM

The onboard RAM access time from the VMEbus (assertion of DS0*/DS1* to assertion of DTACK*) is typically:

- 720 ns for 12.5 MHz modules (MVME133)
- 540 ns for 16.67 MHz modules (MVME133-1
- 500 ns for 20 MHz modules (MVME133A-20).

ROM/PROM/EPROM/EEPROM Cycle Times

All ROM/PROM/EPROM/EEPROM accesses require six MPU clock cycles (three minimum + three wait cycles) to complete.

MVME133 VMEbus Cycle Times

The terms used in this section are:

N Total number of wait cycles incurred. Must always be rounded up to the next integer.

T MPU clock period.

ta Time in nanoseconds from assertion of DS0*/DS1* to the assertion of DTACK*.

tg Delay from assertion of BRX* (driven by the MVME133) to assertion of BGXIN* and negation of AS*

tr Delay from assertion of BR3* (driven by MVME133) to negation of BBSY* and AS*.

The following formula assumes the MVME133 is the current VMEbus master, and all slaves have released DTACK* and BERR*.

For read accesses	$N = 1 + [ta / T]$
For write accesses	$N = 2 + [ta / T]$

The following formula assumes the MVME133 is not the current VMEbus master, the MVME133 is the system controller, and all previous slaves have released DTACK* and/or BERR* when the MVME133 receives VMEbus mastership.

For read accesses	$N = 1 + [(ta + tr + 170\text{ ns}) / T]$	typical
For write accesses	$N = 3 + [(ta + tr + 170\text{ ns}) / T]$	typical

The following formula assumes the MVME133 is not the current VMEbus master, the MVME133 is not the system controller, and all previous slaves have released DTACK* and/or BERR* when the MVME133 receives VMEbus mastership.

For read accesses	$N = 1 + [(ta + tg + 150\text{ ns}) / T]$	typical
For write accesses	$N = 3 + [(ta + tg + 150\text{ ns}) / T]$	typical

MVME133A-20 VMEbus Cycle Times

The terms used in this section are:

N Total number of wait cycles required to complete a VMEbus cycle. Must always be rounded up to the next integer.

tac Time in nanoseconds from assertion of DS0*/DS1* to the assertion of DTACK*.

tg Delay from assertion of BRX* (driven by the MVME133A-20) to assertion of BGXIN* low and negation of AS*.

tr Delay from assertion of BR3* (driven by MVME133A-20) to negation of BBSY* and AS*.

The following formula assumes the MVME133A-20 is the current VMEbus master, and all slaves have released DTACK* and BERR*.

For read accesses	N = 4 + [tac / 50 ns]	typical
For write accesses	N = 5 + [tac / 50 ns]	typical

The following formula assumes the MVME133A-20 is not the current VMEbus master, the MVME133A-20 is the system controller, and all previous slaves have released DTACK* and/or BERR* when the MVME133A-20 receives VMEbus mastership.

For read accesses	N = 4 + [(tac + tr + 120 ns) / 50 ns]	typical
For write accesses	N = 6 + [(tac + tr + 120 ns) / 50 ns]	typical

The following formula assumes the MVME133A-20 is not the current VMEbus master, the MVME133A-20 is not the system controller, and all previous slaves have released DTACK* and/or BERR* when the MVME133A-20 receives VMEbus mastership.

For read accesses	N = 4 + [(tac + tg + 100 ns) / 50 ns]	typical
For write accesses	N = 6 + [(tac + tg + 100 ns) / 50 ns]	typical

MVME133 VMEbus Arbitration Time

When the MVME133 is configured as the system controller and the MVME133 is not requesting VMEbus mastership, the delay from negation of BBSY* and assertion of BR3* to assertion of BG3OUT* is 150 ns typical and 230 ns maximum

When the MVME133 is not configured as the system controller and the MVME133 is not requesting VMEbus mastership, the delay from assertion of BGXIN* to assertion of BGXOUT* is 130 ns typical and 180 ns maximum

MVME133A-20 VMEbus Arbitration Time

When the MVME133A-20 is configured as the system controller and the MVME133A-20 is not requesting VMEbus mastership, the delay from negation of BBSY* and assertion of BR3* to assertion of BG3OUT* is 120 ns typical and 170 ns maximum

When the MVME133A-20 is not configured as the system controller and the MVME133A-20 is not requesting VMEbus mastership, the delay from assertion of BGXIN* to assertion of BGXOUT* is 100 ns typical and 140 ns maximum

System Considerations

The following three sections discuss the system considerations.

MVME133 Memory Sizing

The MVME133 accesses the VMEbus as a longword port (32-bit data) when the MPU executes a longword aligned cycle with [A24] = 0, and as a word port (16-bit data) when the MPU executes a cycle other than longword aligned with [A24] = 0.

This has impact on memory sizing routines. To size the longword portion of VMEbus memory, the sizing routine should use longword aligned transfers with [A24] = 0. To size the word portion of VMEbus memory, the sizing routine should use something other than longword aligned transfers with [A24] = 0. To isolate the word only portion of the VMEbus memory, The sizing routine should subtract the portion that responds to longword aligned cycles from the portion that doesn't respond to longword aligned cycles.

For example, if $100000 through $400000 respond to longword aligned accesses and $100000 through $7FFFFE respond to word accesses, locations $500000 through $7FFFFE are word only locations and should be accessed as bytes or words only, or they should be accessed in the range $1500000 through $17FFFFF; i.e., with [A24] = 1.

Sources of BERR*

There are three sources of bus error exceptions on the MVME133. They are Local Bus Timeout [LTO], Qualified VMEbus BERR* [QVBERR*], and RMW-LOCK.

[LTO] occurs whenever an MPU access does not complete within 13 ms to 15 ms (at 20 MHz operation). If the system is configured properly, this should only happen if software accesses a non-existent location within the onboard range or if something prevents the MVME133 from becoming the VMEbus master.

[QVBERR*] occurs when the BERR* signal line is asserted on the VMEbus while the MVME133 is the VMEbus master. VMEbus BERR* should occur only if an initialization routine samples a non-existent device to determine its presence, or if software accesses a non-existent device within the VMEbus range, or tries to access a device on the VMEbus incorrectly (such as asserting LWORD* to a 16-bit module), or if a hardware error occurs on the VMEbus, or if a VMEbus slave reports an access error (such as parity error).

RMW-LOCK occurs when there is a VMEbus deadlock during an MPU RMW cycle. As noted in the *VMEbus Accesses* section in this chapter, whenever a VMEbus deadlock occurs, the multiport arbiter breaks the lock by asserting [BERR*] and [HALT*] at the same time. This sequence indicates to the MPU that it should abort the current cycle. When the MPU aborts the current cycle, it relinquishes local bus mastership to the VMEbus, and the VMEbus in turn executes a RAM cycle. If the MC68020 is executing an RMW cycle when the VMEbus deadlock occurs, it does not relinquish local bus mastership until it completes all portions of the RMW cycle. The RMW cycle cannot complete until the MVME133 obtains VMEbus mastership, and the MVME133 cannot obtain VMEbus mastership because another VMEbus module is master and is waiting at the MVME133 shared memory. This lockup condition is called RMW-LOCK.

When the multiport arbiter detects a VMEbus deadlock condition and [RMC*] from the MPU is asserted, it asserts [BERR*] without asserting [HALT*] to force the onboard MPU to relinquish the local bus, thus breaking the RMW-LOCK condition, but causing a bus error exception.

If the system is constructed so that RMW cycles is never be generated to the VMEbus range, and that no multiple address RMW cycles can happen to the onboard RAM from the VMEbus side, header J5 can be jumpered so that RMW cycles to onboard RAM do not require VMEbus mastership. This removes RMW-LOCK bus errors.

Because different conditions can cause bus error exceptions, the software must be able to distinguish the source. The MVME133 provides two status bits in the MC68901 MFP to help the software determine the bus error exception source. The status bits are [LTO] and [QVBERR*]. RMW-LOCK is not provided as a status bit, but software can detect it if a bus error exception occurs and neither [LTO] nor [QVBERR*] is set.

Generally, the bus error handler can interrogate [LTO] and [QVBERR*] and proceed with the result. However, an interrupt can happen during the execution of the bus error handler before an instruction can write to the status register to raise the interrupt mask. If the interrupt service routine causes a second bus error, the status that indicates the source of the first bus error may be lost. The software must be aware of this and must be written to deal with it.

Use of RMW Instructions

The MC68020 RMW Instructions are TAS, CAS, and CAS2. These instructions cause indivisible cycle sequences to occur on the MC68020 local bus. TAS and single address CAS perform one read and then one write to the same address. Multiple address CAS and CAS2 perform reads and writes to multiple addresses. The VMEbus defines single address indivisible cycles as READ-MODIFY-WRITE cycles. The VMEbus does not define multiple address indivisible cycles. A scheme has been devised to allow indivisible multiple address cycles on the VMEbus. It is not part of the VMEbus specification. It is implemented on the MVME133 when J5 pins 1-2 are connected. The scheme has the following rules:

1. Locations that are accessed by multiple address indivisible cycles are called Multiple Address Interlock (MAI) locations.
2. All devices that access MAI locations must use indivisible cycle instructions; i.e., TAS, CAS, or CAS2 of MC68020.
3. Any device that executes an indivisible cycle instruction must obtain VMEbus mastership before executing the first cycle of the instruction, and retain VMEbus mastership until it has completed the last cycle of the instruction.

Rule number 1 is a definition, rule number 2 is a software requirement, and rule number 3 is taken care of automatically by the MVME133 requester if J5 pins 1-2 are connected.

The MVME133 does not support the above scheme when J5 pins 1-2 are not connected. In this configuration, the MVME133 does not obtain VMEbus mastership before executing indivisible cycle instructions. J5 pins 1-2 must only be disconnected if the software never executes RMW cycles within the VMEbus range. The advantage of using this jumper option is that, when it is used properly, RMW-LOCKs never occur.

Detailed Description

The following paragraphs describe in detail the theory of operation for the MVME133. During this discussion, sheets referenced belong to the schematic diagram. Refer to *SIMVME133 Support Information.*

MPU, Clocks, and Local Timeout (Sheet 5)

The devices in the modules are:

- The MVME133 has a 12.5 MHz MC68020
- The MVME133-1 has a 16.67 Mhz MC68020
- The MVME133A-20 has a 20 MHz MC68020.

The MC68020 is a full 32-bit processor with 32-bit registers, 32-bit data, and 32-bit addresses. Its advanced architecture, enhanced addressing modes, and on-chip cache are advancements over its predecessors in the MC68000 family of chips.

The frequency of master crystal oscillator Y3 is:

- 25 MHz (MVME133)
- 33.33 MHz (MVME133-1)
- 40 MHz (MVME133A-20)

The local bus timeout generator aborts any cycle that does not complete within:

- 21 to 24 ms (MVME133)
- 16 to 18 ms (MVME133-1)
- 13 to 15 ms (MVME133A-20)

Abort is accomplished by asserting [BERR*] to the MPU. Refer to the *Sources of BERR** section in this chapter for details.

Map Decoder and DSACKs Generator (Sheet 6)

The operation of the map decoder and a detailed discussion of the various memory maps in the MVME133 are given in Chapter 3. This includes the main memory map, coprocessor interface register map, and shared memory map. The MVME133 series also include the [A24] and [A25] functions.

Multiport Arbiter, Refresh, and Shared Memory Map Decoder (Sheet 7)

The 1Mb of onboard dynamic RAM is accessible by the onboard MPU, the refresh circuitry, and the VMEbus. Each of these devices requests and is granted the RAM by the multiport arbiter.

Because the local address and data busses are used to access the onboard dynamic RAM, any device that uses the RAM must first become the local bus master. The

MPU arbitration logic (BR*, BG*, BGACK*) is utilized by the multiport arbiter to transfer local bus mastership from the current master to the next requester.

During normal operation, the MPU is the local bus master. When either the VMEbus or the refresh circuitry requests use of the RAM, the multiport arbiter asserts [BR*] to the MPU. The MPU responds by asserting [BG*], finishing its current cycle if one is in progress, and then giving up local bus mastership. The multiport arbiter asserts [BGACK*], negates [BR*], and then grants local bus mastership to the highest priority requesting device (refresh has higher priority than VMEbus). The granted device executes one RAM cycle (read, write, or refresh), and then Relinquishes local bus mastership. If another device is requesting local bus mastership, the multiport arbiter grants it to that device. Otherwise, it negates [BGACK*] and the MPU resumes local bus mastership.

Local Accesses

The DRAM array appears as a 32-bit port to the onboard MPU. MPU to DRAM accesses complete in four clock cycles (3 + 1 wait). Once the MPU starts a RAM cycle, it does not normally wait for the multiport arbiter because it would already be the local bus master. There is one exception. When the MPU begins a Read-Modify-Write (RMW) cycle to onboard RAM and J5 pins 1 to 2 are connected, the RAM sequencer requires the VMEbus requester to obtain VMEbus mastership before the first access of the RMW cycle can occur. The *Use of RMW Instruction* section in this chapter has further details on using RMW instructions.

VMEbus Accesses

When DS0* and/or DS1* are asserted on the VMEbus, the MVME133 shared memory (VMEbus slave) map decoder looks at the value of the address lines and the address modifiers. If they are in the selected shared memory address range and if the MVME133 is not the current VMEbus master, the shared memory map decoder requests the multiport arbiter for local bus mastership. (Refer to Chapter 3 for details of the shared memory map.) The RAM read or write cycle occurs when the multiport arbiter has granted local bus mastership. When the RAM sequencer asserts the DTACK* signal on the VMEbus, the multiport arbiter grants local bus mastership to the refresh circuitry if a refresh request is pending. If a refresh is not pending, the multiport arbiter returns the local bus mastership to the MPU at the end of the VMEbus shared memory cycle. If the VMEbus master is executing a RMW cycle to the RAM, the multiport arbiter does not restore local bus mastership to the MPU until both the read and write cycles are completed. If the MPU is the current local bus master and is executing a cycle that requires the VMEbus when the shared memory map decoder requests local bus mastership, a

VMEbus deadlock condition occurs. To break this VMEbus deadlock condition, the multiport arbiter signals a retry to the MPU by asserting both BERR* and HALT*. The MPU responds by aborting the current cycle, then relinquishing local bus mastership so that the multiport arbiter can grant it to the VMEbus. When the VMEbus has finished with the RAM, the multiport arbiter returns local bus mastership to the MPU. The MPU then retries the aborted cycle.

However, if the MPU is executing an RMW cycle, it does not give up the local bus mastership. Instead of indicating a retry to the MPU, the multiport arbiter must assert [BERR*] to break the deadlock condition. This creates some software implications which are covered in *Sources of BERR** section in this chapter.

The RAM appears to the VMEbus as a 16-bit port for transfers with LWORD* negated, and as a 32-bit port for transfers with LWORD* asserted. The MVME133 supports misaligned transfers between the local RAM and the VMEbus.

Refresh

The dynamic RAMs must have each of their 256 rows refreshed once every 4 ms. To accomplish this, the refresh timer requests the multiport arbiter for a RAM refresh cycle every:

- 15 μs (MVME133)
- 10 μs (MVME133A-20).

When the multiport arbiter issues a refresh grant, the RAM sequencer performs a CAS-before-RAS refresh cycle to the RAM.

Normally, when any device requests the use of the RAM while the MPU is the current local bus master, the multiport arbiter waits for the MPU to relinquish bus mastership before giving a grant to the new device. The multiport arbiter may, however, issue a refresh grant without waiting for the MPU to release the local bus if the arbiter detects a refresh request while the MPU is executing a cycle to devices other than onboard RAM or the MPU is waiting for VMEbus mastership before RMW to the local RAM. This allows the RAMs to be properly refreshed during long periods when the MVME133 is requesting VMEbus mastership, or when it is accessing very slow or non-existent devices.

RAM Sequencer and Address Multiplexers (Sheets 8 and 20)

The RAM sequencer generates RAS and CAS timing and multiplexes rows and columns for RAM. MPU accesses to onboard dynamic RAM take four MPU clock periods. Refer to the *Local Access* and *VMEbus Access* sections in this chapter. VMEbus access time is typically 700 ns for the MVME133, 540 ns for the MVME133-1, and 500 ns for the MVME133A-20.

DRAM Array (Sheets 9 and 10)

The onboard dynamic RAM (DRAM) uses thirty-two 256K x 1 dynamic RAM ZIPs (zigzag-inline-packages), making a total of 1Mb of local DRAM. It is accessible by the onboard MPU, the refresh circuitry, and the VMEbus (by another VMEbus master), as described in the *Multiport Arbiter, Refresh, and Shared Memory Map Decoder* section in this chapter.

VMEbus Data Buffers (Sheet 11)

Refer to Figure 4-2 and the *Data Bus Structure* section in this chapter.

MVME133 VMEbus Master Interface and Address Buffers (Sheets 12 and 20)

The MVME133 has an A24/D32 master interface. Whenever the MVME133 executes a VMEbus cycle (read, write, or interrupt) and its VMEbus requester has obtained VMEbus mastership, it drives the VME address bus with its local address bus and the VMEbus address modifiers with AM0 = [FC0], AM1 = [FC1], AM3 = high, AM4 = low in short I/O range or high in standard range, and AM5 = high Refer to the *MVME133 Memory Maps and Map Decoder* section in Chapter 3 and Table 3-2 for cycle types and responding devices.

MVME133 VMEbus Data Width

As a VMEbus master, the MVME133 performs 32-bit data transfers only on longword-aligned accesses or if the VMEbus is a 32-bit data system. J17 is jumpered to indicate that the system is 16-bit or 32-bit data.

In a system where there are both 16-bit and 32-bit data, you may configure J17 so that MPU address line [A24] dynamically indicates to the MVME133 master interface the data width of the VMEbus. Refer to Table 4-2.

Table 4-2. Using [A24] to set VBEbus Data Width

MPU Cycle Type	LWORD*	Data Width
Not Longword aligned	negated	VMEbus = 16-bit data port
Longword aligned and [A24 = 0	asserted	VMEbus = 32-bit data port
Longword aligned and [A24] = 1	negated	VMEbus = 16-bit data port

MVME133 Accessing the VMEbus

Whenever the MVME133 executes a VMEbus cycle (read, write, or interrupt acknowledge) and its VMEbus requester has obtained VMEbus mastership, it drives the VME address bus with its local address bus and the VMEbus address modifiers to indicate proper address space, and asserts IACK* if this is an interrupt acknowledge cycle. It asserts LWORD* on longword-aligned transfers only if J17 indicates that the VMEbus is a 32-bit port (either statically or dynamically with [A24]). Refer to Table 3-2 and the *Memory Maps and Map Decoder* section in Chapter 3 for cycle types and responding devices.

Once A01 through A24. AM0 through AM5, IACK* and LWORD* are driven to their appropriate levels, the MVME133 asserts AS*. If it is a read cycle, the MVME133 negates WRITE* and enables D00 through D15 onto [D16] through [D31] if LWORD* is negated or D00 through D15 onto [D00] through [D15] and D16 through D31 onto [D16] through [D31] if LWORD* is asserted. After the appropriate delay, the MVME133 asserts DS0* and/or DS1* as appropriate. If the cycle is a write cycle, the MVME133 asserts WRITE*, and enables [D16] through [D31] onto D00 through D15 if LWORD* is negated and A01 is high, [D00] through [D15] onto D00 through D15 if LWORD* is negated and A01 is low, or [D00] through [D15] onto D00 through D15 and [D16] through [D31] onto D16 through D31 if LWORD* is asserted. After the appropriate delay, the MVME133 asserts DS0*/DS1*. Refer also to Figure 4-2 and the *Data Bus Structure* section in this chapter.

If the cycle terminates normally with DTACK* asserted, the onboard DSACKs generator circuit asserts [DSACK1*] and [DSACK0*] if LWORD* is asserted or [DSACK1*] if LWORD* is negated

If the cycle terminates with BERR* asserted, the BERR generator circuit asserts [BERR*] to the MPU. Once the handshake has occurred (either DTACK* or BERR*), the MPU negates [AS*] and [DS*] and the MVME133 completes the cycle by disabling the data bus drivers and negating DS0*/DS1* and AS*.

The above sequence is altered slightly when the MPU executes RMW cycles. When the MPU starts an RMW cycle, the VMEbus master interface checks to see if it is a single or multiple address RMW by examining SIZ1 and SIZ0. If it is a multiple address RMW cycle, the VMEbus master interface operates normally. Refer to the *Operation of the VMEbus Requester* section in this chapter. If it is a single address RMW cycle, the VMEbus master interface leaves AS* asserted during the entire time from the beginning of the RMW read cycle to the end of the RMW write cycle. This makes a single address RMW cycle from the MPU appear on the VMEbus as a VMEbus-defined READ-MODIFY-WRITE cycle.

MVME133A-20 VMEbus Master Interface and Address Buffers (Sheets 12 and 20)

The MVME133A-20 has an A32/D32 VMEbus master interface for buffering data, address, and control; word data manipulation to accommodate MC68020 and VMEbus data handling differences; and interrupt handling and control of misaligned transfers. However, it may be used with devices that have A24 or A32, and D16 or D32 interfaces by the use of jumpers on headers J18 and J17 (refer to Chapter 2) and by observing the following requirements.

MVME133A-20 VMEbus Address Size

By properly jumpering J18, you can configure the MVME133 to operate in a mixed 32-bit and 24-bit address system, or in a fully 32-bit address system. Refer to the *Address Size Select Header* section in Chapter 2 for details. The MVME133 VMEbus memory map is directly affected by the address option as shown in Table 4-3. (Refer also to Table 3-3.)

Table 4-3. MVME133A-20 VMEbus Memory Map In A Mixed A24/A32 System And In An A32 System

Address Range	VMEbus Activity Type
$00000000-$000FFFFF	No VMEbus activity, onboard DRAM area
$00100000-$00EFFFFF	VMEbus standard (24-bit) address space in a mixed A24/A32 system; extended (32-bit) address space in an A32 system
$00F00000-$FFEFFFFF	VMEbus extended (32-bit) address space
$FFF00000-$FFFBFFFF	No VMEbus activity, local resource area
$FFFC0000-$FFFEFFFF	VMEbus extended (32-bit) address space
$FFFF0000-$FFFFFFFF	VMEbus short I/O (16-bit) address space

MVME133A-20 VMEbus Data Width

As a VMEbus master, the MVME133 performs 32-bit data transfers only on longword-aligned accesses and provided the VMEbus is a 32-bit data system. J17 is jumpered to indicate that the system is 16-bit or 32-bit data.

In a system where there are both 16-bit and 32-bit data, you may configure J17 so that MPU address line [A24] dynamically indicates to the MVME133 master interface the data width of the VMEbus. Refer to Table 4-4.

Table 4-4. MVME133A-20 Using [A24] To Set VBEbus Data Width

MPU Cycle Type	LWORD*	Data Width
Not Longword aligned	negated	VMEbus = 16-bit data port
Longword aligned and [A24 = 0	asserted	VMEbus = 32-bit data port
Longword aligned and [A24] = 1	negated	VMEbus = 16-bit data port

Thus, you can place all 32-bit data devices in any of the 128 16-Mb blocks where [A24] is low ($00000000 to $00FFFFFF, $02000000 to $02FFFFFF, $04000000 to $04FFFFFF, ... $FC000000 to $FCFFFFFF, $FE000000 to $FEFFFFFF), and all 16-bit data devices at any of the other 128 16-Mb blocks where [A24] is high ($01000000 to $01FFFFFF, $03000000 to $03FFFFFF, ... $FF000000 to $FFFFFFFF). The *VMEbus Data Width Select Header* section in Chapter 2 shows the proper configuration of J17.

MVME133A-20 Accessing the VMEbus

Whenever the MVME133A-20 executes a VMEbus cycle (read, write, or interrupt acknowledge) and its VMEbus requester has obtained VMEbus mastership, it drives the VME address bus with its local address bus and the VMEbus address modifiers to indicate proper address space. It also asserts IACK* if this is an interrupt acknowledge cycle. It asserts LWORD* on longword-aligned transfers only if J17 indicates that the VMEbus is a 32-bit port (either statically or dynamically with [A24]). Refer to Table 3-2 and the *Memory Maps and Map Decoder* section in Chapter 3 for cycle types and responding devices.

When A01 through A31, AM0 through AM5, IACK* and LWORD* are driven to their appropriate levels on the VMEbus, the MVME133A-20 asserts AS*. If it is a read cycle, the MVME133A-20 negates WRITE*, enables D00 through D15 onto [D16] through [D31] if LWORD* is negated, or D00 through D15 onto [D00] through [D15] and D16 through D31 onto [D16] through [D31] if LWORD* is asserted, and asserts DS0* and/or DS1* as appropriate. If the cycle is a write cycle, the MVME133A-20 asserts WRITE*, and enables [D16] through [D31] onto D00 through D15 if LWORD* is negated and A01 is high, [D00] through [D15] onto D00 through D15 if LWORD* is negated and A01 is low, or [D00] through [D15] onto D00 through D15 and [D16] through [D31] onto D16 through D31 if LWORD* is asserted. After the appropriate delay, the MVME133A-20 asserts DS0*/DS1*. Refer also to Figure 4-2 and the *Data Bus Structure* section in this chapter.

If the cycle terminates normally with DTACK* asserted, the onboard DSACKs generator circuit asserts [DSACK1*] and [DSACK0*] if LWORD* is asserted or [DSACK1*] if LWORD* is negated. If the cycle terminates with BERR* asserted, the BERR generator circuit asserts [BERR*] to the MPU. Once the handshake has occurred (either DTACK* or BERR*), the MPU negates [AS*] and [DS*] and the MVME133 completes the cycle by disabling the data bus drivers and negating DS0*/DS1* and AS*.

The above sequence is altered slightly when the MPU executes RMW cycles. When the MPU starts an RMW cycle, the VMEbus master interface checks to see if it is a single or multiple address RMW by examining SIZ1 and SIZ0. If it is a multiple address RMW cycle, the VMEbus master interface operates normally. (Refer to the *Operation of the VMEbus Requester* section in this chapter.) If it is a single address RMW cycle, the VMEbus master interface leaves AS* asserted during the entire time from the beginning of the RMW read cycle to the end of the RMW write cycle. This makes a single address RMW cycle from the MPU appear on the VMEbus as a VMEbus-defined READ-MODIFY-WRITE cycle.

VMEbus Requester (Sheet 13)

The VMEbus requester is used to obtain and relinquish mastership of the VMEbus. It can request VMEbus mastership on any one of the four request levels depending on the configurations of J3 and J4, and it fully supports the bus-grant daisy-chain. It requests mastership of the VMEbus whenever the MVME133 is not the current bus master, and either the map decoder or the interrupt handler indicates that the MPU is executing a cycle that requires the VMEbus or the MPU is starting to execute an RMW sequence to the onboard RAM with J5 pins 1-2 connected.

The VMEbus requester operates in the Release-On-Request (ROR) mode. Once the MVME133 has obtained VMEbus mastership, the VMEbus requester maintains mastership until another VMEbus module requests VMEbus mastership and then only if an RMW sequence is not in process. It releases the VMEbus in one of two different ways, depending on the state of the MVME133 at the time.

If the MVME133 is in the middle of a VMEbus cycle (AS* already asserted) when the VMEbus requester decides to relinquish VMEbus mastership, the MVME133 negates BBSY* immediately. The transfer of VMEbus mastership occurs when the VMEbus master interface deactivates and negates AS*

If the MVME133 is not in the middle of a VMEbus cycle when the VMEbus requester decides to relinquish VMEbus mastership, the VMEbus master interface releases all of the VMEbus lines. The VMEbus requester negates BBSY* to complete the transfer of VMEbus mastership.

Refer to the *Accessing the VMEbus section* in this chapter.

VMEbus System Controller and Interrupter (Sheet 14)

The following two sections discuss the system controller and the interrupter.

System Controller and SYSRESET*

The system controller implements a global VMEbus timeout that asserts BERR*, enables Global SYSCLK (16 MHz), enables Level 3 VMEbus arbiter, and asserts the IACK* daisy-chain driver. All of these MVME133 system controller functions and the SYSRESET* driver are enabled/disabled by header J1. The position of the jumper on J1 appears as the SYSCON bit in the module status register. (Refer to the *Status and Control* section in this chapter)

The global bus timeout circuit starts the timing upon detecting assertion of DS0* and/or DS1*. If DS0* and/or DS1* are asserted longer than the timeout period, the circuit asserts BERR*. The timeout count can be set for:

- 82µs to 92 µs (MVME133) with J8 pins 1-2 connected
- 61µs to 69 µs (MVME133-1) with J8 pins 1-2 connected
- 50µs to 57µs (MVME133A-20) with J8 pins 1-2 connected
- Infinity with J8 1-2 pins not connected

The SYSCLK driver drives a periodic 16 MHz clock onto the SYSCLK line on the VMEbus if the system controller on the MVME133 is enabled.

The level 3 arbiter is designed to meet the VMEbus specification requirements, ignore possibly erroneous signals on BBSY*, and rearbitrate if no VMEbus master responds to a grant within 1 ms for the MVME133, or 0.8 to 2.5 ms for the MVME133-20. The IACK* daisy-chain driver is designed to meet the VMEbus specification requirements.

Although SYSRESET* is not a VMEbus system controller function, the MVME133 enables/disables its SYSRESET* function at the same time that it enables/disables its system controller functions. When configured as the system controller, the MVME133 asserts the SYSRESET* signal line when the front panel RESET switch is pressed, a watchdog timeout occurs, the RRESET* line is asserted, or a power up occurs.

NOTE

> MVME133 does not fully implement SYSRESET* timing of a VMEbus power monitor.

VMEbus Single-Level Interrupter

The VMEbus single-level interrupter generates interrupt requests on IRQ3* and provides the value $FF as its status ID byte. The interrupter is an 8-bit interrupter and consequently responds to all sizes of interrupt acknowledge cycles. It drives IRQ3* whenever the MPU reads the real-time-clock with [A25] = 1 and whenever the MPU performs a read access to any location between $FFFB8000 through $FFFBFFFF (the real-time clock) with [A15] high. The state of the interrupter is reflected as the [OIRQ] = GPIO6 bit of the Multi-Function Peripheral (MFP) GPIO port. (Refer to *Status and Control* section in this chapter). A typical sequence for interrupting is:

a. Verify that the [OIRQ] bit is 0.

b. Set up the MFP to interrupt the MPU when [OIRQ] transitions from 1 to 0 to indicate that the interrupt has been acknowledged.

c. Read any register in the real-time clock with [A25] = 1 (for example, $2FB0000). (MVME133)

   Perform a read access to any address between $FFFB8000 and $FFFBFFFF. (MVME133A-1)

d. Continue with other processing until the [OIRQ] interrupt occurs.

e. The VMEbus interrupt has now been acknowledged.

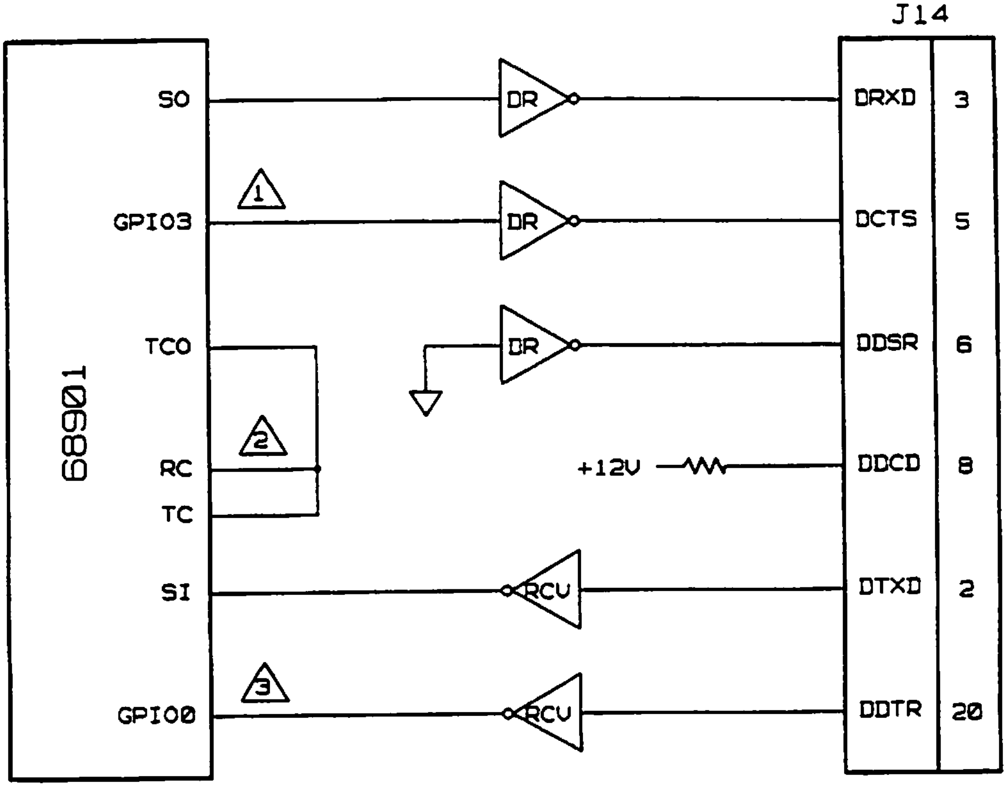
f. Continue with normal processing.

Timers, Debug Port, and Status/Control (Sheet 15)

The MVME133 uses the MFP MC68901 chip for its front panel debug port, tick timers, watchdog timer, and status and control information The MC68901 can interrupt the MPU on level 5 (Refer to the *Interrupt Handler* section in this chapter). Its interrupt sources are from the timers, the debug port, and the GPIO (status) bits.

Debug Port

The front panel debug port (through J14) is a minimal implementation of a to-terminal-only RS-232C serial port. (Refer to Appendix B for a discussion of RS-232C signals.) The debug port uses DRXD as its transmit data output and DTXD as its receive data input, drives DDSR, pulls DDCD to +12 Vdc, controls DCTS with a software bit, and monitors DRTS with another software bit, providing minimal flow control. See Figure 4-3. The baud rate generator for the serial port is timer C of the MC68901 MFP. The XTAL input to the MC68901 is 1.230769 MHz. The baud rates supported are programmed as shown in Table 4-5.



NOTE:

1 GPIO3 PROGRAMMED AS OUTPUT.

2 TC AND RC ARE DRIVEN BY OUTPUT OF TIMER C.

3 GPIO0 PROGRAMMED AS INPUT.

Figure 4-3. Debug Port (To Terminal Only)

Table 4-5. Debug Port Baud Rates Available with XTAL = 1.23 MHz

Baud Rate	Clock Mode	Pre-scale	Timer C Count	Actual Rate	Percent Error
9600	x16	4	1	9615.4	0.16
4800	x16	4	2	4807.7	0.16
2400	x16	4	4	2403.8	0.16
1200	x16	4	8	1201.9	0.16
600	x16	4	$10	601.0	0.16
300	x16	4	$20	300.5	0.16
110	x16	4	$57	110.5	0.47

Timers

There are four timers in the MC68901 MFP. They are assigned as:

Timer A - Software tick timer. The tick timer is capable of generating a periodic interrupt. Refer to Appendix D for an example of its setup.

Timer B - Tick timer overflow/watchdog timeout. The watchdog timer is capable of generating a local/system reset when timer B output is high after a programmable interval.

Timer C - Baud rate generator for the front panel serial debug port.

Timer D - Delay mode only. Unassigned by hardware.

Status and Control

The MC68901 MFP has eight General Purpose I/O (GPIO) pins. The MVME133 uses five of these pins as status inputs and three of them as control outputs. After a reset, the MC68901 makes all of the GPIO pins inputs. Therefore, after each reset, the software should initialize the control bits and make them outputs. MVME133 hardware defaults the control lines to high when they are not programmed as outputs. The assignment of pins GPIO0-GPIO7 is:

GPIO0 - (MVME133) Input: Connected to [DDTR*]. General Purpose I/O Interrupt Port (GPIP). Bit 0 = 0 when DRTS is high on the debug RS-232C interface. Bit 0 = 1 when DDTR is low on the debug RS-232C interface. Bit 0 of the Interrupt Pending Register B (IPRB) may be initialized by software to detect the transitions of DDTR.

(MVME133A-20) Input: Connected to [DRTS*]. General Purpose I/O Interrupt Port (GPIP). Bit 0 = 0 when DRTS is high on the debug RS-232C interface. Bit 0 = 1 when DRTS is low on the debug RS-232C interface. Bit 0 of the Interrupt Pending Register B (IPRB) may be initialized by software to detect the transitions of DRTS

GPIO1 - Input: Connected to [QVBERR*]. If BERR* is asserted when the MVME133 executes a cycle on the VMEbus, [QVBERR*] is asserted until the end of the cycle.

Because [QVBERR*] always negates at the end of error cycle, GPIP bit 1 always reads as 1 by the time software reads it. However, software may initialize IPRB bit 1 to latch the fact that [QVBERR*] has asserted. IPRB bit 1 may then be read and cleared by software.

GPIO2 - Input: Connected to [LTO]. When the MPU performs an access which does not complete within 20 ms, the MVME133 asserts [BERR*] to the MPU. [LTO] is asserted during that cycle and negates after the end of the cycle.

Because [LTO] always negates at the end of the timeout cycle, GPIP bit 2 always reads as 0 by the time software reads it. However, software may initialize IPRB bit 2 to latch the fact that [LTO] has asserted. IPRB bit 2 then may be read and cleared by software.

GPIO3 - Control: Connected to [DCTS*]. When GPIP bit 3 = 0, DCTS is asserted on the debug RS-232C interface. When GPIP bit 3 = 1 or when it is programmed as input, DCTS is negated on the debug RS-232C interface.

GPIO4 - Control: Connected to [IE*]. When GPIP bit 4 = 1 or when it is programmed as input, no interrupt requests reach the MPU. When GPIP bit 4 = 0, interrupt requests may reach the MPU.

GPIO5 - Control: Connected to [BRDFAIL]. When GPIP bit 5 = 1 or when it is programmed as input, the FAIL indicator is lit. Also, if the MVME133 is not the system controller, it asserts the SYSFAIL* line on the VMEbus during this time. When GPIP bit 5 = 0, the SYSFAIL* line is not driven by the MVME133 and the FAIL indicator is not lit.

GPIO6 - Input: Connected to [OIRQ]. When [OIRQ] = 1, the MVME133 is driving IRQ3* on the VMEbus. When [OIRQ] = 0, the MVME133 is not driving IRQ3* on the VMEbus.

[OIRQ] transitions from 0 to 1 when the MVME133 reads the real-time clock with [A15] high. [OIRQ] transitions from 1 to 0 when the MVME133 IRQ3* is acknowledged on the VMEbus. Transitions on [OIRQ] may be detected and latched in Interrupt Pending Register A (IPRA) bit 6. [OIRQ] is cleared by reset.

GPIO7 - Input: Connected to [SYSFAIL]. When SYSFAIL* is asserted, GPIP bit 7 = 1. When SYSFAIL* is negated, GPIP bit 7 = 0. Transitions on SYSFAIL* may be detected and latched in IPRA bit 7.

In addition to the MC68901 MFP GPIO bits, the MVME133 has eight status bits that are read only, have no latching mechanism, and cause no interrupts (with one exception). Collectively, these bits are called the Module Status Register (MSR). Because of hardware savings, the MSR and the GPIO are grouped together and appear as a 16-bit word port to the MPU. (Refer to Table 3-3 and the *Main Memory Map* section in Chapter 3.) Therefore, although the MSR ignores all write accesses, a write to the MSR affects the GPIO.

The GPIO appears on the lower byte of the word, and the MSR appears on the upper byte. The bit assignments for the MSR are:

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
ACFAIL	SYSCON	PWRUP*	SRBIT4	SRBIT3	SRBIT2	SRBIT1	SRBIT0

ACFAIL - This bit is 1 when VMEbus ACFAIL* is asserted. This bit is 0 when ACFAIL* is negated. [ACFAIL] is also an input to the interrupt handler.

SYSCON - This bit is 1 if the module is the VMEbus system controller. This bit is 0 when the module is is not the VMEbus system controller.

PWRUP* - This bit is set to 0 only by power up reset when power is first applied to the MVME133. A read cycle to the real-time clock resets this bit to a 1. Refer to the *Reset* section in this chapter.

SRBIT4 - This bit is 0 when J15 pins 9-10 are connected, and is 1 when they are open.

SRBIT3 - This bit is 0 when J15 pins 7-8 are connected, and is 1 when they are open.

SRBIT2 - This bit is 0 when J15 pins 5-6 are connected, and is 1 when they are open.

SRBIT1 - This bit is 0 when J15 pins 3-4 are connected, and is 1 when they are open.

SRBIT0 - This bit is 0 when J15 pins 1-2 are connected, and is 1 when they are open.

Interrupt Handler, Reset, Abort, and Status LEDs (Sheet 16)

The following sections discuss the interrupt handler and the LED indicators.

Interrupt Handler

The interrupt handler gives the onboard MPU the ability to sense and respond to all onboard interrupts, all seven VMEbus interrupts, VMEbus ACFAIL*, VMEbus SYSFAIL*, and the ABORT switch

All VMEbus interrupts are enabled/disabled using header J12. All interrupts are disabled when the [IE*] bit is 1 (high). ABORT is enabled/disabled using J10. All interrupts that go through the MC68901 MFP are enabled in the MC68901. The Z8530 and the real-time clock may be enabled/disabled individually.

When the MPU initiates an interrupt acknowledge cycle, the interrupt handler determines the acknowledge level by examining [A01] - [A03] and asserts [AVEC*] to indicate to the MPU to generate the interrupt vector internally if the acknowledge cycle was for VMEbus ACFAIL*, ABORT switch, or the real-time clock. The interrupt handler initiates a vector fetch cycle to the appropriate device if the acknowledge cycle is for the Z8530, MC68901 MFP, or VMEbus.

If both onboard and VMEbus interrupts are asserted on the acknowledge level, the interrupt handler acknowledges the onboard interrupt. VMEbus ACFAIL* and ABORT switch are both on level 7 and have the same interrupt offset vector. The software handler routine for this autovector must interrogate the [ACFAIL] bit in the MSR to determine the actual interrupt source. Table 4-6 summarizes the interrupt sources on the MVME133 (in descending order of priority) and the associated interrupt vectors.

Table 4-6. Interrupt Sources and Vectors

Interrupt Source	Vector Source	Vector Number	Vector Offset	Level
ABORT*	auto	31	$7C	7
VMEbus ACFAIL*	auto	31	$7C	7
VMEbus IRQ7*	VMEbus	supplied	4 x vector	7
Z8530 SIO (Serial Ports)	Z8530	programmable (NOTE 1)	4 x vector	6
VMEbus IRQ6*	VMEbus	supplied	4 x vector	6
MC68901 MFP (Multi-function peripheral)	MC68901	programmable (NOTE 2)	4 x vector	5
VMEbus IRQ5*	VMEbus	supplied	4 x vector	5
MM58274A RTC (Real-Time Clock)	auto	28	$70	4
VMEbus IRQ4*	VMEbus	supplied	4 x vector	4
VMEbus IRQ3*	VMEbus	supplied	4 x vector	3
VMEbus IRQ2*	VMEbus	supplied	4 x vector	2
VMEbus IRQ1*	VMEbus	supplied	4 x vector	1

NOTES: 1. Refer to Appendix C for an example of setting up serial port B of the Z8530.

2. Refer to Appendix D for an example of setting up timer A of the MC68901.

Reset

All resets wait until the MPU is between cycles before starting. There six sources of reset on the MVME133. They are:

1. SYSRESET* (VMEbus system reset): Resets all onboard devices.
2. Power Up Reset: Resets all onboard devices and drives SYSRESET* if this module is system controller.

3. Front Panel RESET Switch: Resets all onboard devices and drives SYSRESET* if this module is system controller.
4. Watchdog Timeout: Resets all onboard devices and drives SYSRESET* if the MVME133 is system controller.
5. RRESET* (remote reset): Resets all onboard devices and drives SYSRESET* if the MVME133 is system controller.
6. MC68020 RESET Instruction: Resets only the Z8530 and the MC68901 MFP.

ABORT and RESET Switches

Refer to Chapter 3 for information on these front panel switches.

FAIL, HALT, and RUN Indicators

Refer to Chapter 3 for information on these front panel LED indicators.

ROM/PROM/EPROM/EEPROM Sockets and Real-Time Clock (Sheet 17)

ROM/PROM/EPROM/EEPROM

The MVME133 has four 28-pin ROM/PROM/EPROM/ EEPROM sockets that are organized as two banks with two sockets per bank. Each bank appears as a 16-bit word port to the MPU and can be separately configured. If the devices are ROM, PROM, or EPROM, the sockets may be configured for 8K x 8, 16K x 8, 32K x 8, or 64K x 8. If the devices are EEPROMs, the sockets may be configured for 2K x 8, 8K x 8, or 32K x 8x .

Figure 4-4 shows the definitions of the ROM/PROM/EPROM/EEPROM socket pins, depending upon the configuration used.

Configuration								Configuration				
1	2	3	4	5				5	4	3	2	1
+5V	+5V	A15	NC	A14	1		28	+5V	+5V	+5V	+5V	+5V
A12	A12	A12	A12	A12	2	SOCKET	27	WE*	WE*	A14	A14	VIH
A7	A7	A7	A7	A7	3		26	A13	A13	A13	A13	A13
A6	A6	A6	A6	A6	4		25	A8	A8	A8	A8	A8
A5	A5	A5	A5	A5	5		24	A9	A9	A9	A9	A9
A4	A4	A4	A4	A4	6		23	A11	A11	A11	A11	A11
A3	A3	A3	A3	A3	7		22	OE*	OE*	OE*	OE*	OE*
A2	A2	A2	A2	A2	8		21	A10	A10	A10	A10	A10
A1	A1	A1	A1	A1	9		20	CE*	CE*	CE*	CE*	CE*
A0	A0	A0	A0	A0	10		19	DQ7	DQ7	D7	D7	D7
D0	D0	D0	DQ0	DQ0	11		18	DQ6	DQ6	D6	D6	D6
D1	D1	D1	DQ1	DQ1	12		17	DQ5	DQ5	D5	D5	D5
D2	D2	D2	DQ2	DQ2	13		16	DQ4	DQ4	D4	D4	D4
GND	GND	GND	GND	GND	14		15	DQ3	DQ3	D3	D3	D3

NOTES:
1. Socket A14 = board [A15]
2. Socket A15 = board [A16]
3. See Schematic Diagram in *SIMVME133 Support Information.*
4. Refer to *ROM/PROM/EPROM/EEPROM Size Headers* in Chapter 2.
5. MVME133BUG uses 64K x 8 EPROMs.

Configuration	J6 (Bank 1) or J7 (Bank 2) Connections	Description
1	2 to 4	8K x 8 or 16K x 8 ROM/PROM/EPROM
2	1 to 3, and 2 to 4	32K x 8 ROM/PROM/EPROM
3	1 to 3, and 4 to 6	64K x 8 ROM/PROM/EPROM (Factory Configuration)
4	3 to 5	2K x 8 or 8K x 8 EEPROM
5	1 to 4, and 3 to 5	32K x 8 EEPROM

Figure 4-4. ROM/PROM/EPROM/EEPROM Sockets Configurations

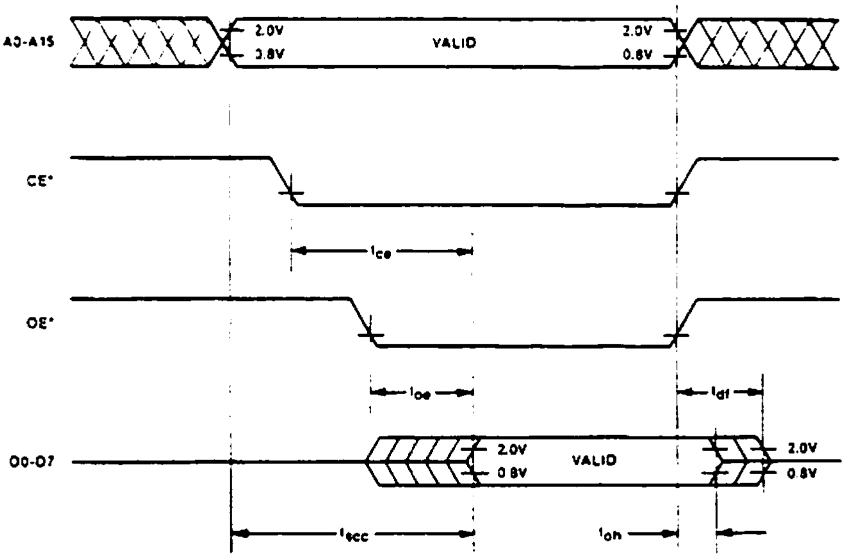
The ROM/PROM/EPROM/EEPROM devices must meet the read timings shown in Figure 4-5. The EEPROMs are guaranteed the write timings shown in Figure 4-6.

Consider the following when using EEPROMs on the MVME133:

1. The MVME133 provides no protection against inadvertent writes to EEPROMs that might happen during power on/off transitions. Most devices provide some level of internal protection. In order to gain "absolute protection", devices with additional "software protection" are recommended.
2. Writes to the bank must always be 16-bit wide. Any access to one byte of the bank also accesses the other byte. Thus, byte-wide writes cause unintended data to be written to the other byte.
3. There are several different algorithms for erasing/writing to EEPROMs, depending on the manufacturer. The MVME133 supports only those devices which have a static RAM compatible erase/write mechanism.
4. The EEPROMs must allow wired-OR on the RDY/BSY* pin (for 2K x 8 and 8K x 8 devices). The MVME133 does not monitor the status of the RDY/BSY* pins.

Real-Time Clock

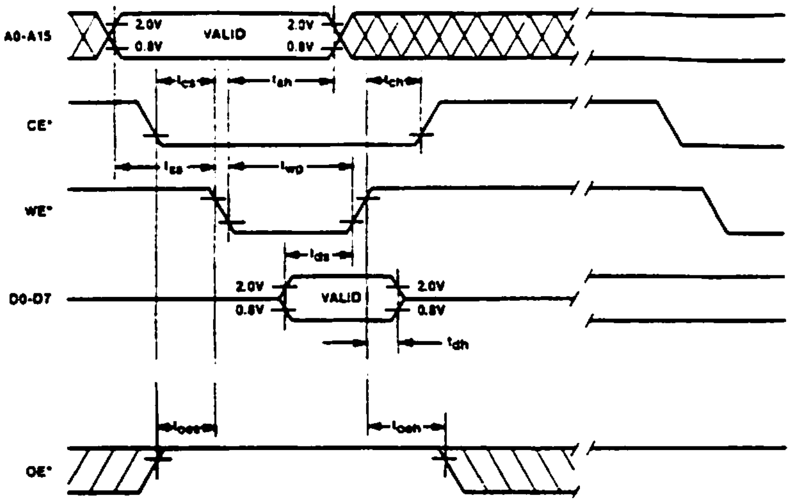
The real-time clock on the MVME133 is an MM58274. It provides a timekeeping function from tenths of seconds to tens of years in independently accessible registers, an hours counter programmable for 12-hour or 24-hour operation, independent interrupting timer, and its own onboard crystal controlled oscillator. The MM5824 counters are arranged as 4-bit words and can be randomly accessed for reading and setting, has a 4-bit data bus that is connected to [D24] through [D27], and is capable of generating interrupts to the MPU on level 4.



		Time in ns					
		133 (12.5 MHz)		133-1 (16.67 MHz)		133A-20 (20 MHz)	
Symbol	Description	Min	Max	Min	Max	Min	Max
tacc	Address valid to data valid	-	390	-	295	-	245
tce	CE* low to data valid	-	350	-	265	-	220
toe	OE* low to data valid	-	275	-	200	-	160
toh	Address invalid, CE* or OE* high to data not valid	0	-	0	-	0	-
tdf	CE* or OE* high to data high impedance	-	80	-	60	-	50

NOTE: The MVME133/133-1/133A-20 does not guarantee a maximum transition time on address and data lines during the time that CE* is high.

Figure 4-5. Required ROM/PROM/EPROM/EEPROM Read Timings



		Time in ns					
		133 (12.5 MHz)		133-1 (16.67 MHz)		133A-20 (20 MHz)	
Symbol	Description	Min	Max	Min	Max	Min	Max
tas	Address valid to WE* low	86	-	66	-	55	-
tcs	CE* asserted to WE* low	44	-	34	-	30	-
toes	OE* negated to WE* low	93	-	63	-	48	-
tah	Address valid after WE* low	324	-	234	-	190	-
twp	WE* low pulse width	230	-	170	-	140	-
tds	Data valid to WE* high	245	-	185	-	155	-
tdh	WE* high to data not valid	93	-	63	-	45	-
toeh	WE* high to OE* low	221	-	161	-	150	-
tch	WE* high to CE* high	58	-	38	-	25	-
tre	RDY/BSY* high to first write to device (Note 1)	126	-	96	-	-	-

NOTE: 1. Assumes that software waits for RDY/BSY* to go high before writing to the device again.

2. The MVME133/133-1/133A-20 does not guarantee a maximum transition time on address and data lines during the time that CE* is high.

Figure 4-6. Guaranteed EEPROM Write Timings

Read accesses to the real-time clock negate (clear) the [PWRUP*] status bit. Also, a read cycle to the real-time clock with [A15] = 1 causes the VMEbus interrupter to request a VMEbus level 3 interrupt. Refer to the *VMEbus Single-Level Interrupter* section in this chapter.

MC68881 Floating Point Coprocessor (Sheet 18)

The MC68881 Floating Point Coprocessor (FPC) is a full implementation of the IEEE standard for binary floating-point arithmetic. The MC68881 provides a logical extension to the MC68020 MPU and operates at the same frequency as the MPU. (12.5 MHz for the MVME133, 16.67 MHz for the MVME133-1, and 20 MHz for the MVME133A-20)

The MC68881 appears as a 32-bit data port to the MPU, and imposes no delays on CE*, [DSACK0*], or [DSACK1*]. Accesses to the MC68881 other than a read to the response or save CIR require three MPU clocks (no wait cycle). Reading the response or save CIR is performed in five MPU clock cycles. Refer to Chapter 3 for the memory map of the registers in the MC68881 floating point coprocessor. Refer to the MC68881 Floating-Point Coprocessor User's Manual for further details.

Multiprotocol Serial Ports (Sheet 19)

The MVME133 uses the Z8530 to implement its two multiprotocol serial ports, providing multifunction support for handling the large variety of serial communications protocols available. The Z8530 can be programmed to satisfy special serial communication requirements, and follow standard formats such as byte-oriented synchronous, bit-oriented synchronous, or asynchronous. Protocol variations are supported within each operating mode by checking odd or even parity, character insertion or deletion, CRC generation and checking, and break and abort generation and detection. The Z8530 can be programmed to support many other protocol-dependent features.

Port A of the Z8530 is connected to onboard RS-485 drivers and receivers. Port B of the Z8530 is connected to onboard RS-232C drivers and receivers. Because of its internal structure, there are several means of obtaining the baud rate clocks for each of the two serial channels. Each channel within the Z8530 has a programmable baud rate generator. The Baud Rate Generator (BRG) input can be from the RTXC input or from PCLK. The hardware on the MVME133 allows the RTXC pin for each channel to be connected to an external clock source or to the onboard 1.23 MHz clock. Tables 4-8 through 4-10 show the values in the Z8530 time constant register that are required to create some common baud rates. Refer to Chapter 2.

Table 4-7. Baud Rates Available with BRG Clock = RTXC Pin = 1.23 MHz

Baud Rate	Clock Mode	Time Constant Register Value	Actual Baud Rate	Percent Error
19200	x16	0	19231	0.16
9600	x16	2	9615	0.16
4800	x16	6	4808	0.16
2400	x16	$E	2404	0.16
1200	x16	$1E	1202	0.16
600	x16	$3E	601	0.16
300	x16	$7E	300	0.16
110	x16	$15E	109	0.67
64000	x1	8	61538	3.85
56000	x1	9	55944	0.10
48000	x1	$B	47337	1.38
38400	x1	$E	38461	0.16

The Z8530 DPLL input can be either the BRG output or the RTXC pin. The DPLL operates at 32 times the data rate for NRZI and at 16 times the data rate for FM. Tables 4-8 through 4-10 give some of the data rates that are achievable with the MPU operating at 12.5 MHz in the MVME133, 16.67MHz in the MVME133-1 and 20 MHz in the MVME133A-20.

Table 4-8. Baud Rates Available with Clock = PCLK = 3.125 MHz (MVME133)

Baud Rate	Theoretical 32 X Clock Frequency	Time Constant Register Value	Actual Frequency	Percent Error
24414	N/A	0	781248	N/A
16276	N/A	1	520832	N/A
12207	N/A	2	390624	N/A
Baud Rate	**Theoretical 16 X Clock Frequency**	**Time Constant Register Value**	**Actual Frequency**	**Percent Error**
48000	768000	0	781250	1.7
32552	N/A	1	520832	N/A
38400	N/A	2	390625	N/A

Table 4-9. Baud Rates Available with Clock = PCLK = 4.167 MHz (MVME133-1)

Baud Rate	Theoretical 32 X Clock Frequency	Time Constant Register Value	Actual Frequency	Percent Error
32552	N/A	0	104667	N/A
26042	N/A	1	694444	N/A
19200	N/A	2	520833	N/A
Baud Rate	**Theoretical 16 X Clock Frequency**	**Time Constant Register Value**	**Actual Frequency**	**Percent Error**
76800	x16	0	104667	1.7
52083	N/A	1	694444	N/A
38400	N/A	2	520833	N/A

Table 4-10. Baud Rates Available with BRG Clock = PCLK = 5.00 MHz (MVME133A-20)

Baud Rate	Clock Mode	Time Constant Register Value	Actual Baud Rate	Percent Error
38400	x32	0	39063	1.7
26042	x32	1	26042	N/A
19200	x32	2	19531	1.7
76800	x16	0	78125	1.7
52083	x16	1	52083	N/A
38400	x16	2	39063	1.7

If frequencies other than the ones available with 1.230769 MHz as the BRG clock source are required, the frequency of 1.230769 MHz can be changed by reprogramming U18, PALSC, to divide the 16 MHz by a value other than 13.

NOTE

You must be aware that both ports of the Z8530 and the MC68901 may be using the 1.230769 MHz signal, and changing that frequency may make it impossible to create a desired frequency on the other port of the Z8530 and/or on the MC68901 MFP debug port.

RS-485 Port

Port A of the Z8530 uses RS-485/RS-422A drivers and receivers. The RS-485 signals are routed to P2 rows A and C. An external cable may be connected to P2 and you must make a crossover cable to convert from the cable pinout of P2 on the MVME133 to the pinout of your serial network. The connector used to interface to the RS-485 network should take shielding into consideration.

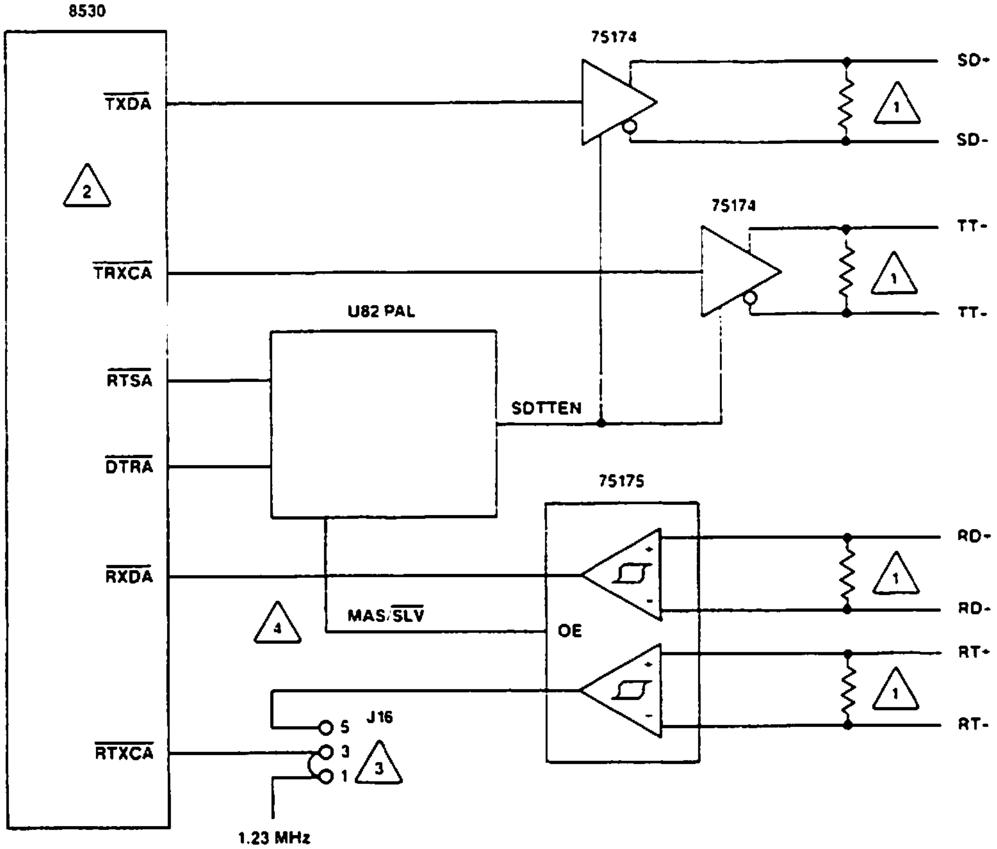
The RS-485 port can be configured by software to be either master or slave and half or full duplex by controlling DTR/REQA (DTRA) and RTSA of port A. The functions of these two control bits are defined by the programmed PAL U82. As shipped from the factory, U82 defines them as DTRA indicates master when it is high (1) and slave when low (0), and RTSA enables the RS-485 drivers when it is low (0). You may change the functions of RTSA and DTRA by reprogramming U82. Table 4-11 and Figures 4-7 and 4-8 show the possible configurations for the RS-485 port with the default program in U82. Refer to Appendix A for U82 program details.

Table 4-11. RS-485 Port Configurations

DTRA	RTSA	Configuration	Description
high	low	Master (drivers on)	TXDA drives SD+/-, RD+/- drives RXDA, TRXCA drives TT+/-, RT+/- drives RTXCA if J16 pins 3-5 are connected.
high	low	Half duplex send	TXDA drives SD+/-, (RD+/- drives RXDA), TRXCA drives TT+/-, (RT+/- drives RTXCA if J16 pins 3-5 are connected).
high (NOTE)	high (NOTE)	Master (drivers off) (NOTE)	SD+/- not driven, RD+/- drives RXDA, TT+/- not driven, RT+/- drives RTXCA if J16 pins 3-5 are connected.
low	low	Slave (drivers on)	TXDA drives RD+/-, SD+/- drives RXDA, TRXCA drives RT+/-, TT+/- drives RTXCA if J16 pins 3-5 are connected.
low	low	Half duplex receive	(TXDA drives RD+/-), SD+/- drives RXDA, (TRXCA drives RT+/-), TT+/- drives RTXCA if J16 pins 3-5 are connected.
low	high	Slave (drivers off)	RD+/- not driven, SD+/- drives RXDA, RT+/- not driven, TT+/- drives RTXCA if J16 pins 3-5 are connected.
low	high	Half duplex receive	RD+/- not driven, SD+/- drives RXDA, RT+/- not driven, TT+/- drives RTXCA if J16 pins 3-5 are connected.

NOTE: RESET causes DTRA (DTR/REQA) and RTSA to enter this state.

MVME133 has only clocks and data; the RS-485 port has no hardware handshakes.



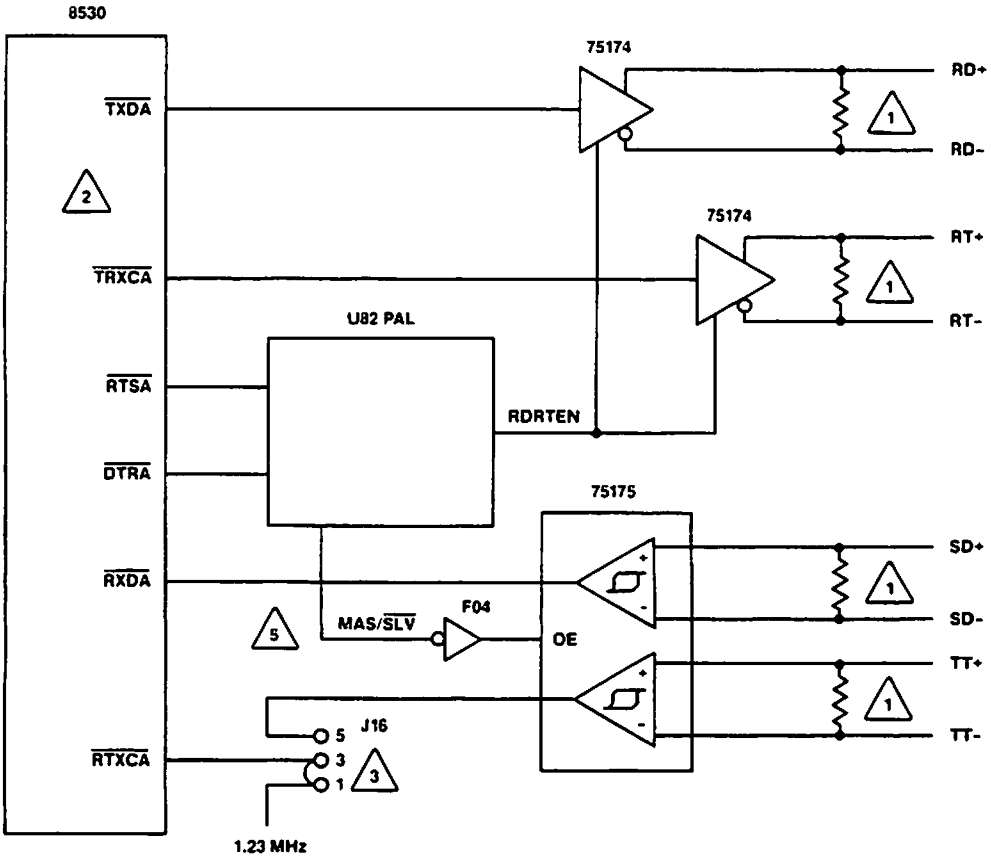
NOTES: 1 PART OF RESISTOR PACK SIP. 120 OHMS.

2 AUTO-ENABLE BIT FOR PORT 'A' MUST BE CLEARED.

3 FACTORY CONFIGURATION.

4 $\overline{\text{DTRA}}$ = 1 TO INDICATE 'MASTER'.

Figure 4-7. RS-485 Port Configured as Master (To DCE)



NOTES: 1 PART OF RESISTOR PACK SIP, 120 OHMS.

2 AUTO-ENABLE BIT FOR PORT 'A' MUST BE CLEARED.

3 FACTORY CONFIGURATION.

5 $\overline{\text{DTRA}}$ = 0 TO INDICATE 'SLAVE'.

Figure 4-8. RS-485 Port Configured as Slave (To DTE)

RS-232C Port

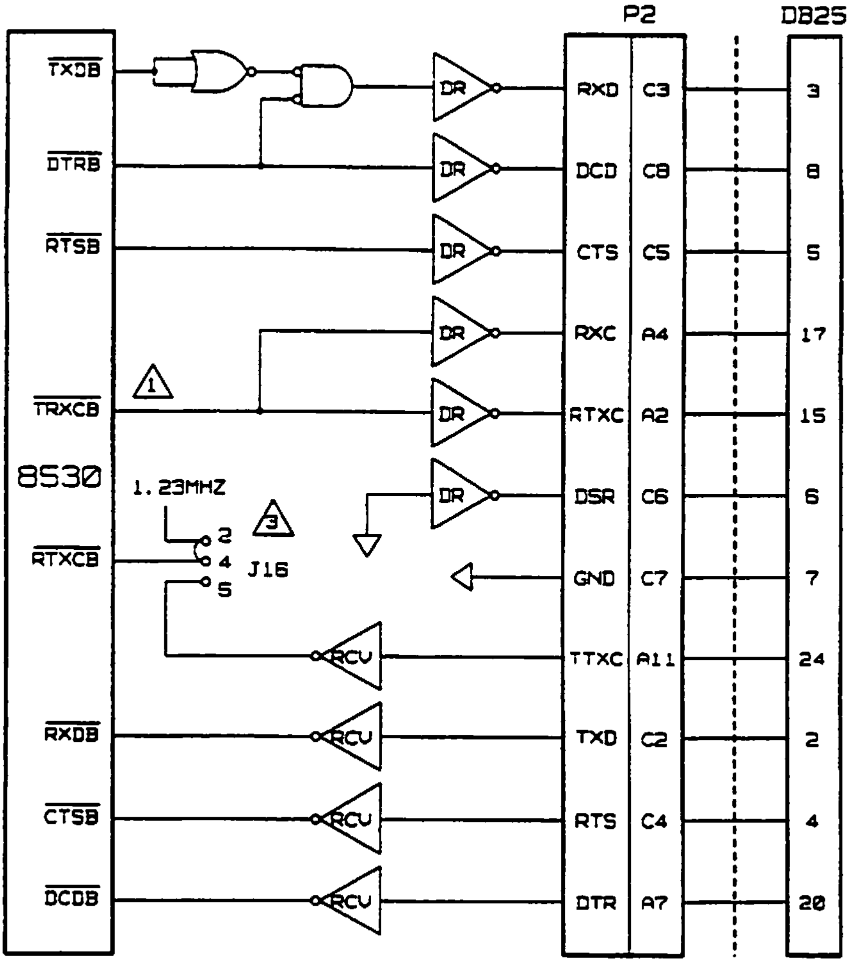
Port B of the Z8530 uses RS-232C drivers and receivers. All of the buffers and the configuration headers are on the MVME133. This port may be configured either as a DTE or as a DCE by using J13. (Refer to chapter 2.) Figure 4-9 shows the DCE configuration of the port, and Figures 4-10 through 4-12 show the DTE configurations of the port. The MVME133 also provides an external ability to disable the TXDB pin of the Z8530. When the DTRB pin of the Z8530 is high, the TXDB pin is disabled to the RS-232C port, and when the DTRB pin is low, the TXDB pin is enabled to the RS-232C port. The DTRB pin is also an RS-232C signal line. DTRB is set high by a reset to the Z8530.

The RS-232C lines are routed to P2 rows A and C. An external cable may be connected to P2 and a DB-25 connector crimped directly onto it. For shielding purposes, the DB-25 should be mounted to a back panel that is mounted to the chassis, and connection should be made between the back panel and the shielding metal of the DB-25.

Refer to Appendix C for an example of setting up software for serial port B.

CAUTION

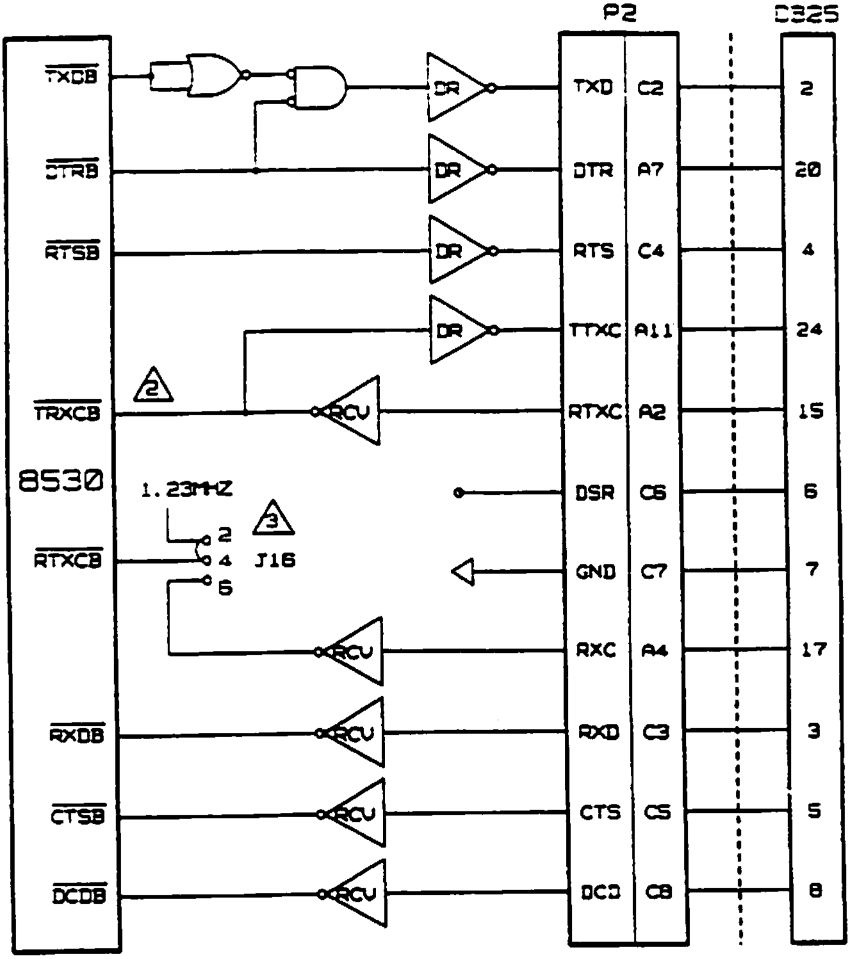
Set up serial port B for the same hardware characteristics (J13 and J16) as used for the software characteristics (such as in Appendix C) or the port will not operate properly.



NOTE: △1 TRXCB MUST BE PROGRAMMED AS OUTPUT

△3 FACTORY CONFIGURATION

Figure 4-9. RS-232C Port Configured as DCE (To Terminal)



NOTE: 2 TRXCB MUST BE PROGRAMMED AS INPUT

3 FACTORY CONFIGURATION

Figure 4-10. RS-232C Port Configured as DTE (To Modem)

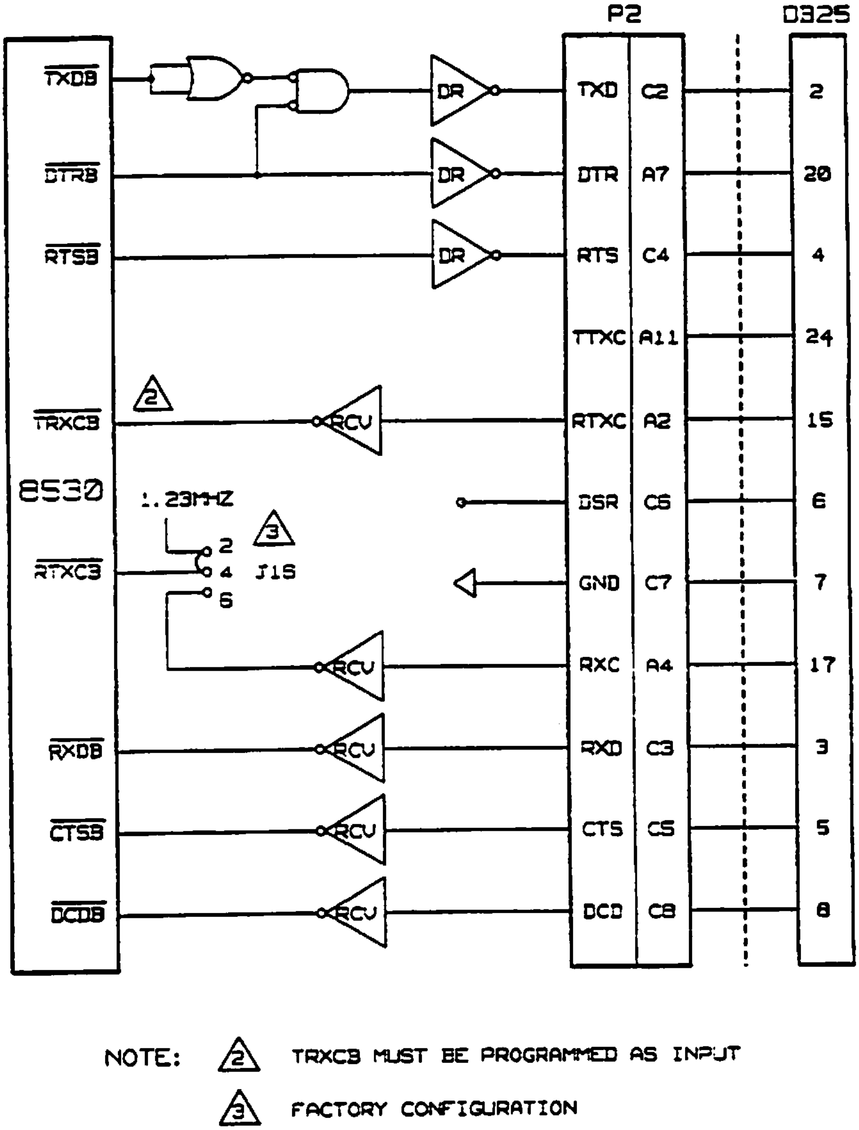
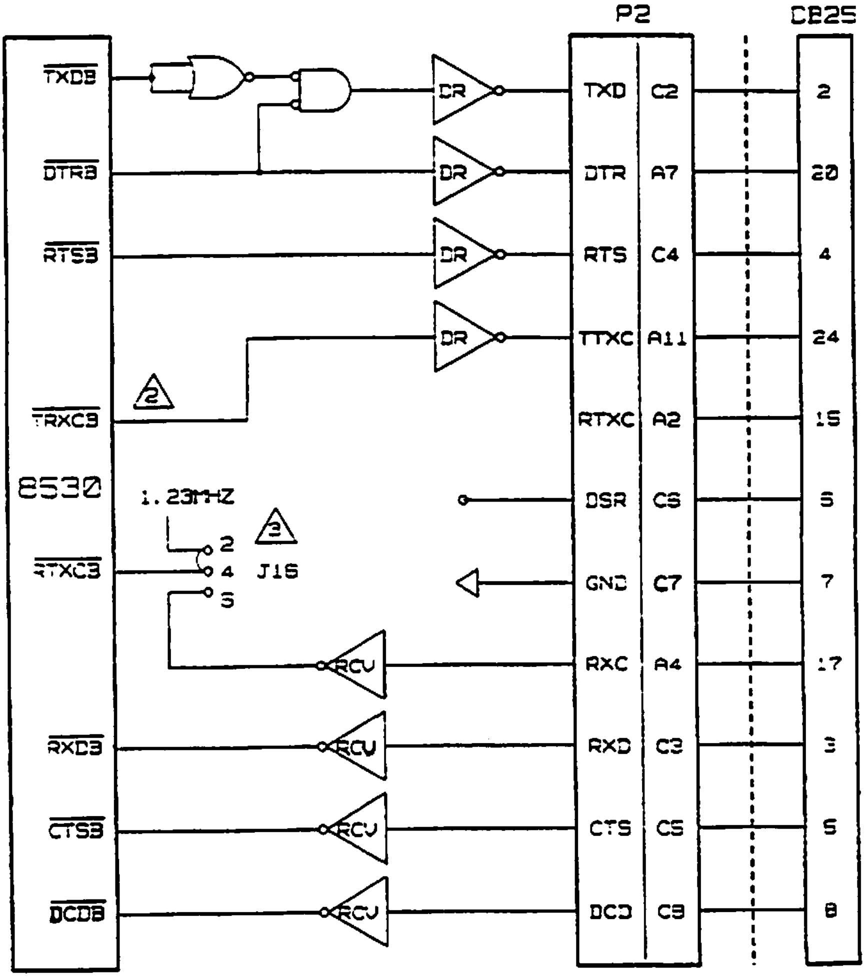


Figure 4-11. RS-232C Port Configured as DTE (To Modem) with TTXC Not Used



NOTE: △2 TRXCB MUST BE PROGRAMMED AS OUTPUT

△3 FACTORY CONFIGURATION

Figure 4-12. RS-232C Port Configured as DTE (To Modem) with RTXC Not used

APPENDIX A
U82 PROGRAMMABLE ARRAY LOGIC PROGRAM

The programmable array logic (PAL) chip U82 is a 20-pin I.C. used for address selection for extended addressing and for defining functions RTSA* and DTRA* of the RS-485 port (port A) of the Z8530 chip.

The pinout for U82 is:

Pin Number	Signal Mnemonic	Pin Number	Signal Mnemonic
1	VA26	11	/RTSA
2	VA25	12	SLVDREN
3	VA24	13	/SPIO
4	VA29	14	/DCDA
5	VA30	15	/EXTEST
6	VA31	16	/CTSA
7	VA28	17	MASDREN
8	VA27	18	MASTER
9	/DTRA	19	/EXTADR
10	GND	20	VCC

Outputs for U82 all depend on /EXTEST being true (pin 15 high, no external test being performed). The output equations when /EXTEST is high are:

U82 PROGRAMMABLE ARRAY LOGIC PROGRAM

Equation	Remarks
EXTADR = /VA31*/VA30*/VA29*/VA28 */VA27*/VA26*/VA25*/VA24	Base address is $00XXXXXX.
/MASTER = DTRA	DTRA* = 1 implies "master".
/MASDREN = /MASTER + /RTSA	Not "master" or disabled. RTSA* = 0 implies enable.
/SLVDREN = MASTER + /RTSA	Not "slave" or disabled. RTSA* = 0 implies enable.
CTSA = /VCC	Always deactivated.
DCDA = /VCC	Always deactivated.
SPIO = /VCC	Spare – driven high.

APPENDIX B
RS-232C INTERCONNECTIONS

The RS-232C standard is the most widely used interface between terminals and computers or modems, and yet it is not fully understood. This is because all the lines are not clearly defined, and many users do not see the need to conform for their applications. A system should easily connect to any other. Many times designers think only of their own equipment, but the state-of-the-art is computer-to-computer or computer-to-modem operation.

The RS-232C Standard was originally developed by the Bell System to connect terminals via modems. Therefore, several handshaking lines were included. In many applications these are not needed, but since they permit diagnosis of problems, they are included in many applications.

Table A-1 lists the standard RS-232C interconnections. To interpret this information correctly it is necessary to know that RS-232C is intended to connect a terminal to a modem. When computers are connected to computers without modems, one of them must be configured as a terminal and the other as a modem. Because computers are normally configured to work with terminals, they are said to be configured as a modem. Also, the signal levels must be between +3 and +15 volts for a high level, and between -3 and -15 volts for a low level. Any attempt to connect units in parallel may result in out of range voltages and is not allowed by the RS-232C specifications.

Table B-1. RS-232C Interconnections

Pin Number	Signal Mnemonic	Signal Name and Description
1		Not used.
2	TXD	TRANSMIT DATA - data to be transmitted is furnished on this line to the modem from the terminal.
3	RXD	RECEIVE DATA - data which is demodulated from the receive line is presented to the terminal by the modem.
4	RTS	REQUEST TO SEND - RTS is supplied by the terminal to the modem when required to transmit a message. With RTS off, the modem carrier remains off. When RTS is turned on, the modem immediately turns on the carrier.
5	CTS	CLEAR TO SEND - CTS is a function supplied to the terminal by the modem which indicates that it is permissible to begin transmission of a message. When using a modem, CTS follows the off-to-on transition of RTS after a time delay.
6	DSR	DATA SET READY - data set ready is a function supplied by the modem to the terminal to indicate that the modem is ready to transmit data.
7	SIG-GND	SIGNAL GROUND - common return line for all signals at the modem interface.
8	DCD	DATA CARRIER DETECT - sent by the modem to the terminal to indicate that a valid carrier is being received.
9-14		Not used.
15	TXC	TRANSMIT CLOCK - this line clocks output data to the modem from the terminal.

Table B-1. RS-232C Interconnections (cont'd)

Pin Number	Signal Mnemonic	Signal Name and Description
16		Not used.
17	RXC	RECEIVE CLOCK - this line clocks input data from a terminal to a modem.
18,19		Not used.
20	DTR	DATA TERMINAL READY - a signal from the terminal to the modem indicating that the terminal is ready to send or receive data.
21		Not used.
22	RI	RING INDICATOR - RI is sent by the modem to the terminal. This line indicates to the terminal that an incoming call is present. The terminal causes the modem to answer the phone by carrying DTR true while RI is active.
23		Not used.
24	TXC	TRANSMIT CLOCK - Same as TXC on pin 15.
25	BSY	BUSY - A positive EIA signal applied to this pin causes the modem to go off-hook and make the associated phone busy.

NOTES: 1. High level = +3 to +15 volts. Low level = -3 to -15 volts.

2. RS-232C is intended to connect a terminal to a modem. When computers are connected to computers without modems, one of the computers must be configured as a modem and the other as a terminal.

There are several levels of conformance that are appropriate for typical RS-232C interconnections. The bare minimum requirement is the two data lines and a ground. The full version of RS-232C requires 12 lines and accommodates automatic dialing, automatic answering, and synchronous transmission. A middle-of-the-road approach is illustrated in Figure A-1.

One set of handshaking signals frequently implemented are RTS and CTS. CTS is used in many systems to inhibit transmission until the signal is high. In the modem application, RTS is turned around and returned as CTS after 150 microseconds. RTS is programmable in some systems to work with the older type 202 modem (half duplex). CTS is used in some systems to provide flow control to avoid buffer overflow. This is not possible if modems are used. It is usually necessary to make CTS high by connecting it to RTS or to some source of +12 volts such as the resistors shown in Figure A-1. It is also frequently jumpered to an MC1488 gate which has its inputs grounded (the gate is provided for this purpose). Another signal used in many systems is DCD. The original purpose of this signal was to tell the system that the carrier tone from the distant modem was being received. This signal is frequently used by the software to display a message like CARRIER NOT PRESENT to help the user to diagnose failure to communicate. Obviously, if the system is designed properly to use this signal, and it is not connected to a modem, the signal must be provided by a pullup resistor or gate as described before (see Figure A-1). Many modems expect a DTR high signal and issue a DSR. These signals are used by software to help prompt the operator about possible causes of trouble. The DTR signal is used sometimes to disconnect the phone circuit in preparation for another automatic call. It is necessary to provide these signals in order to talk to all possible modems (see Figure A-1). Figure A-1 is a good minimum configuration that almost always works. If the CTS and DCD signals are not received from the modem, the jumpers can be moved to artificially provide the needed signal. Figure A-2 shows a way that an RS-232C connector can be wired to enable a computer to connect to a basic terminal with only three wires. This is because most terminals have a DTR signal that is ON and can be used to pullup the CTS, DCD and DSR signals. Two of these connectors wired back-to-back can be used. It must be realized that all the handshaking has been bypassed and possible diagnostic messages do not occur. Also, the TX and RX lines may have to be crossed since TX from a terminal is outgoing but the TX line on a modem is an incoming signal.

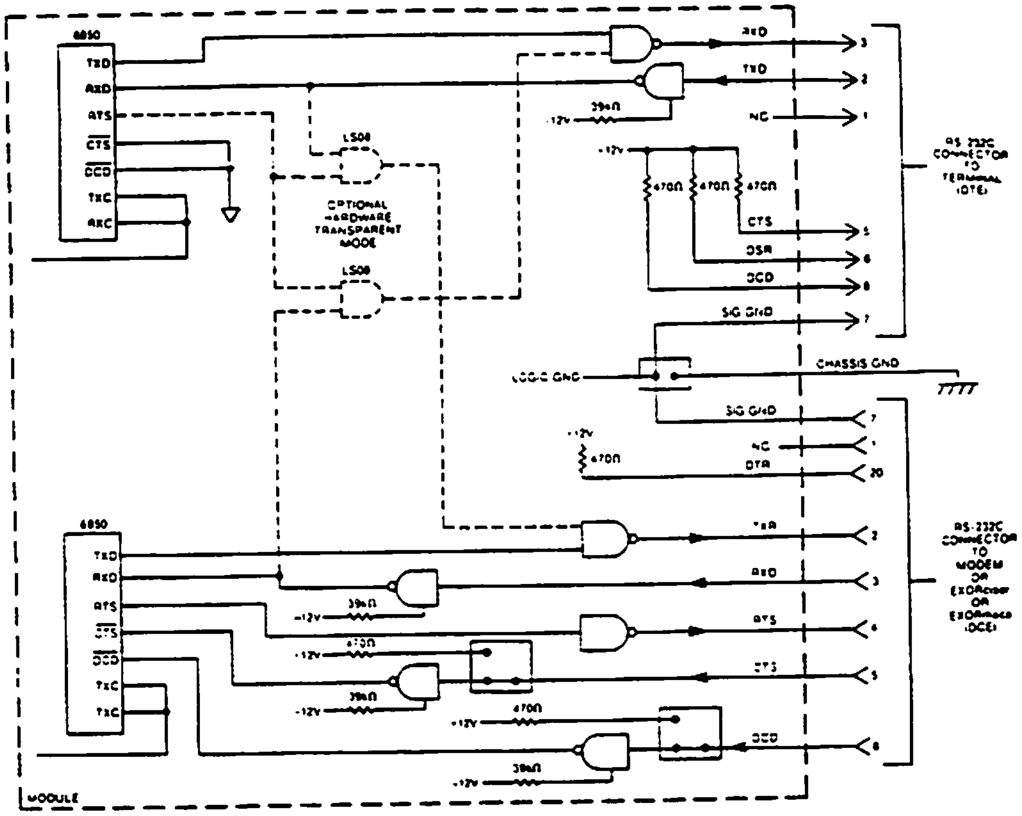


Figure B-1. Middle-of-the-Road RS-232C Configuration

Another subject that needs to be considered is the use of ground pins. There are two pins labeled GND. Pin 7 is the SIGNAL GROUND and must be connected to the distant device to complete the circuit. Pin 1 is the CHASSIS GROUND, but it must be used with care. The chassis is connected to the power ground through the green wire in the power cord and must be connected to the chassis to be in compliance with the electrical code. The problem is that when units are connected to different electrical outlets, there may be several volts difference in ground potential. If pin 1 of the devices are interconnected with a cable, several amperes of current could result. This not only may be dangerous for the small wires in a typical cable, but could result in electrical noise that could cause errors. That is the reason that Figure A-1 shows no connection for pin 1. Normally, pin 7 should only be connected to the CHASSIS GROUND at one point and, if several terminals are used with one computer, the logical place for that point is at the computer. The terminals should not have a connection between the logic ground return and the chassis.

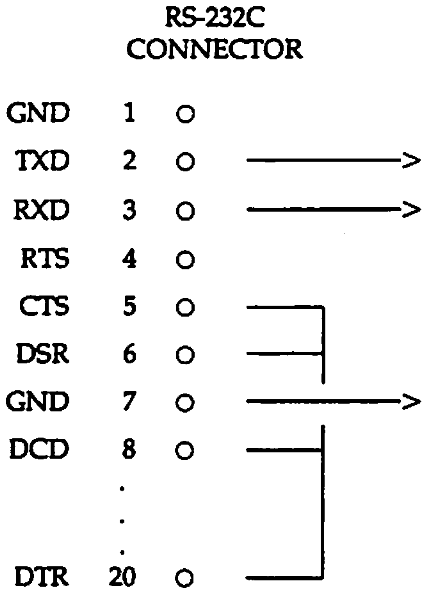


Figure B-2. Minimum RS-232C Connection

APPENDIX C
Z8530 SERIAL PORT B SETUP EXAMPLE

This example sets up port B (the RS-232C port) of the Z8530 as follows:

- 9600 baud, asynchronous only
- Interrupt on Received Character, Transmitter Buffer Ready, and External Status Change with common interrupt vector.

SETUP:

Move #$30 into SIOB_WR0	($FFFA0000)	Clear receiver error status.
Move #$10 into SIOB_WR0	($FFFA0000)	Clear external status interrupts.
Move #$09 into SIOB_WR0	($FFFA0000)	Select register 9.
Move #$40 into SIOB_WR0	($FFFA0000)	Reset channel B.
Move #$0A into SIOB_WR0	($FFFA0000)	Select register 10.
Move #$00 into SIOB_WR0	($FFFA0000)	Make sure NRZ format is set.
Move #$0E into SIOB_WR0	($FFFA0000)	Select register 14.
Move #$82 into SIOB_WR0	($FFFA0000)	Disable baud rate generator.
Move #$04 into SIOB_WR0	($FFFA0000)	Select register 4.
Move #$44 into SIOB_WR0	($FFFA0000)	Divide by 16, no parity, one stop bit.
Move #$03 into SIOB_WR0	($FFFA0000)	Select register 3.
Move #$C1 into SIOB_WR0	($FFFA0000)	Receiver: eight bits, receiver enabled.
Move #$05 into SIOB_WR0	($FFFA0000)	Select register 5.
Move #$EA into SIOB_WR0	($FFFA0000)	Transmitter: eight bits, transmitter enabled, RTS on, DTR on.
Move #$0C into SIOB_WR0	($FFFA0000)	Select register 12.
Move #$02 into SIOB_WR0	($FFFA0000)	Lower byte of time constant.
Move #$0D into SIOB_WR0	($FFFA0000)	Select register 13.
Move #$00 into SIOB_WR0	($FFFA0000)	Higher byte of time constant.
Move #$0B into SIOB_WR0	($FFFA0000)	Select register 11.

Z8530 SERIAL PORT B SETUP EXAMPLE

Move #$56 into SIOB_WR0	($FFFA0000)	RX clock = BR Generator output, TX clock = BR Generator output, TRXC = output = BR Generator output.
Move #$0E into SIOB_WR0	($FFFA0000)	Select register 14.
Move #$81 into SIOB_WR0	($FFFA0000)	BR Generator clock source = RTXC pin.
Move #$01 into SIOB_WR0	($FFFA0000)	Select register 1.
Move #$11 into SIOB_WR0	($FFFA0000)	Interrupt on all Received Character or Special Condition. Also enable external interrupts.
Move #$0F into SIOB_WR0	($FFFA0000)	Select register 15.
Move #$80 into SIOB_WR0	($FFFA0000)	Enable Break/Abort interrupts.
Move #$02 into SIOB_WR0	($FFFA0000)	Select register 2.
Move #$80 into SIOB_WR0	($FFFA0000)	Interrupt vector number. ($80 => vector offset = $200.)
Move #$09 into SIOB_WR0	($FFFA0000)	Select register 9.
Move #$08 into SIOB_WR0	($FFFA0000)	Master interrupt enable. Status information NOT to be included in the vector passed to the MPU.

NOTE

To minimize overhead in the interrupt handling routine, status information may be selected to be included in the vector(s). The vector, then, points directly at the appropriate handling routine according to the interrupt cause. If the Vector-Status-Include (VSI) is set and the content in the vector register is $80, then the vector passed to the MPU is:

$80 (vector offset = $200) for Channel B Transmitter Buffer Empty or

$82 (vector offset = $208) for Channel B External Status Change or

$84

$86 (vector offset = $218) for Channel B Special Received Character.

For this example, place the address of the common interrupt handler at offset $200 in the vector table.

INTERRUPT HANDLER:

Move #$03 into SIOB_WR0	($FFFA0000)	Select register 3.
Read from SIOB_RR0	($FFFA0000)	Read the Read Register 3 for interrupt cause.

Investigate the interrupt pending bits to determine the cause. Branch to the appropriate handling routine.

TRANSMIT A CHARACTER:

If Transmitter Buffer Empty interrupt is desired, it must be enabled before outputting a character or else the interrupt will not occur.

Move #$01 into SIOB_WR0	($FFFA0000)	Select register 1.
Move #$13 into SIOB_WR0	($FFFA0000)	Enable transmitter interrupt.
Move output character into SIOB_TXDATA	($FFFA0001)	Transmit a character.

TRANSMITTER BUFFER EMPTY INTERRUPT HANDLER:

Move #$01 into SIOB_WR0	($FFFA0000)	Select register 1.
Move #$11 into SIOB_WR0	($FFFA0000)	Disable transmitter interrupt.
Move #$38 into SIOB_WR0	($FFFA0000)	Reset highest Interrupt-Under-Service (IUS).

```
Are there more characters to output?
    If Yes, go do TRANSMIT A CHARACTER.
    If No, return from exception.
```

RECEIVED CHARACTER INTERRUPT HANDLER:

Move #$01 into SIOB_WR0	($FFFA0000)	Select register 1.

Z8530 SERIAL PORT B SETUP EXAMPLE

Read from SIOB_RR0	($FFFA0000)	Read the Read Register 1 to check for status.

Check for framing error, receiver overrun, and parity errors.

Read from SIOB_RXDATA	($FFFA0001)	Read received character.
Move #$38 into SIOB_WR0	($FFFA0000)	Reset highest IUS.

EXTERNAL STATUS CHANGE INTERRUPT HANDLER:

Break	– either start of break or end of break.
CTS	– a transition has occurred on the CTS input pin.
DCD	– a transition has occurred on the DCD input pin.

Move #$00 into SIOB_WR0	($FFFA0000)	Select register 0.
Read from SIOB_RR0	($FFFA0000)	Read the Read Register 0 for status.
Move #$10 into SIOB_WR0	($FFFA0000)	Reset external status interrupt.

Take actions as necessary.

If break bit is low, which is the end of a break, a null character is still in the receive buffer. It should be read and discarded.

Read data from SIOB_RXDATA	($FFFA0001)	Read null character.

Return from exception.

APPENDIX D
MC68901 MFP TIMER A SETUP EXAMPLE

The following example sets up the MC68901 MFP timer A (software tick timer) to interrupt the MPU periodically every 10 msec.

SETUP:

Clear bit #5 of MFP_IERA	($FFF80007)	Disable timer A interrupts.
Move #$10 into MFP_TACR	($FFF80019)	Reset and stop timer A.
Move #$7B into MFP_TADR	($FFF8001F)	Load count down value. (Refer to Table 1 in this Appendix.)
Move #$06 into MFP_TACR	($FFF80019)	Delay mode, prescaler = 100.
Move #$68 into MFP_VR	($FFF80017)	Set starting vector at $60. Set software interrupt mode.

NOTE

The vector passed to the MPU for the timer A interrupt is $6D => vector offset = 4 x $6D = $1B4.

Move #$DF into MFP_IPRA	($FFF8000B)	Clear timer A interrupt pending bit (bit #5 of IPRA).
Move #$DF into MFP_ISRA	($FFF8000F)	Clear timer A interrupt-in-service bit (bit #5 of ISRA).
Set bit #5 of MFP_IMRA	($FFF80013)	Unmask timer A interrupts.
Set bit #5 of MFP_IERA	($FFF80007)	Enable timer A interrupts.

TIMER A INTERRUPT HANDLER:

MC68901 MFP TIMER A SETUP EXAMPLE

Read MFP_ISRA ($FFF8000F) Read interrupt-in-service register A.

Investigate MFP_ISRA to determine if it was actually from timer A.

Take actions as necessary.

Move #$DF into MFP_ISRA ($FFF8000F) Clear timer A interrupt-in-service bit (bit #5 of ISRA).

Return from exception.

COUNTDOWN CALCULATION:

The countdown values used during setup may be calculated using the following equation:

$$CD = (TI \times TO) / PS$$

where: CD = countdown value to be loaded into timer data register.
TI = timer input frequency in Hertz = 1,230,769 Hertz.
TO = tick timer interrupts interval in seconds.
PS = prescaler value (4, 10, 16, 50, 64, 100, or 200).

Table D-1 contains the values for PS and CD for some selected interrupts intervals.

Table D-1. Prescaler and Countdown Values for Selected Interrupts Values

TO			CD	
µs	Sec	PS	Hex	Decimal
1.0	0.0010	10	$7B	123
5.0	0.0050	50	$7B	123
10.0	0.0100	100	$7B	123
20.0	0.0200	100	$F6	246
40.0	0.0400	200	$F6	246
41.6	0.0416	200	$00	256

INDEX

When using this index, keep in mind that a page number indicates only where referenced material begins; it may extend to the following page or pages.