PME 68-1B Manual

Rev. 2

Issue 5

Publication No. 421/HH/23144/000

Radstone Technology plc
Water Lane,
Towcester,
Northants.,
NN12 7JN

Telephone: 0327 50312
Telex: 31628 RADSTN G

January 1990

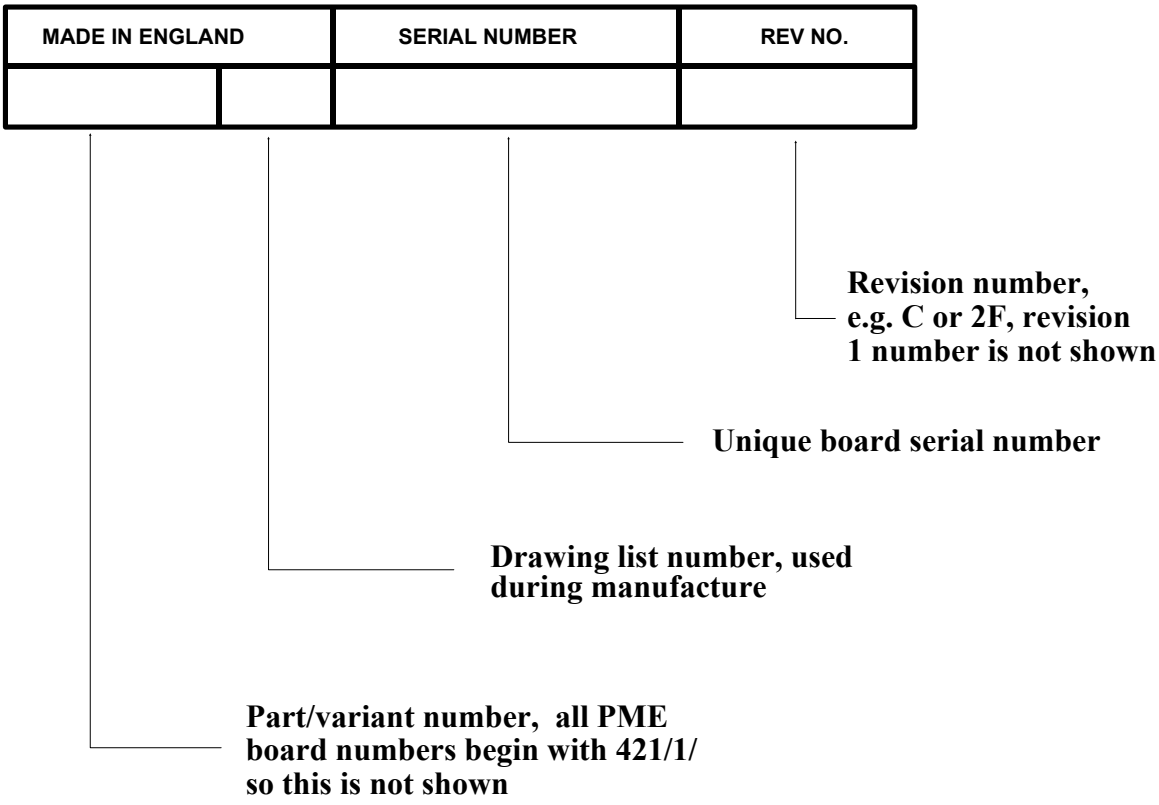
Trademarks

RADSTONE the RADSTONE symbol, PME, and PLUM are trademarks of Radstone Technology plc.

VERSAdos is a trademark of Motorola.

Board Identification

All Radstone baoards can be identified by a label fitted to the component side of P1.



This publication describes the PME 68-1B at Revision state Rev.2 (all letter codes).

Table of Contents

Chapter 4 - Configuration

Chapter 5 - PME68/Monitor

Chapter 1 - Introduction

The PME 68-1B CPU is a single board computer which combines a powerful 16 bit microprocessor, the Motorola 68000, with 128k or 512k bytes of dynamic RAM (DRAM), up to 256k bytes of EPROM or EPROM + SRAM, a programmable real time clock, plus three serial and one parallel input/output (I/O) interfaces. It provides reliable, high speed processing at low cost, in addition to providing 3 interfaces; typically two would be used for a host computer and terminal, and the third for a remote interface for a second terminal, printer, or back-up peripheral.

Access to all these on-board devices is provided by an EPROM resident Monitor, (PME68 or PLUM) which is supplied with the board. This powerful software package can be used for program development and debugging.

The PME 68-1B is supported by Radstone VERSAdos operating system. This software provides for Realtime, Multiuser, Multitasking systems.

The PME 68-1B is a sophisticated VMEbus CPU board which may be used in a multiprocessor environment. It may also be used as a single board computer using its on-board I/O and timer functions.

Variants

The nature of the board variant is indicated by the last three digits of the part number. e.g. ---/-/-----/XXX

The first digit:

/1XX= Standard board with PME68/Monitor
/2XX= VERSAdos + VERSAdos support with PLUM Monitor
/3XX= Special Variants

The third digit identifies processor speed and DRAM combination:

XX0 =	68000 8 MHz.	128k DRAM
XX1 =	68000 10 MHz.	128k DRAM
XX2 =	68000 8 MHz.	512k DRAM
XX3 =	68000 10 MHz.	512k DRAM

Figure 1 The PME 68-1B Board

Photograph not available in PDF

PME68-1B is available with an 8 MHz or 10 MHz, 68000 CPU and various capacities of DRAM. The standard variants are:

Order No.	Description	Part No.
PME68-1B/100	8 MHz 68000, 128k bytes of DRAM	421/1/23144/100
PME68-1B/101	10 MHz 68000, 128k bytes of DRAM	421/1/23144/101
PME68-1B/102	8 MHz 68000, 512k bytes of DRAM	421/1/23144/102
PME68-1B/103	10 MHz 68000, 512k bytes of DRAM	421/1/23144/103

This manual provides a general operating description of all variants of the PME 68-1B and includes information for installation and troubleshooting in Chapter 4, plus details of the PME68/Monitor in Chapter 5.

Certain 'application specific' boards are factory configured for individual users. For reasons of confidentiality, detailed reference is not made to these boards, but default (factory set) link information is given.

Details of the 68000 Processor and the Parallel I/O Interface and Timer are given in the respective Motorola data sheets available on request from your supplier.

Accessing the VME bus after a power-up reset

Occasionally, after a power-up reset, access to the VME bus may be denied. To overcome this feature a dummy access to any PIT register should be made prior to accessing the VME bus.

WARNING: See Chapter 4 before installing the battery.

Figure 2 The PME 68-1B Front Panel

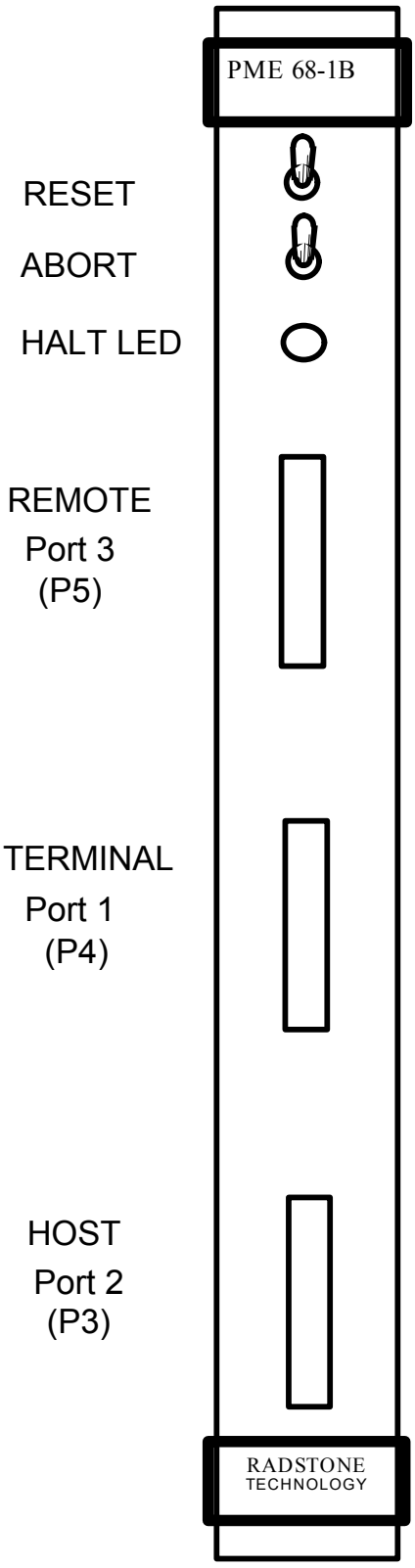


Figure 3 PME 68-1B Functional Block Diagram

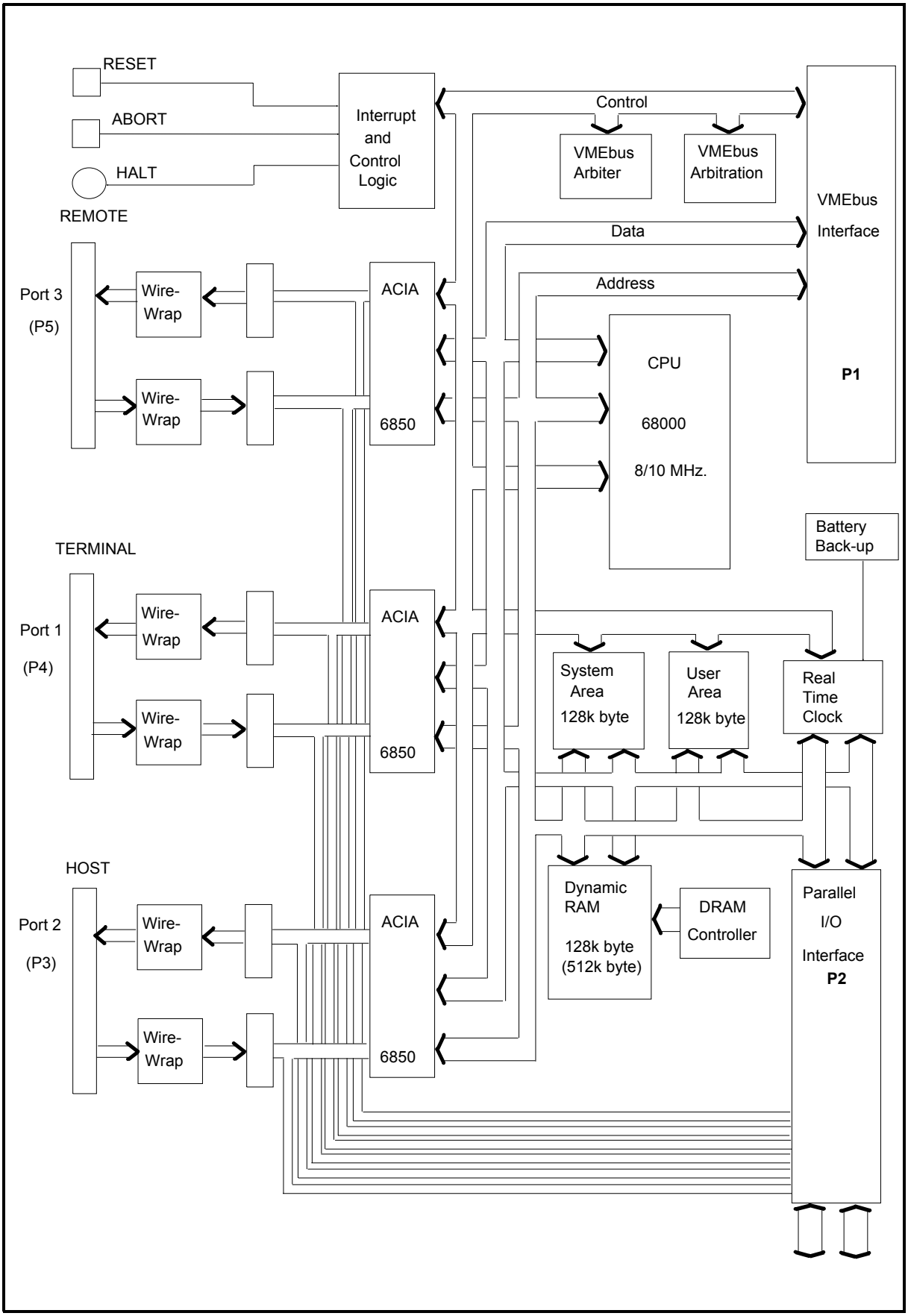
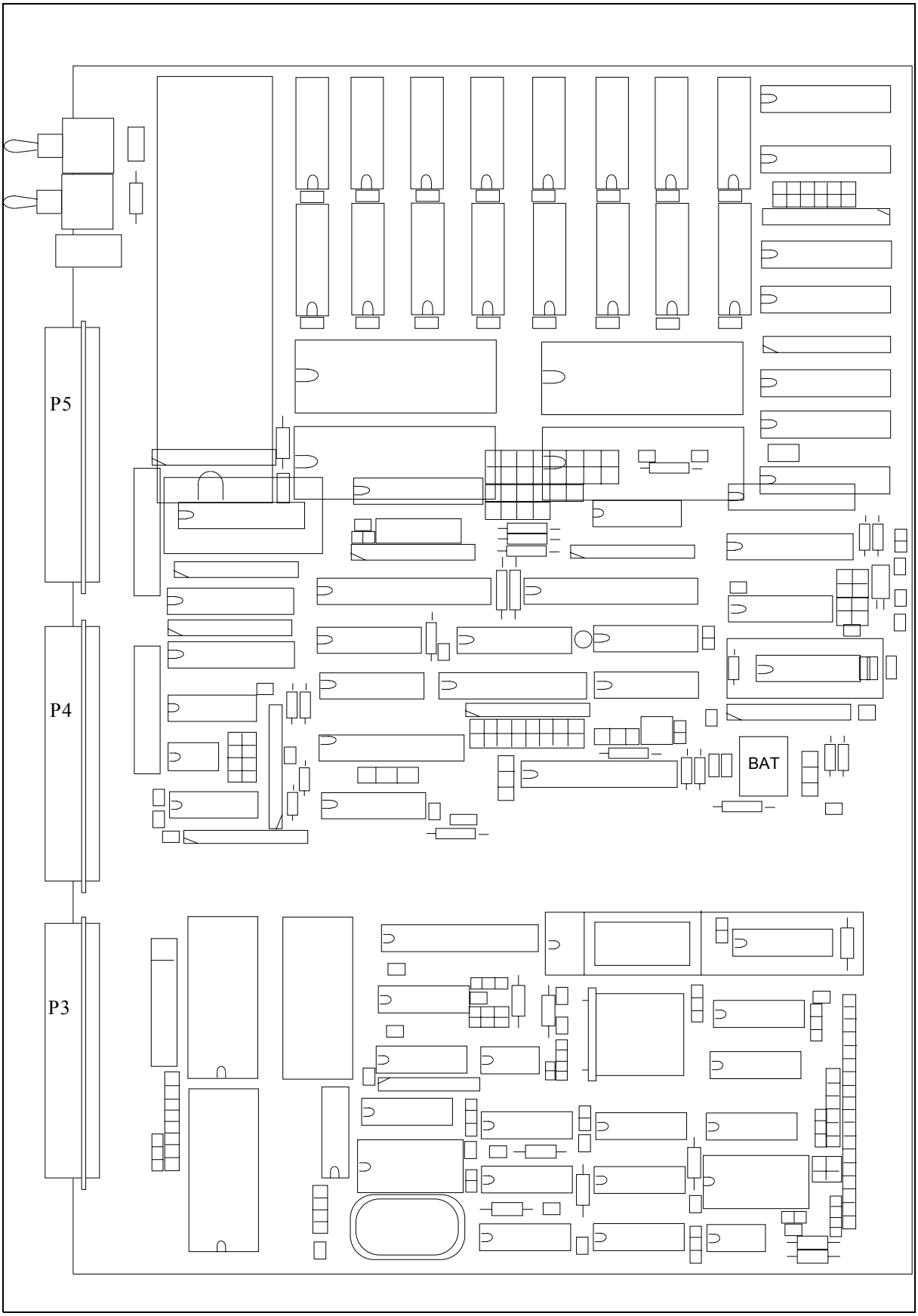


Figure 4 Component Layout Diagram



Operational Overview

The operation of the PME 68-1B board is based around an 8 MHz or 10 MHz 68000 microprocessor unit with an asynchronous 16-bit data bus and 24-bit address bus. The address bus provides a direct memory addressing range of 16M bytes. The CPU communicates with on-board memory and I/O devices via an on-board local bus.

The address range from $000008 to $0FFFFF (1M byte) is assigned to on-board memory and I/O devices. All other addresses are external on the VMEbus.

The PME 68-1B board contains a number of features that allow it to act as the CPU of a powerful system, one of a number of CPUs in a multi-processor system, or as a single board computer.

Features

The PME 68-1B board contains the following features:

- Motorola 68000 CPU with 8 MHz clock, variants 100 and 102 or with 10 MHz clock, variants 101 and 103
- Fully implemented VMEbus interface
- 128k bytes of DRAM using 64k bit devices
  512k bytes DRAM using 256k bit devices. Access time 280ns. Distributed refresh 300ns every 15µs
- Up to 128k bytes of EPROM for system firmware
- Up to 64k bytes of SRAM/or 128k byte EPROM for the user
- Real time clock with battery back-up
- Three RS-232C serial interfaces, 110 to 38400 baud
- Up/down load to/from a host computer (S Record format)
- Parallel interface and 24-bit timer with 5 bit prescaler
- Local interrupt handling via auto-interrupt vectors
- Freely selectable address range for short I/O address accesses
- Single level bus arbiter (Can be disabled to use an external arbiter)
- Powerful Monitor firmware including a line by line assembler/disassembler, register/memory manipulation routines and special I/O handling routines
- RESET and ABORT switches
- Transparent mode

The hardware specification of the PME 68-1B is given in Chapter 2.

I/O Facilities

PME68-1B boards have three RS-232C interfaces, identified as Terminal (Port 1), Host (Port 2) and Remote (Port 3) as shown in Figures 2 and 3.

Port 1 (connector P4) may be connected to a standard terminal to load and debug user programs under control of the Monitor firmware.

Port 2 (connector P3) may be used for up/down loading of user programs and data, or in the transparent mode to directly interface the Port 1 connected terminal to a host computer.

Port 3 (connector P5) is a universal port capable of supporting: a serial printer, a floppy disk system, a second terminal (multiuser station), or an external user task.

A 64-pin male connector (P2) located at the rear of the board, gives access to the Parallel Interface and Timer module (PIT). The PIT provides 24 I/O lines and 4 control lines plus a 24-bit timer with 5-bit prescaler. P2 can be configured to give access to all selected signals of the 3 serial ports.

Memory

The board is normally supplied with a 64k byte EPROM containing the PME68/ Monitor. PLUM is supplied with variants 2XX or on request. Two additional JEDEC compatible sockets are available and will accept either additional EPROMs or SRAMs. Details of the firmware options are available from your supplier.

On-board DRAM is provided in the form of high performance, 64k bit or 256k bit devices.

Real Time Operation

The programmable Real Time Clock can be used in conjunction with a multiuser/ multitasking operating system for real time applications.

Chapter 2 - Specification

General

Microprocessor	Motorola 68000L Type 8 MHz on Variants 100 and 102 10 MHz on Variants 101 and 103	
Bus Compliance	VMEbus Rev. C.1	
Data	D08, D16	
Address	A16, A24	
Parallel I/O and Timer	68230 type PIT 16 data lines and 8 control lines, configurable as a Centronics type parallel interface Timer, one 24-bit timer with 5-bit prescaler	
Serial I/O	Three 6850 type ACIAs configured as RS-232C interfaces. Strap selectable baud rates 110 to 9600 baud or 440 to 38400 baud	
Real Time Clock	58167A programmable real time clock, optional battery back-up using a 3.0V lithium battery, such as a Varta CR 1/3 N	
DRAM	128k bytes or 512k bytes	
EPROM	128k byte system area part occupied by monitor Up to 128k bytes user definable; alternatively up to 64k byte SRAM	
Power requirements	+ 5V 2.8A + 12V 220mA -12V 200mA	
Board dimensions	Double Eurocard	234 x 160 mm 9.2 x 6.3 inches
Weight	0.55 kg	1.21 lb

The PME 68-1B is designed to meet the following environmental specifications:

Operating

Temperature Range	0℃ to + 55℃ (ambient)
Cooling Requirements	A linear airflow of not less than 500 ft/min across the board is recommended
Relative humidity	Up to 95% (non-condensing)
Thermal shock	± 5℃ per minute
Altitude	-300 to + 3,000m (-1,000 to + 10,000 feet approx)
Vibration	5-100Hz. with 2g acceleration
Mechanical Shock	20g for 6ms (half sine) when mounted in a suitable racking system

MTBF

The calculated mean time between failures for the PME 68-1B is 80,000 hrs. The failure rates used in this calculation have been derived from the British Telecom Reliability Handbook using method HRD4, MIL-HDBK-217D and in-house data.

Storage

Temperature	-55℃ to + 85℃
Relative humidity	0 to 95% (non-condensing)
Thermal shock	± 10℃ per minute
Altitude	-300 to + 16,000m (-1,000 to 50,000 feet approx)
Vibration	0 to 500Hz, 2g acceleration
Mechanical Shock	20g for 6 ms (half-sine)

Chapter 3 - Functional Description

This Chapter describes the features and function of the PME 68-1B board. Configuration details are given in Chapter 4, and the PME68/Monitor details in Chapter 5. Details of the PLUM Monitor are given in the Radstone Manual, publication number 651/HH/19050/000.

Copies of the circuit diagrams are available by completing the form at the back of this manual. Please contact your supplier for PAL information.

Hardware Overview

Front Panel

The front panel of the PME 68-1B is shown in Figure 2 and contains:

(1) Reset Switch; a toggle switch, that is user defined to carry out a general VMEbus reset or a reset of all on-board devices only.

(2) Abort Switch; a toggle switch used to abort the user program and return to the resident Monitor.

(3) Halt Indicator; indicates the state of the CPU.

(4) Remote Interface Connector; used for connecting external equipment, e.g. serial printer.

(5) Terminal Interface Connector; used to communicate with a standard terminal.

(6) Host Interface Connector; used to interface the terminal (connected in 5 above) directly to a host computer.

(7) Ejector handles for easy insertion and retrieval of the board when rack mounted.

68000 Hardware Description

Processor

The 68000 processor contains sixteen 32-bit registers, one 32-bit program counter and a 16 bit status register. The first eight registers (D0 to D7) are used as data registers for byte (8-bit), word (16-bit) and long word (2 x 16-bit) data operations. The set of seven address registers (A0 to A6) and the supervisor stack pointer may be used as software stack pointers and base address registers. In addition, all 16 registers may be used for word and long word address operations, or as index registers. Table 1 shows the vector layout of the 68000.

Features of the 68000 CPU include:

- 8 x 32-bit data registers
- 7 x 32-bit address registers
- 32-bit supervisor stack pointer or 32-bit user stack pointer
- 16-bit status register
- 16Mbyte direct addressing range
- 56 powerful instruction types
- 14 different addressing modes
- 5 main data types
- Memory mapped I/O
- Asynchronous bus structure

For further details refer to the Motorola 68000 Data Sheet available from your supplier.

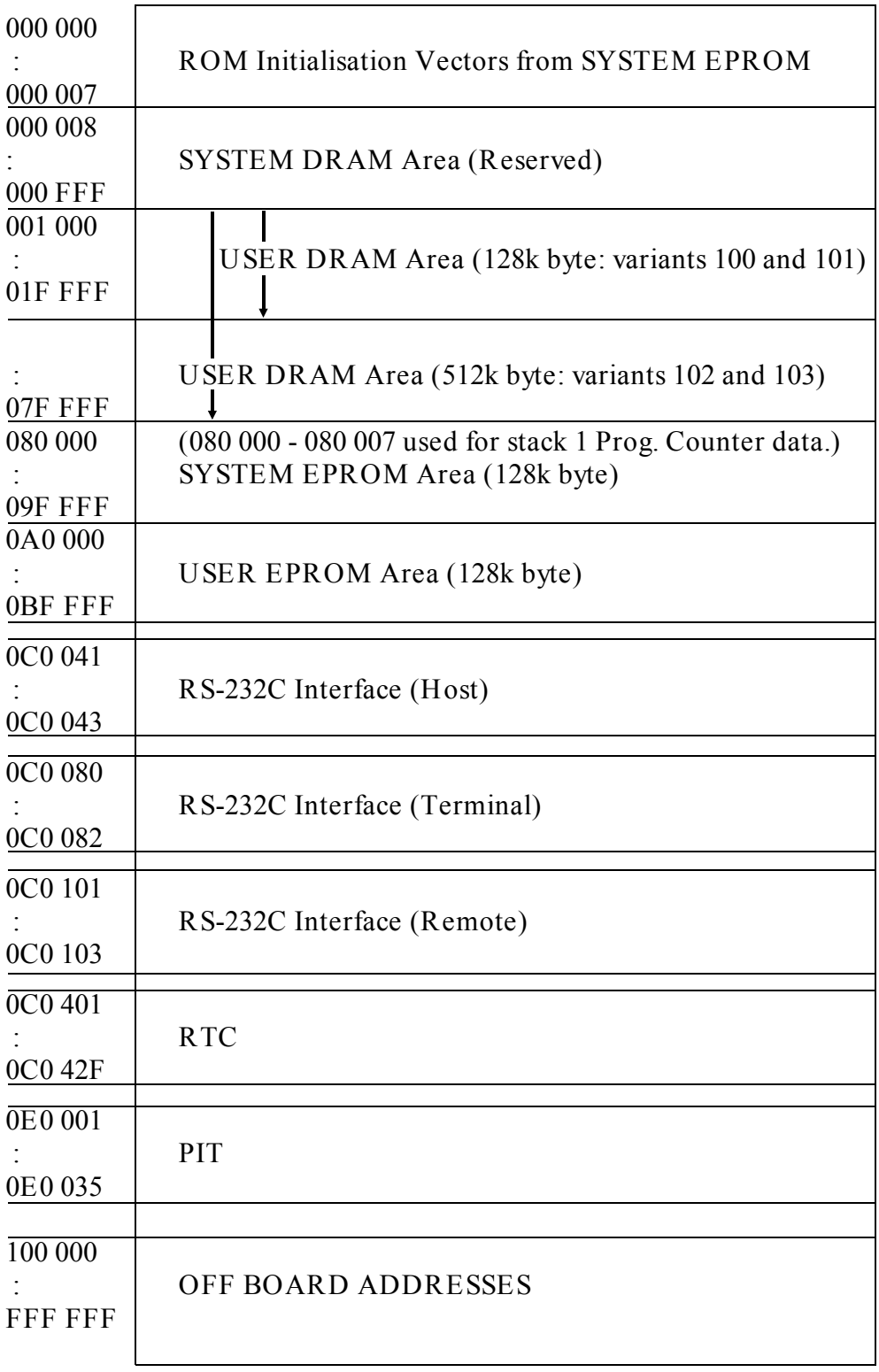
Table 1 68000 Vector Table

Vector No.(s)	Address Decimal	Hex	Space	Assignment
0	0	000	SP	Reset: Initial SSP
-	4	004	SP	Reset: Initial PC
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12*	48	030	SD	Unassigned (Reserved)
13*	52	034	SD	Unassigned (Reserved)
14*	56	038	SD	Unassigned (Reserved)
15	60	03C	SD	Uninitialised Interrupt Vector
16 to 23*	64	040	SD	Unassigned (Reserved)
	95	05F		
24	96	060	SD	Spurious Interrupt
25	100	064	SD	Level 1 Interrupt Auto-vector
26	104	068	SD	Level 2 Interrupt Auto-vector
27	108	06C	SD	Level 3 Interrupt Auto-vector
28	112	070	SD	Level 4 Interrupt Auto-vector
29	116	074	SD	Level 5 Interrupt Auto-vector
30	120	078	SD	Level 6 Interrupt Auto-vector
31	124	07C	SD	Level 7 Interrupt Auto-vector
32 to 47	128	080	SD	Trap Instruction vectors
	191	OBF		
48 to 63*	191	0C0	SD	Unassigned (Reserved)
	255	0FF		
65 to 255	256	100	SD	User Interrupt Vectors
	1023	3FF		

SP = Supervisor Program
SD = Supervisor Data
* = Vector numbers 12, 13, 14, 16 to 23 and 48 to 63 are reserved by Motorola for future enhancements. No user peripheral devices should be assigned to these numbers.

Memory

Figure 5 Memory Map of the PME 68-1B



RAM Area

The on-board dynamic RAM area is used for the exception vector table of the CPU, as a scratch pad RAM for the Resident Monitor and for user program/data. The RAM area has a capacity of 128k bytes on PME68-1B/100 and /101 and 512k bytes on /102 and /103. Typical access time is 280ns if no refresh cycle is taking place.

RAM Refresh

Refresh of the dynamic RAM is accomplished by performing a memory cycle lasting 15 microseconds to each of the 128 row addresses, at intervals of 2ms. The use of a "RAS Only Refresh" signal results in a substantial reduction in operating power. For real time operations the refresh is fully asynchronous to the LOCAL and VMEbuses. The refresh process is completely transparent to the user.

When a refresh request is pending, a delay of 300ns (maximum) is required before attempting to access the memory. This delay is due to refresh having a higher priority.

RAM Timing

The DRAMs are high performance devices ideal for real-time applications. Device access time is typically 150ns with a board access time of 285ns minimum. This time is extended if a refresh cycle is pending.

RAM Summary

Start address	$000008
End address a)	$01FFFF (128k bytes)
End address b)	$07FFFF (512k bytes)
Boundary a)	$020000 (128k bytes)
Boundary b)	$080000 (512k bytes)
Access modes	Byte or Word Read or Write
Usable data bits	D0 to D7 and D8 to D15
Access time	285ns (min) 320ns (typ) 600ns (max) with Refresh

EPROM/SRAM Area

The PME 68-1B contains four sockets for 28-pin, JEDEC compatible devices. Two are defined as the System Area and contain EPROMs holding the standard Monitor etc. The two remaining sockets are user definable and can contain EPROM or SRAM devices.

Access Speed Selection

To enable the user to fit a range of devices, the EPROM areas have a jumper area to allow access time speeds to be selected. This provides a range of access times. The access time selected will apply to both system and user areas. Details of selection are provided in Chapter 4.

System Area

The board contains jumpers which allow the System Area to be configured to use the following devices:

EPROM 2732	4k x 8 bit,	8 kbytes total
EPROM 2764	8k x 8 bit,	16 kbytes total
EPROM 27128	16k x 8 bit,	32 kbytes total
EPROM 27256	32k x 8 bit,	64 kbytes total
EPROM 27512	64k x 8 bit,	128 kbytes total

The PME68/Monitor is supplied as two 2764 type EPROMs labelled 0547 and 0548. The 0547 EPROM contains the upper byte and 0548 the lower byte. These devices are located in board positions J24 and J40 respectively. For details of PLUM contact your supplier.

During the power-up phase the Initial Stack Pointer and Initial Program Counter are down mapped from the System EPROMs to locations $000000 to $000003 and $000004 to $000007 respectively in the on-board RAM. They are then read from these locations by the CPU.

The System Area may be re-configured to take 2732, 27128, 27256 or 27512 type EPROMs. Refer to Chapter 4, for details of jumper settings required for these devices.

System Area Summary

Start address	$080000
End address	$09FFFF
Boundary	$020000
Boot address	$000000 to $000007
Access modes	Byte or Word Read Only
Access time	See Access Speed Selection in Chapter 4 (jumper selectable)

User Area

The two sockets in the user area may be configured for EPROMs, containing application programs, or for SRAMs.

The user area can be configured to use the following devices with various access times. (See configuration detail in Chapter 4).

EPROM	2732	4k x 8 bit	8k bytes total
EPROM	2764	8k x 8 bit	16k bytes total
EPROM	27128	16k x 8 bit	32k bytes total
EPROM	27256	32k x 8 bit	64k bytes total
EPROM	27512	64k x 8 bit	128k bytes total
SRAM	6264	8k x 8 bit	16k bytes total
SRAM	62256	32k x 8 bit	64k bytes total

Chip-selection for the upper (D8 to D15) and the lower socket (D0 to D7) is organized byte wide. This allows byte manipulation of the area when used with static RAMs. Refer to Chapter 4 for details of jumper settings to accommodate particular types.

User Area Summary

Start address	$0A0000
End address	$0BFFFF
Boundary	$020000
Access modes	Byte or Word transfers Read or Write on SRAM Read only on EPROM
Usable data bits	D0 to D7 and D8 to D15
Access time	See Access Speed Selection in Chapter 4 (jumper selectable)

Local I/O and Control Devices

PME68-1B boards contain three serial I/O Interfaces (Ports 1 to 3), a Real Time Clock with a battery back-up and a Parallel Interface and Timer Module.

Serial I/O

The board contains three independent serial I/O channels. The clock input for both the transmit and receive baud rate of each of the three channels is selectable to one of eight baud rate clocks driven by a Motorola 14411 baud rate generator. Baud rates are:

38400	600
19200	440
9600	300
2400	150
4800	110
1200	

For circuit and configuration details refer to Chapter 4.

Each serial I/O interface has a jumper field for changing the I/O signal assignment of the female, 25-pin, D-type sub-connector assigned to it. The serial I/O controllers are 6850 devices. These are Asynchronous Communication Interface Adaptor (ACIA) chips. Selected signals associated with each port may be configured to be available at P2, link information for this is given in Chapter 4.

Terminal Interface

One RS-232C interface is used to communicate via Port 1 (connector P4) with a standard terminal. The transmission format is initially set as follows:

8 data bits
1 stop bit
No parity
No protocol
9600 baud

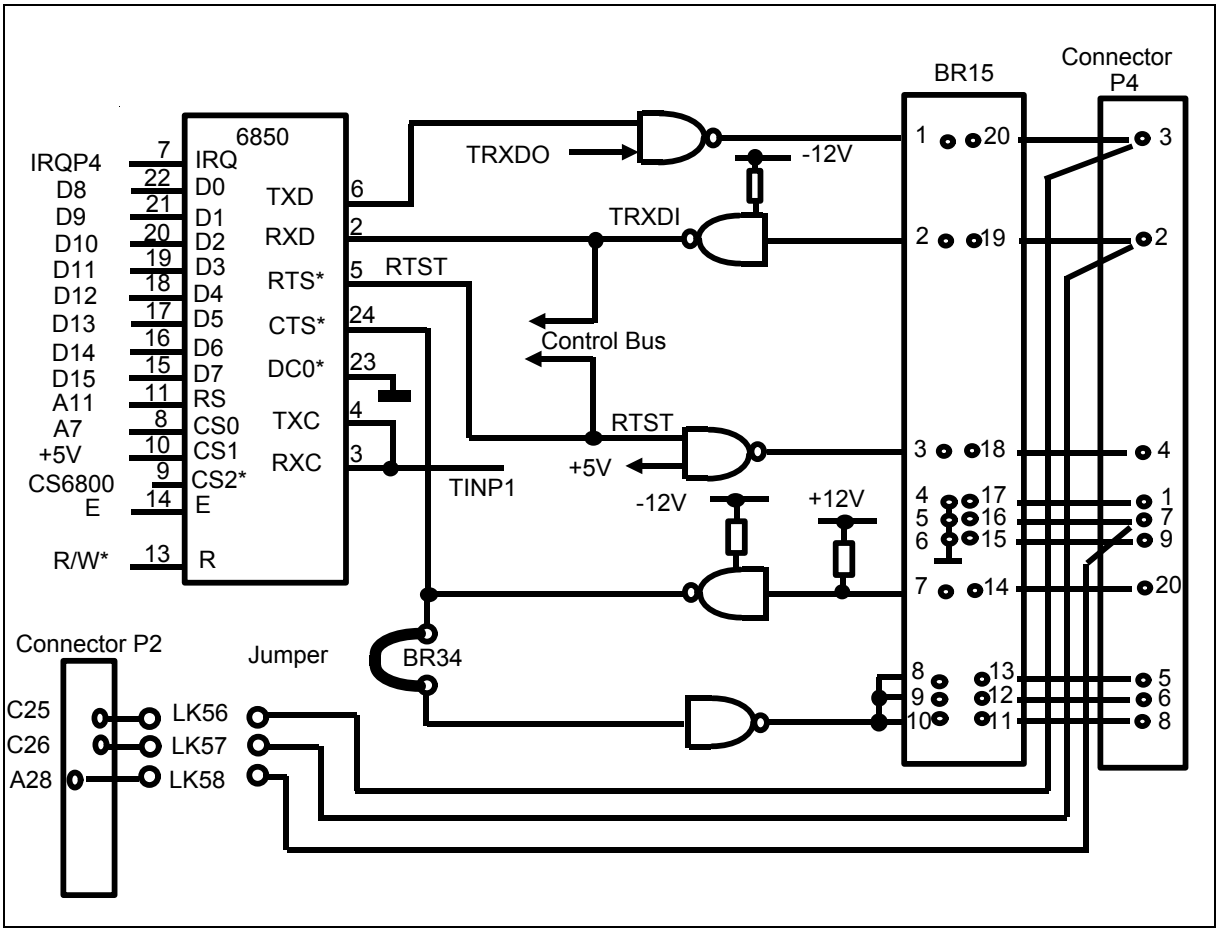
The terminal must initially be set to conform to these parameters.

The baud rate is selected by on-board jumpers (see Chapter 4) and the other parameters by the Port Format command (see Chapter 5).

The default signal assignment is contained in Table 2; this may be re-defined by the user (see Chapter 4).

The terminal interface can interrupt the 68000 CPU on level 4. The interrupt vector used is auto-interrupt vector (# 28/$000070); this cannot be changed. A diagram showing the terminal interface is given below as an example to illustrate the use of the 6850 device. More detailed board and configuration information is provided in Chapter 4.

Figure 6 Terminal Interface



The interface is enabled in Transparent Mode by activating the RTST line.

Address Assignment of Terminal Interface

DEVICE: 6850 ACIA (Terminal)

Address	Mode	Description
0C0080	R	Status Register
0C0080	W	Control Register
0C0082	R	Receive Data Register
0C0082	W	Transmit Data Register

ITable 2 Default Signal Assignment - Terminal/Remote Interfaces

P4/P5 Pin	Input	Output	Signal
1	X	X	Protective GND connected to signal GND
2	X		Transmit Data (TXD)
3		X	Receive Data (RXD)
4		X	Request to Send (RTS)
5		X	Clear to Send (CTS)
6		X	Data Set Ready (DSR)
7	X	X	Signal GND
8	X		Data Carrier Detect (DCD)
9	X	X	Signal GND
20	X		Data Terminal Ready (DTR)

Remote Interface

Port 3 is an RS-232C compatible interface on connector P5. It is defined as a Remote or Universal Interface.

Typical applications for this interface are:

Serial printer

Second terminal (multi-user applications)

Serial link to a back-up medium

The Remote Interface can interrupt the CPU on level 3; the interrupt vector used is auto-vector (# 27/$000006C).

The transmission form and signal assignment default settings and the method of re-selection are the same as for the Terminal interface.

Address Assignment - Remote Interface

Address	Mode	Description
0C0101	R	Status Register
0C0101	W	Control Register
0C0103	R	Receive Data Register
0C0103	W	Transmit Data Register

Host Interface

Port 2 (connector P3) may be used for up/down loading of user programs and data, using the Monitor Dump/Load command, or in the transparent mode, to interface the terminal connected to Port 1 (connector P4) directly to a host computer.

In the configuration diagram Figure 6, both the PME 68-1B and the host are set to receive data from the terminal on the Transmit Data Line (TXD) and echo it via the Receive Data Line (RXD) back to the terminal. The PME 68-1B polls the terminal ACIA registers.

A standard method of communication between the terminal connected to P4 and a host computer is known as 'Transparent Mode'. In this mode all transmitted characters are sent directly from the terminal to the host computer and no correction or modification is done by the PME68-1B. Figure 7 is a schematic of this configuration.

When the user types a valid stop character sequence, the CPU recognises it, stops transmission of the following characters from the terminal to the host computer and returns the system to the Monitor program. The stop character sequence is therefore the last sequence that the PME 68-1B board sends to the host.

The baud rate of each module, terminal, PME 68-1B terminal interface, and host computer interface must be set to the same level. The baud rate of the host ACIA can remain unaltered as the transparent mode does not use this ACIA for transfers.

The host interface can interrupt the PME 68-1B CPU on level 2, the interrupt vector used is the auto-interrupt vector (# 26/$000068). If a connected device cannot drive the CTS signal high, a connection must be made between pin 5 and pin 20 of connector P3.

A detailed description of the data format and Dump/Load (DU/LO) commands is given in Chapter 5 for the PME68/Monitor and in the separate PLUM manual for the PLUM monitor.

Fig. 7 Configuration with a Host Computer

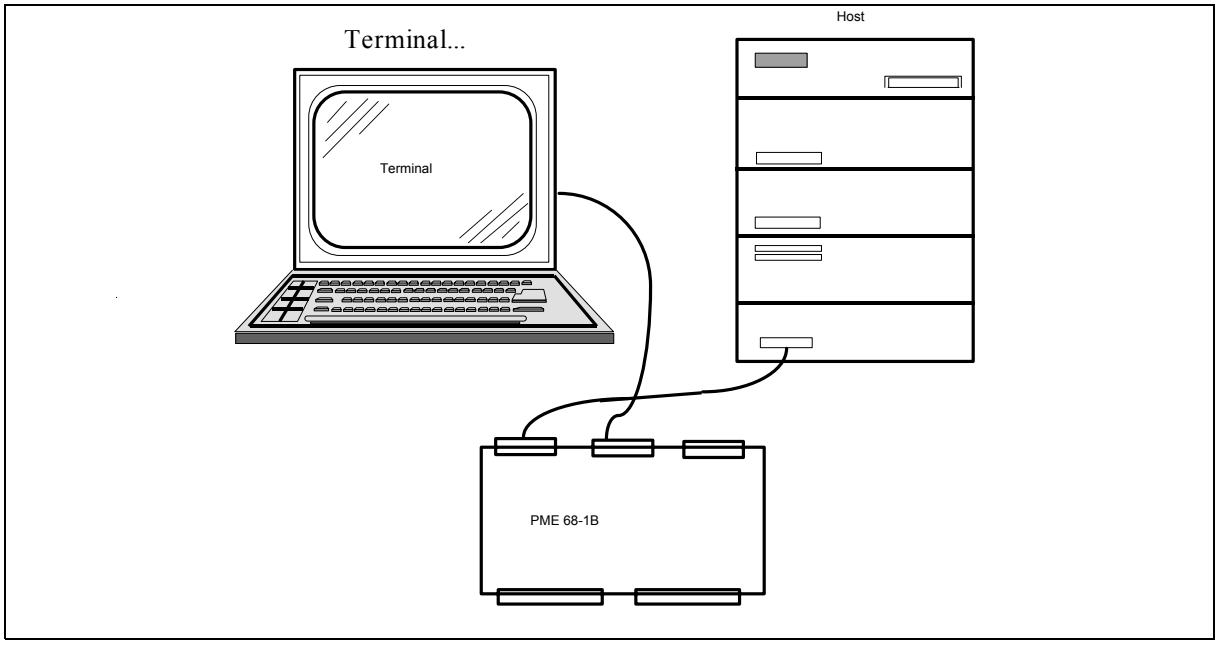


Table 3 lists the default signal assignment of the Host interface. This can be re-defined by the user; see Chapter 4

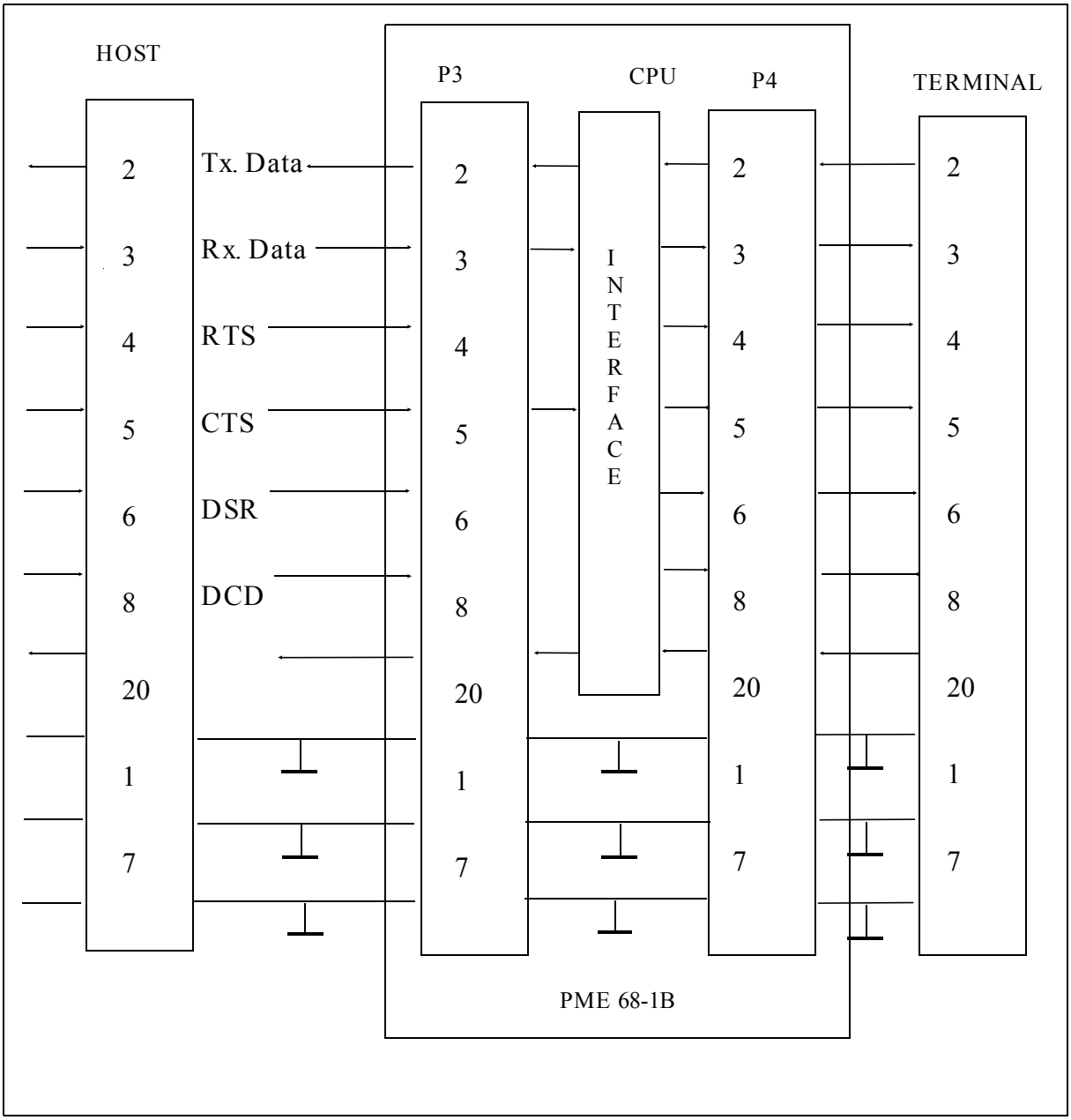
Address Assignment - Host Interface

Address	Mode	Description
0C0041	R	Status Register
0C0041	W	Control Register
0C0043	R	Receive Data Register
0C0043	W	Transmit Data Register

Table 3 Signal Assignment - Host Interface

P3 Pin	Input	Output	Signal
1	X	X	Protective GND connected to signal GND
2		X	Transmit Data (TXD)
3	X		Receive Data (RXD)
5	X		Clear to Send (CTS)
7	X	X	Signal GND
9	X	X	Signal GND
20	X		Data Terminal Ready (DTR)

Fig. 8 Interfacing with a Host Computer in Transparent Mode



Serial I/O Interface Summary

Terminal Interface (Port 1 Connector P4)

Start address	$0C0080
End address	$0C0082
Access mode	Byte Only Read and Write
Usable Data Bits	D8 to D15
Access time	1000ns(min) 2000ns(max)
Interrupt request level	4
Interrupt handling	Fixed IRQ vector (# 28) Address: $000070

Host Interface (Port 2 Connector P3)

Start address	$0C0041
End address	$0C0043
Access mode	Byte Only Read and Write
Usable data bits	D0 to D7
Access time	1000ns(min) 2000ns(max)
Interrupt request level	2
Interrupt handling	Fixed IRQ vector (# 26) Address: $000068

Serial I/O Interface Summary contd.

Remote Interface (Port 3 Connector P5)

Start address	$0C0101
End address	$0C0103
Access mode	Byte Only Read and Write
Usable data bits	D0 to D7
Access time	1000ns(min) 2000ns(max)
Interrupt request level	3
Interrupt handling	Fixed IRQ vector (# 27) Address: $00006C

ACIA Access Time

The ACIA is a controller device from the 6800 family. Access cycles are controlled by the Processor signals VMA*, VPA* and the E signal.

To initiate a transfer, the Processor must receive a VPA* signal from the decoding logic. When the CPU is synchronised to the E clock signal, the VMA signal is asserted to indicate to the I/O devices that a transfer is beginning. Synchronisation requires additional time (1000ns maximum).

Real Time Clock

The programmable Real Time Clock can be used in conjunction with a multi-user/multi-task operating system for real time applications. It can be used as a calendar, a real time counter and for time measurement.

The RTC is a 58167A device; this is a metal gate CMOS circuit with an access time of approximately 1100ns.

This requires a special delay of the DTACK* signal to the CPU for an access of the RTC.

The RTC has the following features:

- 1/10000 of a second through month counter
- 24 hour clock
- 56 bits of RAM with comparator to compare RTC data with RAM data
- Interrupt output with 8 possible interrupt signals
- Power-down mechanism disables all input and output signals
- Status register to indicate rollover during a read cycle
- 32758 Hz crystal oscillator
- Battery back-up (Refer to Chapter 4 for details.)

Addressing the RTC

The access address of the RTC register is $0C0401 to $0C042F. Only single byte transfers to and from the RTC on data bits D0 to D7 are allowed. The register model of the RTC is given in Table 4.

RTC Interrupts

The RTC can be used to interrupt the on-board CPU. This feature is selected by a jumper inserted during manufacture in field B200 and LK 60 pin 2-3. The RTC interrupt can be disabled, refer to Chapter 4, RTC Interrupts for further details.

The RTC interrupt request level is fixed at level 6, the highest maskable interrupt level. This level is shared by RTC and PIT PC3 ; selection is made by LK 60 as described in Chapter 4. The on-board interrupt control logic decodes the RTC/PIT PC3 interrupt request and forces the auto-interrupt vector, # 30/$000078, (after the interrupt has been acknowledged) on level 6. This vector is fixed and reserved for the RTC/PIT PC3.

Table 4 Register Model of the RTC

Address	Mode	Description
0C0401	R/W	Counter - Ten thousandths of seconds
0C0403	R/W	Counter - Hundredths and tenths of seconds
0C0405	R/W	Counter - Seconds
0C0407	R/W	Counter - Minutes
0C0409	R/W	Counter - Hours
0C040B	R/W	Counter - Day of the week
0C040D	R/W	Counter - Day of month
0C040F	R/W	Counter - Month
0C0411	R/W	RAM - Ten thousandths of seconds
0C0413	R/W	RAM - Hundredths and tenths of seconds
0C0415	R/W	RAM - Seconds
0C0417	R/W	RAM - Minutes
0C0419	R/W	RAM - Hours
0C041B	R/W	RAM - Day of week
0C041D	R/W	RAM - Day of month
0C041F	R/W	RAM - Month
0C0421	R	Interrupt Status Register
0C0423	R/W	Interrupt Control Register
0C0425	W	Counters - Reset
0C0427	W	RAM - Reset
0C0429	R/W	Status Bit
0C042B	W	GO Command
0C042D	W	Standby Interrupt
0C042F	W	Test Mode

RTC Summary

Access address: $0C0401 - $0C042F
Access mode: Byte Mode (odd only)
Usable data bits: D0 to D7

Interrupt level: 6 (auto-interrupt vectoring)

Interrupt vector: Fixed: # 30
Address: $000078

Parallel I/O Interface

The Parallel Interface and Timer module (PIT 68230) is used to provide powerful asynchronous parallel I/O on PME 68-1B boards. The PIT provides 24 I/O lines, 4 control lines and a 24-bit timer with a 5-bit prescaler. A Centronics type interface can be configured. Figure 9 is a hardware connection example.

The PIT is accessed via P2, the 64-pin male VMEbus connector. Access via a synchronous bus structure and the non-multiplexed data/address bus allows the PIT to communicate without any CPU wait-states.

The PIT includes the following features:

- Mode selection
  - Bit I/O
  - Unidirectional 8-bit and 16-bit
  - Bidirectional 8-bit and 16-bit
- Selectable handshaking options (This is done via the software and reference should be made to a PIT data sheet.)
- 24 bit programmable timer with 5-bit prescaler
- Software programmable timer modes
- 2 selectable interrupt sources
- Registers are readable and writeable
- All registers are directly addressable

PIT Addressing

All PIT registers are directly addressable and are readable or read/writeable depending on the register selected (see Table 5).

Only single byte transfers to or from the PIT on data bits D0 to D7 are allowed (i.e. odd addresses).

Caution: The PIT interrupts the processor on auto-interrupt vector number 5 and does not supply the processor with a vector number during interrupt acknowledge cycles. So, programming the Port Interrupt Vector Register (PIVR) has no effect.

Table 5 Parallel I/O Interface Registers

Address	Mode	Affected By Reset	Using
0E0001	R/W	Y	Port General Control Register (PGCR)
0E0003	R/W	Y	Port Service Request Register (PSRR)
0E0005	R/W	Y	Port A Data Direction Register (PADDR)
0E0007	R/W	Y	Port B Data Direction Register (PBDDR)
0E0009	R/W	Y	Port C Data Direction Register (PCDDR)
0E000B	R/W	Y	Port Interrupt Vector Register (PIVR)
0E000D	R/W	Y	Port A Control Register (PACR)
0E000F	R/W	Y	Port B Control Register (PBCR)
0E0011	R/W**	N	Port A Data Register (PADR)
0E0013	R/W**	N	Port B Data Register (PBDR)
0E0015	R	N	Port A Alternate Register (PAAR)
0E0017	R	N	Port B Alternate Register (PBAR)
0E0019	R/W	N	Port C Data Register (PCDR)
0E001B	R/W*	Y	Port Status Register (PSR)
0E0021	R/W	Y	Timer Control Register (TCR)
0E0023	R/W	Y	Timer Interrupt Vector Register (TIVR)
0E0027	R/W	Y	Counter Preload Register High (CPRH)
0E0029	R/W	N	Counter Preload Register Middle(CPRM)
0E002B	R/W	N	Counter Preload Register Low (CPRL)
0E002F	R	N	Count Register High (CNTRH)
0E0031	R	N	Count Register Middle (CNTRM)
0E0033	R	N	Count Register Low (CNTRL)
0E0035	R/W*	Y	Timer Status Register (TSR)

* A Write to this register may perform a special status reset operation.
** Affected by Read Cycle. (mode dependent e.g. port interface or timer.)

PIT Interrupts

The PIT is able to interrupt the CPU. This allows it to operate in fully asynchronous mode as well as being able to use the timer as a time base for multi-tasking software.

The general purpose 24-bit timer, with its 5-bit prescaler, can be used as an output for programmable frequencies, with internal or external clocks, as a watchdog and as a normal time base.

The PIT can interrupt under parallel operation, or if port C is used in timer mode, under timer operation. A special pin is used for each and is selectable using BR18 links 1, 2 and 3. Both interrupts are enabled under default conditions.

The PIT interrupt signal lines (PC3 + PC5) can both be linked to interrupt request level 5. Auto-interrupt vectoring is used, the fixed interrupt vector number for the PIT (level 5) is # 29. The PIT interrupt signal line PC3 can also be linked to interrupt at level 6 if the RTC interrupt is disabled.

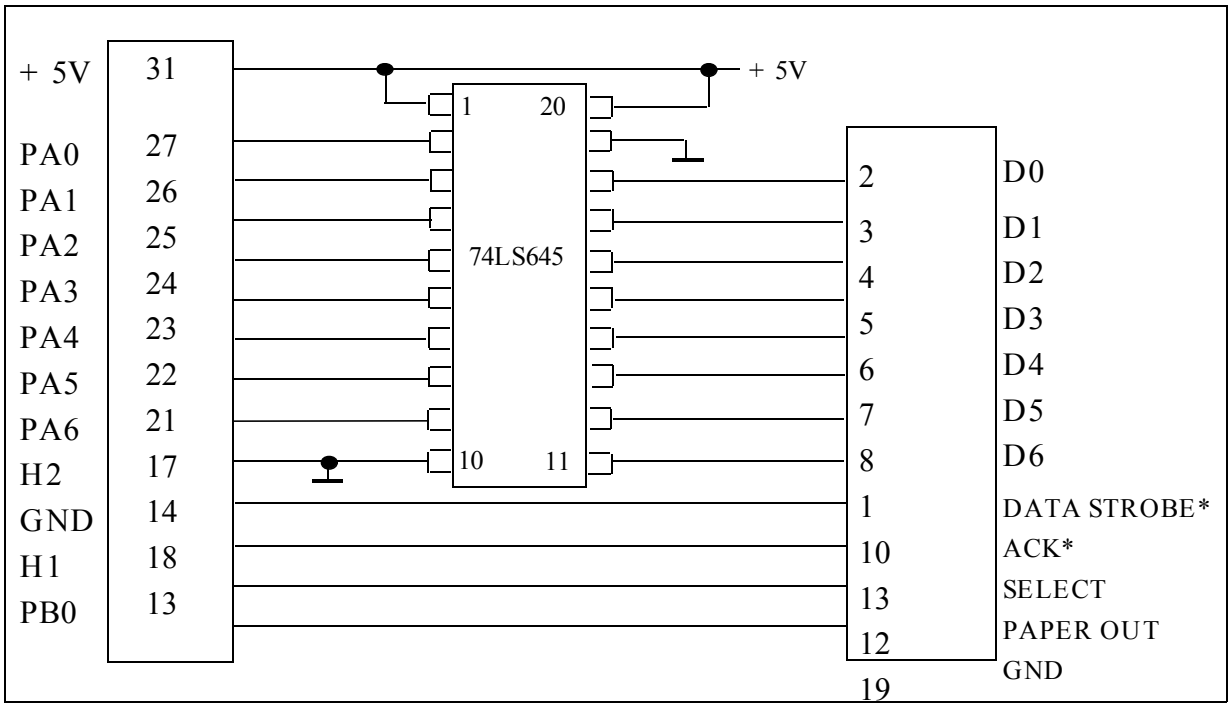
Address $000074 is the start address of the PIT handling routine (for level 5). Please refer to the PIT 68230 data sheet (available from your supplier) which provides programming information for the PIT.

Centronics Type Interface

The PIT can be used as a Centronics-type parallel interface. This allows connection of a Centronics compatible printer with the handshake protocol under full software control. See Chapter 4 for further details.

Figure 9 below shows how the PIT may be user configured as a Centronics type interface. The diagram shows how a 74LS645 can be employed to give extra buffering on the data and strobe lines. This allows cable length to be extended to approximately 5m (16 ft).

Figure 9 Example of a Centronics Type Interface



PIT Summary

Access address:	$0E0001 - $0E0035
Access mode:	Byte Mode (odd only)
Usable data bits:	D0 to D7
Interrupt level:	5 (auto-interrupt vectoring)
Interrupt vector:	Fixed # 29 Address: $000074

Interrupt Handling

The on-board CPU is able to handle 7 different prioritised interrupt request levels. Interrupt priority levels are numbered from one to seven, level seven being the highest priority. The status register contains a three bit mask which indicates the current priority of the processor. The interrupt daisy-chain of the VMEbus allows an unlimited number of peripheral devices to interrupt the CPU.

Interrupt requests arriving at the processor do not force immediate exception processing as they are only detected between instruction executions. If the priority of the incoming interrupt request is lower than or equal to the current processor priority, execution continues with the next instruction; interrupt exception processing is not carried out and the incoming request is ignored.

If the priority of the incoming interrupt is greater than the current processor priority, the exception processing sequence is started.

On-board Interrupt Sources

PME68-1B boards contain six possible on-board interrupt sources; the three serial I/O Controllers (ACIAs), the ABORT switch, the Parallel Interface and the Real Time Clock.

Table 6 lists these devices, their interrupt level and the default settings of the auto-interrupt vectors (if required).

Table 6 On-board Interrupts

Device	Interrupt Level	Note	Auto-interrupt Vectoring	Default Vector	Address
ABORT Switch	7		YES	# 31	$00007C
RTC/PIT PC3	6	(a)	YES	# 30	$000078
PIT 68230	5		YES	# 29	$000074
ACIA Terminal	4		YES	# 28	$000070
ACIA Remote	3		YES	# 27	$00006C
ACIA Host	2		YES	# 26	$000068

Notes:

(a) These two Interrupt Request Signals are tied together.

The RTC and PIT PC3 interrupt are tied together on auto-interrupt vector # 30; this eases software control of the interrupt scheme. The RTC contains an Interrupt Status Register which indicates if it is the interrupt source.

Software ABORT Switch

The ABORT switch on the front panel generates a non-maskable interrupt to the CPU on level 7. This interrupt uses auto-interrupt vector # 31.

The switch can be used for debug purposes (refer to 'Abort Switch' in Chapter 5); alternatively, the switch can be used for self test if special routines are built in.

ACFAIL

Providing an ACFAIL monitor is fitted, the VMEbus ACFAIL signal if driven low indicates that the main input power is not within the defined voltage limits.

In the default state an active ACFAIL signal will generate a HALT* signal to the cpu causing it to enter the HALT state. This is indicated by the red HALT LED on the front panel. A link may be removed so that ACFAIL has no affect on the cpu (see Chapter 4).

Interrupt Exception Sequence

Once the interrupt exception sequence has started, a copy of the Status Register (SR) is saved on the stack, the privilege state of the processor is set to supervisor, and the processor priority level is set to the level of the interrupt being acknowledged.

The processor fetches the vector number from the interrupting device, classifies the reference as an interrupt acknowledge and displays the level number of the interrupt being acknowledged on the address bus using the address signals A1, A2 and A3 as listed in Table 7.

Table 8 shows the conversion of the interrupt vector into the start address of the interrupt service routine.

Table 7 Interrupt Acknowledge Level Code

IRQ Level	A3	A2	A1	FC0	FC1	FC2	
7	1	1	1	1	1	1	
6	1	1	0	1	1	1	
5	1	0	1	1	1	1	
4	1	0	0	1	1	1	
3	0	1	1	1	1	1	
2	0	1	0	1	1	1	
1	0	0	1	1	1	1	
-	0	0	0	1	1	1	Not assigned

Note: FC0 to FC2 are function code signals from the 68000.

The vector from the interrupting device is placed onto data bits D0 to D7 and is translated by the CPU into the address of the interrupt handling routine.

The content of the interrupt vector is loaded into the program counter to start the interrupt handling routine, (the normal vector table of the CPU is given in Table 1).

Table 8 Interrupt Vector Conversion

Data Bit	Data Bus
D0-D7	Peripheral Vector Number V0-V7
D8-D15	ignored

Address Bit	Translated Address Bit
A0-A7	A0= 0 A1= 0 A2= V0 A3= V1 A4= V2 A5= V3 A6= V4 A7= V5
A8-A15	A8= V6 A9= V7 A10 : = 0 A15

Table 9 Auto-interrupt Vector Table

Absolute Address	Vector Number	Corresponding Interrupt Level
$000060	# 24	Spurious Interrupt
$000064	# 25	Level 1 Auto-interrupt Vector
$000068	# 26	" 2 " "
$00006C	# 27	" 3 " "
$000070	# 28	" 4 " "
$000074	# 29	" 5 " "
$000078	# 30	" 6 " "
$00007C	# 31	" 7 " "

Priority level 7 cannot be disabled by the interrupt priority mask; it is therefore a 'non-maskable interrupt'.

VMEbus Interrupt Handling

All on-board interrupts have a higher priority in the internal interrupt acknowledge daisy chain than VMEbus interrupts at the same interrupt level.

VMEbus interrupt signals, IRQ1 to IRQ7, can be enabled/disabled separately using jumpers. Refer to Chapter 4 for further details. Default condition is all interrupts enabled.

VMEbus Interrupt Acknowledge Daisy Chain

The VMEbus specification defines 7 interrupt request signals IRQ1* to IRQ7*, plus 3 special control signals which allow an unlimited number of interrupt sources in the system. Each of the seven VMEbus IRQ signals may be shared by two or more interrupter modules. The Interrupt Acknowledge (IACK*) Daisy Chain is used to ensure that only one of these modules places its interrupt vector on data bits D0 to D7 at the required time.

An active Interrupt Acknowledge (IACK*) signal informs all cards within the system that the current read cycle is an interrupt vector acquisition. This signal is connected at slot 1 of the motherboard to the Interrupt
Acknowledge In (IACKIN*) signal (Figure 10). The IACKIN* signal passes through each board on the bus.

If a board receives an active IACKIN* signal and has not produced an interrupt on that level, its control logic must pass the signal through its own IACKOUT* signal to inform the next board (if it is the interrupt requester), that it can place its interrupt vector onto the data bus.

A functional diagram of the Interrupt Acknowledge Daisy Chain is shown below.

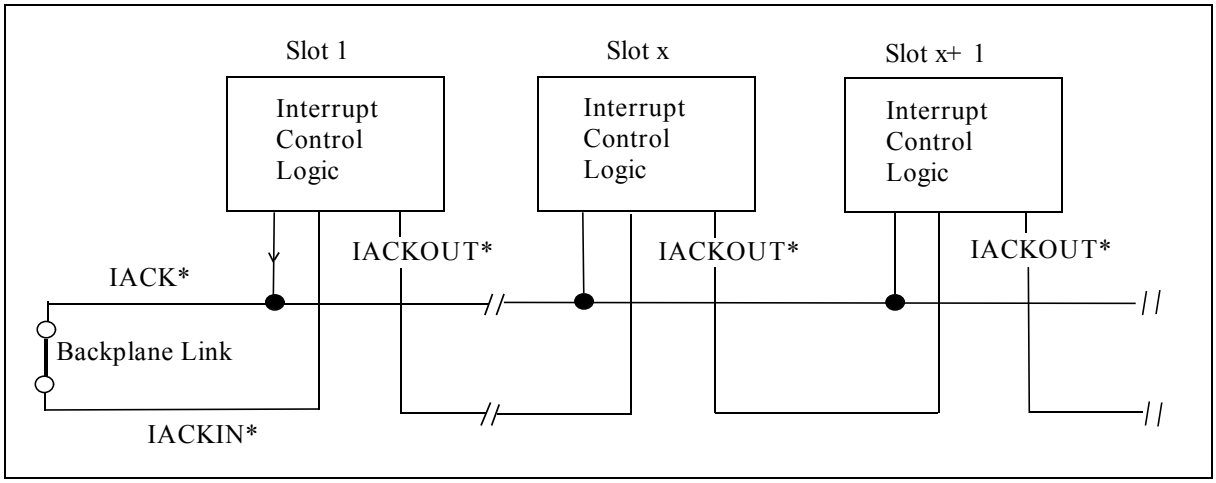
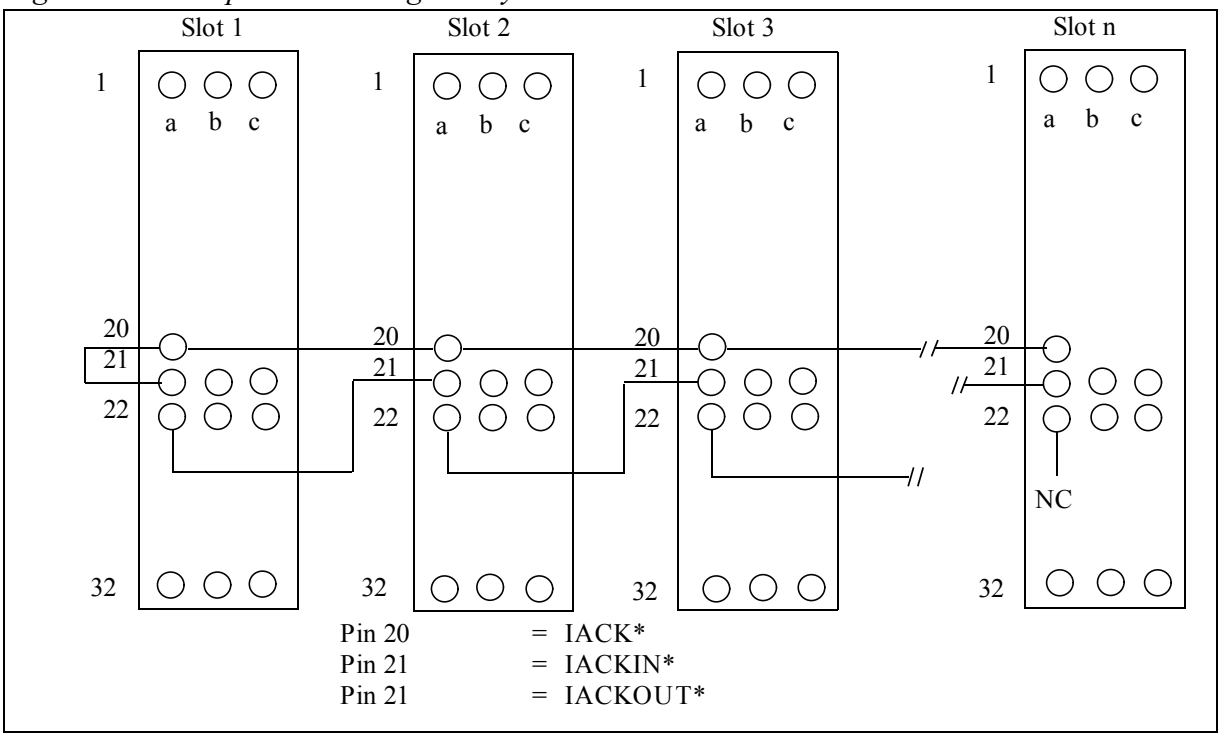


Figure 10 The Interrupt Acknowledge Daisy Chain with Slot 1 Interrupt Handler.
The low driven IACK* signal is wired to slot 1 of the VME motherboard and runs down the IACK daisy-chain.

Figure 11 Interrupt Acknowledge Daisy Chain



The PME 68-1B has an on-board interrupt handler which can be configured to respond to any group of request levels. Alternatively, the interrupt handler can be disabled and the 68-1B used with an external interrupt handler.

VMEbus Arbitration

The VMEbus is designed to allow multi-master and multi-processor applications. Only the current VMEbus master is able to carry out read or write transfers with other modules. This requires a special handshake scheme to define which module receives bus mastership; this is the function of the VMEbus Arbiter.

The Arbiter is used to control the Data Transfer Bus (DTB). The DTB is the transport medium for all data and includes the address, address modifier, strobe control and acknowledge signals.

The Arbiter must reside in slot number 1 of the system because bus arbitration is daisy-chained from slot 1 to 2 to 3 etc. Each system can have only one arbiter. This may be a special card, another CPU card, or the one located on the PME68-1B.
Three arbiter types are defined in the VMEbus specification.

(1) A four-level Bus Arbiter with a priority scheme

(2) A four-level Bus Arbiter with a round-robin scheme

(3) A one-level Bus Arbiter

The PME 68-1B provides a one-level Bus Arbiter only.

A one (single) level Arbiter only honours requests on a single, predefined, Bus Request (BR) line and relies on the daisy-chain structure for priority determination.

The VMEbus specification defines four BR levels, BR0* to BR3* and specifies that BR3* should be used by one-level Arbiters. This is the default setting of the PME 68-1B Arbiter but the user may select any other BR level if required; see Chapter 4.

Using PME 68-1B as the System Bus Arbiter

A bus arbitration scheme with a minimum of system overhead can be built using the on-board bus Arbiter. This uses four types of VME signal: "Bus Request", "Bus Grant In and Out", "Bus Clear" and "Bus Busy".

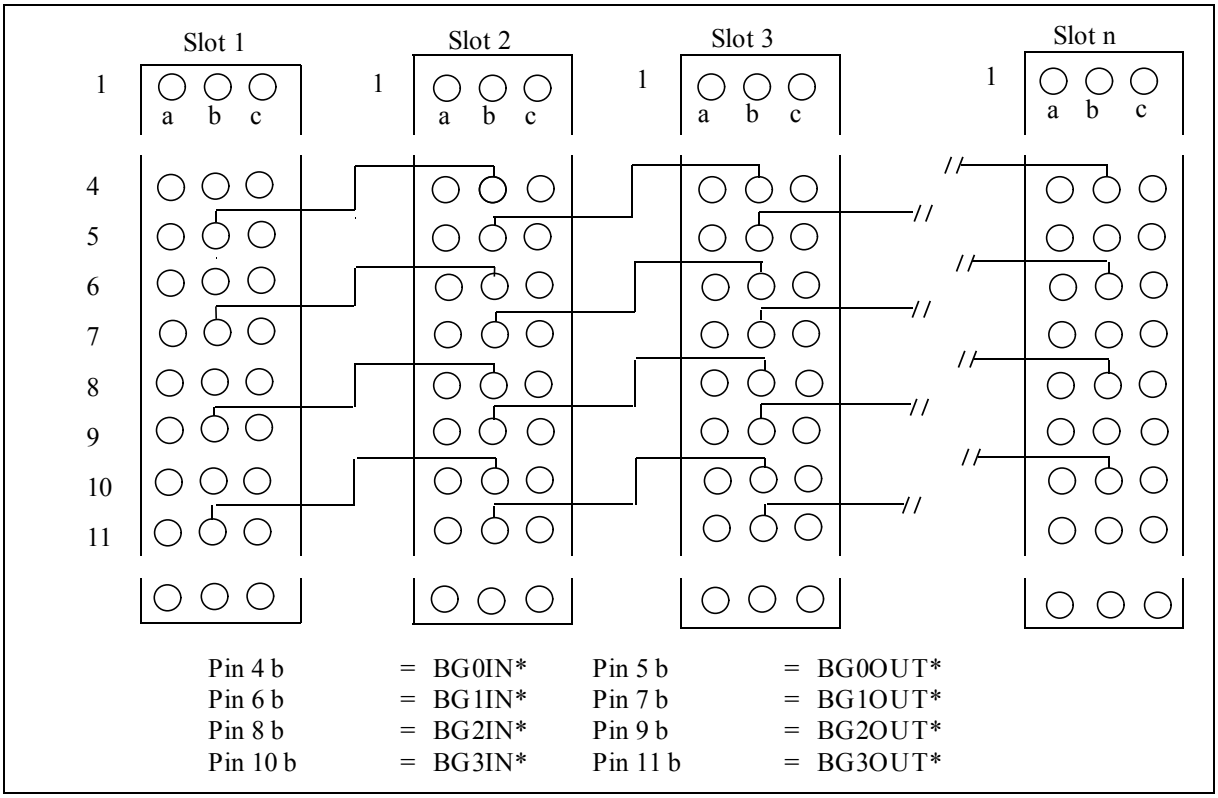
The BR* signal is driven low by a VME module to inform the Arbiter that it requires control of the DTB (Data Transfer Bus).

When the current bus master releases control of the bus and a bus request is pending, a new arbitration cycle is initiated.

The Arbiter then starts the bus grant daisy-chain by driving a Bus Grant Out (BGxOUT*) signal low. This BGxOUT* signal is the BGxIN* signal to the next slot in the motherboard. It informs the board in this slot that it can become Bus Master if it has requested DTB control. If the board has not requested DTB control, it must pass on the BGxIN* signal to the next slot, and so on.

When the DTB requester receives the BGxIN* signal, it acquires DTB control by driving Bus Busy (BBSY*) low and de-activating its BR* signal. This informs the Arbiter that DTB acquisition is complete. Figure 12 shows the bus grant daisy-chain.

Figure 12 Bus Grant Daisy Chain



Bus Release Functions

Two release functions are defined in the VMEbus specification:

RWD Release When Done

ROR Release On Request

The PME68-1B provides a RAT (Release After Time-Out) function and, when configured in slave arbitration mode, a release on BCLR.

The maximum continuous time available as bus master is defined by on-board jumpers as 30, 60 or 300µs. An arbitration cycle on every bus cycle can also be selected. Refer to Chapter 4 for further details.

VMEBUS Interface

When the PME 68-1B is the current VMEbus master and the current access address is higher than or equal to $100000 (an off-board address), a VMEbus transfer is initiated.

SYSCLK Signal

A Texas 74S241 driver is installed on PME 68-1B boards; this provides the 16 MHz, SYSCLK VMEbus signal.

The SYSCLK driver can be isolated from the VMEbus via a jumper. Refer to Chapter 4 for further details.

SYSRESET* Signal

A jumper defines whether the board drives, receives or ignores the VMEbus signal SYSRESET*. The RTC RESET is fixed to VME SYSRESET*. Refer to Chapter 4 for further details.

RESET Switch

The RESET switch is a toggle switch located on the front panel (Figure 2). It can be used to carry out a RESET of all devices on the PME 68-1B board: CPU, PIT, ACIA: or produce a VME SYSRESET* signal. This carries out a general RESET of all boards and devices within the system, including 68-1B RTC, and is the default condition.

SYSFAIL* Signal

The VMEbus specification defines a SYSFAIL* signal which informs all cards that one (or more) card(s) in the system has failed. On PME 68-1B boards the user can enable or disable the use of this signal. When enabled, an incoming SYSFAIL* will cause the CPU to jump directly to the halt state and the HALT LED on the front panel to light. When disabled the SYSFAIL* signal is ignored.

Address Modifier Codes

The VMEbus defines the address modifier signals. These in turn define the type of VME access being executed.

Table 10 Address Modifier Codes

Hex Code	Address Modifier 5	4	3	2	1	0	Function
3F	H	H	H	H	H	H	Standard Supervisory Block Transfer
3E	H	H	H	H	H	L	Standard Supervisory Program Access
3D	H	H	H	H	L	H	Standard Supervisory Data Access
3C	H	H	H	H	L	L	Reserved
3B	H	H	H	L	H	H	Standard Non-Privileged Block Transfer
3A	H	H	H	L	H	L	Standard Non-Privileged Program Access
39	H	H	H	L	L	H	Standard Non-Privileged Data Access
38	H	H	H	L	L	L	Reserved
30-37	H	H	L	X	X	X	Reserved
2F	H	L	H	H	H	H	Reserved
2E	H	L	H	H	H	L	Reserved
2D	H	L	H	H	L	H	Short Supervisory I/O Access
2C	H	L	H	H	L	L	Reserved
2B	H	L	H	L	H	H	Reserved
2A	H	L	H	L	H	L	Reserved
29	H	L	H	L	L	H	Short Non-Privileged I/O Access
28	H	L	H	L	L	L	Reserved
20-27	H	L	L	X	X	X	Reserved
10-1F	L	H	X	X	X	X	User Defined
0F	L	L	H	H	H	H	Extended Supervisory Block Transfer
0E	L	L	H	H	H	L	Extended Supervisory Program Access
0D	L	L	H	H	L	H	Extended Supervisory Data Access
0C	L	L	H	H	L	L	Reserved
0B	L	L	H	L	H	H	Extended Non-Privileged Block Transfer
0A	L	L	H	L	H	L	Extended Non-Privileged Program Access
09	L	L	H	L	L	H	Extended Non-Privileged Data Access
08	L	L	H	L	L	L	Reserved
00-07	L	L	L	X	X	X	Reserved

Function Codes

The 68000 microprocessor contains 3 function code signals, FC0 to FC2. These indicate the state of the processor, i.e. USER or SUPERVISOR mode, and the type of cycle currently being executed.

Table 11 Possible Processor States and Cycle Types

FC2	FC1	FC0	Processor State/Cycle Type
LOW	LOW	LOW	Reserved
LOW	LOW	HIGH	User Data
LOW	HIGH	LOW	User Program
LOW	HIGH	HIGH	Reserved
HIGH	LOW	LOW	Reserved
HIGH	LOW	HIGH	Supervisor Data
HIGH	HIGH	LOW	Supervisor Program
HIGH	HIGH	HIGH	Interrupt Acknowledge

Short I/O Address Modifier Code

An address comparator is used to select a 64k byte range from the 68000's 16M byte address space.

The 64k byte range is jumper selectable within the range $100000 to $FFFFFF. $000000-$100000 are employed by on-board memory and addresses. Refer to Chapter 4.

Bus Error Function

A time-out counter is used to provide an error handling function. The external VMEbus Error (BERR*) signal is ANDed with the internal signal so that the PME68-1B, when acting as the VMEbus master, will respond if any device on the bus raises a BERR* or does not respond within a user specified time of up to 2.5ms . For example, a BERR* signal could be generated by a dynamic memory using Error Detection and Correction Logic (EDC) if an uncorrectable error is detected.

If the PME 68-1B is bus master and BERR* becomes active, the CPU aborts the current cycle and enters an error handling routine. Available time-out settings are:

25µs 32µs 250µs 320µs 2.5ms

The default time-out value of 2.5ms can be altered by changing the settings of jumpers. Refer to Chapter 4 for details.

Connectors

VMEbus P1 Connector

Pin	Row A	Row B	Row C
1	D00	BBSY*	D08
	D01	BCLR*	D09
3	D02	ACFAIL*	D10
4	D03	BG0IN*	D11
5	D04	BG0OUT*	D12
6	D05	BG1IN*	D13
7	D06	BG1OUT*	D14
8	D07	BG2IN*	D15
9	GND	BG2OUT*	GND
10	SYSCLK	BG3IN*	SYSFAIL*
11	GND	BG3OUT*	BERR*
12	DS1*	BR0*	SYSRESET*
13	DS0*	BR1*	LWORD*
14	WRITE*	BR2*	AM5
15	GND	BR3*	A23
16	DTACK*	AM0	A22
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*	SERCLK	A17
22	IACKOUT*	SERDAT*	A16
23	AM4	GND	A15
24	A07	IRQ7*	A14
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11
28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	-12V	+ 5VSTDBY	+ 12V
32	+ 5V	+ 5V	+ 5V

P2 Pin Assignments

Pin	Row A	Row B	Row C
1	PC0		PC4
2	PC1		PC5
3	PC2		PC6
4	PC3		PC7
5	GND		
6	PB7		
7	PB6		
8	PB5		
9	PB4		
10	PB3		
11	PB2		
12	PB1		
13	PB0		
14	GND		
15	H4		
16	H3		
17	H2		
18	H1		
19	GND		
20	PA7		
21	PA6		
22	PA5		
23	PA4		
24	PA3		
25	PA2		LK56
26	PA1		LK57
27	PA0		LK53
28	LK58		LK55
29	LK50		LK54
30	LK51		LK52
31	+ 5V		+ 5V
32	+ 5V		+ 5V

Note: PA0 - PA7, PB0 - PB7, PC0 - PC7, H1 - H4 are all signals from the PIT (J50)

LK50(RX), 51(TX), 52(SIG GND) refer to the Remote interface
LK53(TX), 54(RX), 55(SIG GND) refer to the Host interface
LK56(TX), 57(RX), 58(SIG GND) refer to the Terminal interface

P3 Pin Assignments

Pin	Signal
1	Protective GND connected to signal GND
2	Transmit Data (TXD)
3	Receive Data (RXD)
5	Clear to Send (CTS)
7	Signal GND
9	Signal GND
20	Data Terminal Ready (DTR)

P4/P5 Pin Assignments

Pin	Signal
1	Protective GND connected to signal GND
2	Transmit Data (TXD)
3	Receive Data (RXD)
4	Request to Send (RTS)
5	Clear to Send (CTS)
6	Data Set Ready (DSR)
7	Signal GND
8	Data Carrier Detect (DCD)
9	Signal GND
20	Data Terminal Ready (DTR)

See also pages 20 and 22

Chapter 4 - Configuration

Preparation For Use

To communicate with the board a terminal must be connected to Port 1 (connector P4). This allows the PME68/Monitor to be used for program development, up/download of programs etc. The default set-up of all RS-232C interfaces is:

(1) 8 data bits
(2) 1 stop bit
(3) No parity
(4) No protocol
(5) 9600 baud

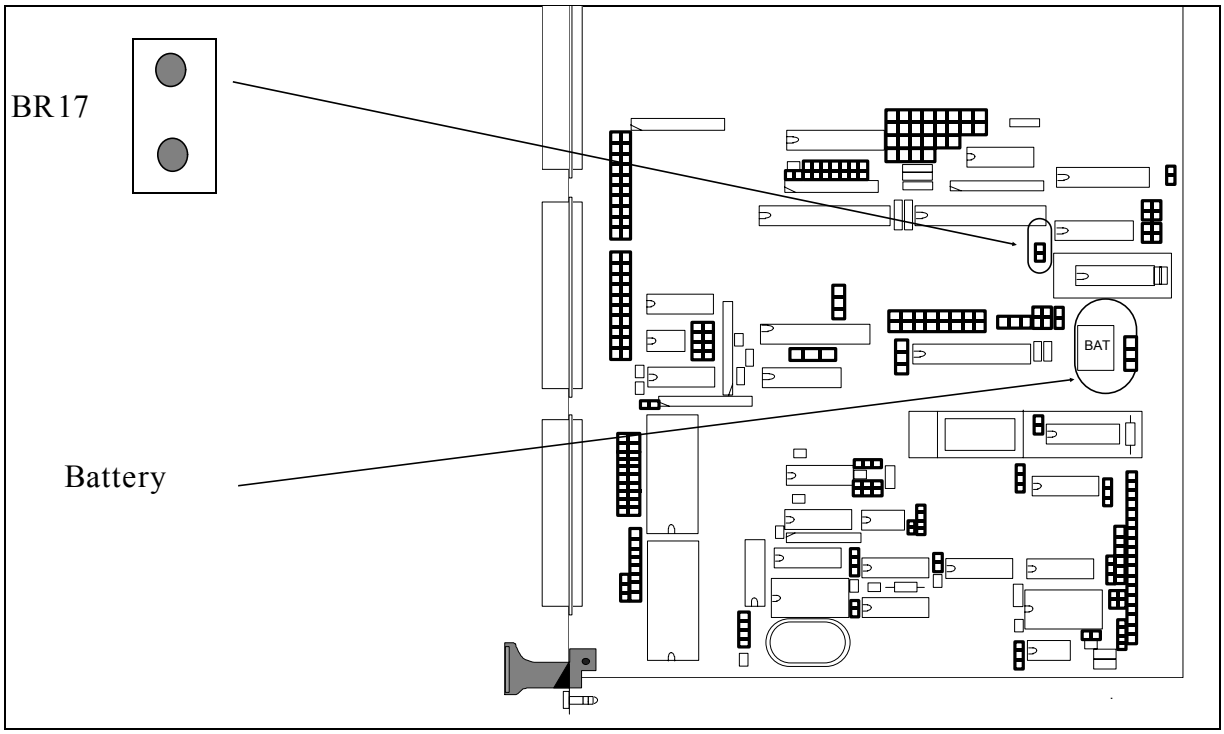
No special control or escape sequences are used in the monitor.

A standard RS-232C interface cable is fitted with two 25-pin, male connectors. The user should check the signal line assignment of all peripheral devices with those of the PME68-1B to avoid signal conflicts and to protect the output driver of the board.

Fitting a Battery

WARNING: Before fitting the battery in position ensure that the link BR17 is removed to disconnect the + 5V standby supply. If this is not done the battery will explode.

Figure 13 BR17 and Battery Location



Manual Soldering:

The terminal should be soldered at a point as far as possible from the spot-welds to the battery.

With a soldering tool temperature of 350° C the soldering operation should be completed within 5 seconds, with a tool temperature of 250° C the operation should be completed within 10 seconds.

Fitting:

Check the polarity carefully. The style of battery employed on the PME 68-1B has two connections at one end and a single connection at the other end. The board is fitted with corresponding plated-through holes.

In general:

Solder should not be applied directly to the battery.

Do not stack batteries in contact with one another and take particular care to avoid short circuits at all times.

Do not heat the battery

Do not incinerate

Do not attempt to charge the battery

No attempt should be made to open the battery or pierce the casing.

Batteries should be stored in a dry location at normal room temperature with minimal temperature variations. Corrosion and loss of performance will result if stored in damp or hot conditions.

Do not expose batteries to direct sunlight or rain.

Use a charge prevention diode if there is a possibility that a battery could receive a charging current from a source of power. (The PME 68-1B has been fitted with charge prevention diodes.)

The batteries should not be positioned close to other heat emitting components. (The battery on the PME 68-1B has been positioned near the edge of the board at a location well away from potentially hot components.)

To measure the battery voltage it is necessary to use a meter or measuring instrument with an impedance or input impedance greater than 1M Ohm.

Link Settings

PME68-1B contains a number of links which enable the board to be configured to suit system requirements. Other parameters can be changed by replacing PAL devices.

CAUTION: Components may be damaged by incorrectly fitted, or illegal combinations of links.

The board is supplied in a default state (Table 12) which will allow it to run in a system after the initialisation sequence has been entered. Each board includes spare links. The board link areas are shown in Figure 14.

NOTE: The link settings shown here are for general guidance; some links listed will not be present on all board variants.

Table 12 Link Settings

Link	Pin connections
B33	1-2
B200	1-2
B201	1-2
B202	1-2
BR2	1-3, 2-4
BR5	1-3
BR6	1-2
BR7	2-3
BR12	1-3
BR14	1-20, 2-19, 3-18, 4-17, 5-16, 6-15, 7-14, 8-13, 9-12, 10-11
BR15	1-20, 2-19, 3-18, 4-17, 5-16, 6-15, 7-14, 8-13, 9-12, 10-11
BR16	1-16, 2-15, 3-14, 4-13, 5-12, 6-11, 7-10, 8-9
BR17	OUT
BR19	OUT
BR21	1-2
BR22	1-4, 2-3
BR24	1-2
BR26	1-2
BR27	1-2
BR32	1-2
BR33	1-2
BR34	1-2
BR35	1-2
BR36	2-3
BR37	1-2
BR38	1-2
BR39	1-2
BR40	1-2
BR41	2-3
BR42	2-3

Table 12 Links settings Applicable to All Variants contd.

WA1	1-14, 2-13, 3-12, 4-11, 5-10, 6-9, 7-8	
WB1	See WB2	
WB2	WB2# 1-2-WB1# 1	
WC1	WC1# 1-WC2# 1, WC1# 2-WC2# 2	None or PME68/Monitor
WC2	WC1# 4-WC2# 2	
WC1	WC1# 1-WC2# 1,WC1# 4-WC2# 2,WC1# 2-3	PLUM Monitor
WD1	WD1# 1-WD2# 1, WD1# 2-WD2# 2, WD1# 4-WD1# 2	
WD2	See WD1	
WF1	WF1# 1-WF2# 1&2&3	
WF2	See WF1	
WF3	See WF1	
WH1		OUT
WH2		
WH3		
WH4		
WI1	WI1# 1-WI1# 3	
WI2	No link	
WI3	No link	
WI4	WI4# 2-3	
WK1	WK1# 1-2, WK1# 4-WK2# 4	
WK2	See WK1	

Table 13 Variant Link Settings

Variant Description	Standard -/10-	Versados -/20-	Wide Temp -/30-	User Specific Configuration -/31	-/32-	-/33
BR18	1-3, 2-3		1-3, 2-3	1-3, 2-3	1-3, 2-3	1-3, 2-3
LK50 - 55	OUT	OUT	OUT	IN	IN	OUT
LK56 - 58	OUT	OUT	OUT	OUT	IN	OUT
LK60	2-3	1-2	2-3	2-3	2-3	2-3
BR20	1-3	1-2	1-3	1-3	1-3	1-3
BR28	1-2, 3-4	3-4	1-2, 3-4	1-2, 3-4	1-2, 3-4	1-2, 3-4
WL1	3-WL2# 3	3-WL2# 3	3-WL2# 3	3-WL2# 3	3-WL2# 3	3-WL2# 3
WL2	See WL1	See WL1	See WL1	See WL1	See WL1	See WL1
WL3	OUT	OUT	OUT	OUT	OUT	OUT

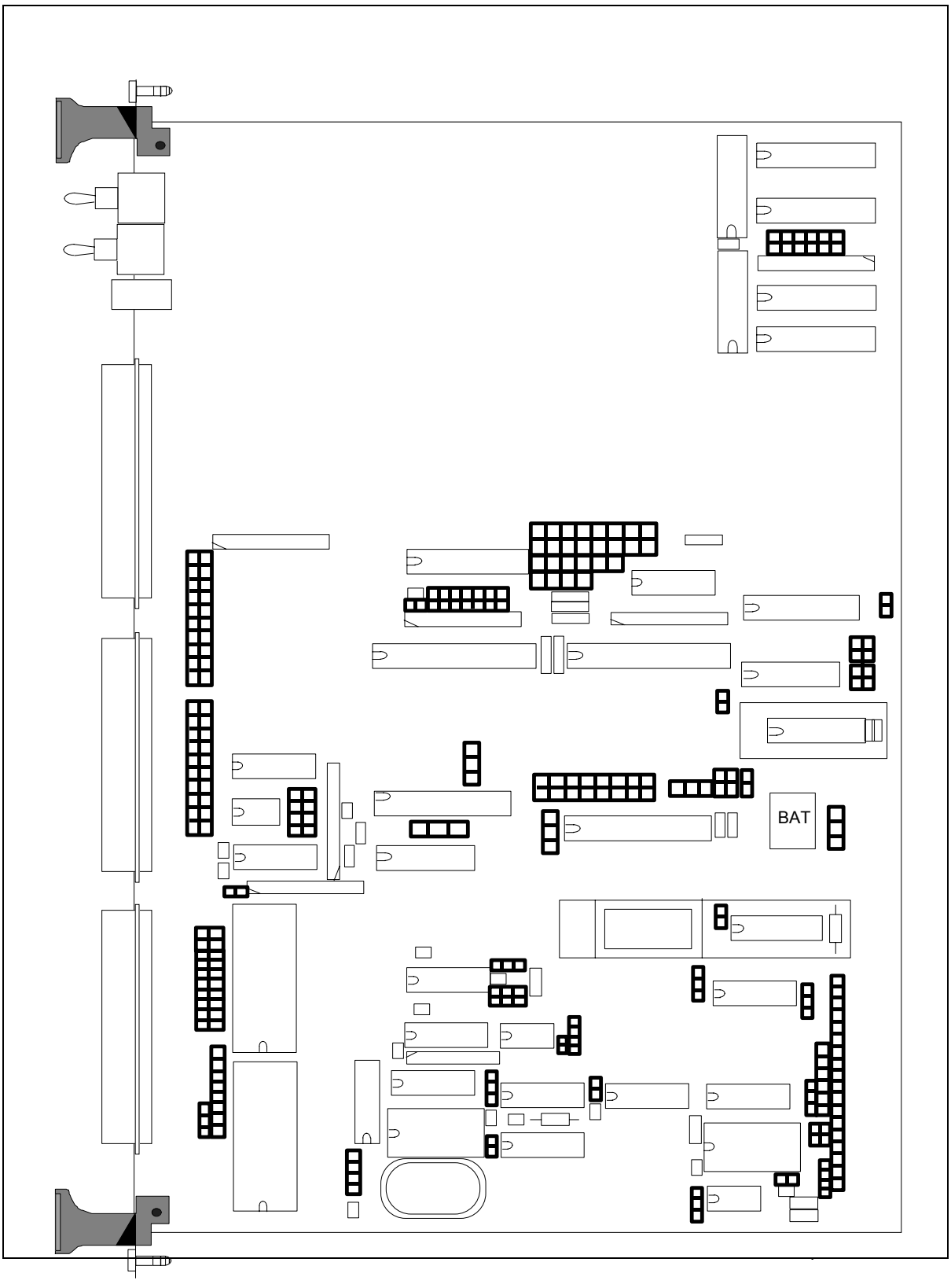
Table 14 Variant Features

Variant	-/--0	-/--1	-/--2	-/--3
8MHZ 68000	Y		Y	
10MHZ 68000		Y		Y
128Kbyte DRAM	Y	Y		
512Kbyte DRAM			Y	Y
BR4	2-3	1-2	2-3	1-2

Table 15 Standard Variant Default Settings

Function	Setting	Links Fitted
RTC Interrupt	Interrupt Level 6	B200 1-2, LK60 2-3
Baud Rate Selection	9600	BR12 # 1-3, WF1# 1toWF2# 1,2&3,
BERR*	Enabled	BR27 # 1-2, BR28# 1-2,# 3-4, WK1 # 1-2
Time-out Counter for BERR*	2.5ms	BR26 # 1-2, WK1# 1-2, WK1 # 4 to WK2# 4
SYSRESET*	Drives SYSRESET*	BR2 # 1-3,# 2-4
Access Speed Selection (System / User Areas)	Access times of 188ns to 250ns	BR6 # 1-2, BR7 # 2-3
Host Interface(P3) RS232 Set as DTE Asserting DTR from CTS	Tx, Rx, CTS, DTR, COM, GND, Pin 9 com.	BR35 # 1-2 BR12A # 1-16,# 2-15,# 3-14, # 4-13,# 5-12,# 6-11,# 7-10, # 8-9
Remote Interface(P5) RS232 Set as DCE Asserting CTS,DSR, DCD from DTR	GND, Rx, Tx, RTS, CTS, DSR, DCD Pin 9 com.	B33 # 1-2 BR14 # 1-20,# 2-19,# 3-18, # 4-17,# 5-16,# 6-15,# 7-14, # 8-13,# 9-12,# 10-11
Terminal Interface(P4) RS232 Set as DCE Asserting CTS,DSR, DCD from DTR	GND, Rx, Tx, RTS, CTS, DSR, DCD, COM, DTR Pin 9 com.	BR34 # 1-2 BR15 # 1-20,# 2-19,# 3-18, # 4-17,# 5-16,# 6-15,# 7-14, # 8-13,# 9-12,# 10-11
Parallel I/O Interface and Timer		BR18 # 1-3,# 2-3
ACFAIL	Halts on-board CPU	BR20 # 1-3
SYSCLK	Enabled	BR21 # 1-2
VMEbus Interrupt	Level 1 to 7 enabled	WA1 # 1-14,# 2-13,# 3-12, # 4-11,# 5-10,# 6-9,# 7-8
Slave Bus Arbitration	Enabled	WB2 # 1 to WB1 # 1, WI1 # 1 to WI1# 3, WI4# 2-3
System Area	PROM 2764 (PME68) PROM 27256 (PLUM)	WC1 and WC2 as indicated by Table 12
User Area	PROM 2764	WD1 # 1 to WD2 # 1, WD1 # 2 to WD2 # 2, WD1 # 4 to WD2 # 4
Time-Out for Bus-Release Functions	300µs	WL1 # 3 to WL2 #
5V Standby	Not Driven	BR17 OUT
SYSFAIL	Ignored	BR19 OUT

Figure 14 Link Areas



Blank Page

RAM/PROM/EPROM Area

System Area

The board is normally supplied with two 2764 EPROM devices installed in the system area (J24 and 40) for the PME68/Monitor, or alternatively two 27256 EPROMs for PLUM. These can be replaced with other devices; wire-wrap areas WC1 and WC2 provide the connections for address and control signals as detailed in Table 17.
Figure 15 shows the location of the EPROMs and wire-wrap area and Figure17 is the wiring diagram.

Table 16 RAM/EPROM Pins

RAM/EPROM	2732	2764/128	27256	27512	6264	62256
PIN 1	O/C	V_{cc}	V_{cc}	A16	O/C	A15
PIN 26	V_{cc}	A14	A14	A14	V_{cc}	A14
PIN 27	O/C	V_{cc}	A15	A15	R/W*	R/W*

System Connections

Table 17 System Area Jumper Field Settings

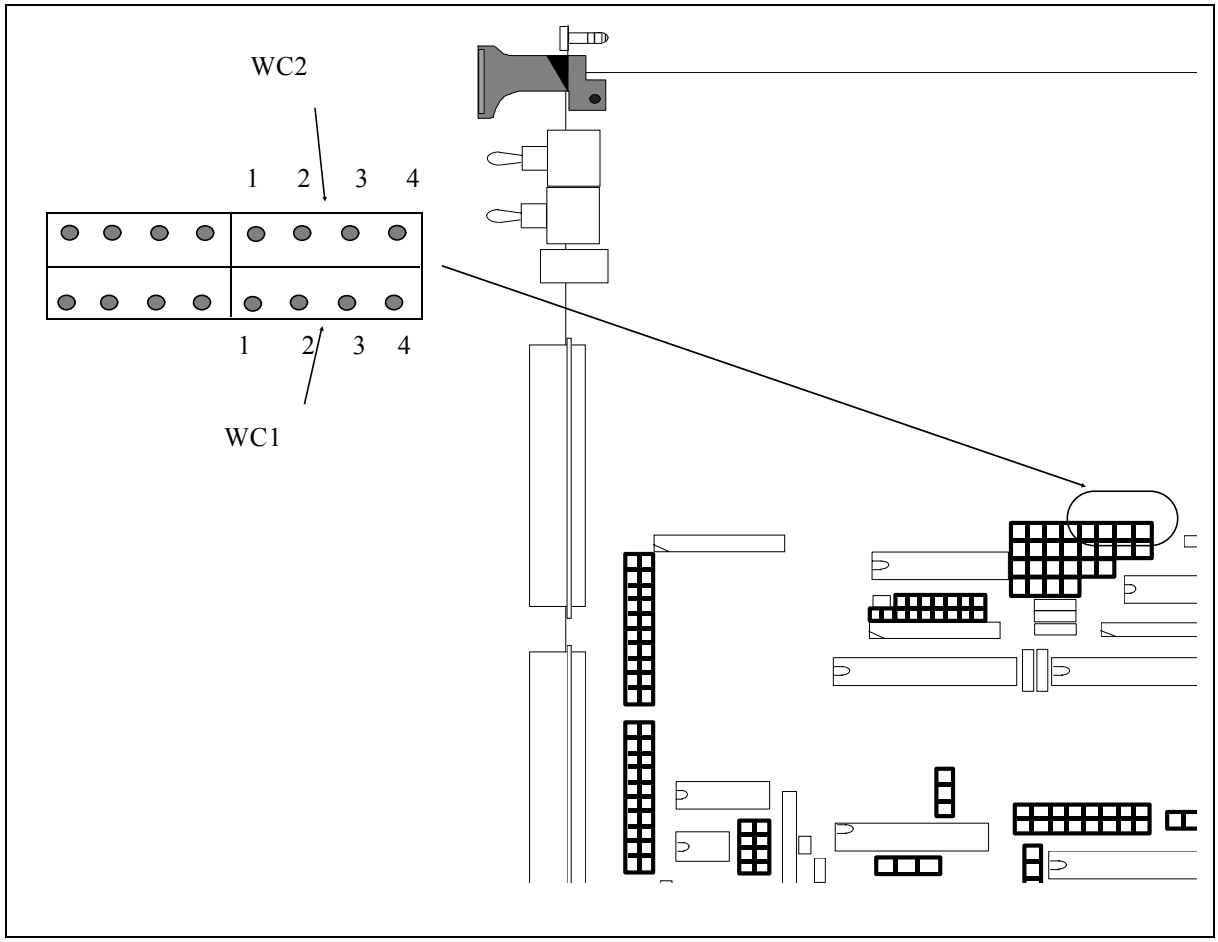
Device Type	Device Pin	WC1	to	WC2	Connection	
2732	26	WC1# 1	-	WC2# 2	V_{cc}	
	27	WC1# 2			O/C	
	1	WC1# 4			O/C	
2764 and 27128	26	*WC1# 1	-	WC2# 1	A14	
	27	WC1# 2	-	WC2# 2	V_{cc}	Default
	1	WC1# 4	-	WC2# 2	V_{cc}	
27256	26	WC1# 1	-	WC2# 1	A14	
	27	WC1# 2	-	WC1# 3	A15	
	1	WC1# 4	-	WC2# 2	V_{cc}	
27512	26	WC1# 1	-	WC2# 1	A14	
	27	WC1# 2	-	WC1# 3	A15	
	1	WC1# 4	-	WC2# 4	A16	

6264 and 62256 N/A for system area

* Not essential for 2764

NOTE: 2732 is a 24-pin device and should reside between pins 3 and 26 inclusive of J24 and J40

Figure 15 Location of System EPROM Area



User Area

The user area (J25 and 41) will accept two devices (of the same type) from those detailed in Table 16. Address and control lines are defined by wire-wrap areas WD1 and WD2. The standard (default) devices are a 2764/27128 EPROM.

Table 18 defines the jumper field settings required for these devices. Figure 16 shows their location and Figure 17 is the wiring diagram of the wire wrap areas.

Table 18 User Area Jumper Settings

Device Type	Device Pin	WD1	to	WD2	Connection	
2732 EPROM	26	WD1# 1	-	WD2# 2	V_{cc}	
	27	WD1# 2	-		O/C	
	1	WD1# 4	-		O/C	
2764/27128 EPROM	26	WD1# 1	-	WD2# 1	A14	
	27	WD1# 2	-	WD2# 2	V_{cc}	Default
	1	WD1# 4	-	WD2# 2	Vcc	
27256 EPROM	26	WD1# 1	-	WD2# 1	A14	
	27	WD1# 2	-	WD1# 3	A15	
	1	WD1# 4	-	WD2# 2	V_{cc}	
27512	26	WD1# 1	-	WD2# 1	A14	
	27	WD1# 2	-	WD1# 3	A15	
	1	WD1# 4	-	WD2# 4	A16	
6264 SRAM	26	WD1# 1	-	WD2# 2	V_{cc}	
	27	WD1# 2	-	WD2# 3	R/W*	
	1	WD1# 4	-		O/C	
62256 SRAM	26	WD1# 1	-	WD2# 1	A14	
	27	WD1# 2	-	WD2# 3	R/W*	
	1	WD1# 4	-	WD1# 3	A15	

* Not needed for 2764

NOTE: 2732 is a 24-pin device and should reside between pins 3 and 26 inclusive of J25 and J41.

Figure 16 Location of the User Area

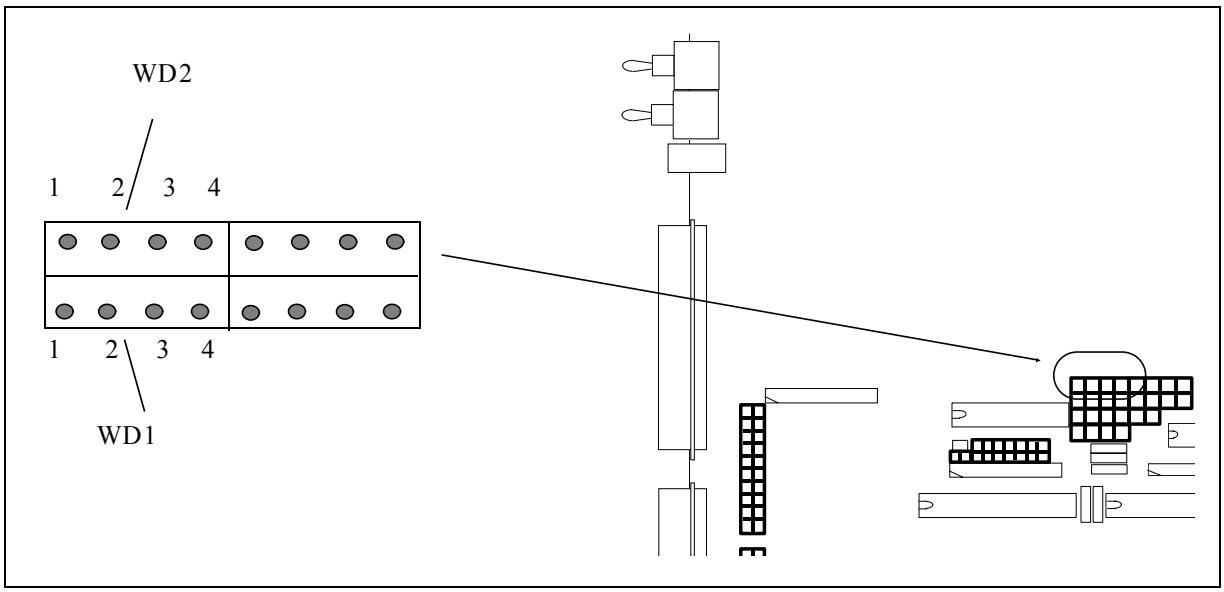
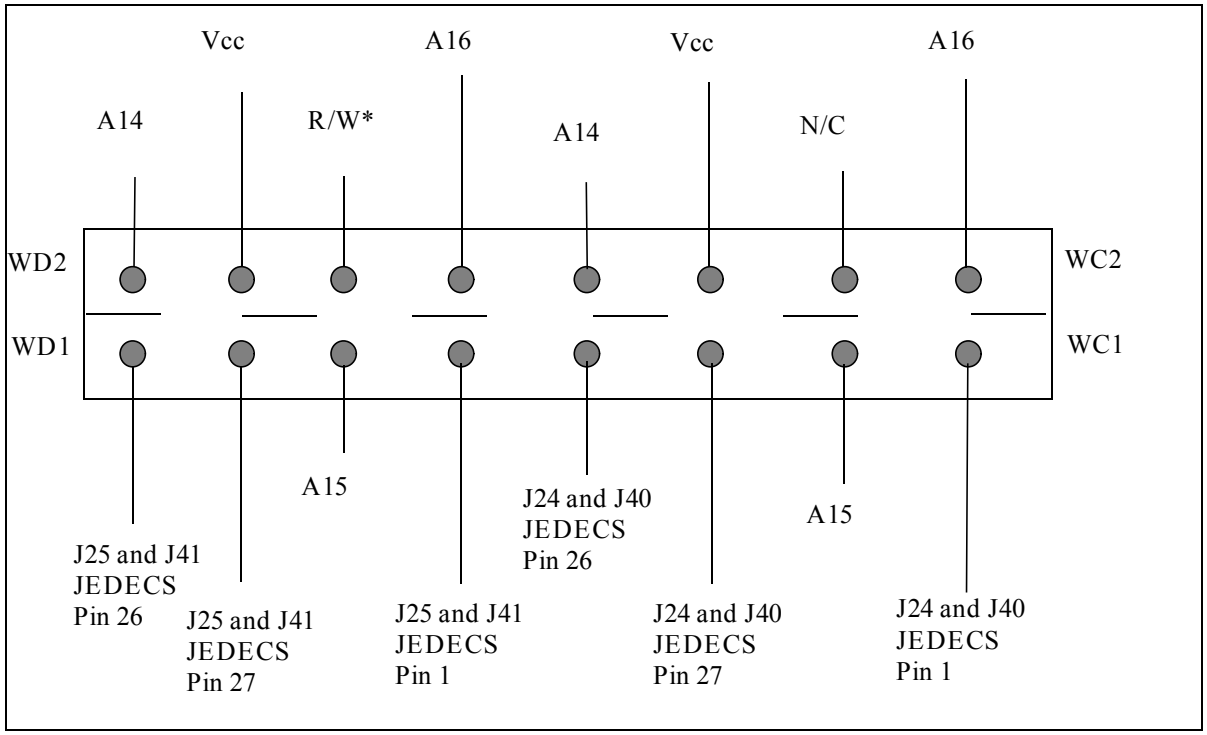


Figure 17 Jumper Fields for System/User Areas



Note: The wire-wrap pins for addresses A14 - A16, are based on a traditional approach which identifies the least significant address bit as A1. Care must be exercised when dealing with memory chips and other peripheral devices which may identify the least significant address bit as A0.

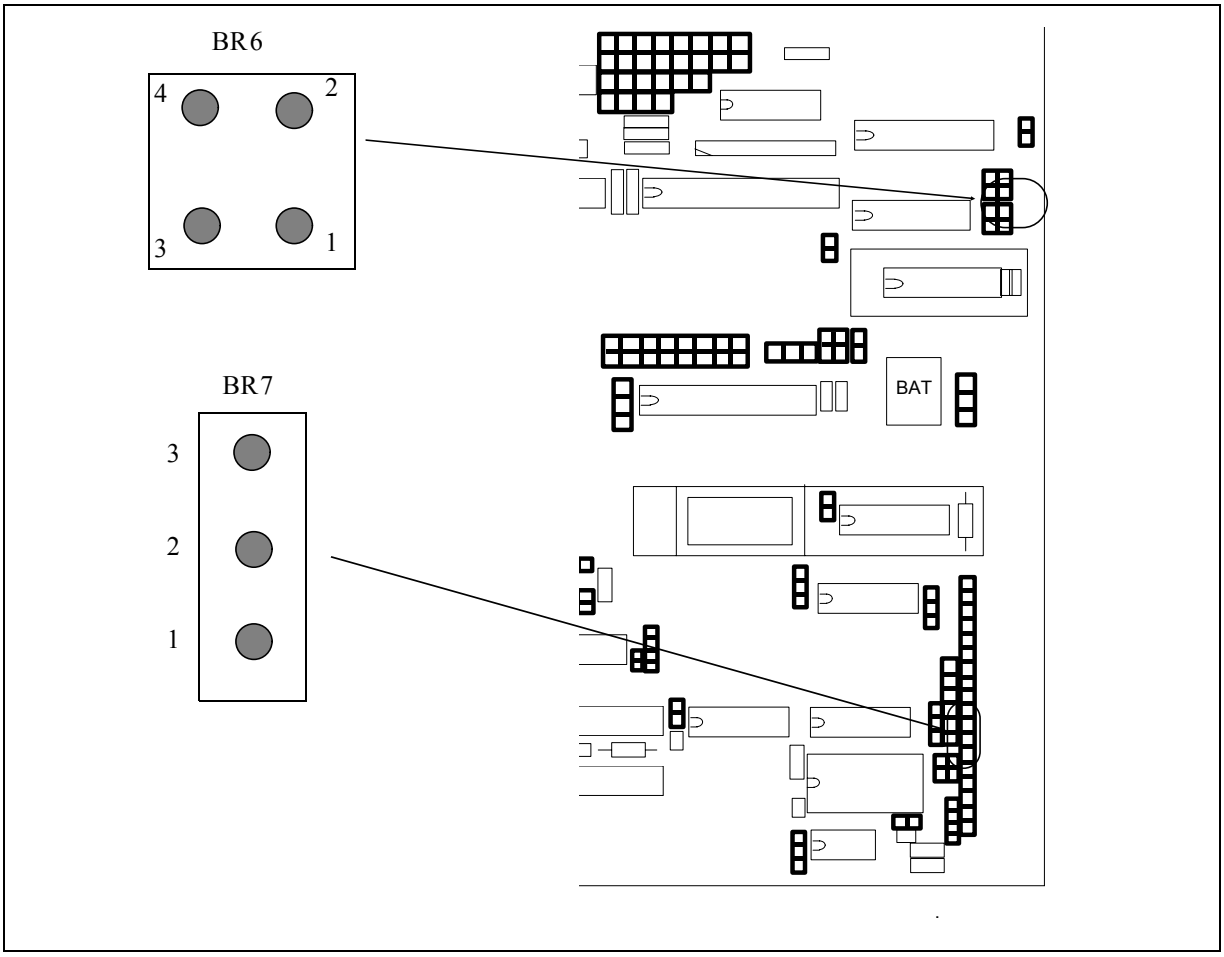
Access Speed Select

Jumper fields BR6 and BR7 are provided to enable the access speed of the EPROM/RAM areas to be adjusted to suit the devices fitted, see Table 19. Figure 18 shows the location of BR6 and BR7.

Table 19 Access Speed Selection

Jumper Closed on BR7	Jumper Closed on BR6	Access (ns) Min	Times Max.	Device Access Times (ns)
2 - 3	2 - 3	62	125	60
2 - 3	2 - 4	125	188	125
2 - 3	2 - 1	188	250	180 Default
1 - 2	2 - 3	125	250	125
1 - 2	2 - 4	250	375	250
1 - 2	2 - 1	375	500	375

Figure 18 Location of the Speed Selection Jumpers



Local I/O and Control Devices

Terminal Interface

A hardware diagram of the interface is given in Figure 20.

Default setting is defined in Chapter 3 and the default pin assignment in Table 2.

Changing the baud rate is described later in the section headed Baud Rate Selection, other transmission parameters are changed using the PME 68 Monitor PF command.

The signal assignment is defined by jumper block BR15, see Figure 19.

Figure 19 Location of Terminal Interface Jumper

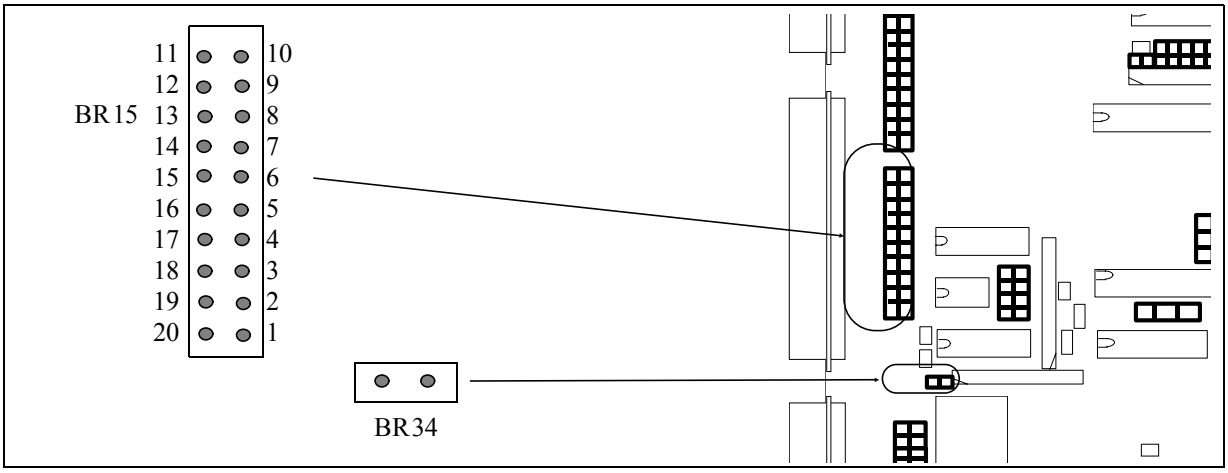
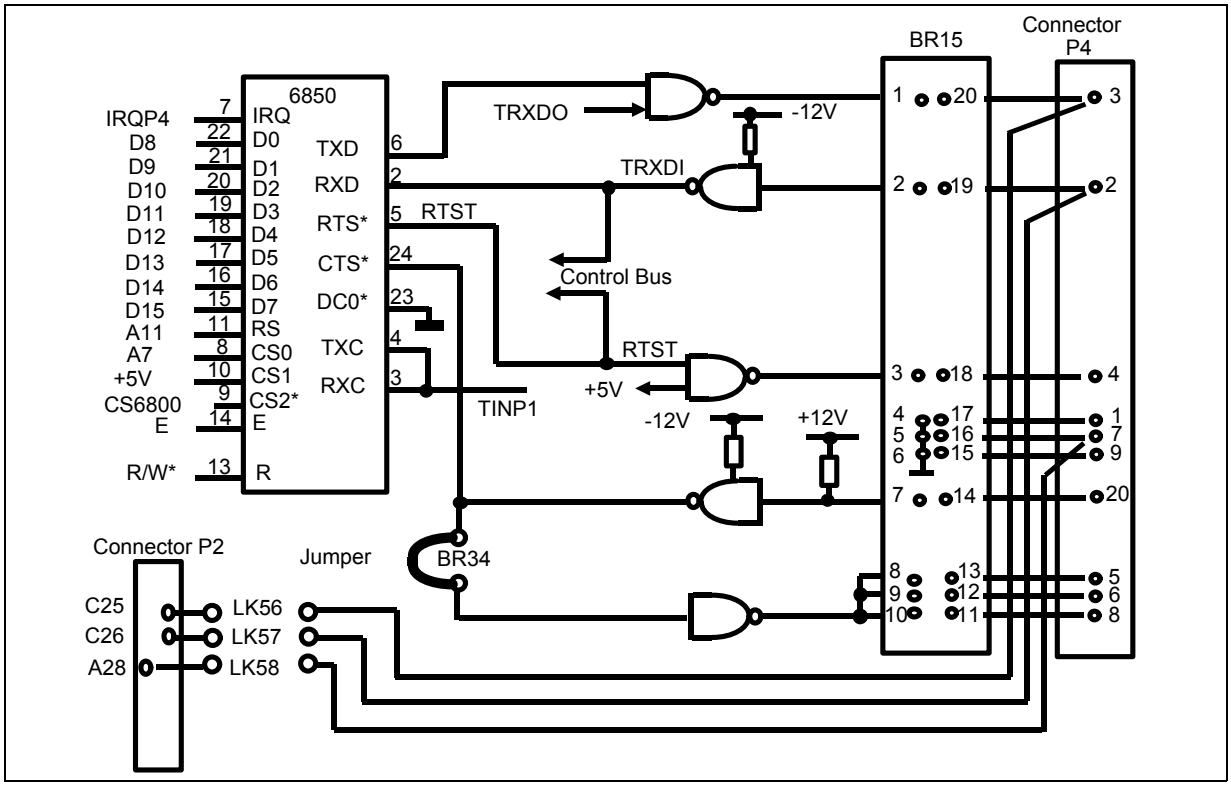


Figure 20 Terminal Interface



Remote Interface

The transmission format default settings and the method of re-selection is the same as the Terminal Interface. A hardware diagram is given in Fig. 22. The signal assignment of the pins of P5 are controlled by jumper block BR14 see Fig. 21. Default settings are as Table 2 in Chapter 3.

Figure 21 Location of Remote Interface Jumper

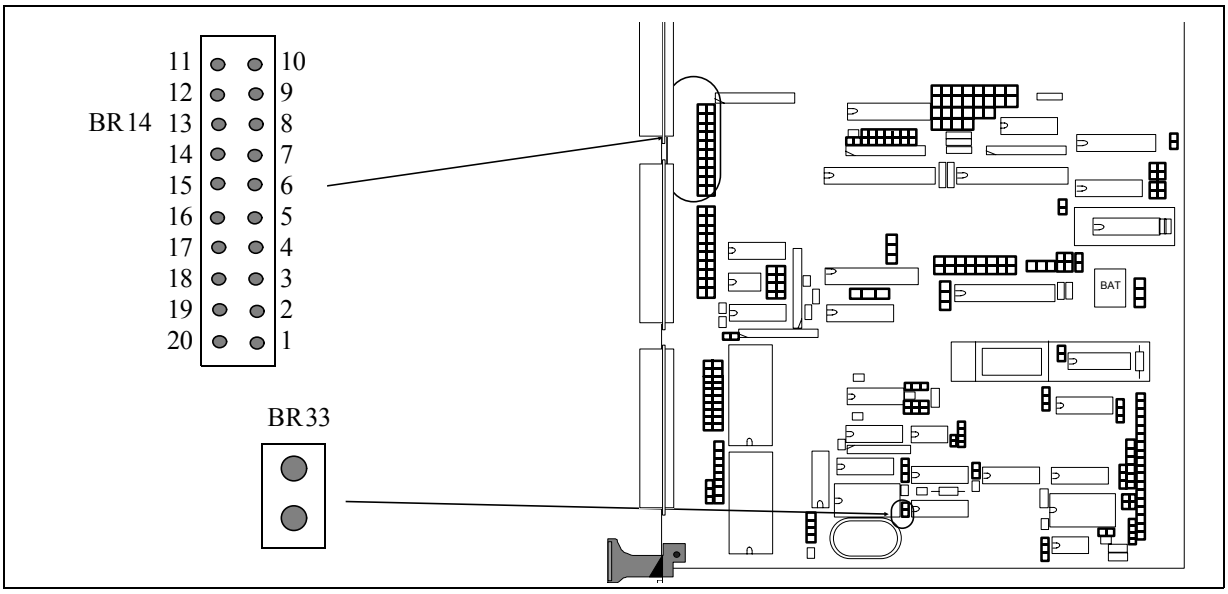
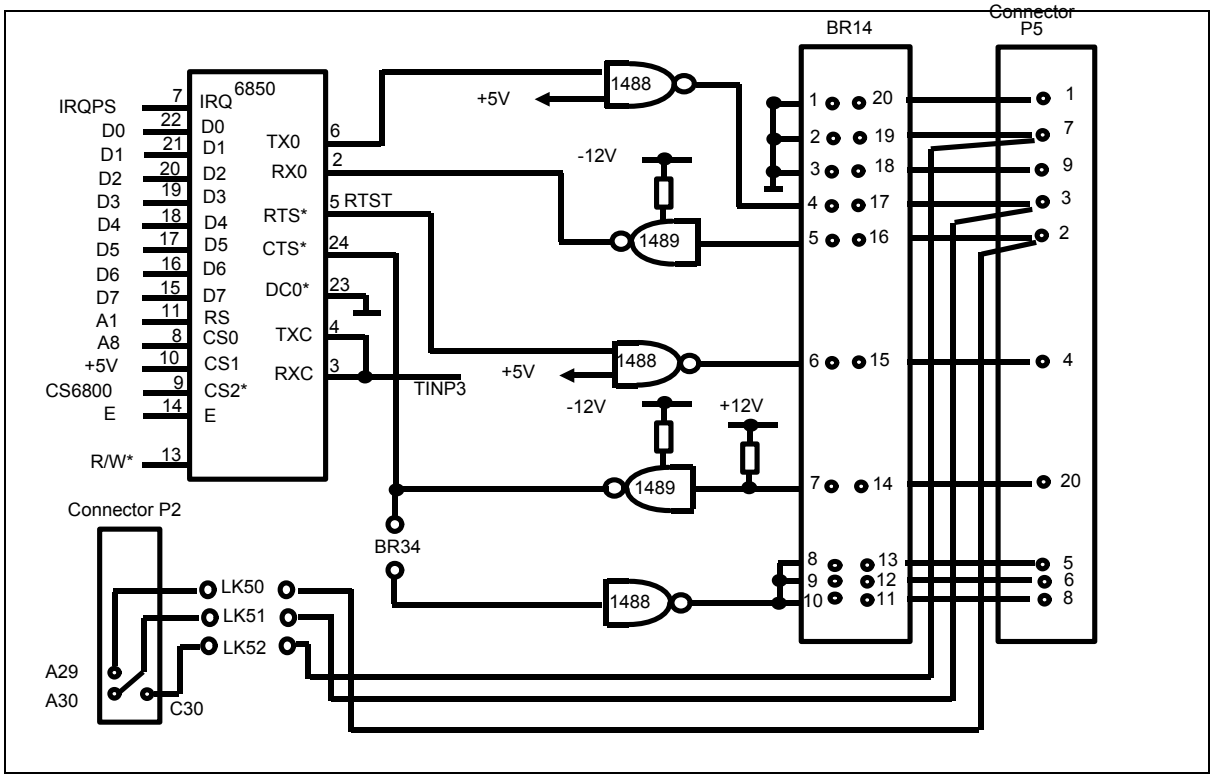


Figure 22 Remote Interface



Host Interface

Port 2 (connector P3) may be used for up/down loading of user programs and data or, in the transparent mode, to interface the terminal connected to Port 1 (connector P4) directly to a host computer. A diagram of the host interface is given in Figure 24 and the location of jumper block BR16 is shown below in Figure 23.

Figure 23 Location of Host Interface Jumpers

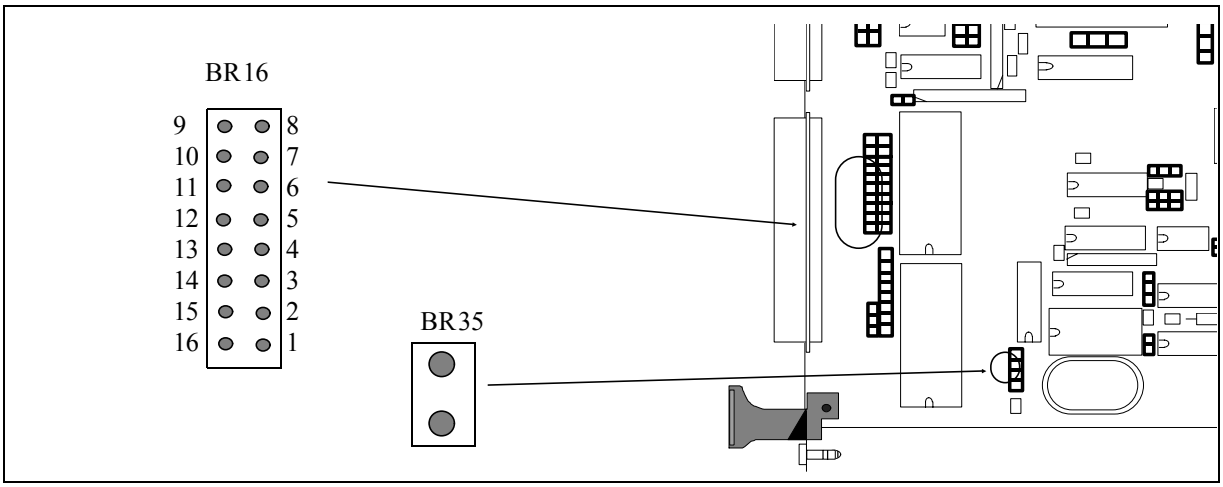
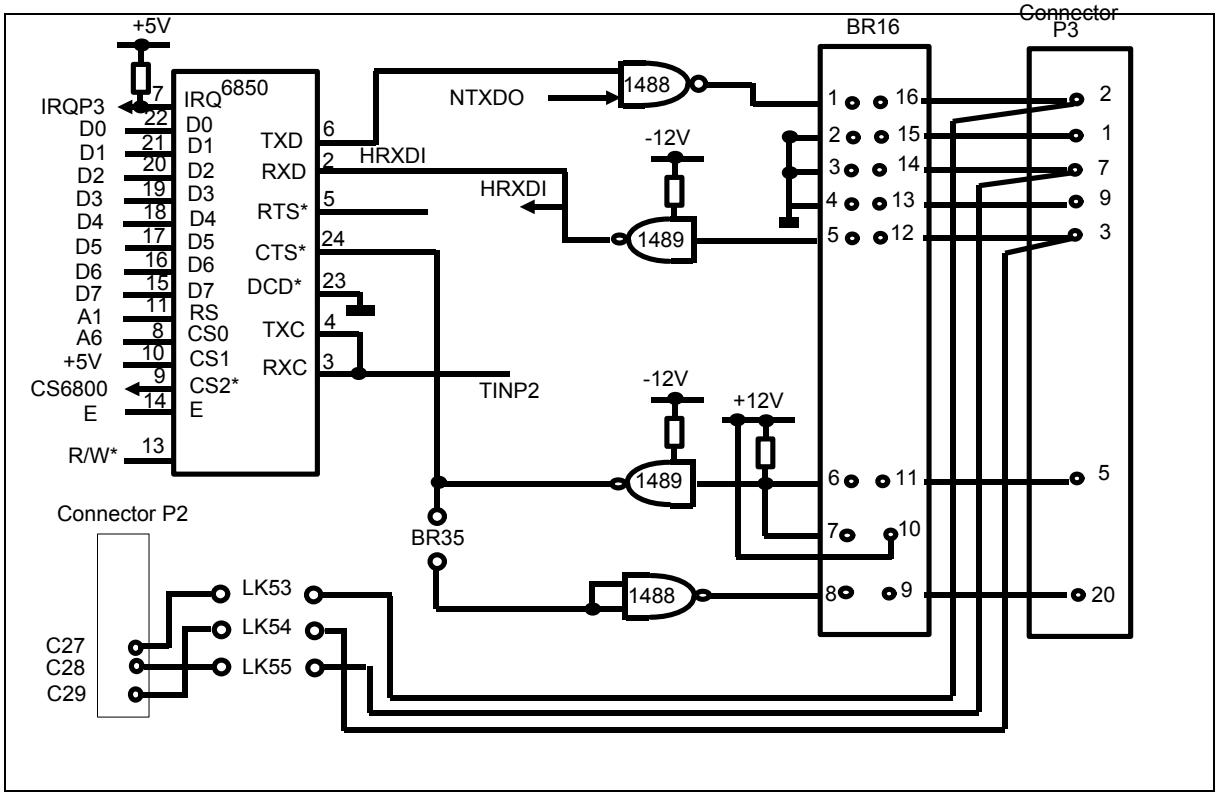


Figure 24 Host Interface



Baud Rate Selection

Tables 20 and 21 define the baud rate values available; Figure 26 shows the circuit diagram of the baud rate selection devices. Figure 25 shows the location of the baud rate selection areas.

Jumper field BR12 defines the ratio of the baud rates. A jumper between pin 1 and 3 (default condition) make the baud rates listed in Table 20 available. A jumper between pins 1 and 2 make the baud rates detailed in Table 21 available. Any number of connections can be made in areas WF1 and WF2.

Table 20 Baud Rate Selector (BR12 1-3)

Jumper Pin	Baud Rate	
WF1# 1	9600	TINP1, TINP2, TINP3 (9600 baud Default)
WF1# 2	4800	
WF1# 3	2400	
WF1# 4	1200	
WF3# 1	600	
WF3# 2	300	
WF3# 3	150	
WF3# 4	110	

Table 21 Baud Rate Selector (BR12 1-2)

Jumper Pin	Baud Rate
WF1# 1	38400
WF1# 2	19200
WF1# 3	9600
WF1# 4	4800
WF3# 1	2400
WF3# 2	1200
WF3# 3	600
WF3# 4	440

Figure 25 Location of the Baud Rate Selection Areas

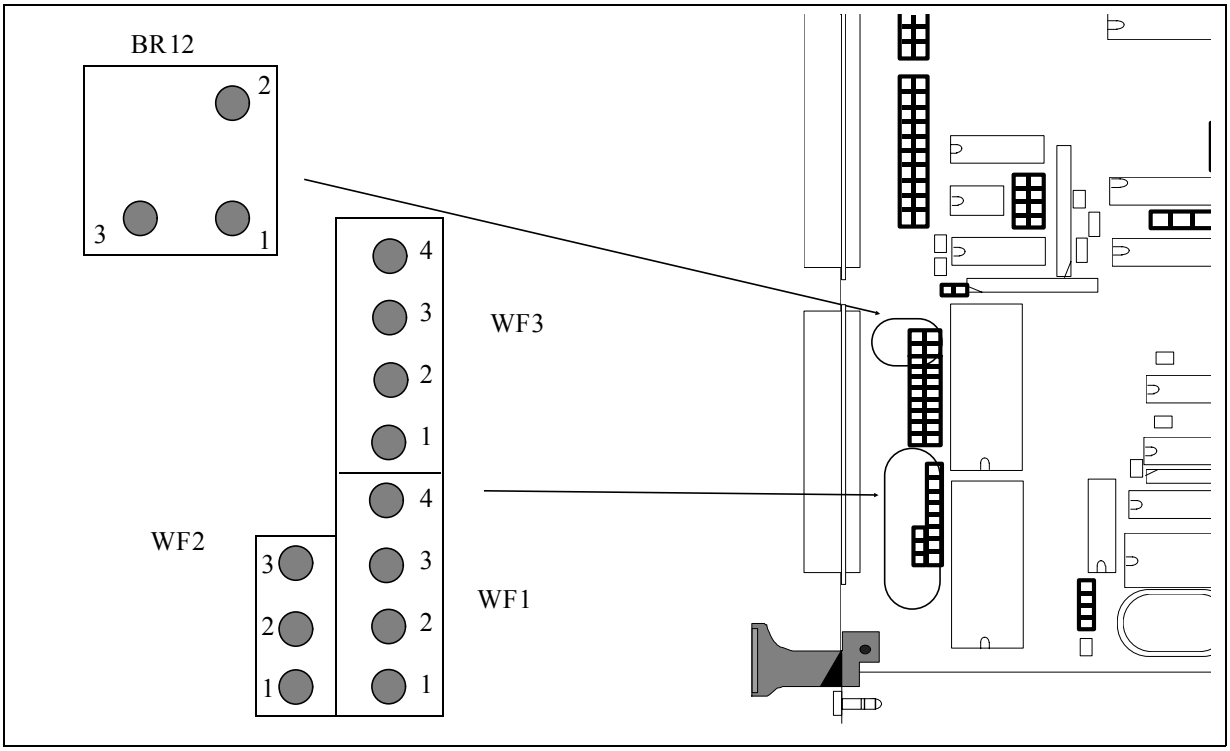
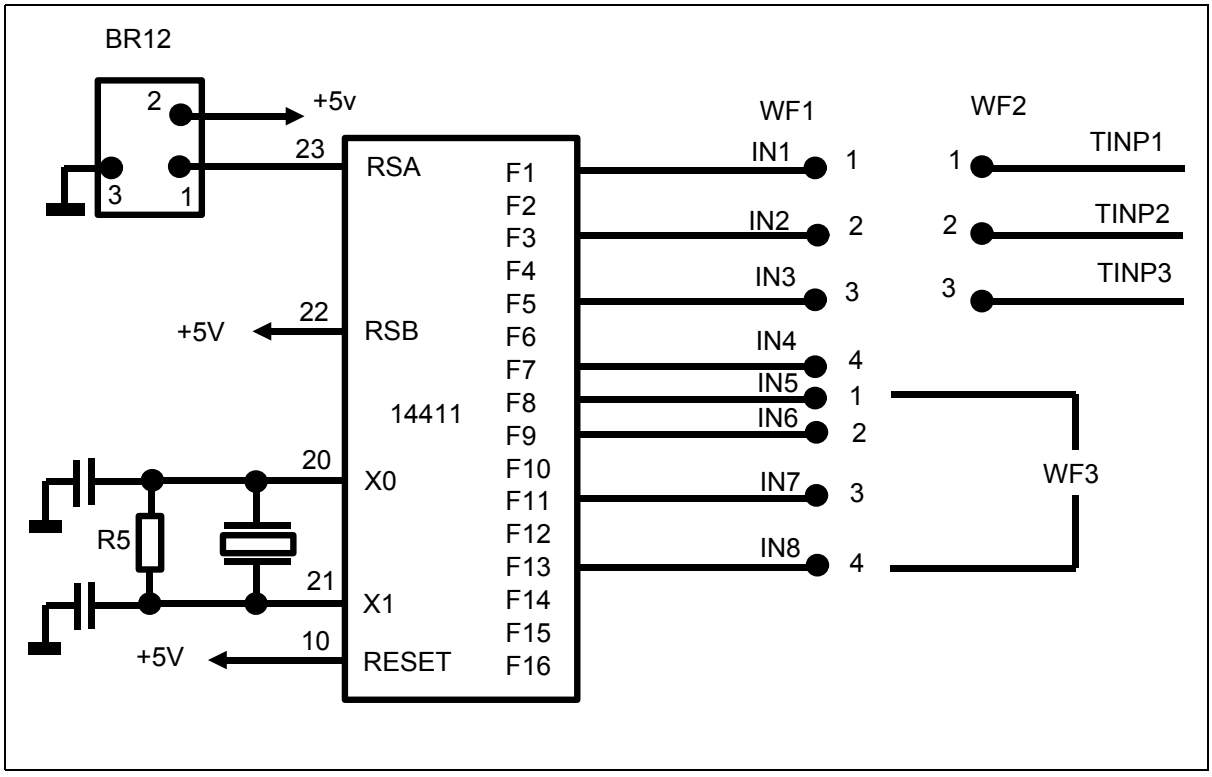


Figure 26 Baud Rate Selection Devices



RTC Interrupts

The RTC interrupt level is selectable between:

RTC interrupt

or

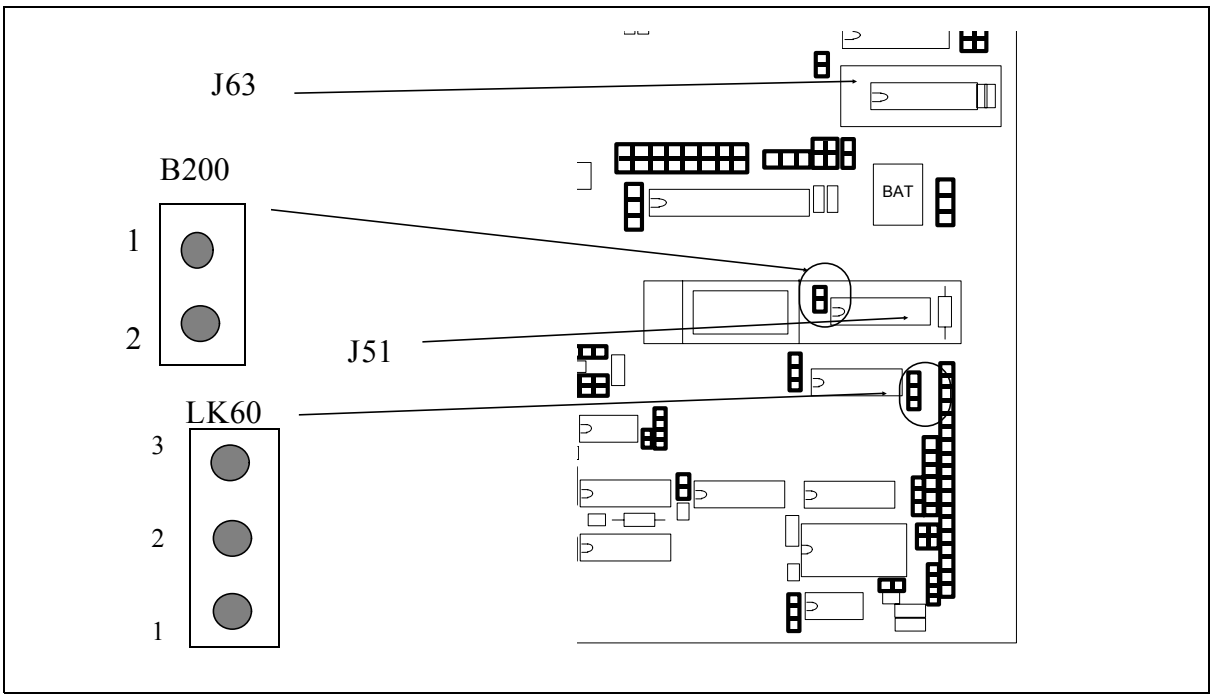
PIT PC3 interrupt

These options are set by LK60 and B200

	LK60	B200	
RTC only	2-3	IN	
PIT PC3 only	1-2	IN	*

* NOTE.
PIT PC3 by default interrupts on level 6; so for this combination BR18 Pin 2-3 should be removed. (See Figure 28)

Figure 27 Location of the RTC Jumper and RTC Chip.



Parallel I/O Interface and Timer

Centronics Type Interface

The PIT can be used as a Centronics type parallel interface. This allows connection of a Centronics compatible printer with the handshake protocol under full software control.

This application uses 8 input and output signals. The output signals are driven by a 74LS645-1 device with a drive capacity of 48mA. This allows a cable length of approximately 5 metres (16 feet) to the printer.

The signal assignment of the PIT is given in Figure 29. The location of BR18 jumper is shown in Figure 28. Chapter 3 covers the parallel interface in detail. This example is of a minimum configuration which uses the following signals only:

Signal	Type	Signal	Type
D0	Output	Data Strobe*	Output
D1	Output	Acknowledge*	Input
D2	Output	Select	Input
D3	Output	Paper Out	Input
D4	Output		
D5	Output		
D6	Output		

Figure 28 BR18 Location

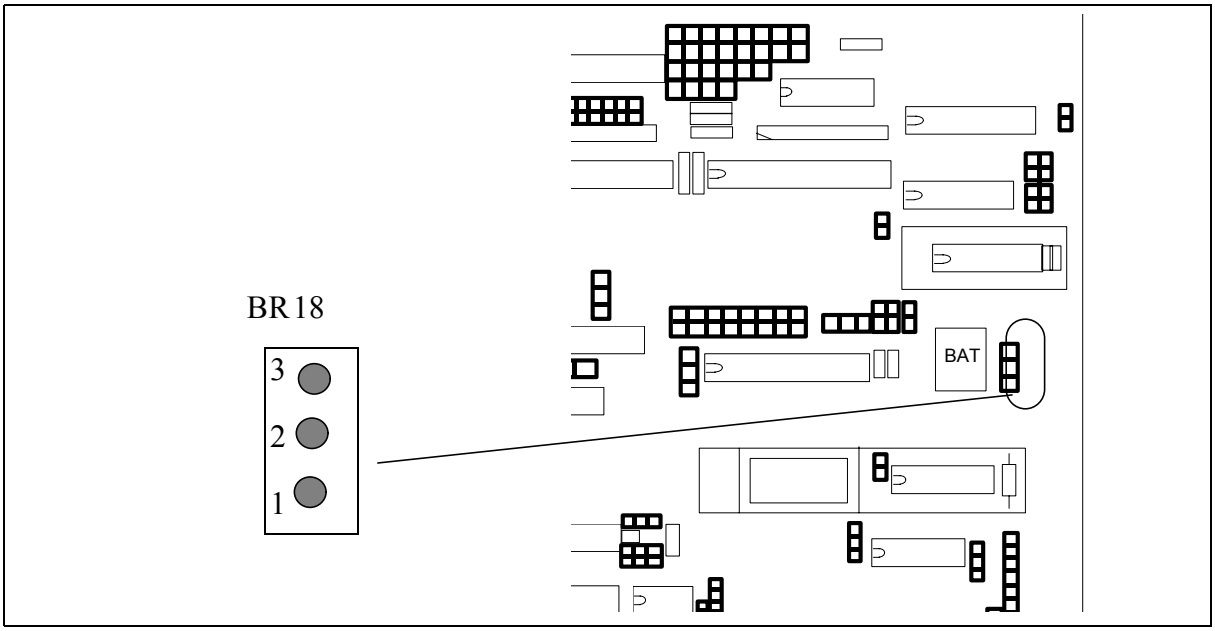
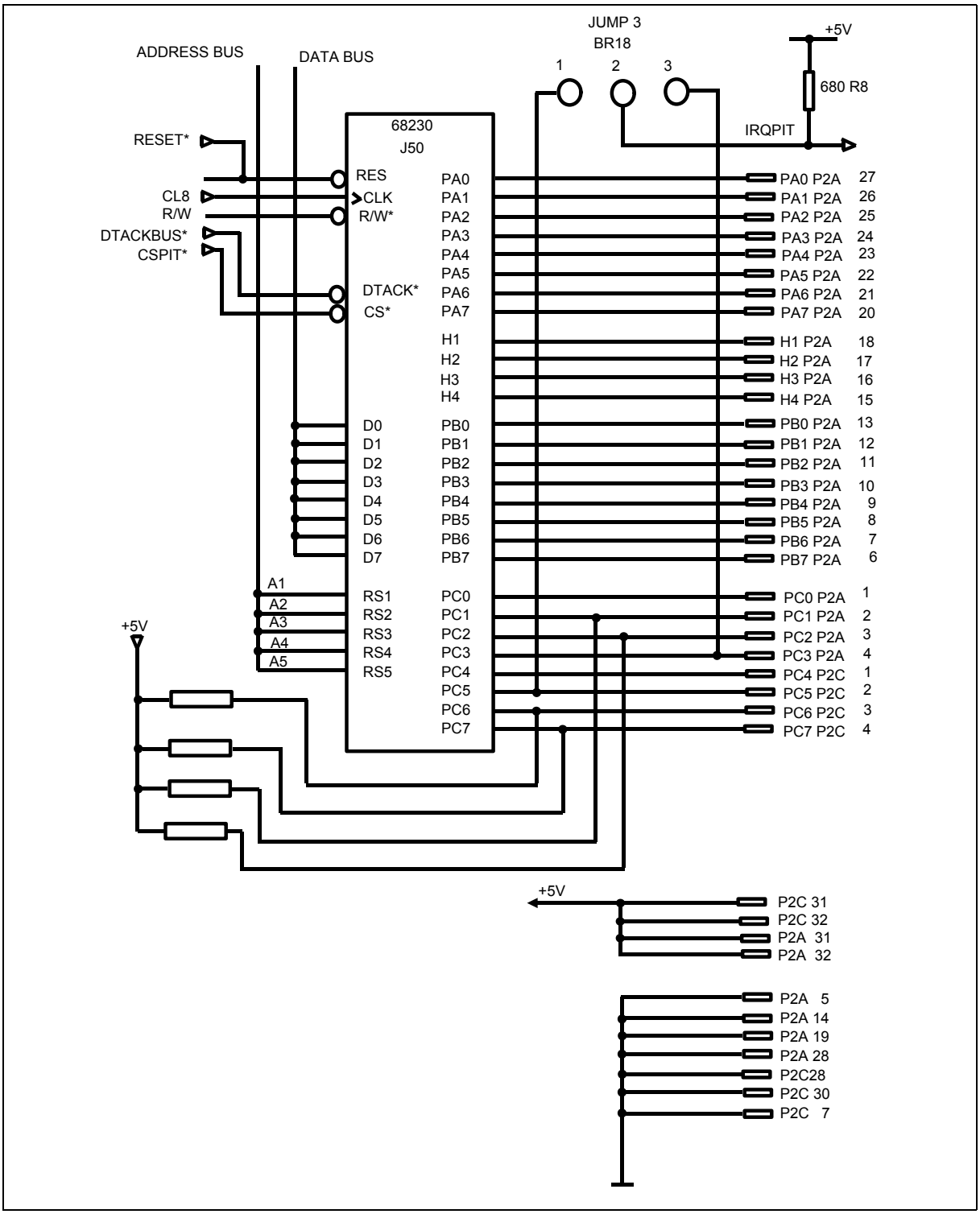


Figure 29 PIT Signal Assignment



Interrupt Handling

ACFAIL Interrupt

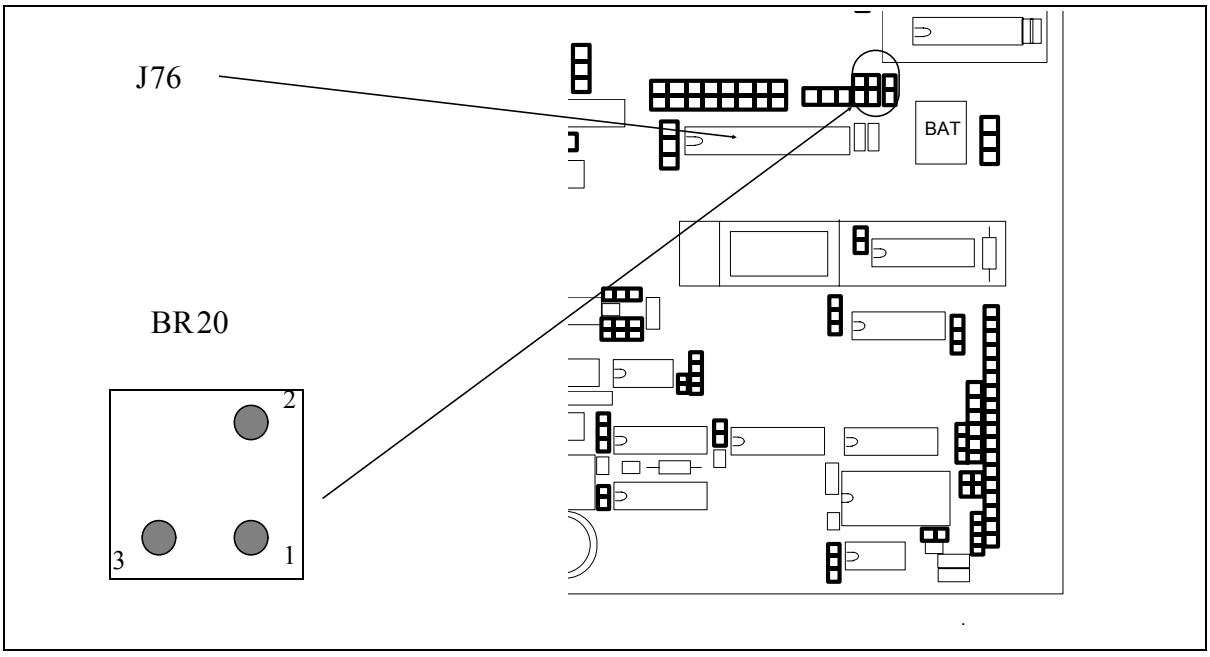
The position of the ACFAIL jumper field is BR20, shown on the location diagram Figure 30.

The pins of BR20 are connected as follows:

BR20	Connected to
1	ACFAIL* (from Receiver)
2	IRQ6
3	HALT (of 68000 CPU)

If a connection is made between pins 1 and 3, the on-board CPU will stop all activities immediately after the ACFAIL signal is asserted. The Red HALT LED on the front panel will light.

Figure 30 Location of the ACFAIL Jumper Field



If the link is removed, ACFAIL will have no effect.

VMEbus Interrupt Handling

All VMEbus interrupt signals, IRQ1 to IRQ7, can be enabled/disabled separately using jumper field WA1 to WA7, as shown in Figure 32. The position of the jumpers are shown below in Figure 31.

When a jumper is inserted an incoming IRQ signal will be acknowledged by the CPU; removing a jumper disables the equivalent IRQ signal. In the default condition, all IRQ signals will be acknowledged.

Figure 31 Location of VMEbus Interrupt Jumper

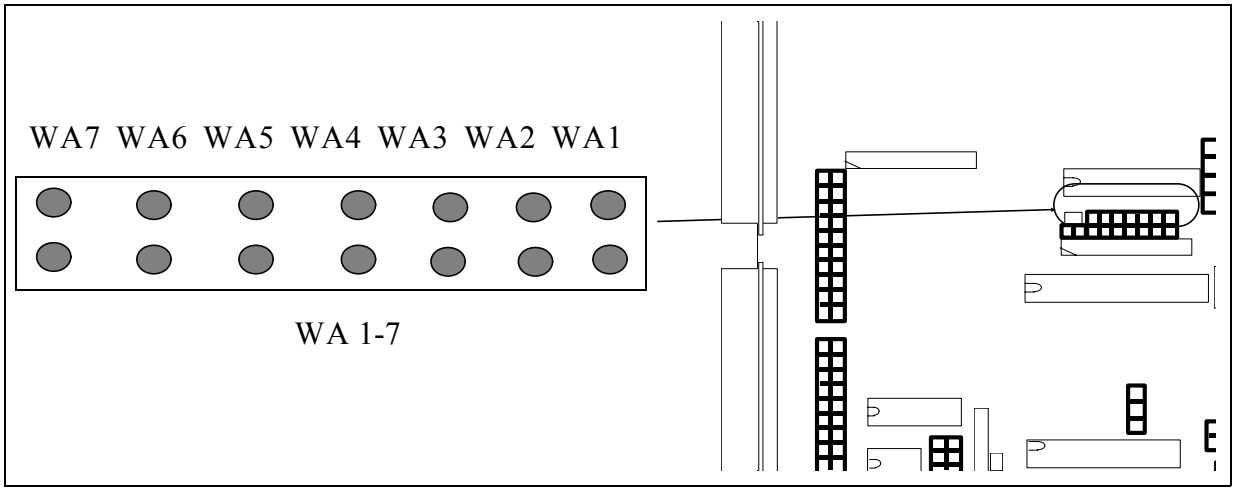
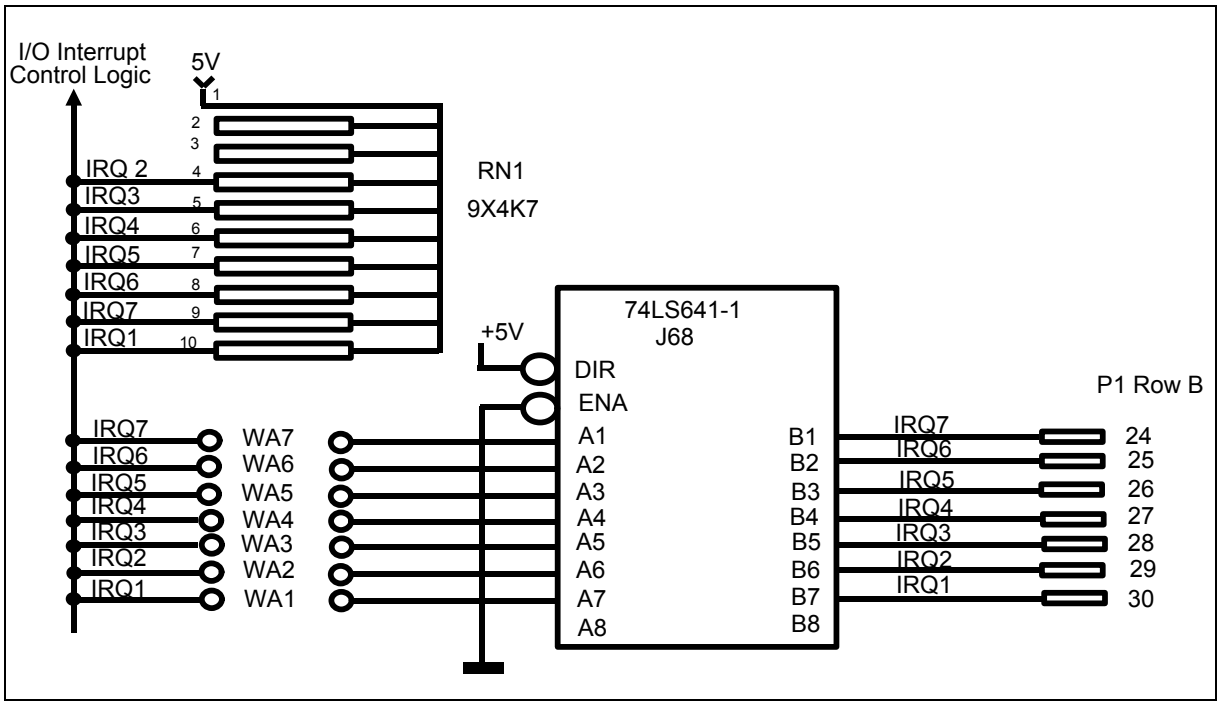


Figure 32 VMEbus Interrupt Jumper Field



VMEbus Arbitration

The board may be configured to use the on-board, single-level arbiter as the system arbiter, or to react correctly to off-board arbitration.

On-Board DTB Slave Bus Arbitration

The default slave Bus Request level is 3. Connections are made as listed in Table 21.

Table 21 Slave Arbitration Default Connections

Connections between:				
WI1	# 1	and	WI1	# 3
WI4	# 2	and	WI4	# 3
WB1	# 1	and	WB2	# 1

Figure 34 shows BRx*, BGxIN and BGxOUT jumper fields, the location of the jumpers is shown below in Figure 33.

Figure 33 Location of On-board DTB Slave Bus Arbitration Jumpers

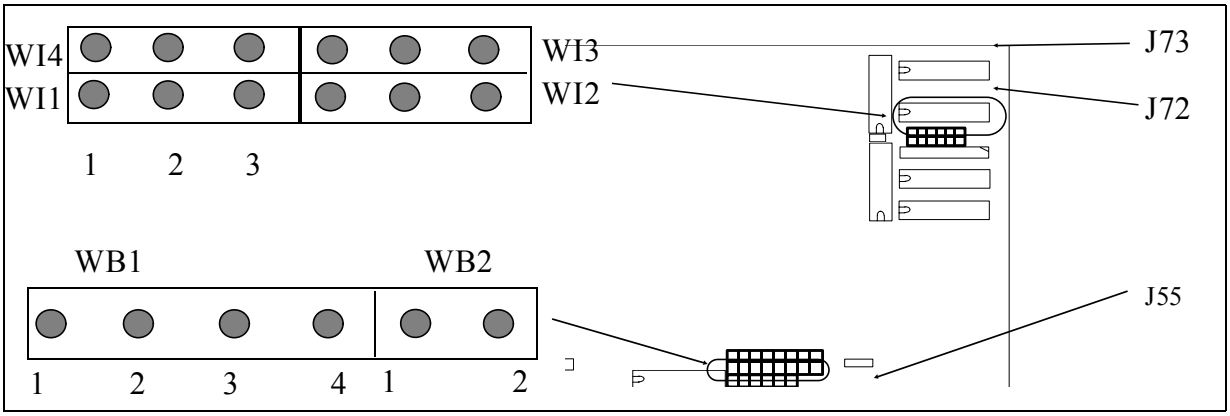
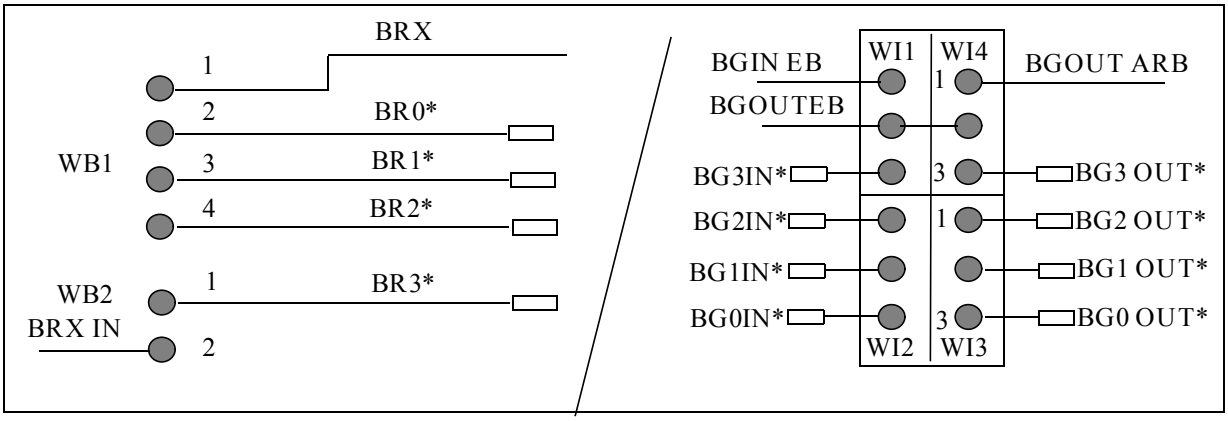


Figure 34 Jumper Fields for Slave Bus Arbitration



Off-Board Arbitration

In a four level bus arbitration scheme with an external arbiter, the bus request levels require the following connections: (See Figure 33 for positions of WB1, WB2, WI1# /2/3/4)

BR0*		**BR2***	
WB1# 1 to WB1# 2	WI2# 3 to WI3# 1	WB1# 1 to WB1# 4	WI1# 3 to WI4# 3
WI1# 1 to WI2# 3	WI2# 2 to WI3# 2	WI1# 1 to WI2# 1	WI2# 2 to WI3# 2
WI4# 2 to WI3# 3	WI1# 3 to WI4# 3	WI4# 2 to WI3# 1	WI2# 3 to WI3# 3
BR1*		**BR3***	
WB1# 1 to WB1# 3	WI2# 1 to WI3# 1	WB1# 1 to WB2# 1	WI2# 1 to WI3# 1
WI1# 1 to WI2# 2	WI1# 3 to WI4# 3	WI1# 1 to WI1# 3	WI2# 2 to WI3# 2
WI4# 2 to WI3# 2	WI2# 3 to WI3# 3	WI4# 2 to WI4# 3	WI2# 3 to WI3# 3

Bus-Release Functions

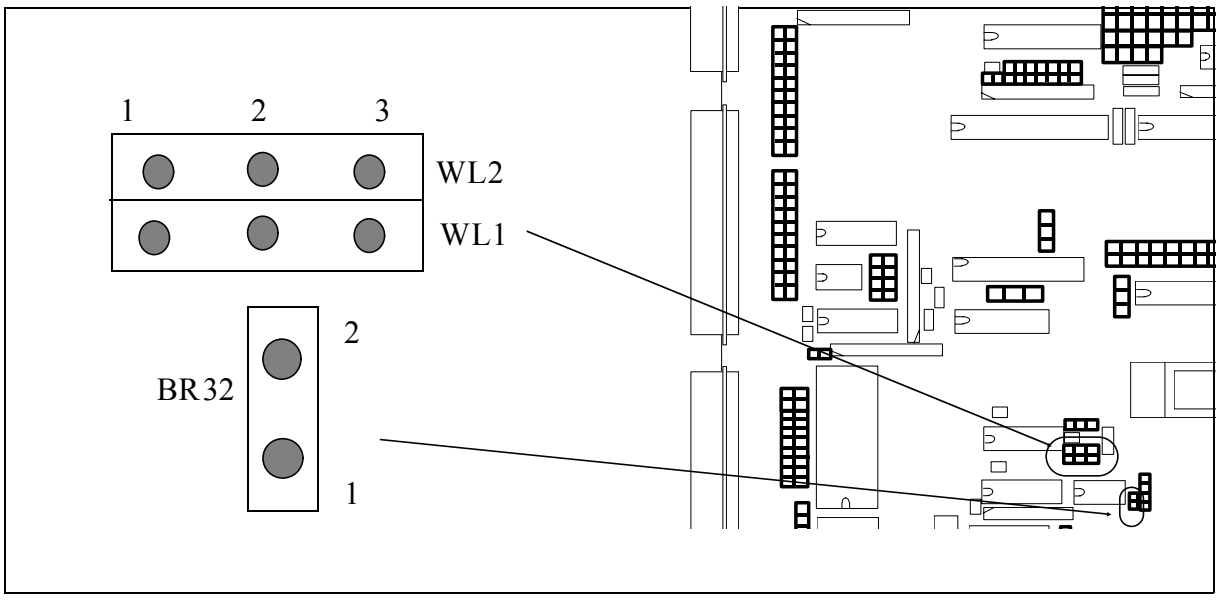
PME68-1B provides for a RAT (Release After Time-Out) function and in slave arbitration mode, release on BCLR.

The maximum continuous time available as bus master is defined by the setting of jumper fields WL1 and WL2 (BR32 must be closed). The location of these jumpers is shown in Figure 35.

Time Out (typical)	Connected Between	
30µs	WL1# 1	WL2# 1
60µs	WL1# 2	WL2# 2
300µs	WL1# 3	WL2# 3 (Default Setting)

If WL2# 1 is connected to WL3# 1, every transfer on the VMEbus causes an arbitration cycle.

Figure 35 Location of Bus Release Time Jumpers



VMEBUS Interface

SYSCLK Signal

The on-board SYSCLK driver can be isolated from the VMEbus via a jumper in position BR21.

Jumper Inserted	SYSCLK connected to VMEbus
No Jumper	SYSCLK signal not connected to VMEbus

Figure 36 shows the location of jumper BR21.

SYSRESET* Signal

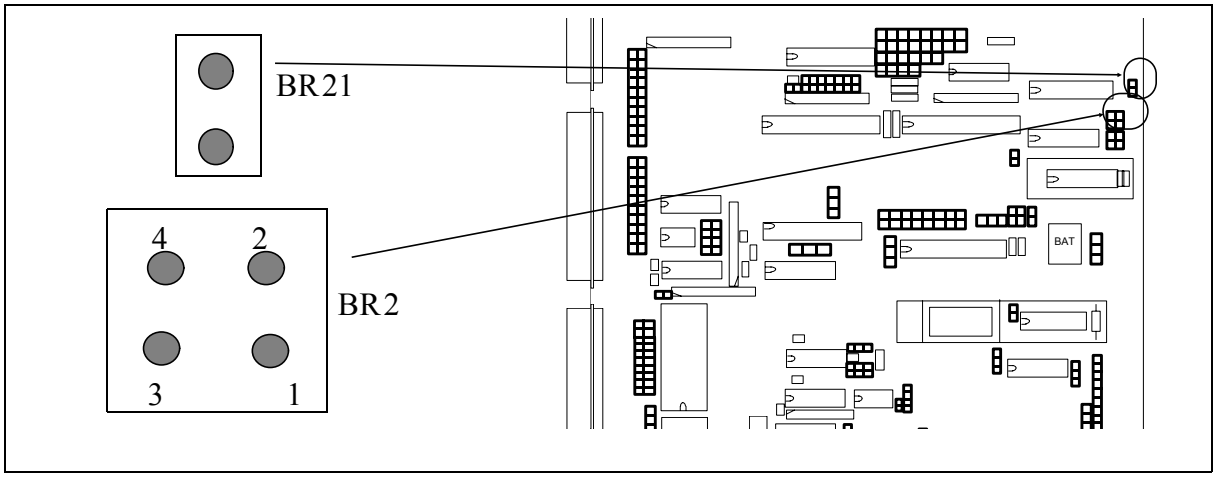
Jumper field BR2 defines whether the board drives, receives, or ignores the VMEbus signal SYSRESET*.

Table 22 lists all permitted jumper settings and the corresponding modes. Figure 36 shows the location of jumper field BR2.

Table 22 SYSRESET Settings*

BR2 Connection Between	Effect on PME68-1B
# 1-# 2, # 3-# 4	Receives SYSRESET*
# 1-# 3, # 2-# 4	Drives SYSRESET* (default condition)
No Connections	SYSRESET* ignored (except by RTC)

Figure 36 Location of Jumper Field BR2 and BR21



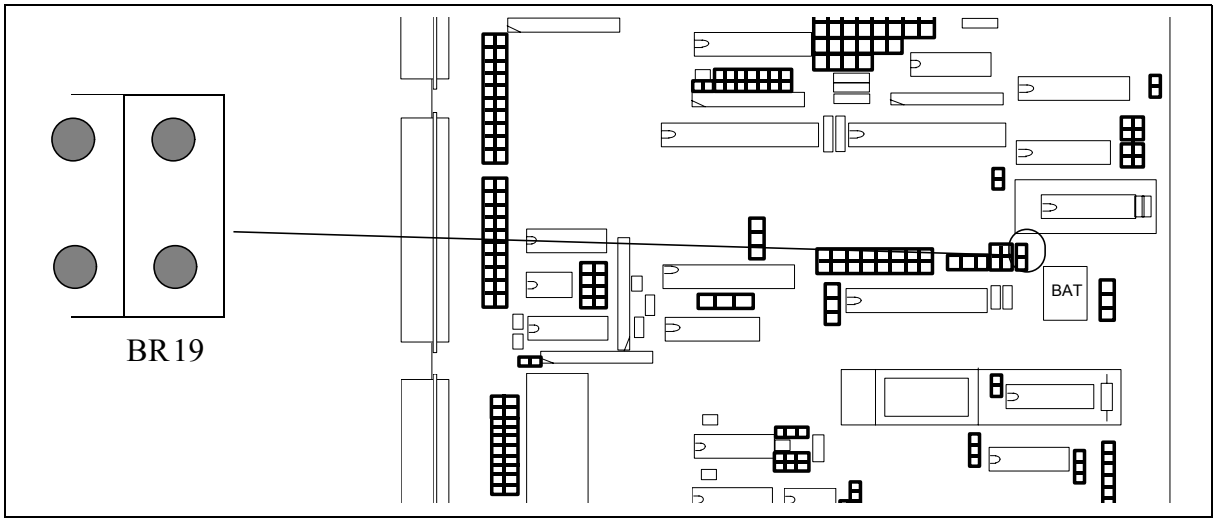
SYSFAIL* Signal

The VMEbus SYSFAIL* signal informs all cards that one or more cards in the system has failed. On PME 68-1B boards this input signal is directly connected to jumper field BR19.

If the jumper is installed an active SYSFAIL* signal causes the on-board CPU to jump directly to the HALT state and the red HALT LED on the front panel to light.

If no jumper is inserted the SYSFAIL* signal is ignored.

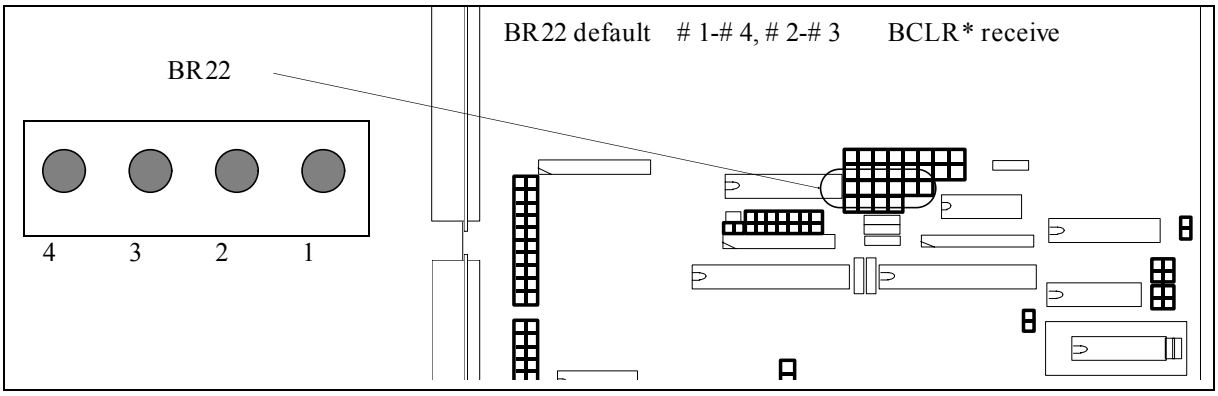
Figure 37 Location of SYSFAIL Jumper Field BR19*



BCLR* Signal

The PME 68-1B only supports single-level arbitration, therefore, a BCLR* signal is not generated. However, the BCLR* input signal is received in the board default state but can be disabled by removing BR22 links.

Figure 38 Location of Jumper Field BR22
If the jumper BR22 is configured to receive BCLR, an active BCLR will force the 68-1B to release the bus at the end of the current cycle.



Short I/O Address Modifier Code

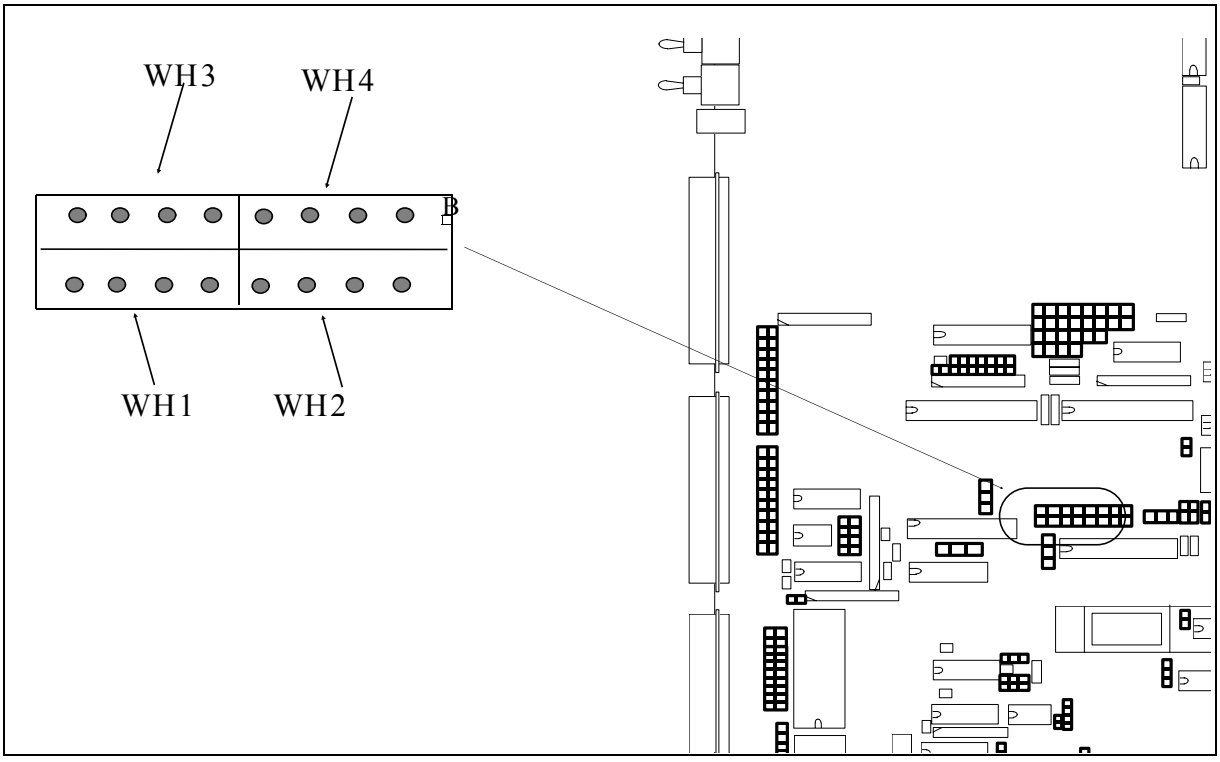
The 64k byte range is jumper selectable within the range $100000 to $FFFFFF, as defined by jumper fields WH1, WH2, WH3 and WH4.

For valid decoding, jumper settings are:

OUT	the corresponding address signal must be high (1).
IN	the corresponding address signal must be low (0).

Table 23 lists the jumper positions, the corresponding address signals and the default condition. Figure 40 illustrates the jumper circuit arrangement. The position of the short I/O jumpers are shown on the location diagram Figure 39.

Figure 39 Location of Short I/O Jumpers



The binary code set on WH1-WH3 and WH2-WH4 is compared with the binary code of address bits A16-A23 by the 74LS688. The AM4 signal is generated when A16-A23 matches the link setting. The settings therefore determine the two most significant HEX digits for the start address of the I/O short address range. e.g. 2A0000 to 2AFFFF.

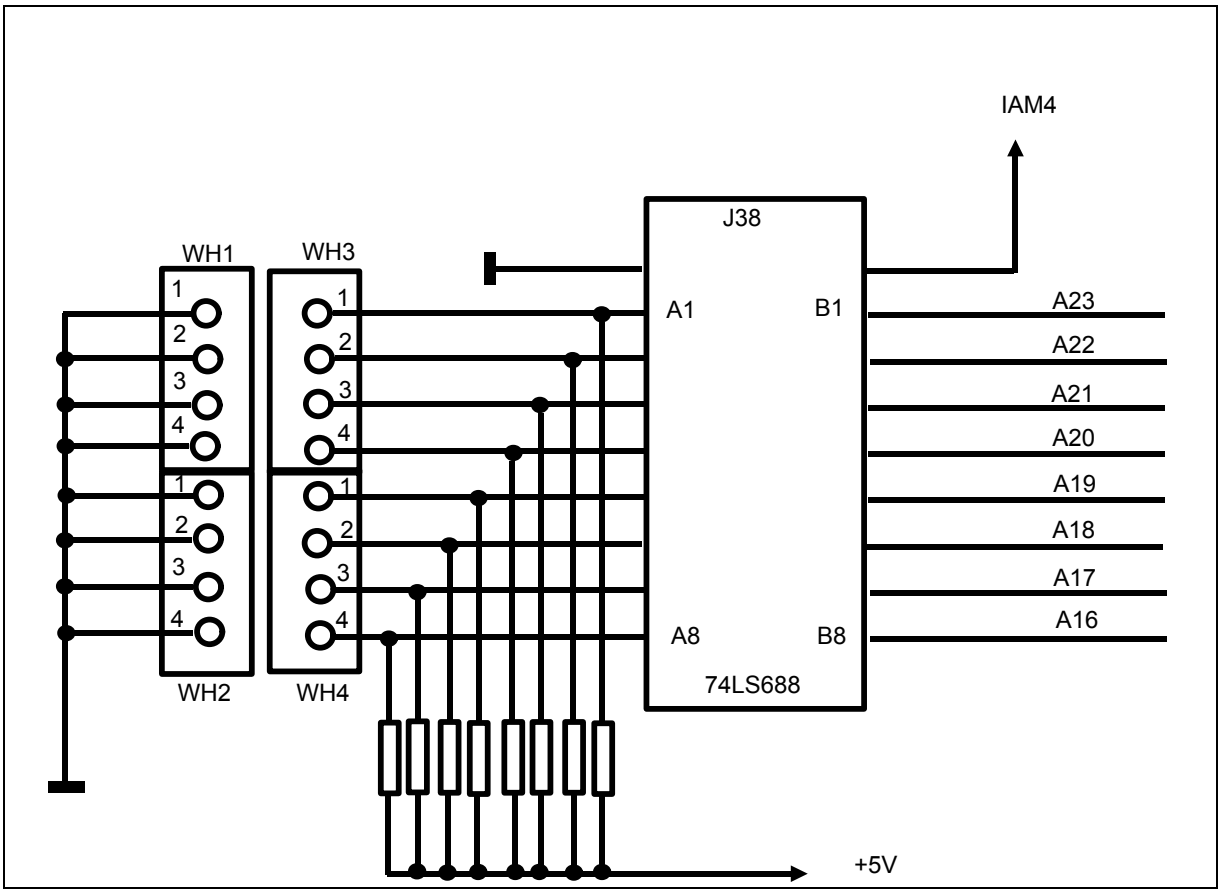
Table 23 Short I/O Jumper Settings

Jumper Positions	Corresponding Address Signal	Default Setting
WH1# 1 / WH3# 1	A23	OUT
WH1# 2 / WH3# 2	A22	OUT
WH1# 3 / WH3# 3	A21	OUT
WH1# 4 / WH3# 4	A20	OUT
WH2# 1 / WH4# 1	A19	OUT
WH2# 2 / WH4# 2	A18	OUT
WH2# 3 / WH4# 3	A17	OUT
WH2# 4 / WH4# 4	A16	OUT

Default Short I/O Start Address : $FF0000

End Address : $FFFFFF

Figure 40 Short I/O Comparator



Bus Error Function

The internal time-out value (BERRINT) can be altered by changing the settings of jumpers BR26 and WK1/WK2. The initial timing is set by WK1/WK2, BR26 alters this value by a factor of 10.

The other jumpers involved in the chain must be left in their default positions, i.e.

BR27 # 1 to # 2
BR28 # 1 to # 2 and # 3 to # 4
WK1# 1 to WK1# 2

Table 24 lists possible time-out values and the jumper settings required. Figure 41 shows the location of the jumper fields.

Table 24 Time-Out Counter Settings

	BR26	
	1 - 2 (E)	2 - 3 (CL8)
WK1 # 3 to WK2 # 3	250µ5P255Ds	25µs
WK1 # 2 to WK2 # 2	320µs	32µs
WK1 # 4 to WK2 # 4	2.5ms**	250µs

** is the default setting

For example, to set BERRINT to 25µs the jumpers must be set as follows:

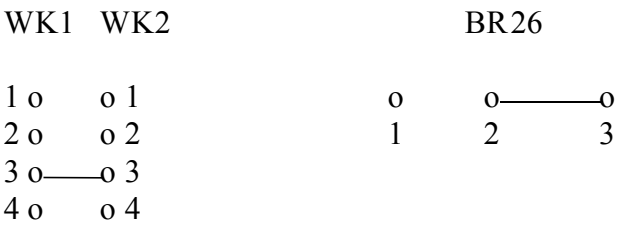
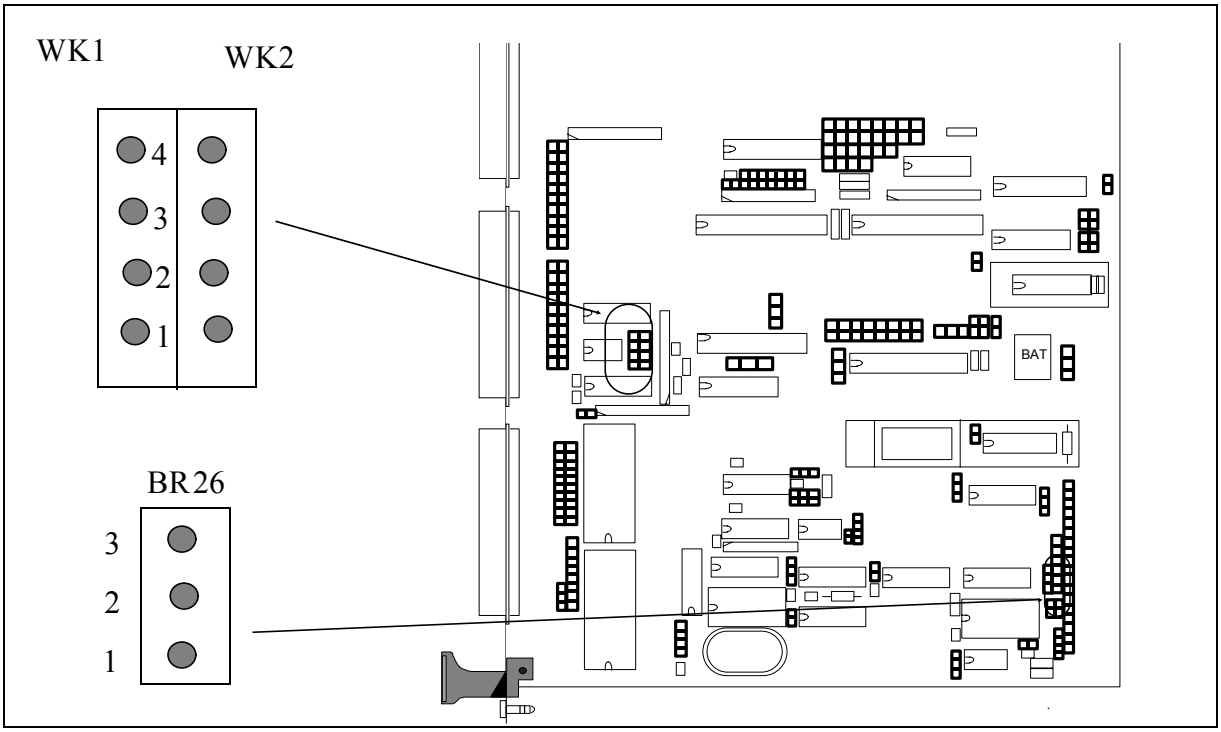


Figure 41 Location of BERR Jumpers*



Chapter 5 - PME68/Monitor

Software Capabilities

The PME68/Monitor resides in on-board EPROM. This 16k byte package provides a user interface to the hardware of the PME 68-1B CPU board. The system monitor controls communication with a terminal and exercises all other elements of the system as well as providing complete debug capability and I/O control. An assembly/disassembly utility supports user program development and serves as an interface for user or application software such as macro-assemblers and high level language interpreters.

During program development, a dynamic assembler/disassembler function is used. In this mode the source program is not saved. Each instruction is translated into the proper op-code and stored in memory on a line-by-line basis at the time of entry. The assembler source statement is composed of operation and operand fields. Line numbers, labels, and comments are not allowed.

When displaying an instruction, the firmware disassembles the op-code and displays the instruction mnemonic and operands. Editing is done by re-entering a source statement.

If a higher level of assembly capability is required the software can be developed on any suitable host computer and down/up-loaded via the PME 68-1B serial Port 2.

The PME68/Monitor has the following capabilities and features:

Block fill memory - from add1 through add2 with data
Block move - move from add1 through add2 to add3
Set/display breakpoint
Block search - search add1 through add2
Block test of memory
Data conversion
Display formatted registers
Dump memory to object file
Execute program
Go direct
Execute program: temporary breakpoint
Display monitor commands
Load object file
Display memory
Modify memory
Set memory - starting at add with data1, data2,...
Remove breakpoint
Detach printer
Offset
Attach printer
Set/display port format
Transparent mode
Trace
Trace: temporary breakpoint
Verify memory/object file
Display/set address register
Display/set data register
Display/set offset register
Display/set program counter
Display/set status register
Display/set supervisor stack
Display/set user stack
Disassemble memory location
Disassemble/assemble memory location

General System Overview

Power-Up Sequence

During the power-up routine, initialisation of hardware is performed as described in this Manual.

The 68000 CPU fetches the initial Supervisor Stack Pointer and start address from locations $0 and $4. These are the first 8 bytes of the system EPROM area; they are down mapped to provide a defined start following a reset or during power-up. They pass control to the Monitor firmware which performs all the necessary initialisation routines.

$0000 – $0003 Initial Supervisor Stack

$0004 – $0007 Initial Start Address

RESET Switch

RESET is provided by Switch 1 (SW1) on the PME 68-1B front panel. Pressing this switch causes all programs or processes to terminate, resets the CPU and the parallel I/O devices, and restarts the resident firmware. The RESET switch may be used if 'all else fails'.

ABORT Switch

The ABORT switch is SW2 on the CPU front panel. It causes an interrupt at the highest priority level (IRQ level 7 of the CPU); it cannot be masked or disabled. Control is immediately transferred via the interrupt service routine to the Monitor main control routine. ABORT also causes the contents of the 68000 internal registers to be displayed. ABORT differs from RESET in that the processor register and memory contents are not modified and the status of the processor registers and peripherals is not changed.

Using the abort function or trace breakpoints is the best way to debug software as the user can interrupt the processor without destroying the present state of the system.

Vectors and Errors

Exception vectors are memory locations from which the processor fetches the address of a routine which will handle an exception. All exception vectors are two words in length (32 bit). See the following table:

Vector Number(s)	Address Dec	Hex	Space	Assignment
0	0	000	SP	Reset: Initial SSP
-	4	004	SP	Reset: Initial PC
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12	48	030	SD	(Unassigned, reserved)
13	52	034	SD	(Unassigned, reserved)
14	56	038	SD	(Unassigned, reserved)
15	60	03C	SD	Uninitialised Interrupt Vector
16 : 23	64 95	040 05F	SD	(Unassigned, reserved)
24	96	060	SD	Spurious Interrupt
25	100	064	SD	Level # 1 Interrupt Auto Vector
26	104	068	SD	Level # 2 Interrupt Auto Vector
27	108	06C	SD	Level # 3 Interrupt Auto Vector
28	112	070	SD	Level # 4 Interrupt Auto Vector
29	116	074	SD	Level # 5 Interrupt Auto Vector
30	120	078	SD	Level # 6 Interrupt Auto Vector
31	124	07C	SD	Level # 7 Interrupt Auto Vector
32 : 47	128 191	080 0BF	SD	TRAP Instruction Vectors
48 : 63	192 255	0C0 0FF	SD	(Unassigned, reserved)
64 : 255	256 1023	100 3FF	SD	User Interrupt Vectors

Interrupt Level Assignment

The PME68/Monitor has the following interrupt level vector assignments:

Default Interrupt Levels of the VMEBus	Exception Vector (68000 CPU)	Comment
Spurious Interrupt	$ 60	INTR TRAP ERROR
IRQ7*	$ 7C	AV # 7 TRAP ERROR
IRQ6*	$ 78	AV # 6 TRAP ERROR
IRQ5*	$ 74	AV # 5 TRAP ERROR
IRQ4*	$ 70	AV # 4 TRAP ERROR
IRQ3*	$ 6C	AV # 3 TRAP ERROR
IRQ2*	$ 68	AV # 2 TRAP ERROR
IRQ1*	$ 64	AV # 1 TRAP ERROR

NOTE: The USER INTERRUPT vectors from $100 - $3FF are preset with the message: ???? TRAP ERROR.

The hardware interrupt level assignment is described in Chapter 3.

As the Monitor shares resources with the user application program, the contents of individual vectors must be saved and restored to ensure proper function of the firmware.

The vectors are re-initialised during a RESET sequence.

If a Trap through one of these locations occurs, a message is output identifying the vector address, i.e.

BUS TRAP ERROR
ADDR TRAP ERROR
UT0 TRAP ERROR
AV7 TRAP ERROR

and the internal registers of the CPU are displayed.

Return to the Monitor

Return to the PME68/Monitor may be initiated via software. A useful means of doing this from a user application program is:

```
        +
        +
        +
-- User Program --
        +
        +
        +
        TRAP # 14
```

i.e. Adding a TRAP # 14 instruction at the end of the user program.

The other user Trap locations, from UT0 to UT13 and UT15, are preset with the appropriate error handling routine, they can be used by application programs.

A coldstart of the Monitor (complete reinitialisation) is done via an indirect jump to the initial start address ($4). Coldstart calls (i.e. pressing the RESET switch) should be handled with care as the old status of the system is destroyed and breakpoints may be lost from application programs.

The HALT Indicator

A red LED on the front panel indicates the status of the CPU, it is hardwired to the HALT signal line of the processor. During normal operation the LED should not light. If a failure occurs only a RESET can restart the CPU.

Note: The LED is not accessible by software.

PME68/Monitor Memory Map

Certain areas are not available to the user because the firmware employs some of the board's facilities. System RAM area must be accessed with care to ensure proper operation of PME/Monitor.

$000000 : : $000007	System EPROM
$000008 : : $000FFF	System RAM
$001000 : : $01FFFF	User RAM
$020000 : : $080007	Reserved
$080008 : : $09FFFF	System EPROM
$0A0000 : : $0BFFFF	User EPROM
$0C0000 : : $100000	I/O devices

The memory map of the PME 68-1B is shown in Chapter 3.

Operation of the PME68/Monitor

Operating Procedure

Once the module has been set up and power applied, automatic CPU reset circuitry gives program control to the monitor. Once the initialisation sequence is complete the system is ready to communicate with a user terminal attached to Port 1 (connector P4) of the CPU. The terminal must be set initially at 9600 Baud, the user may change the transmission format of the RS-232C interface using the monitor command PF described in this chapter. Initial termination setting is:

Baud rate hardwired: 9600 Baud

Format:	8 bit
	1 Stop Bit
	No Parity
	No Protocol

For further details refer to Chapter 3.

A prompt message is output to the terminal at system start up:

PME68/MONITOR v.r.

where v.r is the version/revised number. Whenever the monitor "prompt" (>) appears, a valid command may be entered at the terminal.

System Operation

The user can now enter any command supported by the firmware. A standard input routine controls the system while a line of input is typed in. Command execution starts after the line has been entered, followed by a carriage return.

The following I/O devices are supported by the firmware:

Port 1	User Terminal	(ACIA 6850)
Port 2	Host	(ACIA 6850)
Port 3	Parallel Printer	(PIT 68230)

The parallel Centronics type printer interface system must be configured by the user (refer to Chapter 4). Other devices, such as the RTC and the serial REMOTE interface are not supported by the firmware but are free for user applications. Examples of programs for these devices are shown in Address Assignment of I/O Devices later in the chapter.

If a command causes the firmware to access any unused address (i.e. no memory or I/O device is located at that address), a bus trap error will occur. The contents of all CPU registers are displayed and a return to the standard input routine is performed. This also occurs if an attempt is made to write to a reserved memory area.

Terminal Control Characters

Several keys are used as command line edit and control functions. It is best to be familiar with these functions before exercising the system. Some of these functions are terminal dependent, such as:

a) DELETE (RUBOUT) key or CTRL H - deletes the last character entered on the terminal

b) CTRL X - cancels the entire line

c) CTRL D - re-displays the entire line

d) RETURN (carriage return)-enters the command line and causes processing to begin

e) BREAK - aborts commands that perform any console I/O and return to the input.

Command Line Format

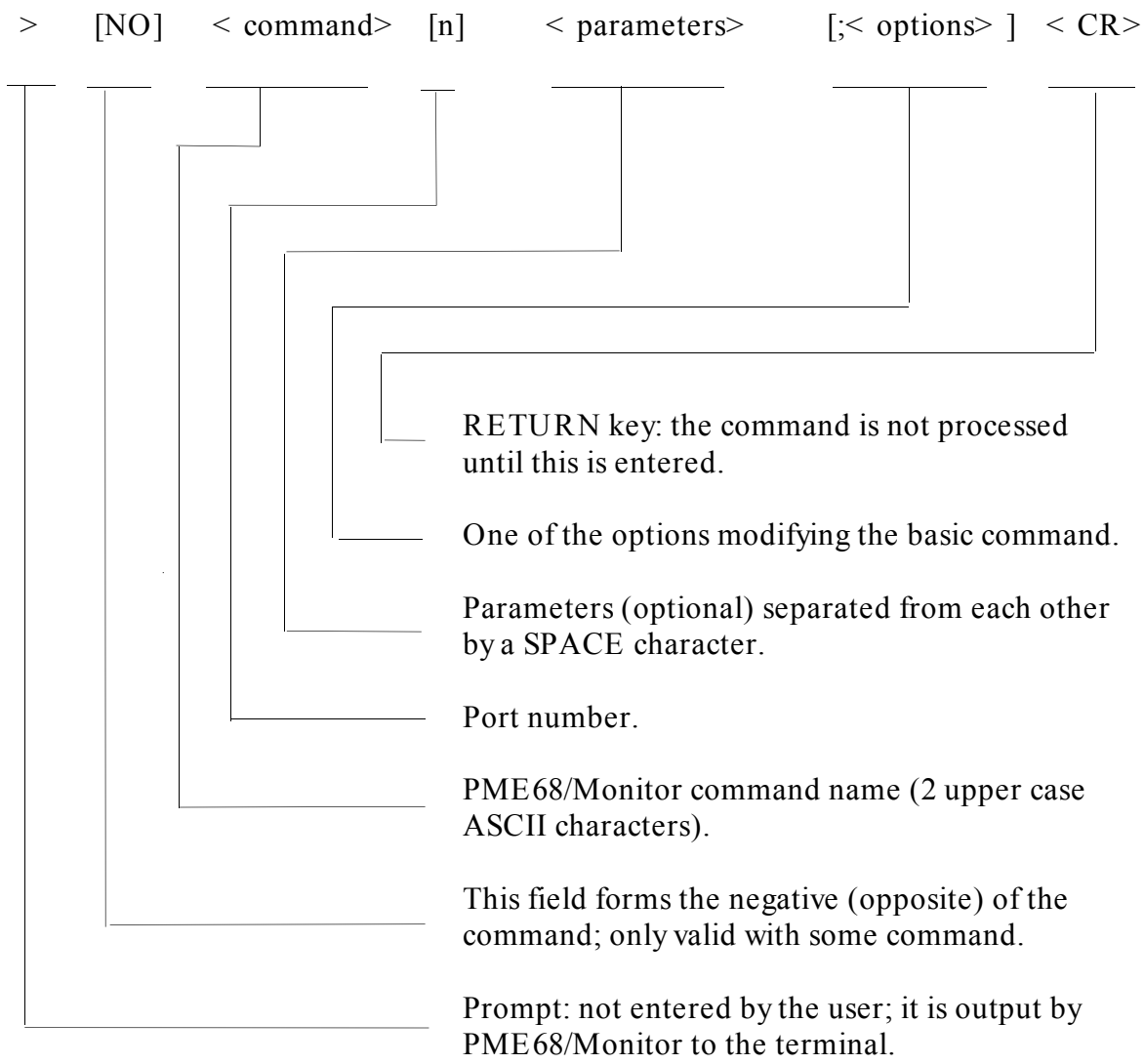
Notation

Commands are entered in the same way as most buffer organised computer systems. A standard input routine controls the system while the user enters a line. The DELETE (RUBOUT) key or CTRL H deletes the last character entered. Processing begins only after carriage return < CR> has been entered.

Many primitive commands can be altered using the options field. This provides several extensions to the primitive commands.

Some of these commands are set/reset pairs. Instead of having separate primitive commands, NO is added as the first two characters of the command. For example, the set breakpoint command is BR, while the remove breakpoint command is NOBR.

The command format is:



The source or destination for several commands can be redirected to different I/O devices by specifying the logical unit number of the desired device.

Valid destinators for devices supported by the monitor are:

Port 1	-	User Terminal (default)
Port 2	-	Host
Port 3	-	Printer (optional)

The printer output destinator is supported, the PIT is pre-set by the firmware as a Centronics type printer interface.

The user must initialise the PIT within the application program; if not specified, the standard output is to Port 1, the terminal.

Monitor Commands

Monitor commands are presented in alphabetical order, examples are used to highlight possible applications.

COMMAND	DESCRIPTION
BF < address1> < address2> < data>	Block Fill memory - from add1 through add2 with data
BM < address1> < address2> < address3>	Block Move - move from add1 through add2 to add3
BR[< address> [< ;count>] ....]< CR>	Set/display Breakpoint
BS < address1> < address2> < data> < CR>	Block Search - search add1 through add2 for data
BT < address1> < address2> < CR>	Block Test of memory
DC < expression> < CR>	Data Conversion
DF < CR>	Display Formatted registers
DU[n] < address1> < address2> [< string>]< CR>	Dump memory to object file
GO[< address>] < CR>	Execute program
GD[< address>] < CR>	Go Direct
GT < address> < CR>	Exec prog: temp. breakpoint
HE < CR>	Display monitor commands
LO [n] [;< options>] < CR>	Load Object file
MD [n] < address> [< count>] < CR>	Memory Display
MM [n]< address> [< data>][;< options>] < CR>	Modify Memory
MS < address> < data1> < data2.> < .CR>	Memory Set-starting at addr with datal, data2, ..
NOBR [< address>] < CR>	Remove breakpoint
NOPA < CR>	Detach printer
OF < CR>	Offset
PA < CR>	Printer Attach
PF [n] < CR>	Set/display Port Format
TM [< exit character>] < CR>	Transparent Mode
TR [< count>] < CR>	Trace
TT < address> < CR>	Trace: Temp. breakpoint
VE [n] [< = string>] < CR>	Verify memory/object file
.A0 - .A7[< expression>] < CR>	Display/set address register
.D0 - .D7[< expression>] < CR>	Display/set data register
.R0 - .R7[< expression>] < CR>	Display/set offset register
.PC[< expression>] < CR>	Display/set program counter
.SR[< expression>] < CR>	Display/set status register
.SS[< expression>] < CR>	Display/set supervisor stack
.US[< expression>] < CR>	Display/set user stack
MD [n]< address> [< count>];DI < CR>	Disassemble memory location
MM [n]< address> ;DI < CR>	Disassemble/Assemble memory location
* string < CR>	Send message via Host ACIA

NAME: **BLOCK FILL MEMORY** **BF**

COMMAND SYNTAX: BF < address1> < address2> < data> < CR>

FUNCTION: The BF command is used to fill a block of memory with a specified value starting at a word boundary (even address). The data value is word sized only. The memory block begins at < address1> and ends at < address2> .

EXAMPLE:
```
> BF 1000 1100 AA55
PHYSICAL ADDRESS= 00001000 00001100
```

NAME: **BLOCK MOVE** **BM**

COMMAND SYNTAX: BM < address1> < address2> < address3> < CR>

FUNCTION: The BM command is used to move (duplicate) a block of memory from one area to another.

address1 = start address of the source block

address2 = end address of the source block

address3 = start address of the destination memory block.

EXAMPLE:
```
> BM 4000 4100 1000
PHYSICAL ADDRESS= 00004000  00004100
PHYSICAL ADDRESS= 00001000
>
```

NAME: **SET/DISPLAY BREAKPOINT BR**

COMMAND SYNTAX: BR[< address> [; < count>] ....] < CR>

FUNCTION: The BR command enables portions of a user program to be executed in real-time and halted if a user-specified instruction address (breakpoint) is encountered. The breakpoint, specified within the BR command, is placed into a breakpoint table maintained by the Monitor (unless the table is whenever the Monitor gives control to the target program (ie with the GO, GT, TR and TT commands). When a breakpoint is encountered, the display format is displayed on the terminal

Up to eight breakpoints may be set at any one time.

The count, if specified, is the number of times the address is to be encountered before the program is halted. The default value is one (1) (ie the program is halted as soon as the address is encountered). The range of count is a 32 bit integer. If no address is specified, the addresses of all breakpoints in the table are displayed on the terminal. The illegal instruction $4AFB is used for the breakpoint. If Monitor control is lost and the RESET push-button is used to regain control, breakpoints may be left in the user target program.

See also: NOBR

EXAMPLE

```
> BR 1000 1100 1200;5

BREAKPOINTS
001000          001000
001100          001100
001200          001200;5

> NOBR 1000 1100

BREAKPOINTS
001200          001200;5

> NOBR

BREAKPOINTS

>
```

NAME: **BLOCK SEARCH** **BS**

COMMAND SYNTAX: BS < add1> < addr2> < data> [< mask>][< ;option>] < CR>
BS address1 address2 'string' CR

FUNCTION: The BS command is used to search a memory block for a specified data pattern. The data may be either a literal (ASCII) string or a number. Occurrences are indicated by displaying their addresses on the terminal.

The mask, if specified, is logically ANDed with the data. If the search finds data that matches after the mask is ANDed, the data from memory before applying the AND mask is displayed.

The following data pattern size options are supported:

;B data is byte-sized (default form)

;W data is word-sized (two bytes)

;L data is long-word-sized (four bytes).

EXAMPLE:

```
> MS 1100 'FIND ME'

> MD 1100
001100 46 49 4E 44 20 4D 45 FF .....

> BS 1000 1200 'FIND ME'
PHYSICAL ADDRESS= 000010000 00001200
001100 'FIND ME'

> BS 1000 1200 46494E44;L
PHYSICAL ADDRESS= 00001000 00001200
001100 46494E44

> BS 1000 1200 44
PHYSICAL ADDRESS= 00001000 00001200
0010A7 44
0010AF 44
0010C7 44
001103 44
001155 44

>
```

NAME: **BLOCK TEST OF MEMORY BT**

COMMAND SYNTAX: BT < address1> < address2> < CR>

FUNCTION: The BT command is a destructive test of a block of memory beginning at < address1> and ending at < address2> . If this test runs without detecting errors, the memory tested will be set to zeros.

If a fault is found, a message is displayed on the terminal indicating the address, the data stored, and the data read from the failing memory.

EXAMPLE:

```
> BT 700 1FFE
PHYSICAL ADDRESS= 00000700 00001FFE

> BT 3000 4000
FAILED AT 3100 WROTE= FFFE READ= FFF0

>
```

NAME: **DATA CONVERSION DC**

COMMAND SYNTAX: DC < expression> < CR>

FUNCTION: A specified expression is converted into hexadecimal and decimal and displayed on the terminal. Expressions may be entered in hexadecimal, decimal or mixed format. The display will show both types. Values input may take one of two forms:

& number	decimal
$ number	hexadecimal (default form)

EXAMPLE:

```
> DC & 120
$78= &120

> DC &15+ $4-$13
$0= &0

> DC -1000
$FFFFF000= -$1000= &4096
```

NAME: **DISPLAY FORMATTED REGISTERS DF**

COMMAND SYNTAX: DF < CR>

FUNCTION: The DF command displays the CPU registers of the target program. These registers are also displayed whenever the monitor regains program control (ie at breakpoints and when tracing). The instruction pointed to by the program counter is also disassembled.

EXAMPLE:
```
> DF
PC= 004522      SR= 2700= .S7.....US= FFEFDE77 SS= 00000654

D0= 0030064   D1= 00000045  D2= 0000067C  D3= 0000000
D4= 00005000  D5= 00000000  D6= 00000008  D7= 00000041
A0= 0000FFA1  A1= 00005032  A2= 00000454  A3= 00000428
A4= 00000444  A5= 0000058E  A6= 0000058E  A7= 00000654
----------------------04522             2200            MOVE.L D0,D1

>
```

NAME: **DUMP MEMORY DU**

COMMAND SYNTAX: DU[n] < address1> < address2> (< string>) < CR>

FUNCTION: The DUMP command enables memory data to be output to an external device in the form of an ASCII-converted-binary (S-records) object file. This allows the user to save memory-resident programs for subsequent re-loading (using the LO command). string , if specified, is placed into a S0 header record.

S-records are a standard format widely used for the transmission of data. The S-record format is described in detail later in this chapter .

n = 1 Terminal (default)
n = 2 Host
n = 3 Printer (optional)

See also: LO, VE

EXAMPLE:
```
> DU 4000 4010 MONITOR START
PHYSICAL ADDRESS= 00004000 00004010
S01000004DF4E49544F5220535441525419
S113400000000067C00004C5E41F8000843FA06F606
S1044010208B
S9030000FC

> DU2 1000 1200 PROG1

>
```

NAME: **GO EXECUTE PROGRAM** **GO**
G

COMMAND SYNTAX: G[< address>] < CR>
GO[< address>] < CR>

FUNCTION: The GO command is used to execute a program, beginning execution at a specified address. If no address is specified, execution begins at the address contained in the pseudo PC.

The program will execute until:

1. a break point is encountered
2. the user presses either RESET or ABORT
3. an abnormal condition causes exception processing (BUSERROR etc)

Note: If breakpoints with counts are in operation the target program will not run in real time.

See also: BR, TR, TT

EXAMPLE: Refer to command GT

NAME: **GO DIRECT TO EXECUTE PROGRAM** **GD**

COMMAND SYNTAX: GD[< address>] < CR>

FUNCTION: This is the same as the GO command except that breakpoints are not placed into memory. It starts the target program without changing any of the CPU exception vectors. ($0 - $3FF)

Note: If breakpoints with counts are in operation the target program will not run in real-time

See also: BR, GO, GT, TR, TT

EXAMPLE: Refer to command GT

NAME: GO UNTIL BREAKPOINT GT

COMMAND SYNTAX: GT< address> < CR>

FUNCTION: Set a temporary breakpoint at the specified address, begin execution at the address contained in the pseudo PC. The temporary breakpoint is reset when any breakpoint is encountered.

EXAMPLE:

```
> MS 1000 4E71 4E71 4E71 4E71 4E71 4E71

> PC 1000
> BR 100A

BREAKPOINTS
00100A          00100A

> GO
PHYSICAL ADDRESS= 00001000

AT BREAKPOINT:
PC= 00100A      SR= 2700= .S7.....US= FFEFDE77 SS= 00000654
D0= 00300643    D1= 00000045    D2= 0000067C    D3= 00000000
D4= 00005000    D5= 00000000    D6= 00000008    D7= 00000041
A0= 0000FFA1    A1= 00005032    A2= 00000454    A3= 00000428
A4= 00000444    A5= 0000058E    A6= 0000058E    A7= 00000654
----------------------00100A            4E71            NOP

> GO 1000
PHYSICAL ADDRESS= 00001000
AT BREAKPOINT
PC= 00100A      SR= 2700= .S7..... US= FFEFDE77 SS= 00000654
D0= 00300643    D1= 00000045    D2= 0000067C    D3= 00000000
D4= 00005000    D5= 00000000    D6= 00000008    D7= 00000041
A0= 0000FFA1    A1= 00005032    A2= 00000454    A3= 00000428
A4= 00000444    A5= 0000058E    A6= 0000058E    A7= 00000654
----------------------00100A            4E71            NOP

> GD 1004
PHYSICAL ADDRESS= 00001000

> .PC 1000

> GT 1004
PHYSICAL ADDRESS= 00001004
PHYSICAL ADDRESS= 00001000
PC= 001004      SR= 2700= .S7..... US= FFEFDE77 SS= 00000654
D0= 00300643    D1= 00000045    D2= 0000067C    D3= 00000000
D4= 00005000    D5= 00000000    D6= 00000008    D7= 00000041
A0= 0000FFA1    A1= 00005032    A2= 00000454    A3= 00000428
A4= 00000444    A5= 0000058E    A6= 0000058E    A7= 00000654
----------------------001004            4E71            NOP
```

NAME: HELP HE

COMMAND SYNTAX: HE < CR>

FUNCTION: The HELP command displays the monitor command names on the terminal

EXAMPLE:

```
> HE
.PC   .SR     .US   .SS
.D0   .D1     .D2   .D3     .D4   .A5     .D6   .A7
.A0   .A1     .A2   .A3     .A4   .A5     .D6   .A7
.R0   .R1     .R2   .R3     .R4   .R5     .R6

BF    BM      BR    NOBR   BS    BT      DC    DF
DU    C       GD    GO     GT    HE      LO    M
MD    MM      MS    OF     PA    NOPA    PF    T
TM    TR      TT    VE

>
```

NAME: **LOAD OBJECT FILE** **LO**

COMMAND SYNTAX: LO[n][; < OPTION>] < CR>

FUNCTION: The LO command enables an ASCII-converted-binary (S-records) object file to be taken from an external device and loaded into memory. Each byte of data is verified as it is loaded and if not stored correctly, an error message is displayed.

n = 1 Terminal
n = 2 Host

Options:

;C Abort the operation if a checksum error is detected. Default:+ C.

;C Ignore the checksum while loading.

;X Display the object file being loaded from port 2 on the terminal. Default: -X.

= < string> < string> is output before the load operation commences. This is intended to instruct an external device to down-load the file. If used it must be the last option specified.

Any offset stored in RO will be added to the destination address of the S-record

Note: No attempt is made to control the host transmission. If a load error occurs, an error message is displayed on the terminal. During this time, object records from the host system are ignored.

See also: DU, OF, VE.

EXAMPLE:

```
> LO ;= COPY FILE27.MX#
COPY FILE27.MX,#

> LO2 ;= DU2 1000 1200

>
```

NAME: **MEMORY DISPLAY** **MD**

COMMAND SYNTAX: MD [n] < address> [< count>] < CR>

FUNCTION: The MD command is used to display a section of memory beginning at address. The section is count bytes long, if count is not specified, it takes the value 16.

Once entered the MD command will continue with the next 16 lines of output each time a carriage return is entered. Any other command terminates this feature.

The command can be redirected to the following devices:

MD	Port 1	Terminal (default)
MD1	Port 1	Terminal
MD2	Port 2	Host
MD3	Port 3	Printer (optional)

See also: MM, MS

EXAMPLE:
```
> MD 4000 12
004000 00  00  06  7C 00  00  4C 5E 41  F8 00  08  43  FA 06   F6
004010 20  C9 B1  FC 00  00  04  00  6B F6 41  FA 0E  12  43  F8

>
```
Option ..;DI invokes the disassembler function (see command MD ...;DI and Chapter 5 respectively)

```
> MD 200 20;DI

002000      0C00020     CMP.B     # 32,D0
002004      6D28        BLT.S     $ 202E
002006      0C00030     CMP.B     # 48,D0

>
```

NAME **MEMORY MODIFY** **MM**
M

COMMAND SYNTAX: MM[n] < address> [; < options>] < CR>
M [n] < address> [; < options>] < CR>

FUNCTION: The MM command is used to display and change memory. If a data field is specified, memory locations beginning with address are set equal to the value of the data. If a data field is not specified, address and its contents are displayed on the terminal. In either case a sub-command set is entered, and operates as follows:

< address> < present data> ?[< new data>] < CR>	If new data is specified, place in address ; display the address and contents of the next location
< address> < present data> ?[< new data>] < CR>	If new data is specified, place in address ; display the address and contents of the previous location
< address> < present data> ?[< new data>] < CR>	If new data is specified, put it in address ; display address and its new contents
< address> < present data> ?[< new data>].< CR>	If new data is specified, put it in address ; then leave the sub-command mode

Address and present data are displayed by the monitor.
The remainder of the line is input by the user.

MEMORY MODIFY contd.

The following options are supported:

Default data is byte-sized

;W	data is word-sized (two bytes)
;L	data is long word-sized (four bytes)
;0	data is byte-sized; access odd addressed bytes only
;V	data is byte-sized; access even addressed bytes only
;N	do not verify that the data stored correctly. This option is useful when writing data to non-RAM locations such as hardware registers.

See also : MD, MS

Several modes allow modification and verification of data:

[< data>] < CR> update location and go forward
[< data>] < CR> update location and go backward
[< data> =] < CR> update location and do not advance
[< data> .] < CR> update location and terminate

EXAMPLE:

```
> MM 1000;L
001000      12003400    ?200=
001000      00000200    ?
001004      56345678    ?FFFFFFFF
001008      000000FF    ?
001004      FFFFFFFF    ?
001008      000000FF    ?.

> MM 4000;N
004000      ?1234
004001      ?5678
004002      ?.

> MM 1000;V
001000 00   ?12
001002 02   ?34
001004 FF   ?56.
```

NAME: **MEMORY SET** **MS**

COMMAND SYNTAX: MS < address> < data> < CR>

FUNCTION: The MS command is used to alter memory by setting data at the address specified. The data may be a literal (ASCII) string enclosed in apostrophes, or a number with a value no greater than a 32 bit integer.

See also: MM, MD

EXAMPLE:

```
> MS 700 'ABC'

> MS 703 4445

> MS 705 46474849 4A (can be up to 8 characters)

> MD 700
000700 41 42 43 44 45 46 47 48 49 4A .....

>
```

NAME: **REMOVE BREAKPOINT** **NOBR**

COMMAND SYNTAX: NOBR[< address> ....] < CR>

FUNCTION: The NOBR command removes one or more addresses from the breakpoint table; it functions as the reverse of the BR command. The address, if specified, is removed from the table. If no address is specified, all addresses are removed from the table.

See also: BR

EXAMPLE:

```
> BR 1000;5 2000 3000;6 10000

BREAKPOINTS
001000      001000;5
002000      002000
003000      003000;6
010000      010000

> NOBR 10000 1000

BREAKPOINTS
002000      002000
003000      003000;6

> NOBR
```

NAME: **DETACH PRINTER** **NOPA**

COMMAND SYNTAX: NOPA < CR>

FUNCTION: The NOPA command is used to inhibit terminal information being 'echoed' on the printer. It functions as the reverse of the PA command.

See also: PA

NAME: **OFFSET** **OF**

COMMAND SYNTAX: OF < CR>

FUNCTION: The OF command displays the offsets used to assist with program relocatability and position independent code.

Linked segments of target program code each have different load addresses. For user convenience, seven general purpose offsets, .R0 to .R6, are provided. .R7 is always zero to provide a convenient means of entering an address without an offset.

Unless another offset is explicitly used, each monitor command that expects an address parameter automatically adds offset .R0 to the user entered address. For example, the following commands are equivalent:

BR 1000
BR 1000+ R0

EXAMPLE:
```
> .R1 1000

> .R3 3300

> .R4 440000

> .R5 0

> .R6 -1

> OF
R0= 00000000   R1= 00001000   R2= 00000000   R3= 00003300
R4= 00440000   R5= 00000000   R6= FFFFFFFF   R7= 00000000

>
```

NAME: **ATTACH PRINTER** **PA**

COMMAND SYNTAX: PA < CR>

FUNCTION: The PA command enables terminal information to be 'echoed' to the printer: any data displayed on the terminal will also be output to the printer.

A PRINTER NOT READY message will be output if the printer is not on line or no printer is present.

See also: NOPA

NAME: **PORT FORMAT** **PF**

COMMAND SYNTAX: PF[n] < CR>

FUNCTION: The PF command allows the user to display and/or change the characteristics of the serial ports. The number of stop bits and the character NULL padding may be altered.

EXAMPLE:

COMMAND FORMAT	DESCRIPTION
> PF	Display port format information (Port 1 in column 1; Port 2 col 2.)

```
FORMAT=  15 15
CHAR     NULL= 00 00
C/R      NULL= 00 00
OPTIONS@000 4E61
```

> PF2	Change the format of Port 2
FORMAT= 15?11	Alter the number of stop bits
CHAR NULL= 00?	CR
C/R NULL= 00?3	Change the CR NULL padding

```
>
```

Note: The Baud rate for Port 1 and 2 are jumper selectable and default set to 9600 baud.

Format = 15 causes 1 stop bit (default)
Format = 11 causes 2 stop bits

NAME: **TRANSPARENT MODE** **TM**

COMMAND SYNTAX: TM[< exit character>] < CR>

FUNCTION: The TM command is used to enter the transparent mode in which Port 1 and 2 are directly connected. In this mode, the terminal communicates directly with the host.

Note: Both ports must be set to the same baud rate.

The transparent mode is terminated by the receipt of the exit character. If it is not specified in the TM command, the exit character defaults to CTL-A (ASCII $01).

When input, the exit character is sent to the host computer.

EXAMPLE: > TM

EXIT CHAR = $01 = CTL A

> TM

EXIT CHAR = $7C =

>

NAME: **TRACE** **TR**
T

COMMAND SYNTAX: TR[< count>] < CR>
T[< count>] < CR>

FUNCTION: The TR command executes count number of instructions beginning at the address contained in the pseudo PC. After each instruction is executed, the formatted registers are displayed on the terminal. If count is not specified, it assumes the value of 1.

When the trace mode is entered, the prompt changes to include a colon (:). In this mode, entering a carriage return causes one user program instruction to be traced. Entering another command terminates the trace mode.

Breakpoints and breakpoint counts are in effect during trace.

Trace cannot be used to step through interrupts or exceptions: (TRAP, etc)

See also : DF, GO, GT, TT

EXAMPLE: > .PC 1000

```
> T 2
PHYSICAL ADDRESS= 00000000
PC= 001002      R= 2700= .S7.....US= FFEFDE77 SS= 0000067C
D0= 00300643    D1= 00000045    D2= 0000067C    D3= 00000000
D4= 00005000    D5= 00000000    D6= 00000008    D7= 00000041
A0= 0000FFA1    A1= 00005032    A2= 00000428    D3= 00000428
A4= 00000444    A5= 0000058E    A6= 00000454    A7= 0000067C
----------------------001002 67FE BEQ.S $001000

PC= 001004      SR= 2700= .S7.....US= FFEFDE77 SS= 0000067C
D0= 00300643    D1= 00000045    D2= 0000067C    D3= 00000000
D4= 00005000    D5= 00000000    D6= 00000008    D7= 00000041
A0= 0000FFA1 A1= 00005032       A2= 00000454    A3= 00000428
A4= 00000444    A5= 0000058E    A6= 0000058E    A7= 0000067C
--------------------001004 2200 MOVE.L D0,D1

> TR
PHYSICAL ADDRESS= 00001008
PC= 001006 SR= 2700= .S7.....US= FFEFDE77 SS= 0000067C
D0= 00300643    D1= 00000045    D2= 0000067C    D3= 00000000
D4= 00005000    D5= 00000000    D6= 00000008    D7= 00000041
A0= 000FFA1     A1= 00005032    A2= 00000454    A3= 00000428
A4= 0000444     A5= 0000058E    A6= 0000058E    A7= 0000067C
--------------------001006 2A07 MOVE.L D7,D5
```

NAME: **TRACE TO TEMPORARY BREAKPOINT TT**

COMMAND SYNTAX: TT < address> < CR>

FUNCTION: The TT command executes program instructions beginning at the address contained in the pseudo PC, until address is reached or a breakpoint is encountered.

EXAMPLE: > .PC 1000

```
> TT 1008
PHYSICAL ADDRESS= 00001008
PHYSICAL ADDRESS= 00001000
AT BREAKPOINT
PC= 001008      SR= 2700= .S7.....US= FFEFDE77 SS= 0000067C
D0= 00300643    D1= 0000045     D2= 0000067C    D3= 00000000
D4= 00005000    D5= 00000000    D6= 00000008    D7= 00000041
A0= 0000FFA1    A1= 00005032    A2= 00000454    A3= 00000428
A4= 00000444    A5= 0000058E    A6= 0000058E    A7= 0000067C
----------------001008 0C0020 CMP.B # 32,D0

>
```

NAME: **VERIFY** **VE**

COMMAND SYNTAX: [VE[n] [= < string>] < CR>

FUNCTION: The VE command enables the contents of memory to be compared with an ASCII-converted-binary object file read from an external device. Each byte of data from the file is compared with the content of the memory at the location into which it would be loaded. If a mis-comparison is detected, an error message is displayed on the terminal.

< string> , if specified, is output before the verification operation commences, it is intended as a means of instructing an external device to download the file.

n = 1 Terminal
n = 2 Host

Note: No attempt is made to control the host transmission. If a verification error occurs, an error message will be displayed on the terminal. During this time, object records from the host system are ignored.

See also: LO, DU

EXAMPLE:

```
> VE2;= COPY FILE27.MX,#

> VE2;= DU2 1000 1200

> VE2
S113100-.-.-.-20-.-.-.-.
>
```

NAME: **DISPLAY/SET REGISTER** **.A0 - .A7**
.D0 - .D7
.R0 - .R6
.PC
.SR
.SS
.US

COMMAND SYNTAX: .< register> [< expression>]

FUNCTION: Individual registers can be displayed and/or altered. Commands with a leading period, and the registers displayed/altered by these commands are:

.A0 - .A7	address register
.D0 - .D7	data register
.R0 - .R6	relative offset register (software register)
.PC	program counter
.SR	status register (in the 68000)
.SS	supervisor stack pointer
.US	user stack pointer

EXAMPLE:

> .PC .PC= 00001010	Display program counter
> .A7 1300	Set address register 7
> .R5 5500	Set relative offset register 5

NAME: **MEMORY DISPLAY DISASSEMBLER** **MD ...;DI**

COMMAND SYNTAX: MD[n] address [counts];DI CR

FUNCTION: The disassembler is called as an option to the Memory Display command. When the suffix option DI is used with the command the content of the memory location specified will be disassembled.

EXAMPLE:

```
> MD 4000;DI
004000      384F        MOVE.W    A7,A4

> MD 4000 10;DI
004000      384F        MOVE.W    A7,A4
004002      4DF80520    LEA.L     $0520,A6
004006      B0F84580    CMP.W     $00004580,A0
00400A      6674        BNE.S     $00408
00400C      600001F2    BRA.L     $004200

>
```

NAME: MEMORY MODIFY DISASSEMBLER/ASSEMBLER **MM...;DI**
M ...;DI

COMMAND SYNTAX: MM address ;DI < CR>

FUNCTION: The Disassembler/Assembler is called as an option to the Memory Modify command. When the suffix DI is used the content of the location specified will be disassembled and a ? printed after the disassembled line. A new instruction can be entered in assembly language. A leading space must precede the instruction. Labels are not recognised, only explicit values relative to the program counter.

The assembler will assemble and print the new line. If there is an error in the assembly instruction entered, a ? will be printed and the entire line must be re-entered.

To insert line(s), move a portion of the existing code using the BM command and enter the new line(s).

EXAMPLE: BF 4000 4100 FFFF
PHYSICAL ADDRESS= 00004000 00004100

```
> MM 4000;DI
004000      FFFF        DC.W      $FFFF ? MOVEA A7,A4
004000      384F        MOVEA     A7, A4
004002      FFFF        DC.W      $FFFF ? LEA $520,A6
004002      4DF80520    LEA       $520,A6
004006      FFFF        DC.W      $FFFF ? CMPA $4580,A0
004006      B0F84580    CMPA      $4580,A0
00400A      FFFF        DC.W      $FFFF ? BNE.S $4080
00400A      6674        BNE.S     $4080
00400C      FFFF        DC.W      $FFFF ? BRA.S $4200.

>
```

Using the Assembler/Disassembler

Introduction

Included in the PME68/Monitor firmware is an assembler/disassembler. This is a dynamic assembler/editor in which the source program is not saved. Each source line is translated into the proper 68000 machine language code and is stored in memory on a line-by-line basis at the time of entry. To display an instruction, the machine code is disassembled and the instruction mnemonic and operands are displayed. All valid 68000 instructions are translated.

The assembler does not allow line numbers and labels; it processes each line of program as an individual unit. However, it is a powerful tool for creating, modifying, and debugging 68000 code.

The limitations of one-line assemblers are:

(a) Label and line numbers are not allowed. Labels are commonly used to reference other lines and locations in a program. The one-line assembler has no knowledge of other program lines and therefore cannot make the required association between a label and a label definition located on a separate line.

(b) Source lines are not saved. In order to read back a program after it has been entered, the machine code is disassembled and then displayed as mnemonic and operands.

(c) Limited error indication. The one-line assembler will show a question mark (?) under the portion of the source statement where an error probably occurred, or will display the word "ERROR" or another message.

(d) Only one directive (DC.W) is accepted.

(e) No macro handling capability is included.

(f) No conditional assembly can be used.

68000 Assembly Language

The symbolic language used to code source programs for processing by the assembler is called "68000 Assembly Language". This is a collection of mnemonics representing:

(a) Operations - 68000 machine instruction operation codes
- Directive (pseudo-op)

(b) Operators

(c) Special symbols

Directives

The assembly language can contain mnemonic directives which specify auxiliary actions to be performed by the assembler. Directives are not always translated into machine language.

Assembler directives assist the programmer to:

(a) control the assembler output

(b) define data and symbols

(c) allocate storage.

The assembler recognises only one directive, define constant (DC.W). This is used to define data within a program.

Source Program Coding

A source program is a sequence of statements arranged in a logical way to perform a predetermined task. Each source statement occupies a line and must be either an executable instruction or a DC.W directive. Each statement follows a consistent source line format (see also commands MD...;DI and MM...;DI).

For further details on source program coding and on the 68000 instruction set see the 68000 Users Manual (Available from your supplier).

A description on the assembler language is given in the following publication:

68000 Assembler Language Programming
Osborne/McGraw Hill - ISBN 0-931988-62-4

Disassembled Source Line

The disassembled source line may not look identical to the source line entered as the disassembler makes a decision on how to represent a numerical value based on how it interprets the number's use. If the number is determined to be an address or a "would-be" address, it is displayed in hexadecimal; everything else is decimal. For example:

```
MOVE.L    # $1234, $5678
```

disassembles to

```
MOVE.L    # 4660, $00005678
```

For some instructions there are two valid mnemonics for the same op-code, or there is more than one assembly language equivalent. The disassembler may choose a form different from the one originally entered. For example:

BRA is returned for BT
DBF is returned for DBRA

The assembler recognises two forms of mnemonic for two branch instructions. The BT form (branch conditional always true) has the same op-code as the BRA instruction. DBRA (decrement and branch always) and DBF (never true, decrement, and branch) mnemonics are different forms for the same instruction. The assembler will accept both forms.

Mnemonics and Delimiters

The assembler recognises MC68000 instruction mnemonics. Numbers are recognised as both decimal and hexadecimal with decimal being the default (Note, this is the reverse of the monitor commands).

One or more ASCII characters enclosed by apostrophes (') constitute an ASCII string. ASCII strings are left justified and zero filled (if necessary), whether stored or used as immediate operands. Left justification will be to a word boundary if one or two characters are specified; or to a long word boundary if the string contains more than two characters:

```
DC.W      'S'
MOVE.L    # 'ABCD',D1
DC.W      '56'
```

The following register mnemonics are recognised by the assembler:

D0-D7	Data Registers
A0-A7	Address Registers
A7, SP	Either mnemonic represents the system stack pointer of the active system state.
USP	User Stack Pointer. Used only in privileged instructions which are restricted to the supervisory state.
CCR	Condition Code Register (lower 8 bits of SR)
SR	Status Register. All 16 bits may be modified in the supervisor state. Only the lower 8 bits (CCR) may be modified in the user state.
PC	Program Counter. Used only in program counter relative addressing.

Character Set

The character set recognised by the assembler is a subset of ASCII, and is listed below:

(a) Uppercase letters A through Z

(b) Integers 0 through 9

(c) Arithmetic operators: + -

(d) Parentheses ( )

(e) Characters used as special prefixes:

£ (pound sign) or # specifies the immediate form of addressing, $ (dollar sign) specifies a hexadecimal number, & (ampersand) specifies a decimal number, ' (apostrophe) specifies an ASCII literal character

(f) Five separating characters: Space
, (Comma)
. (period)
/ (slash)
- (dash)

(g) The * (asterisk) indicates current location.

Source Code Format

This section describes the functions performed by assemblers in comparison with the ROM resident assembler/disassembler.

Assembler Language Format

Assembler language instructions are separated into a number of fields; delimiters or separators for these fields are commonly spaces, colons, and carriage returns. The standard assembler source line format is as follows, optional items are enclosed in brackets:

[LABEL:] < sp> < OPERATION> < [OPERAND]> < [COMMENT]> < CR>

LABEL:	optional
OPERATION:	68000 opcode or directive
OPERAND:	One or two operands, addresses
COMMENT:	optional

The dynamic assembler does not allow labels and comments. The specific line format looks similar to:

< sp> OPERATION [OPERAND] < CR>

The first < sp> is to be compatible with structured assemblers. A space must be the first character of each line.

Assembler Operation Codes

An operation can be either a valid 68000 instruction or a directive. The only directive recognised by the assembler is DC.W.

The size of the affected data field may be specified by the data size code. If not explicitly specified, a default size is taken.

BYTE	.B	8 bit
WORD	.W	16 bit (default)
LONG	.L	32 bit

Several instructions do not have or do not request a data size specification.

Format: OP [.SIZE]

Valid operations are:

MOVE	.B	move byte
ADD		add word
NOP		no operation

Invalid operations are:

RTS.B	return from subroutine, illegal size spec.
ORG	directive not supported

Operands and Addresses

Format: < sp> [OP1[,OP2]]

The operand field follows the operation field, separated by a space. Zero, one or two operands are possible depending on the instruction. The second operand is separated by a comma as a delimiter.

In this case: OP1 is source
OP2 is destination

DC.W Define Constant Directive

Format: DC.W < operand>

The function of this directive is to define a constant in memory. The DC.W directive can have only one, 16 bit, operand which can contain the actual value (decimal, hexadecimal, or ASCII). Alternatively, the operand may be an expression which can be assigned a numeric value by the assembler. The constant is aligned on a word boundary as specified by word (.W) size.

An ASCII string is recognised when characters are enclosed in single quotes (’). Each character (7 bits) is assigned to a byte of memory, with the eighth bit (MSB) always equal to zero. If one byte is entered, the byte is left justified.

Examples are:

```
DC.W    1234 Decimal number
DC.W    $AAFE Hexadecimal number
DC.W    ’AB’ ASCII string
DC.W    ’TB’+ 1 Expression
```

Entering and Modifying Source Programs

Application programs are entered into the PME 68-1B RAM area using the one-line assembler/disassembler. It is entered in assembly language statements on a line-by-line base. The source code is not saved as it is immediately converted to machine code. This imposes several restrictions on the type of source line that can be entered. Labels and symbols are not allowed; absolute values or addresses or PC relative offsets have to be used.

Editing is accomplished by retyping the new source line. Lines can be added or deleted by moving a block of memory data to free up or delete the appropriate number of locations.

Calling the Assembler/Disassembler

The command MM< address> ; DI transfers control to the resident assembler/ disassembler. The contents of the specified locations are immediately disassembled if they contain a valid instruction, otherwise a DC.W operand is displayed. The ? prompts for a new input.

Example:

```
> MM 2000; DI < CR>
002000 0C000020 CMP.B # 32,D0 ?
```

If nothing is entered but < CR> , the next instruction is displayed.

Program Input

A new instruction may be entered after the ? prompt. If it is a valid 68000 code, the old line is overwritten by the new entry after the < CR> .

```
> MM 1000;DI
001000 FFFF  DC.W $FFF ? MOVE.B # 1,D0 < CR>
```

The next instruction is also displayed.

```
> MM 1000;DI
001000 103C0001  MOVE.B # 1,D0
001004 FFFF  DC.W $FFF ?
```

This procedure is continued until the end of the program. A full-stop '.' terminates the program input phase.

Error Detection

If an invalid command or statement is entered, the assembler returns to the old line and displays an '?' prompt to request a new input. For severe errors an error message is displayed and a return to the monitor prompt is performed.

Disassembled Program Listings

A listing of the program is obtained using the Memory Display (MD) command with the ;DI option. When ;DI is invoked, the number of instructions disassembled and displayed will be equal to the number of instructions whose op-code (first word of any instruction) is contained within the byte count. The DC.W directive will also be displayed for any words of data contained within the byte count.

The output of this command can be redirected to any of the I/O devices supported by the firmware:

MD1	Terminal (default)
MD2	Host
MD3	Printer (user configurable)

Disassembled Program Listings contd.

Example:

> MD2 1000 10;DI < CR> disassembled output to Host
Remember, a disassembled source line may not look identical to the entered statement or instruction, for example this will occur if:

(a) the default data format for constants is decimal; hex constants are returned in decimal notation.

(b) various instruction types are not always separated; CMP and CMPI

(c) PC relative operations are indicated as absolute address references; BRA.S as BRA.S $2000.

Saving Programs

If, after a program has been created and tested, a permanent copy is desired for documentation purposes, or to avoid re-entry the next time it is to be executed, it can be saved. The methods available will depend upon the system hardware.

Upload to a Host

Programs can be saved by uploading to a host computer via Port 2 of the PME 68-1B.

Once a program has been sent, it can be saved on the hosts mass storage media. Uploading to a host requires a program in the host to input the S-records from the RS232C port and save them either in RAM or directly onto the mass storage media (see para. 6.15 for S-record format).

Uploading to a host is another method of saving programs and it also uses the DUMP command. A file is usually uploaded through Port 2. In order to upload successfully, the host must contain a program to input S-records from the RS-232C port and save them. The PME 68-1B CPU board can upload to any host which has an RS-232C port and the required program.

Download from a Host

Files must be in S-record format. As well as retrieval of programs created using the resident assembler/disassembler, the LOAD command is a handy tool for loading 68000 language programs created using the hosts resident or cross assembler. Such assemblers currently exist for many potential host computers. This can be achieved by performing the following sequence:

Down-load From a Host contd.

LO2 ;= DU1 2000 3000 CR

The command after the equal sign is sent to the RADSTONE VME MDS and interpreted as command input.

The VERIFY command should be used to ensure that data has properly loaded.

- > VE ;= DU1 2000 3000 CR
- >

If any differences are found they will be listed (see Monitor Commands for details).

User Application Software Examples

Several devices on the PME 68-1B CPU are free for user applications, some simple examples are given to show how these devices may be programmed. The examples are guidelines only and may be adopted for the users particular purposes. The programs below have been written using the PME/Ideal, ROM resident screen-oriented Editor/68000 Assembler.

Data Transfer (From and To) an ACIA

The following routines are a short example of how the ACIA3 may be programmed in assembler language using a standard 68000 structured label assembler.

Output of One Line to the REMOTE ACIA

A string is transmitted via the ACIA3 until CR ($0D) is detected. TRAP # 14 performs a return to the PME68/Monitor.

```
                        ORG $2000;

            /* OUTPUT ONE LINE TO ACIA3 */

ACIA :      EQU         @$C0101         ;/*ACIA3 CSR */
RESET:      EQU         $03             ;/* RESET */
MODE :      EQU         $15             ;/* MODE  */

START0::    LEA.L       ACIA,A0         ;/* GET BASE INTO A0 */
            MOVE.B      # RESET,0(A0)   ;/* INIT ACIA */
            MOVE.B      # MODE ,0(A0);

            LEA.L       STRING,A1       :/* GET STRING */
NEXT:       BSR.B       OUTCH           ;
            CMPI.B      # $0D,(A1)+     ;
            BNE.B       NEXT            ;

            TRAP        # 14            ;/* MONITOR CALL */

OUTCH:      BTST.B      # 1,0(A0)       ;/* DONE ? */
            BEQ.B       OUTCH           ;
            MOVE.B      (A1),2(A0)      ;/* SEND BYTE */
            RTS

STRING:     DC 'PME 68-1B'              ;/* TEXT */
            DC.B        $0A             ;
            DC.B        $0D             ;;
```

Input of One Line from the Remote ACIA

A string is read from the device, < CR> ($0D) terminates the input. A return to the monitor is performed at the end of the operation by a TRAP # 14 sequence.

```
                                ORG $4000;

                          /* INPUT ONE LINE FROM ACIA3*/

ACIA :    EQU     @$C0101          ;/* ACIA3 CSR */
RESET:    EQU     $03              ;/* RESET */
MODE :    EQU     $15              ;/* MODE */

START1::  LEA.L   ACIA,A0          ;/* GET BASE INTO A0 */
          MOVE.B  # RESET,0(A0)    ;/*RESET */
          MOVE.B  # MODE ,0(A0)    ;

          LEA.L   STRNG,A1         ;
NEXT:     BSR.B   INCH             ;
          CMPI.B  # S0D,(A1)+      ;/* DO UNTIL CR      */
          BNE.B   NEXT             ;

          TRAP    # 14             ;/* MONITOR CALL */

INCH:     MOVE.B  0(A0),D0         ;/* GET STATUS */
          LSR.B   # 1,D0           ;/* DONE ? */
          BCC.B   INCH             ;
          MOVE.B  2(A0),(A1)       ;/* STORE BYTE 2*/
          RTS

STRING:   DS.B    $100             ;;/* TEXT BUFFER */
```

Initialisation of the Real-time Clock

```
                                ORG $3000;

                        /* INITIALISE REALTIME CLOCK */

RTCLOCK   :   EQU     @$C0400       ;/* MM58167A BASE ADDRESS */

COUNT_1000:   EQU     1(A0)         ;/* COUNTER REG OFFSETS */
COUNT_100 :   EQU     3(A0)         ;
COUNT_SECS:   EQU     5(A0)         ;
COUNT_MIN :   EQU     7(A0)         ;
COUNT_HOUR:   EQU     9(A0)         ;
COUNT_DAY :   EQU     11(A0)        ;
COUNT_WEEK:   EQU     13(A0)        ;
COUNT_MONT:   EQU     15(A0)        ;

COUNT_GO  :   EQU     $2B(A0)       ;/* GO COMMAND REGISTER */


HOUR      :   EQU     # 08      ;/* TIME AND DATE */
DAY       :   EQU     # 01      ;
WEEK      :   EQU     # 11      ;
MONT      :   EQU     # 02      ;

NULL      :   EQU     $00       ;
START     :   EQU     $FF       ;/* START COMMAND */

STARTRTC::    LEA.L   RTCLOCK,A0          ;/* GET BASE INTO A0 */

              MOVE.B  # NULL,COUNT_1000 ;/* PRESET COUNTERS */
              MOVE.B  # NULL,COUNT_100  :
              MOVE.B  # NULL,COUNT_SECS      ;
              MOVE.B  # NULL,COUNT_MIN ;
              MOVE.B  # HOUR,COUNT_HOUR;
              MOVE.B  # DAY ,COUNT_DAY ;
              MOVE.B  # WEEK,COUNT_WEEK;
              MOVE.B  # MONT,COUNT_MONT;

              MOVE.B  # START,COUNT_GO ;/* START CLOCK */

              TRAP    # 14                ;;/* MONITOR CALL */
```

Address Assignment of I/O Devices

DEVICE: J10 6850 ACIA (Terminal)

Address:	Mode:	Description:
0C0080	R	Status Register
0C0080	W	Control Register
0C0082	R	Receive Data Register
0C0082	W	Transmit Data Register

DEVICE: J3 6850 ACIA (Host)

Address:	Mode:	Description:
0C0041	R	Status Register
0C0041	W	Control Register
0C0043	R	Receive Data Register
0C0043	W	Transmit Data Register

DEVICE: J4 6850 ACIA (Remote)

(not supported by firmware)

Address:	Mode:	Description:
0C0101	R	Status Register
0C0101	W	Control Register
0C0103	R	Receive Data Register
0C0103	W	Transmit Data Register

DEVICE: J50 68230 PIT (Parallel Interface/Timer)

Address	Mode	Affected By Reset	Affected by 55D Read Cycle	Using
0E0001	R/W	Y	N	Port General Control Register (PGCR)
0E0003	R/W	Y	N	Port Service Request Register (PSRR)
0E0005	R/W	Y	N	Port A Data Direction Register (PADDR)
0E0007	R/W	Y	N	Port B Data Direction Register (PBDDR)
0E0009	R/W	Y	N	Port C Data Direction Register (PCDDR)
0E000B	R/W	Y	N	Port Interrupt Vector Register (PIVR)
0E000D	R/W	Y	N	Port A Control Register (PACR)
0E000F	R/W	Y	N	Port B Control Register (PBCR)
0E0011	R/W	N	**	Port A Data Register (PADR)
0E0013	R/W	N	**	Port B Data Register (PBDR)
0E0015	R	N	N	Port A Alternate Register (PAAR)
0E0017	R	N	N	Port B Alternate Register (PBAR)
0E0019	R/W	N	N	Port C Data Register (PCDR)
0E001B	R/W*	Y	N	Port Status Register (PSR)
0E0021	R/W	Y	N	Timer Control Register (TCR)
0E0023	R/W	Y	N	Timer Interrupt Vector Register (TIVR)
0E0027	R/W	Y	N	Counter Preload Register High (CPRH)
0E0029	R/W	N	N	Counter Preload Register Middle (CPRM)
0E002B	R/W	N	N	Counter Preload Register Low (CPRL)
0E002F	R	N	N	Count Register High (CNTRH)
0E0031	R	N	N	Count Register Middle (CNTRM)
0E0033	R	N	N	Count Register Low (CNTRL)
0E0035	R/W	Y	N	Timer Status Register (TSR)

* A write to this register may perform a special status reset operation.

** Mode dependent.

DEVICE: J63 58167A RTC (Real Time Clock)

(not supported by firmware)

Address	Mode	Description
0C0401	R	Counter - Ten Thousands of Seconds
0C0403	R	Counter - Hundredths and Tenths of Seconds
0C0405	R	Counter - Seconds
0C0407	R	Counter - Minutes
0C0409	R	Counter - Hours
0C040B	R	Counter - Days of Week
0C040D	R	Counter - Days of Month
0C040F	R	Counter - Months
0C0411	R/W	RAM - Ten Thousands of Seconds
0C0413	R/W	RAM - Hundredths and Tenths of Seconds
0C0415	R/W	RAM - Seconds
0C0417	R/W	RAM - Minutes
0C0419	R/W	RAM - Hours
0C041B	R/W	RAM - Days of Week
0C041D	R/W	RAM - Days of Month
0C041F	R/W	RAM - Months
0C0421	R	Interrupt Status Register
0C0423	R/W	Interrupt Control Register
0C0425	W	Counters Reset
0C0427	W	RAM - Reset
0C0429	R	Status Bit
0C042B	W	GO Command
0C042D	R/W	Standby - Interrupt
0C042F	W	Test Mode

Error Messages and Monitor Messages

Error Message	Meaning
PRINTER NOT READY	Printer is not properly connected or cannot receive output
SYNTAX ERROR	Error in command line
ERROR	Error (prefix)
ILLEGAL INSTRUCTION	Instruction used an illegal op-code
TRAP ERROR	See Traps in MC68000 User's Guide
IS NOT A HEX DIGIT	Improper character entered in a field that requires a hexadecimal digit
DATA DID NOT STORE	Data did not go where intended (such as attempting to write to ROM)
INVALID ADDRESS	Too big (1 in bits 24/32 or odd for .W or .L (1 in bit 0)
????	Program does not recognise user's entry.
NOT HEX	Same as IS NOT A HEX DIGIT
FAILED AT..WRITE= .. READ= ..	Read or write command failure output by BT

Monitor Messages	Meaning
>	PME68/Monitor prompt
HELP	HE command displays all valid commands
BRKPTS=	Displayed by BR command
FORMAT=	Displayed by PF command
CHAR NULL=	Displayed by PF command
C/R NULL=	Displayed by PF command
TRANSPARENT EXIT= $01= CTL A	Displayed by TM command
SOFTWARE ABORT	BREAK key has been used
AT BREAKPOINT	Indicates program has stopped at breakpoint
PHYSICAL ADDRESS	Actual address by command
CHECKSUM	Checksum in LO or VE command

Addresses of the Main System Routines

The addresses of system routines listed on the following pages may be used within user written application programs when correctly called.

Label	Function	Address	Remarks
FIXBUF	Initialisation of A5, A6 to Buffer	$81CA0	
FIXADD	Add data (A5) to buffer A5 = BUFFER A6	$800F6	
FIXCRLF		$80106	
OUTCH	Output Char in D0 to terminal	$81E24	D3 = 0 ; GETACIA1
INCH	Read Char into D0 from Terminal without ECHO	$81FB0	D3 = 0 ; GETACIA1
CHRPRINT	Output Char in D0 to Printer	$81D84	
OUTPUT	Output String to Terminal	$81C14	A5 = Bufbegin A6 = Bufend+ 1
OUTPUT21	Output String to Host	$81C34	A5 = Bufbegin A6 = Bufend+ 1
PORTIN1	Read String from Terminal until < CR>	$81CA8	A5= A6 Textbegin
PORTIN20	Read String from Host	$81FDE	A5= A6 Textbegin
PRCRLF	Output String to Printer	$81D70	A5= Bufbegin A6= Bufend+ 1
GETACIA1	ACIA1 Addr into A0	$81C7C	
GETACIA2	ACIA2 Addr into A0	$81C8E	
COLDSTRT	Initialisation of the Monitor	$80146	
WARMSTRT	Return to the Monitor	$80232	resp. TRAP # 14

Note: It must be remembered that not all registers are saved automatically.

Furthermore it should be noted that the use of these subroutines in an illogical sequence could affect the proper operation of the PME 68-1B CPU System Monitor.

S-Record Format

An S-record is a standard format used for transmitting and receiving programs and data. The whole transfer is performed in ASCII characters.

There are ten possible standard S-record types, but only five are used within the PME68/Monitor environment:

S0	Header record
S1	16-bit address Data record
S2	24-bit address Data record
S8	24-bit address End of File/Execution Address record
S9	16-bit address End of File/Execution Address record.

The standard S-record is defined as follows:

Frame	Value	Description
	$0D	Carriage Return
	$0A	Line Feed
	$00	Null
1	$53 (S)	Start of Record
2	$30-$39 (0-9)	Record Type
3,4		Byte Count
5-8		Address (for 16 bit)
5-10		Address (for 24 bit)
5-12		Address (for 32 bit)
:		
:		Data
:		
N-1,N		Checksum

Byte Count - is the sum of bytes including address, data and checksum.

The byte count, address, data and checksum are represented in ASCII encoded hexadecimal; i.e. two frames per data byte, with the most significant digit in the leading frame.

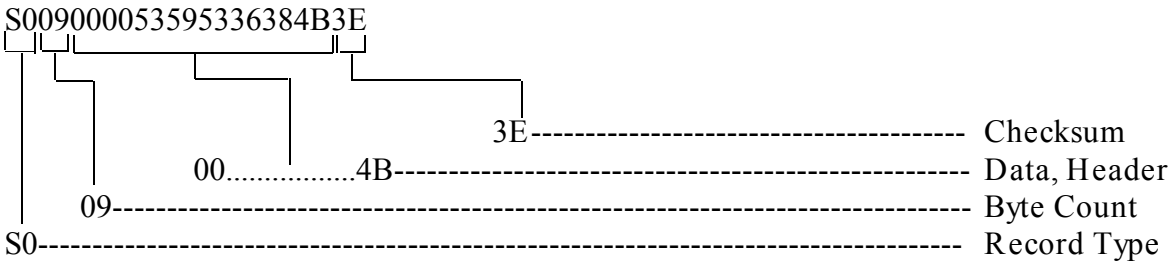
Checksum - is built from byte count, address and data.

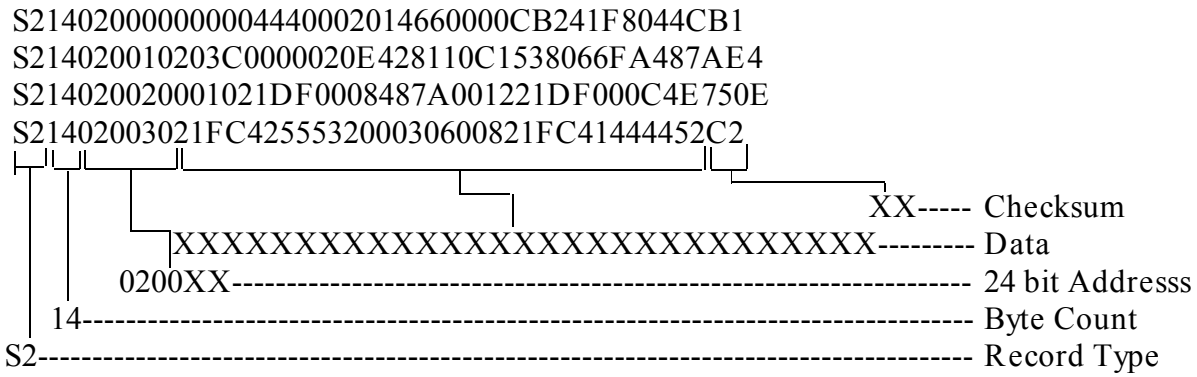
The checksum is the 1's complement of the sum of all 8-bit data/address bytes (not frames) from byte count to the last data byte, inclusive.

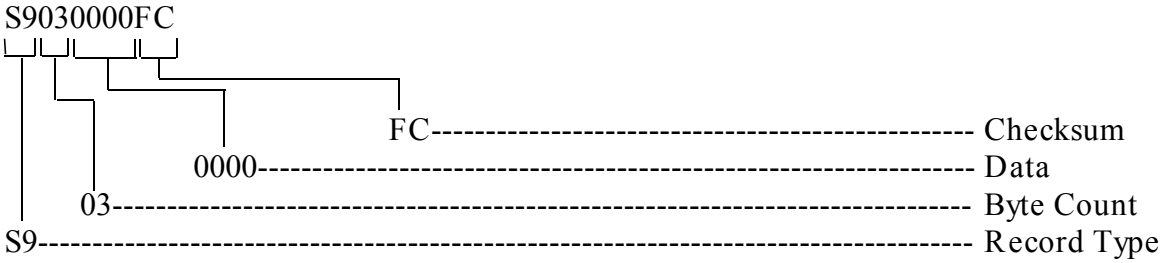
A transfer is normally performed in the following sequence:

1.	Header	String of minimum 32 ASCII Nulls
2.	S0	Header Record
3.	S1 or S2	Data Records
4.	S9 or S8	End Record

S-Record Example







Each line is separated by a minimum of 32 ASCII zeros.

Blank Page