Computing
Surface

SPARC/IO Processing Element (MK401) Users Guide

S1002–10M137.00

meiko

Meiko's address in the US is:

Meiko
130 Baker Avenue
Concord MA01742

508 371 0088
Fax: 508 371 7516

Meiko's address in the UK is:

Meiko Limited
650 Aztec West
Bristol
BS12 4SD

Tel: 01454 616171
Fax: 01454 618188

Issue Status:

Draft	
Preliminary	
Release	x
Obsolete	

Circulation Control: *External*

Contents

Overview *1*

The MK401 SuperSPARC/IO board offers high performance scalar computing power and flexible I/O options. It provides a general purpose compute server capability, typically offering to the user community shared Solaris processing resource and access to disk, networking, and other I/O resources.

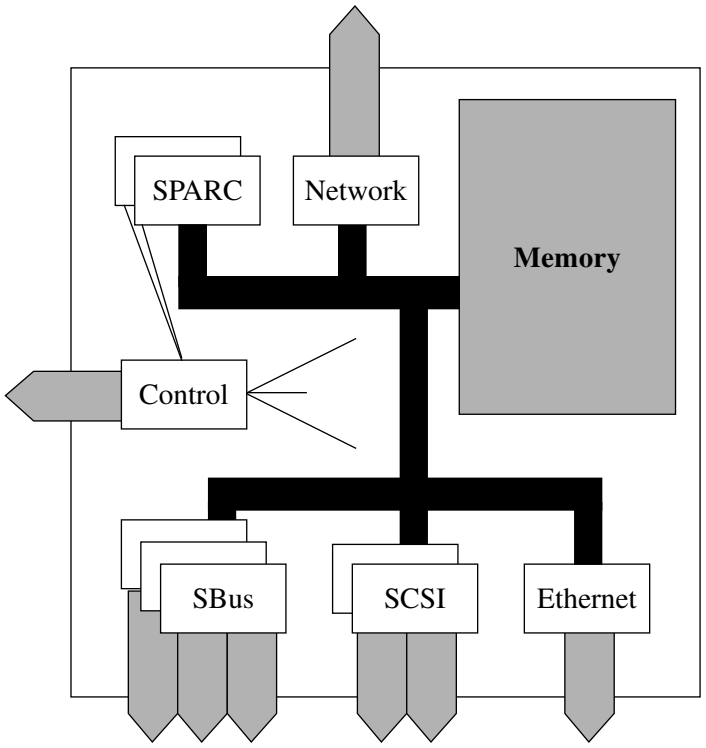
The board design encompasses at the lowest level the principle design objectives for the CS-2, offering a scalable modular construction with easy upgrade options, a reliance on state of the art commodity components, leading edge proprietary network components, and support for system-wide fault tolerance.

In outline the SuperSPARC/IO board offers:

- Dual Superscalar SPARC MBus modules. Easy upgrade to next generation SPARC technology protects investment and extends system life.
- Meiko Elan Communications Processor offering a high bandwidth, low latency interface to the CS-2 data network.
- Up to 512Mbytes of field configurable memory with error correction, detection and logging.
- Dual SCSI-2 controllers for external storage devices.
- Three full size SBus slots for more network connectivity, storage capacity, or other third party options.
- On-board Ethernet controller for use with thin or thick wire Ethernet.

- Interface to the machine-wide control area network (CAN) offering remote diagnostic control and error logging facilities.
- Keyboard, mouse, and dual serial connections.
- Fully compliant with the SPARC Compliance Definition (SCD); applications can be ported between SCD environments without change.

Figure 1-1 Board Overview



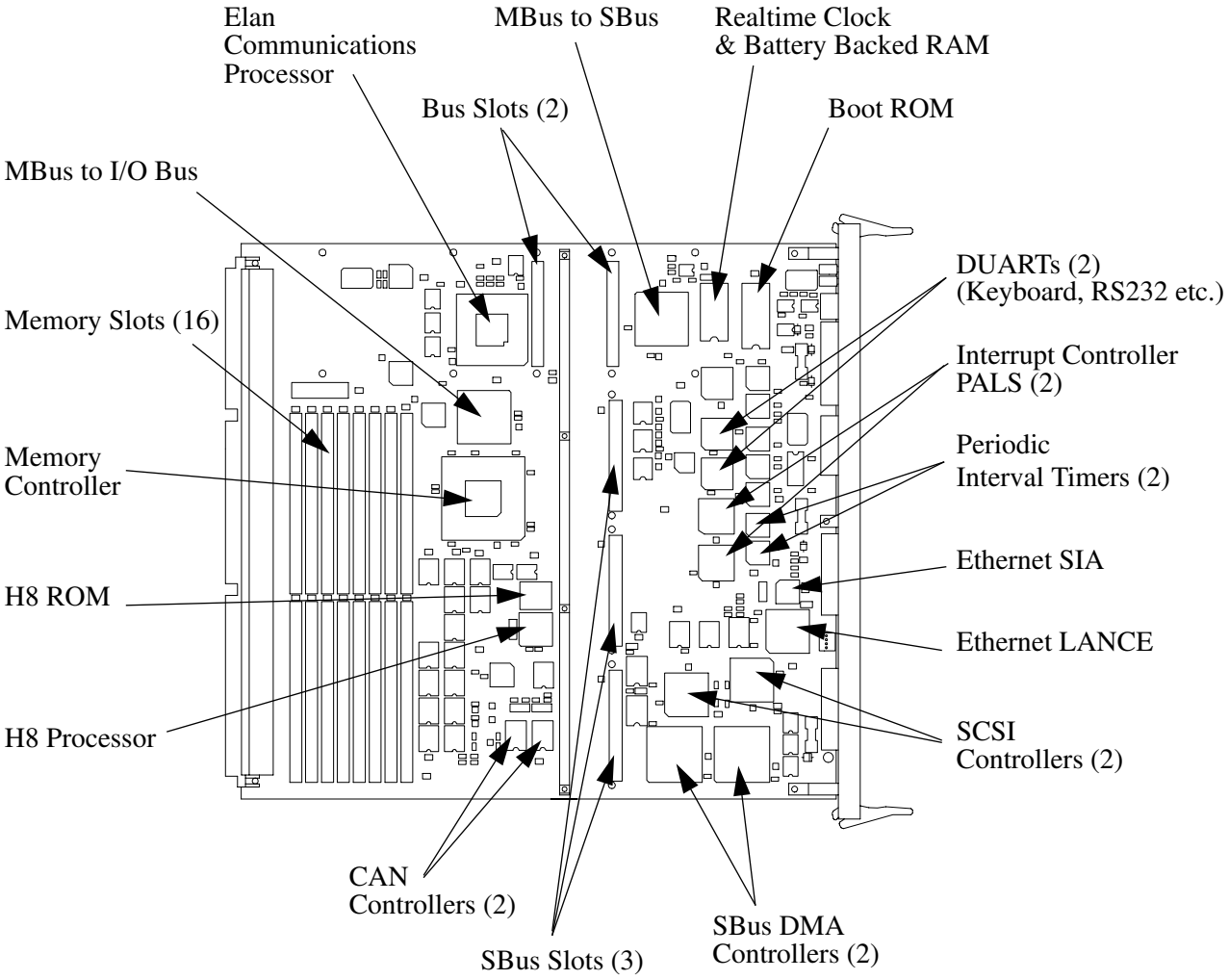
MK401 Board Description *2*

Interconnection of the processing devices and the memory system is via the industry standard SPARC MBus architecture using a 40MHz, level 2 cache coherent implementation. Interfacing the MBus are two additional buses; the SBus connects the major I/O devices (such as the SCSI controllers), and the I/O bus connects minor I/O devices (such as the real time clock).

Running throughout the whole CS-2 system is a control network (CAN) used to distribute status and configuration information, to provide remote control and diagnostics of all processors, and to create remote console connections to the processors. The MK401 has two interfaces to this network; one connected to the I/O bus (and thus providing a direct interface to the SPARC processors) and one via a dedicated micro-controller which provides board control.

The major components and their placement on the MK401 motherboard are shown in Figure 2-1.

Figure 2-1 Board Schematic Showing Major Components



MBus and SBus plug-in modules not shown.

MBus

Two full size MBus sites are provided allowing two SPARC CPUs to be supported either on a single dual processor daughter board, or two single processor boards. The MBus boards simply plug into the slots provided.

The MBus is fully level 2 compliant and runs at 40MHz. The two SPARC processors share the MBus with the Elan Communications Processor, the MBus-to-SBus interface, and the I/O bus controller; the allocation of MBus Id's is:

- MBus id 0 is the I/O bus controller.
- MBus id 4 is MBus to SBus controller.
- MBus id 6 is Elan Communications processor.
- MBus id 8 and 9 are MBus slot 0.
- MBus id 10 and 11 are MBus slot 1.

The use of MBus sockets on the motherboard allows SPARC modules to be easily upgraded as SPARC technology develops. The TI Viking and ROSS Pinnacle SPARC technologies are current options.

Note that MBus modules from different manufacturers may not be compatible. If two MBus cards are fitted they must use the same technology and originate from the same manufacturer.

ROSS Pinnacle Module

Two variants of the Pinnacle MBus module are available; you may use either two single processor modules or one dual processor module. Both variants include external second level cache.

The Pinnacle MBus modules are built upon a tightly coupled set of three ROSS devices: the RT620 HyperSPARC CPU, the RT625 cache controller, memory management, and tagging unit (CMTU), and the RT627 cache data units (CDUs).

Features of the RT620 CPU are:

- SPARC version 8 conformance.
- 90MHz clock rate.

- 4 execution units offering parallel execution of major instruction types: Load/Store, Branch/Call, integer and floating point units.
- Dual instruction fetch per clock cycle.
- 8Kbyte 2-way set-associative on-chip instruction cache.
- Instruction pipelining including a cache stage to accommodate the latency for second level cache accesses on data. Simultaneous accesses to on-chip and second level cache for each instruction fetch.
- High bandwidth 64bit Intra Module Bus (IMB) provides the interface between the CPU and the second level cache. Use of second level cache decouples the processor clock rate from the lower MBus clock rate.

Key features of the RT625 (CMTU) and RT627 (CDU) devices are:

- Full level 2 cache-coherent MBus compatibility.
- Each CDU has integral 16Kbytes x 32bit SRAM. MBus modules use either 2 or 4 CDUs for 128Kbyte or 256Kbyte second level direct-mapped cache.
- Physical cache tagging with virtual indexing allow the cache coherency logic to determine snoop hits and misses without stalling the CPU's access to the cache.
- Both copy-back and write-through cache modes supported.
- 32 byte read buffer and 64 byte write buffer for buffering the 32 byte cache lines in and out of the second level cache.
- SPARC reference MMU offering 64 entry, fully set-associative TLB with 4096 contexts.

Texas Instruments Viking Module

Two variants of the TI Viking MBus module are available. One contains a Viking SPARC processor with direct connection to the MBus. The second includes a Viking processor with additional Cache Controller and 1Mbyte of second level external cache (E-cache).

Key features of the TMS390Z50 SuperSPARC are:

- SPARC Version 8 conformance.
- 3 instructions per cycle, instruction pipelining, 150MIPs peak performance.
- SPARC Integer Unit.
- SPARC Reference MMU. Cached translation lookaside buffers (TLBs). 32bit virtual addresses, tagged with a 16bit context (65,536 contexts), map to 36bit physical addresses.
- Single and Double precision FPU. Tightly coupled to the integer execution pipeline and allowing one floating-point operation and one memory reference to be issued in each clock cycle. The FPU maintains a 4 entry FIFO queue for FP operations.
- 20Kbyte instruction cache, 16Kbyte data cache. The instruction cache is 5-way set associative, physically addressed, and non-writable. The data cache is 4-way associative and physically addressed. Both cache's are coherent with each other and with optional E-cache or MBus. Without E-cache the instruction and data caches operate in write-through mode; otherwise they are copy-back mode.
- Store buffer. A FIFO queue of 8 entries, each 64bits, decouples the instruction execution pipelines from the E-cache or MBus.
- Multiprocessor cache coherent support; highly pipelined and non-multiplexed VBus interface to optional external cache and the TMS390Z55 Cache Controller, or direct connection to the MBus.
- Prefetch buffer.
- Support for system and software debugging including hardware breakpoint.

The external cache is managed by the TMS390Z55 Cache Controller. Key features of this device are:

- Built in support for cache coherent multiprocessing; multiple Viking modules can share a single MBus and remain fully cache coherent.

- High performance VBus interface with the SPARC processor; decouples the SPARC processor from the MBus clock speed allowing higher processor performance. Reduced MBus traffic reduces contention when multiple processors share a single MBus.
- E-cache is direct mapped, copy-back, and unified: there is a single cache location where a particular byte of the physical address space can reside in the cache (direct mapping); writes by the main processor into the E-cache do not propogate to main memory until the cache is flushed or replaced (copy-back); both instructions and data are supplied to the processor from the same cache (unified).
- 1Mbyte SRAM cache.

SBus Interfaces

The MBus to SBus interface supports up to five SBus devices. Device allocation on the MK401 is as follows:

- Devices 0, 1, and 2 are the SBus slots. Slot 0 is nearest the processor modules.
- Device 4 is an SBus DVMA device serving both the Ethernet controller and one of the SCSI-2 controllers.
- Device 5 is an SBus DVMA device serving the second SCSI-2 bus controller.

The Ethernet controller connects to an external Ethernet transceiver via a front panel connection and standard Ethernet drop cable. Connection to either thin or thick wire Ethernet is supported.

The SCSI-2 controllers support an 8bit single ended SCSI bus. External connections are via the backplane connectors (allowing connection to disks within the Processor Module) or via standard high density connectors on the front panel (allowing connection to external SCSI devices). Switchable SCSI terminators are accessible from the board's front panel and are used to terminated the bus when not in use or when SCSI devices are connected to just one end (either the front panel connectors or the backplane connectors); termination of the external buses is also required.

The SBus runs at a clock speed of 20Mhz.

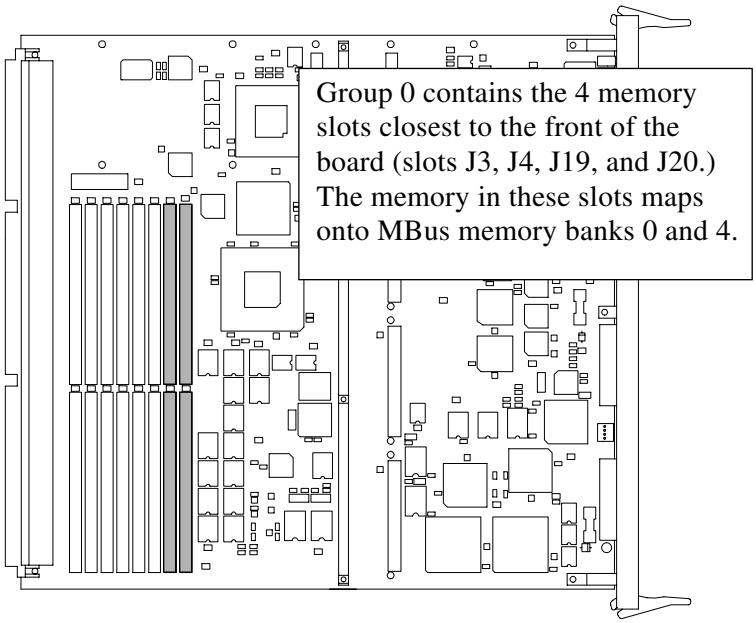
Memory Configurations

The memory controller used by the MK401 offers a 128bit wide data bus for fast memory access, with single bit error correction, 2 bit error detection, and multiple bits within nibble detection. Memory errors are logged by kernel software and propagated via the CAN bus.

Up to 16 memory modules may be fitted to the MK401 offering a maximum of 512Mbytes. Both single and double sided, 4Mbit and 16Mbit SIMM may be used. The 16 sockets on the board are configured as 4 groups of 4 sockets; within each group the SIMMs must be identical, but there is no requirement for the groups to be the same.

Memory is mapped into the MBus address space in 8 banks of 64Mbytes. Bank 0 maps onto one side of the SIMMS in physical group 0. Bank 4 maps onto the second side of the SIMMS in physical bank 0. If less than 64Mbytes of memory is present in a bank it is echoed throughout the 64Mbytes. For each 128bit MBus data access 4 bytes are read from each SIMM.

Figure 2-2 MBus Memory Mapping and Physical Memory Groups



IO Bus

The IO bus is a slave-only bus used for the connection of minor peripherals to the MBus. The following devices are connected to this bus:

- A 512Kbyte EPROM holding bootstrap and diagnostic programs. The bootstrap code initialises the hardware devices, initial page table construction, and the booting of the Unix kernel. Normally only processor 0 in a dual-processor configuration handles the hardware initialisation; processor 1 sleeps until it is woken by processor 0 with a software interrupt.
- A realtime clock module with battery backed SRAM. This holds configuration information and node fault logs, including the log of uncorrectable memory errors. The realtime clock provides year, month, day, hour, minute, and second times.
- Dual DUART devices; one for connection of keyboard and mouse, the other for two general purpose serial ports. In the absence of a keyboard the bootstrap code in the EPROM will usually direct console I/O via serial port A. The serial ports are clocked at 4.9152MHz, with a working capability of 38.4KBaud. Both serial ports share the same 25-way front panel connection; port A has full synchronous/asynchronous operation and a full complement of modem control lines; port B has a limited set of control lines and is asynchronous only.
- Interrupt controllers, one per SPARC processor. These use registers to mask out certain types of interrupt to relieve the SPARC processor from unnecessary interrupt loading, and to share the handling between the two processors. In addition some levels of interrupt are also presented direct to the Elan allowing them to be handled there and thus masked out of both the SPARC processors. The programming of the interrupt controllers is handled by the kernel device drivers.
- Periodic interrupt timers; used for maintaining the kernel clock and for kernel profiling. Two clock devices are used, one per SPARC processor. The lower level (kernel clock) interrupts from one of the clock devices are shared by both SPARC processors to remove the need for the two processor clocks to synchronise.

- A single CAN device provides both SPARCs with an interface to the machine-wide control area network (CAN). The SPARC processors write diagnostic information to this bus, and can also act as X-CAN or G-CAN routers. CAN routers transfer data between two levels of the CAN network; an X-CAN router handles transfers between the modules in a Cluster, and a G-CAN router handles transfers between Clusters. The configuration of a SPARC as a router will cause it's CAN device to generate numerous level 2 interrupts which will impact on processor performance.

Board Control Processor

The MK401 board uses an Hitachi H8/534 micro-controller (commonly referred to by Meiko as the H8) to perform basic node control functions. This controller is a single-chip 16bit RISC microcomputer with integral 2KBytes RAM, 32Kbytes EPROM, 16bit RISC CPU, and a number of I/O ports and timers.

The H8 processor runs independently of the other processors on the board and is used solely for control and diagnostic purposes. It has its own interface to the CAN bus via the second of the board's CAN interface devices. This processor receives diagnostic messages, via the CAN bus, from the local SPARC processors, and interprets incoming control messages, such as board reset, console connections, and network configuration.

Using the MK401 3

This chapter describes the usage of the MK401 in terms of its installation, hardware interfaces, and field serviceable components.

Installation

The MK401 is designed for use solely in a CS-2 Processor Module. The Processor Module supplies the board's power, cooling, and connection to the CS-2 data and control networks. The MK401 is fitted into one of the four vertical board slots behind the Processor Module's removable front panel.

Warning – You must disconnect the power from the Processor Module before removing or installing processor boards.

Warning – The board may be fitted with fragile or static sensitive devices. You must handle with care and observe anti-static precautions.

Removing the Module's Front Panel

The module's front panel is held in position by four clips, one in each corner. To remove the panel pull firmly away from the module.

The module's LED display is fitted to the module by two 50-way connectors. To remove the LED display pull firmly away from the module.

Use the reverse procedure to install the LEDs and front panel.

Installing the Processor Board

Insert the board so that it fits into the guide rails at the top and bottom of the module's board rack, ensuring that the component side is to the left (viewed facing the module). Gently push the board squarely on its front panel. Before pushing the board fully into position fold back the levers at each end of the front panel so that they are at 90^{o} to the board; now push the board (while holding the levers) until the base of the two levers is touching the card cage. To lever the board into its final position push both levers until they lie flat on the board's front panel. Secure the board by tightening the two captive screws.

Use the reverse procedure to remove the board.

Warning – You should take care not to damage the connectors at the rear of the board and on the module's backplane. Ensure that the board mates squarely with the module's backplane.

Warning – When removing or installing a board you should take care not to damage the RFI (copper) seals along the edge of the board's front panel.

Warning – To maintain proper circulation of cooling air and to conform to RFI regulations all board slots must be fitted with a processor board or blanking plate.

Field Serviceable Components

The MK401 has the following field serviceable components.

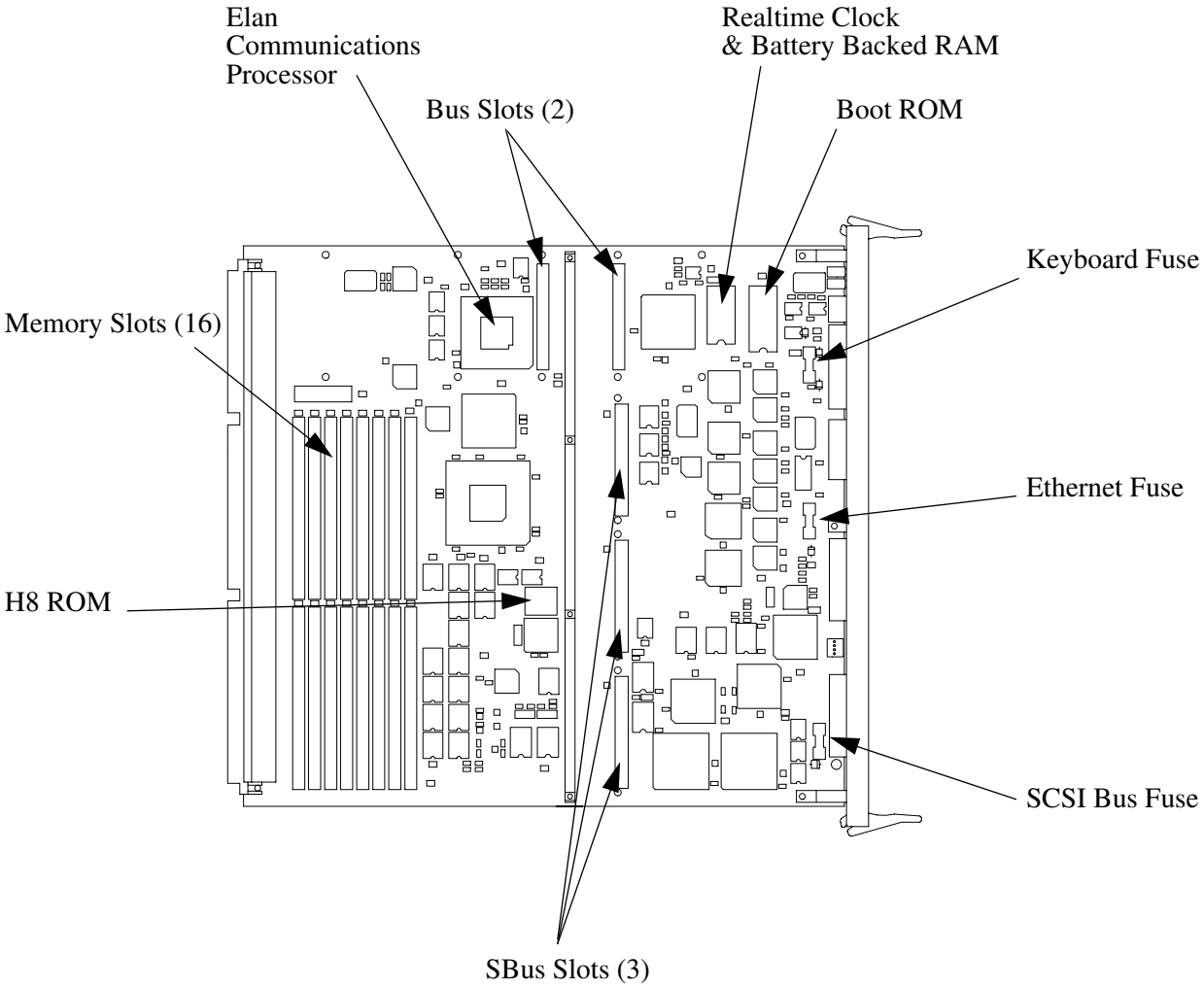
- Superscalar SPARC processor modules fitted to MBus slots.
- Three SBus slots.
- 16 memory slots.
- Boot ROM.
- H8 ROM.
- Realtime clock and non-volatile RAM (NVRAM).
- Fuses.

The location of these components on the MK401 is shown in Figure 3-1.

Warning – User's are not permitted to make mechanical or electrical modifications to the equipment. Meiko is not responsible for regulatory compliance of equipment that has been modified. You will invalidate your warranty if you make unauthorised changes.

Warning – These components are fragile or static sensitive. Handle with care and observe anti-static precautions.

Figure 3-1 Field Serviceable Components



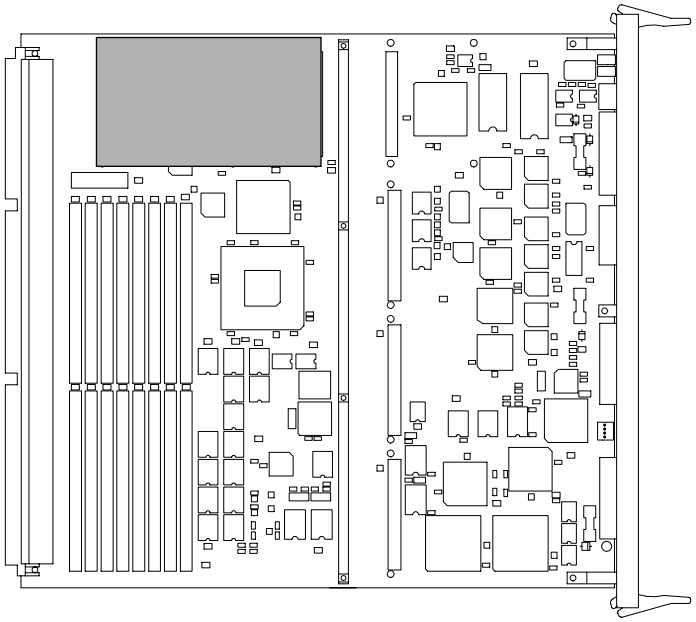
MBus (processor) and SBus (peripheral) plug-in modules are not shown.

Installing MBus Processor Modules

Superscalar SPARC modules may be installed in the two MBus sockets — when using just one module it must be installed in position 0 (the slot nearest the communications processor and the rear of the board). The sockets are polarised to prevent incorrect installation.

Installation is simple — push the processor module into the connector on the MK401 and secure into position with an M3 screw at each corner.

Figure 3-2 Position of MBus Module 0

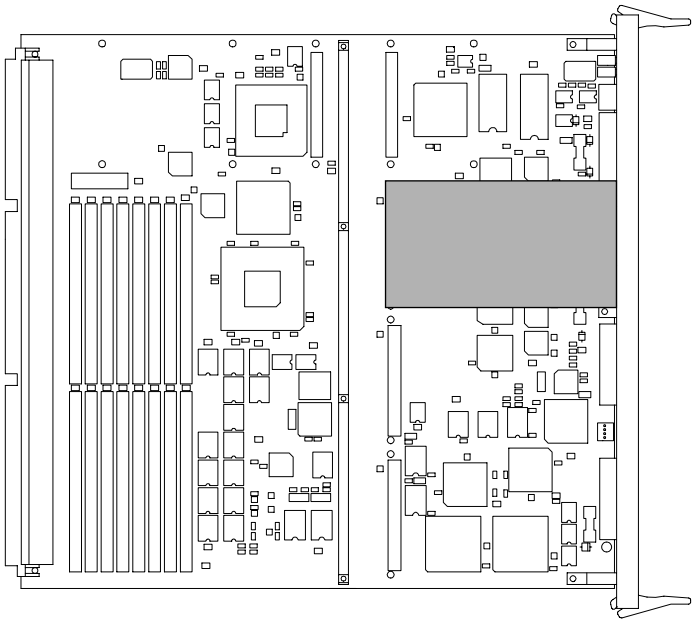


Installing SBus Modules

Three SBus slots are provided and these may be fitted with standard SBus modules; these are plugged into the SBus connectors and secured with two M3 screws. When using SBus cards that have external connections, for example a graphics card, remove the appropriate panel from the front of the MK401 — the panel is held in place by two small screws.

SBus devices are numbered from 0 to 2, device 0 being next to the processor slots. Device 4 is the on-board Ethernet and SCSI bus. Device 5 is the second on-board SCSI bus.

Figure 3-3 Position of SBus Module 0

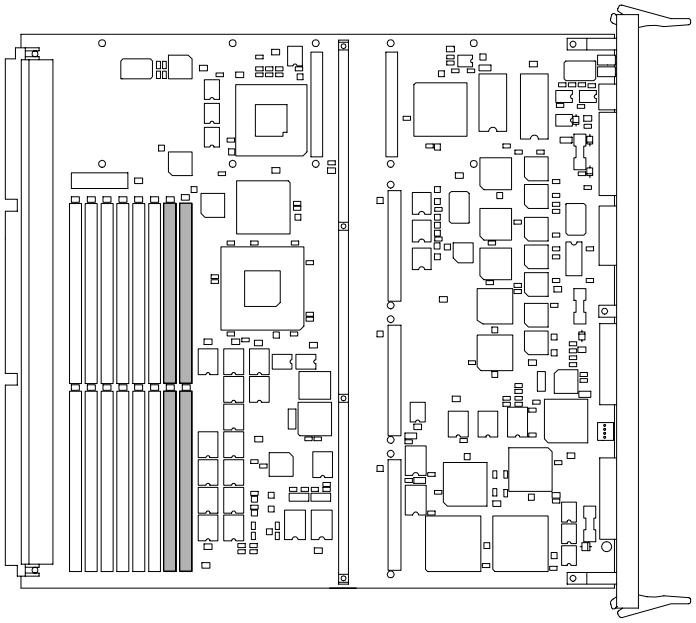


Memory

Up to 16 single in-line memory modules (SIMMs) may be fitted to the MK401 board (JEDEC 36bit SIMMs). The array is constructed of 4 groups of 4 SIMMs, arranged as shown in Figure 3-4. Within each group the SIMMs must be identical, but there is no requirement for the groups to be the same.

SIMMs can use either 4Mbit DRAM or 16Mbit DRAM technology, and be either single or double sided. This gives a minimum memory configuration of 16Mbytes and a maximum of 512Mbytes.

Figure 3-4 Memory slots on an MK401; the shaded region shows one of the 4 groups of 4 memory slots.



Boot ROM and H8 ROM

Both of these ROMs may be upgraded from time to time. They are held in sockets and are readily replaced. Note the position on pin 1 before removing the old device (usually marked by a dot on the packaging).

Realtime Clock and Battery backed RAM

The real time clock and non-volatile RAM device is held in a DIL socket and is easily replaced. Before removing the old device note the position of pin 1 (usually marked by a dot on the packaging). Note that the information within the RAM can only be restored by Meiko's engineers.

Warning – This device contains lithium batteries; never dispose of this device in a fire or attempt to dismantle.

Fuses

There are 3 fuses on the MK401:

Keyboard fuse: 20 × 5mm 1 A quick blow ceramic.

Ethernet fuse: 20 × 5mm 1 A quick blow ceramic.

SCSI bus fuse: 20 × 5mm 500 mA quick blow ceramic.

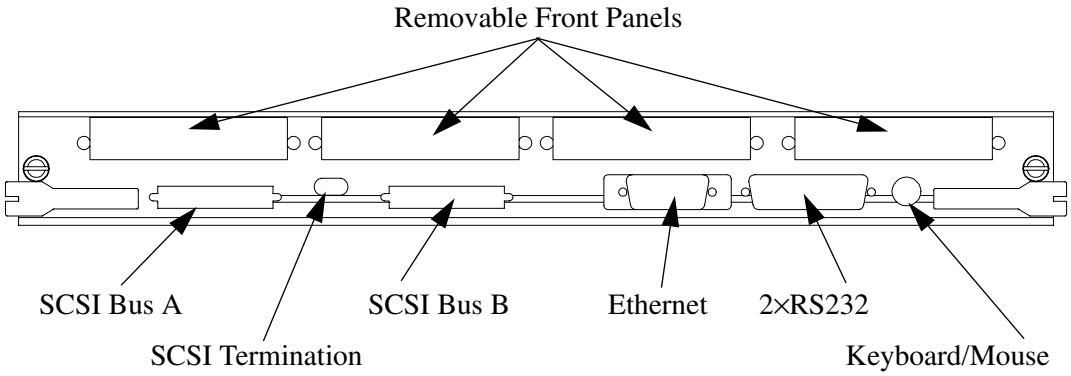
External Connections

External connections are provided for a keyboard/mouse (8 pin circular socket), serial interfaces (2 channels provided by one 25-way D-type connector), Ethernet (15-way D-type connector), and two independent SCSI buses (each via a 50-way miniature connector).

Front Panel Connections

Removable panels provide access to connectors on the optional SBus boards.

Figure 3-5 MK401 Front Panel Connections



RS232 Connections

The two RS232 channels are output via a single 25-way connector. The connections are as shown in the following tables. Signal ground is pin 7, chassis ground in pin 1.

Table 3-1 RS232 Channel A Pinout.

Signal	Input/Output	Pin number
TXD	Out	2
RXD	In	3
RTS	Out	4
CTS	In	5
CSR	In	6
DCD	In	8
DB	In	15
DD	In	17
DA	On	24
DTR	On	20

Table 3-2 RS232 Channel B Pinout.

Signal	Input/Output	Pin number
TXD	Out	14
RXD	In	16
RTS	Out	19
CTS	In	13
DCDB	In	12

Adding SCSI Peripherals

The MK401 includes two SCSI-2 controllers — both SCSI buses are available via connections on the front panel or via the Processor Module's backplane. Up to 3 additional SCSI buses can be added using standard SBus cards.

The disk devices that are optionally fitted into the Processor Modules are connected, via the module's backplane, to the SCSI controllers on your processor boards. The allocation of disks to boards and their controllers, and the number of disk devices is site specific, typical allocations are: 4 disks, one connected to SCSI bus A on each board; 4 disks, 2 connected to SCSI bus A on board's 0 and 1; 4 disks all connected to SCSI bus A on board 0.

SCSI Termination

Two switches on the MK401 front panel allow the SCSI bus termination to be switched; with the switches down termination is on.

If *either* the front panel connections *or* the backplane connections are used, or no devices are attached at all, then the on-board termination must be turned on. Only when both front-panel and backplane connectors are used together should the on-board termination switches be turned off. The end's of a SCSI bus must always be terminated.

External Indicators

Two LEDs (one green, one amber) are included on the board's front panel. The green LED is the heart beat from the board's (H8) CAN controller. The amber light illuminates each time the CAN controller transmits on the CAN bus. Both should flash steadily. These indicators are also displayed on the module's LED display.

The green LED flashes at a slow steady rate (once per second) when operating normally. A quicker flash rate (2×normal) indicates that the board's SPARC processors are not responding; a very quick flash rate (3×normal) indicates that the H8 processor on the module's controller is not responding.

Each processor board within a processor module controls a 4×4 matrix of red LEDs on the module's front panel. The MK401 displays a random pattern on these when running the Boot ROM. When Solaris has been booted a circulating pattern is displayed. The pattern can be changed by user programs and various system commands and daemons.

MBus Address Map *A*

This section identifies the key devices used on the MK401 and lists their mapping into the MBus physical address space.

For detailed information about any of the devices you should refer to the manufacturer's data sheets.

The following notes are associated with the tables:

1. These locations are byte-wide and are mapped into all 4 bytes of a word. Care should be taken to generate correct byte-wide accesses to the least significant byte of the word in order to maintain future compatibility.
2. These locations are byte-wide memory, mapped into contiguous byte locations. Word or halfword accesses will be automatically mapped into several successive byte-wide accesses.
3. These locations are byte sized registers which are only mapped into the most significant byte of a halfword. To ensure comparability with other boards only byte accesses at the correct address should be used.
4. These locations are byte sized registers which are only mapped into the least significant byte of the word. To conserve MBus bandwidth and ensure comparability with other boards only byte wide accesses at the correct address should be used.

5. These locations are halfword sized registers which are only mapped into the least significant halfword of the word. To conserve MBus bandwidth and ensure compatibility with other boards only halfword wide accesses at the correct address should be used.
6. These locations form a double-word register.

All addresses are in hexadecimal, and all locations are word-wide unless otherwise stated in the notes.

DRAM and SBus Slots

MBus Address	Usage	Read/ Write	Note
000000000	64MB Memory in bank 0.	RW	1
004000000	64MB Memory in bank 1.	RW	1
008000000	64MB Memory in bank 2.	RW	1
00c000000	64MB Memory in bank 3.	RW	1
010000000	64MB Memory in bank 4.	RW	1
014000000	64MB Memory in bank 5.	RW	1
018000000	64MB Memory in bank 6.	RW	1
01c000000	64MB Memory in bank 7.	RW	1
020000000 to dfffffff	Unused (MBus Timeout).		
e00000000 to e0fffffff	SBus Slot 0 (connector J11).	RW	
e10000000 to e1fffffff	SBus Slot 1 (connector J12).	RW	
e20000000 to e2fffffff	SBus Slot 2 (connector J13).	RW	

SBus SCSI and Ethernet

The MBus-to-SBus (M2S) device drives two LSI Logic L64853A SBus DMA devices. One (chip B) drives the SCSI-B controller, the other (chip A) controls both the SCSI-A and Ethernet controllers.

SBus DMA chip B and SCSI

An Emulex FAS101 SCSI controller provides the MK401's SCSI-B bus.

The memory map for the SCSI SBus DMA device and the SCSI controller is:

MBus Address	Usage	Read/ Write	Note
e30000000	SBus DMA_B ID register (= 0xfe810102)	R	
e30000004 to e303fffff	Unused (echoes of above)		
e30400000	SBus DMA_B Control/Status Register	RW	
e30400004	SBus DMA_B (Next)AddressCounter	RW	
e30400008	SBus DMA_B (Next)ByteCount	RW	
e3040000c	Reserved for testing M2S		
e30400010 to e307fffff	Unused (echoes of above)		
e30800000	SCSI_B Transfer Count Low	RW	1
e30800004	SCSI_B Transfer Count Mid	RW	2
e30800008	SCSI_B FIFO Data	RW	2
e3080000c	SCSI_B Command	RW	2
e30800010	SCSI_B Status	R	2
e30800010	SCSI_B Destination Bus ID	W	2

MBus Address	Usage	Read/ Write	Note
e30800014	SCSI_B Interrupt	R	2
e30800014	SCSI_B Select/Reselect Timeout	W	2
e30800018	SCSI_B Sequence Step	R	2
e30800018	SCSI_B Synchronous Period	W	2
e3080001c	SCSI_B FIFO Flags	R	2
e3080001c	SCSI_B Synchronous Offset	W	2
e30800020	SCSI_B Configuration 1	RW	2
e30800024	SCSI_B Clock Conversion Factor	W	2
e30800028	SCSI_B Test mode	W	2
e3080002c	SCSI_B Configuration 2	RW	2
e30800030	SCSI_B Configuration 3	RW	2
e30800034	SCSI_B Reserved		
e30800038	SCSI_B Transfer Count High	RW	2
e3080003c	SCSI_B Reserved		
e30800040 to e30bfffff	Unused (echoes of above)		
e30c00000 to e3fffffff	Reserved (Read Undefined)		

SBus DMA chip A, Ethernet and SCSI

An Emulex FAS101 SCSI controller and an Advanced Micro Devices AM7990 Ethernet controller (LANCE) provide the MK401's SCSI-A bus and Ethernet.

The following table identifies the memory mapping for the SBus DMA device, the SCSI, and the Ethernet controllers.

MBus Address	Usage	Read/ Write	Note
e40000000	SBus DMA_A ID register (=0xfe810102)	R	
e40000004 to e403fffff	Unused (echoes of above)		
e40400000	SBus DMA_A Control/Status Register	RW	
e40400004	SBus DMA_A (Next)AddressCounter	RW	
e40400008	SBus DMA_A (Next)ByteCount	RW	
e4040000c	Reserved for testing		
e40400010 to e407fffff	Unused (echoes of above)		
e40800000	SCSI_A Transfer Count Low	RW	2
e40800004	SCSI_A Transfer Count Mid	RW	2
e40800008	SCSI_A FIFO Data	RW	2
e4080000c	SCSI_A Command	RW	2
e40800010	SCSI_A Status	R	2
e40800010	SCSI_A Destination Bus ID	W	2
e40800014	SCSI_A Interrupt	R	2
e40800014	SCSI_A Select/Reselect Timeout	W	2
e40800018	SCSI_A Sequence Step	R	2
e40800018	SCSI_A Synchronous Period	W	2
e4080001c	SCSI_A FIFO Flags	R	2
e4080001c	SCSI_A Synchronous Offset	W	2
e40800020	SCSI_A Configuration 1	RW	2

MBus Address	Usage	Read/ Write	Note
e40800024	SCSI_A Clock Conversion Factor	W	2
e40800028	SCSI_A Test mode	W	2
e4080002c	SCSI_A Configuration 2	RW	2
e40800030	SCSI_A Configuration 3	RW	2
e40800034	SCSI_A Reserved		
e40800038	SCSI_A Transfer High	RW	2
e4080003c	SCSI_A Reserved		
e40800040 to e40bfffff	Unused (echoes of above)		
e40c00000	LANCE Register Data Port	RW	2
e40c00002	LANCE Register Address Port	RW	2
e40c00004 to e5fffffff	Unused (echoes of above)		
e60000000 to e600000ff	Unused (Invalid TLB Entry) (MBus Error?)		
e60000100 to e600001f0	M2S TLB Slices 0 through 15. (on 16-byte boundaries)	RW	
e60000200 to e6fffffff	Unused (Invalid) (MBus Error?)		
e70000000 to effffffff	Unused (SBus Reserved) (MBus Error?)		

Memory Controller

The MK401 uses an LSI Logic L64860 memory controller.

The following table shows the MBus memory maps for the memory controller's control and diagnostic ports.

MBus Address	Usage	Read/ Write	Note
f00000000	Memory Enable	RW	
f00000004	Memory Delay	RW	
f00000008	Fault Status	R(W)	
f0000000c	Video Config.	RW	
f00000010	Fault Address 0	R	
f00000014	Fault Address 1	R	
f00000018	ECC Diagnostics	RW	
f0000001c to f0ffffffff	Unused (Read undefined)		
f00000000 to fefffffff	Unused (No response) (MBus Timeout)		

Boot ROM, Serial Ports, Real Time Clock, Miscellaneous

The following table identifies the memory map for various devices on the I/O bus.

MBus Address	Usage	Read/ Write	Note
ff0000000 to ff007ffff	Boot ROM (512KByte).	R(W)	3
ff0080000 to ff00fffff	Unused (Boot ROM Echo)		
ff0100000 to ff0100007	Serial Port Controller		
ff0100000	Control Registers port B	RW	4
ff0100002	Data Buffer port B	RW	4
ff0100004	Control Registers port A	RW	4
ff0100006	Data Buffer port A	RW	4
ff0100008 to ff01fffff	Unused (Serial Port Echoes)		
ff0200000 to ff0200007	Keyboard and Mouse Port Controller		
ff0200000	Control Registers mouse port	RW	4
ff0200002	Data Buffer mouse port	RW	4
ff0200004	Control Registers keyboard port	RW	4
ff0200006	Data Buffer keyboard port.	RW	4
ff0200008 to ff02fffff	Unused (Keyboard and Mouse Port Echoes)		
ff0300000 to ff0301fff	Real Time Clock module and 8KByte SRAM	RW	3
ff0302000 to ff03fffff	Unused (RTC Echoes)		

MBus Address	Usage	Read/ Write	Note
ff0400000 to ff06fffff	Unused (MBus Error)		
ff0700000	Node Reset Request	W	5
ff0700004 to ff07001ff	Unused (echoes)		
ff0700200	MBus Grant readback	R	5
ff0700204 to ff07003ff	Unused (echoes)		
ff0700400	Physical Slot Identifier	R	6
ff0700404 to ff07005ff	Unused (echoes)		
ff0700600	LED Bargraph	RW	6
ff0700604 to ff07007ff	Unused (echoes)		

Control Area Network Interface

The SPARC processors' interface the Control Area Network via Philips PCA82C200 devices, which are mapped into the MBus address space at the following locations:

MBus Address	Usage	Read/ Write	Note
ff0700800	CAN — Control Register	RW	5
ff0700804	CAN — Command Register	W	5
ff0700808	CAN — Status Register	R	5
ff070080c	CAN — Interrupt Register	R	5
ff0700810	CAN — Acceptance Code Register	RW	5
ff0700814	CAN — Acceptance Mask Register	RW	5

MBus Address	Usage	Read/ Write	Note
ff0700818	CAN — Bus Timing Register 0	RW	5
ff070081c	CAN — Bus Timing Register 1	RW	5
ff0700820	CAN — Output Control Register	RW	5
ff0700824	CAN — Test Register		
ff0700828	CAN — TXBuf Identifier	RW	5
ff070082c	CAN — TXBuf RTR Data Length code	RW	5
ff0700830	CAN — TXBuf Data Byte 1	RW	5
ff0700834	CAN — TXBuf Data Byte 2	RW	5
ff0700838	CAN — TXBuf Data Byte 3	RW	5
ff070083c	CAN — TXBuf Data Byte 4	RW	5
ff0700840	CAN — TXBuf Data Byte 5	RW	5
ff0700844	CAN — TXBuf Data Byte 6	RW	5
ff0700848	CAN — TXBuf Data Byte 7	RW	5
ff070084c	CAN — TXBuf Data Byte 8	RW	5
ff0700850	CAN — RXBuf Identifier	RW	5
ff0700854	CAN — RXBuf RTR Data Length code	RW	5
ff0700858	CAN — RXBuf Data Byte 1	RW	5
ff070085c	CAN — RXBuf Data Byte 2	RW	5
ff0700860	CAN — RXBuf Data Byte 3	RW	5
ff0700864	CAN — RXBuf Data Byte 4	RW	5
ff0700868	CAN — RXBuf Data Byte 5	RW	5
ff070086c	CAN — RXBuf Data Byte 6	RW	5
ff0700870	CAN — RXBuf Data Byte 7	RW	5

MBus Address	Usage	Read/ Write	Note
ff0700874	CAN — RXBuf Data Byte 8	RW	5
ff0700878	CAN — Unimplemented		
ff070087c	CAN — Clock Divider Register	RW	5
ff0700880 to ff0700fff	Unused (echoes of above)		

Interrupt Request Control and Status Registers

The MK401 has many sources of interrupts to the SPARCs. These are assigned priority levels and are passed to the Interrupt Controllers (one per SPARC). These enable the handling of interrupts to be shared between the SPARCs. The controller has the following major features:

- Masks for all hardware generated interrupts.
- Software interrupts on six levels using a multi-processor compatible set/clear register structure. This allows one processor to interrupt the other by writing once to a single memory location, without having to do a read-modify-write cycle which is subject to being broken by the other processor. Supported interrupt levels are 1, 4, 6, 12, 13, and 15.
- Latches triggered by the output of the free running timers (the Periodic Interrupt Timers). These latches capture the timer passing through zero event and cause an interrupt to the processor. When the interrupt handler accepts the interrupt and clears the request the latch is reset.
- Priority encoder to generate the 4bit encoded interrupt level that is passed to the SPARC processor module.

The Periodic Interrupt Timers are AMD 82C54 free running timers, one assigned to each of the SPARC processors. Timer 0 generates the lower priority clock ticks signal, issuing a level 10 interrupt when the timer has counted down from a preset value to 0 (the count down is at a rate of one decrement every 3.2us). Timer 1

also counts down from a preset value but issues a level 14 interrupt and decrements at a rate of 0.8us. Both processors share timer 0 from one of the timer devices (to avoid the need for explicit synchronisation).

MBus Address	Usage	Read/ Write	Note
ff0701000	IRQ PAL 0 — Timer Latches	RW	5
ff0701002	IRQ PAL 0 — Mask Register Read/ Clear	RW	5
ff0701006	IRQ PAL 0 — Mask Register Set	RW	5
ff070100a	IRQ PAL 0 — Software Interrupt Reg Read / Clear	RW	5
ff070100e	IRQ PAL 0 — Software Interrupt Reg Set	W	5
ff0701010 to ff07011ff	Unused (echoes)		
ff0701200	Timer 0 Level10 Processor 0 and 1	RW	5
ff0701204	Timer 0 Level14 Processor 0	RW	5
ff0701208	Timer 0 Spare timer	RW	5
ff070120c	Timer 0 Control register	RW	5
ff0701210 to ff07015ff	Unused (echoes)		
ff0701600	Err and Powerfail Latch	RW	5
ff0701604 to ff0701fff	Unused (echoes)		
ff0702000	IRQ PAL 1 — Timer Latches	RW	5
ff0702002	IRQ PAL 1 — Mask Register Read/ Clear	RW	5
ff0702006	IRQ PAL 1 — Mask Register Set	RW	5

MBus Address	Usage	Read/ Write	Note
ff070200a	IRQ PAL 1 — Software Interrupt Reg Read / Clear	RW	5
ff070200e	IRQ PAL 1 — Software Interrupt Reg Set	W	5
ff0702010 to ff07021ff	Unused (echoes)		
ff0702200	Spare timer	RW	5
ff0702204	Timer 1 Level14 Processor 1	RW	5
ff0702208	Timer 1 Spare timer	RW	5
ff070220c	Timer 1 Control register	RW	5
ff0702210 to ff0702fff	Unused (echoes)		
ff0703000 to ff0703fff	Unused (Read Undefined)		
ff0704000 to ff07fffff	Unused (echoes of above)		

MBus to I/O Bus

The MBus to I/O Bus interface is via an LSI Logic L64851 device.

MBus Address	Usage	Read/ Write	Note
ff0800000	Software interrupts and enable.	RW	
ff0800004	Active level of external interrupts.	RW	
ff0800008	Programmable limit register 0.	RW	
ff080000c	Programmable limit register 1.	RW	
ff0800010	I/O bus devices available.	RW	
ff0800018	Latency delay register.	RW	

MBus Address	Usage	Read/ Write	Note
ff080001C	MBus Id register.	R	
ff0800020	Programmable timer.	R	
ff0800024	Programmable timer.	R	

MBus to SBus Chip, Elan, and MBus Slot Slaves

The MBus-to-SBus controller is an LSI Logic L64852C. The Elan Communications Processors is a Meiko device.

MBus Address	Usage	Read/ Write	Note
ff4fffff0	M2S Virtual Address Table Base Address	RW	
ff4fffff4	M2S IO/MMU Control register	RW	
ff4fffff8	M2S Error/Status register	R	
ff4fffffc	M2S — MBus ID Register	R	
ff5000000 to ff6f7ffff	Unused (MBus Timeout)		
ff6f80000 to ff6ffdfff	ELAN Command port area	RW	
ff6ffe000 to ff6ffffbf	ELAN Hush register area	RW	
ff6ffffc0	ELAN Clock Hi	RW	
ff6ffffc4	ELAN Clock Hi	R	
ff6ffffc8	ELAN Clock Lo	RW	
ff6ffffcc	ELAN Clock Lo	R	
ff6ffffd0	ELAN Alarm	RW	
ff6ffffd4	ELAN Alarm	R	

MBus Address	Usage	Read/ Write	Note
ff6ffffd8	ELAN Interrupt	R	
ff6ffffdc	ELAN Interrupt	R	
ff6ffffe0	ELAN Clock Hi	R	7
ff6ffffe4	ELAN Clock Lo (for 64bit accesses)	R	7
ff6ffffe8	ELAN Main Proc. Interrupt Mask	RW	
ff6ffffec	ELAN Main Proc. Interrupt Mask	R	
ff6fffff0	ELAN Control register	RW	
ff6fffff4	ELAN Control register	R	
ff6fffff8	MBus Port ID register for ELAN Chip	R	
ff6fffffc	MBus Port ID register for ELAN Chip	R	
ff7000000 to ff7ffffff	Unused (MBus timeout)		
ff8000000 to ff9ffffff	Used by MBus slave device in MBus Slot 0		
ffa000000 to ffbffffff	Used by MBus slave device in MBus Slot 1		
ffc000000 to fffffffff	Unused (MBus timeout)		

A

NVRAM Variables ***B***

The battery-backed RAM in the realtime clock module is used to used to store basic machine start-up and communication options.

These parameters may be queried using the Forth Monitor (i.e. at the `ok` prompt):

`printenv`	Display current variable settings.
`setenv` *variable value*	Assign (or reassign) a value to a variable.
`set-default` *variable*	Restore the variables default value.
`set-defaults`	Restore the default values to all variables.

For example:

```
ok setenv output-device can
```

Alternatively the System Administrator can use the `eeprom`(1m) command to view and change the variables direct from a Unix command shell. For example:

```
root@cs2# eeprom output-device=can
```

Some of the parameters (those marked in the following list) may also be modified using the Set function in Pandora's Network and Configuration Views.

B

Variable	Default	Description
sbus-probe-list	43012	Identifies the SBus slots to probe and the probe order.
keyboard-click?	false	If true, enable keyboard click.
keymap	*no default*	Name of custom keymap file.
output-device †	screen	Power-on output device. One of screen, can, ttya, or ttyb. Use can to enable console connections to be grabbed by cancon(1m) and Pandora.
input-device †	keyboard	Power-on input device. One of keyboard, can, ttya, or ttyb. Use can to enable console connections to be grabbed by cancon(1m) and Pandora.
cancon-host	4294967295	Used to record the host of the cancon(1m) remote console connection through a reboot of this processor. Do not change.
elanip-broadcast-high †	4096	Highest Elan Id in network.
elanip-broadcast-low †	0	Lowest Elan Id in network.
ep-btxpktlifetime †	1000	Elan packet characteristics.
ep-btxtimeout †	1000	Elan packet characteristics.
ep-txpktlifetime †	10000	Elan packet characteristics.
ep-txtimeout †	10000	Elan packet characteristics.
ep-bigmsgbcastboxes †	4	Elan packet characteristics.
ep-bigmsgboxes †	32	Elan packet characteristics.
ep-bigmsgsize †	20416	Elan packet characteristics.
ep-smallmsgbcastboxes †	4	Elan packet characteristics.
ep-smallmsgboxes †	32	Elan packet characteristics.
ep-smallmsgsize†	4032	Elan packet characteristics.
elan-boot-id †	0	Elan Id of node that this processor boots from.

Variable	Default	Description
`elan-node-id` †	`0`	Elan Id of this processor.
`elan-node-level` †	`1`	The processor's level in the CS-2 network.
`elan-num-levels` †	`1`	Number of levels in the CS-2 network.
`elan-top-switch` †	`0`	Specifies the level in the network that the processor sees it's topswitch. Usually this is level 0, the real top of the network.
`elan-switch-plane` †	`0`	Switch plane that this processor receives it boot code from when booting via the Elan network.
`ttyb-rts-dtr-off`	`false`	If true, Solaris does not assert RTS/DTR on ttyb.
`ttyb-ignore-cd`	`true`	If true, Solaris ignores carrier-detect on ttyb.
`ttya-rts-dtr-off`	`false`	If true, Solaris does not assert RTS/DTR on ttya.
`ttya-ignore-cd`	`true`	If true, Solaris ignores carrier-detect on ttya.
`ttyb-mode`	`9600,8,n,1,-`	ttyb (baud rate, #bits, parity, #stop, handshake). Baud rate is 110, 300, 1200, 2400, 4800, 9600, 19200, or 38400. #bits is 5, 6, 7, or 8. Parity is `n` (none), `e` (even), `o` (odd), `m` (mark), `s` (space). Handshake is `-` (none), `h` (hardware rts/cts), `s` (software).
`ttya-mode`	`9600,8,n,1,-`	ttyb (baud rate, #bits, parity, #stop, handshake). Baud rate is 110, 300, 1200, 2400, 4800, 9600, 19200, or 38400. #bits is 5, 6, 7, or 8. Parity is `n` (none), `e` (even), `o` (odd), `m` (mark), `s` (space). Handshake is `-` (none), `h` (hardware rts/cts), `s` (software).
`fcode-debug?`	`false`	If true, include name fields for plug-in device Fcodes.
`diag-file` †	`kadb`	The file and arguments to load from the root filesystem when the `diag-switch?` is true; otherwise use the `boot-file` parameter.
`diag-device` †	`elan`	Device to boot from when the `diag-switch?` is true; one of `disk`, `net`, or `elan`. Specify `elan` to boot over the CS-2 data network.

B

Variable	Default	Description
boot-file		The file and arguments to load from the root filesystem (e.g. `kadb -v`, or `/kernel/unix -vr`). No file implies `/kernel/unix`.
boot-device †	elan	Device to boot from; one of `disk`, `net`, or `elan`. Specify `elan` to boot over the CS-2 data network.
auto-boot? †	false	Boot automatically after power-on. Default value is true.
watchdog-reboot?	false	If true, reboot after watchdog reset.
local-mac-address?	false	If true, use the ethernet address taken from the `local-mac-address` parameter; otherwise use the IdPROM.
screen-#columns	80	Number of on-screen columns.
screen-#rows	34	Number of on-screen rows.
selftest-#megs	1	Megabytes of RAM to test on power-up or memory test.
scsi-initiator-id	7	SCSI bus address of host adapter, range 0–7.
cpu-#mhz	40	CPU clock rate.
use-nvramrc?	false	If true, execute the code stored in the nvramrc parameter when the boot ROM starts up.
nvramrc		Forth code to execute when the boot ROM starts-up (but only if `use-nvramrc?` is true).
sunmon-compat?	false	If true, come-up with old style prompt '>'.
security-mode	none	System security level for monitor commands; one of `none`, `command`, or `full`. None allows all commands to be executed. Command allows the continue and boot (without parameters) commands to be executed; others require a password. Full requires a password before any commands may be executed.
security-password	*no default*	The password used with security-mode described above.

Variable	Default	Description
`security-#badlogins`	*no default*	System set variable showing the number of times a bad password was specified.
`oem-logo`	*no default*	Byte array OEM logo (enabled by `oem-logo?`). Create a Forth array containing the logo and then copy into the `oem-logo` field.
`oem-logo?`	`false`	Enables OEM logo defined by `oem-logo`.
`oem-banner`	*no default*	Text displayed in the custom OEM banner alongside the OEM logo. Enabled by `oem-banner?`.
`oem-banner?`	`false`	Enables OEM banner text specified in `oem-banner`.
`hardware-revision`	*no default*	Hardware revision of this board (e.g. Rev D).
`last-hardware-update`	*no default*	Date of board's manufacture or last upgrade (e.g. 25May94).
`testarea`	`0`	Unused.
`mfg-switch?`	`false`	If true, perform repeated self tests.
`diag-switch?`	`false`	Run in diagnostic mode.

† These parameters may be changed using the Set function in Pandora's Network and Configuration Views.

Forth Monitor Commands *C*

The following commands have been added to the Forth Monitor and are in addition to the commands that are normally present on a Solaris system. The additional commands relate to the Control Network (CAN) and Elan network.

CAN Commands

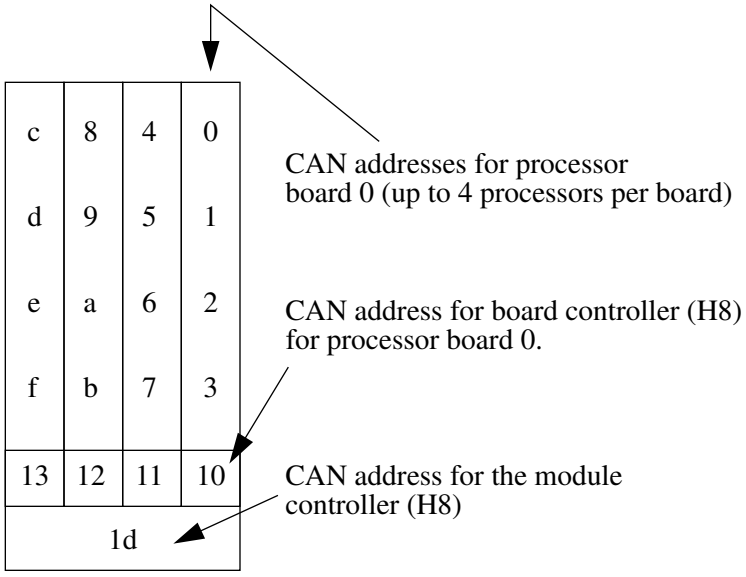
To test and use the CAN bus you need to understand CAN addresses.

Nodes are addressed by their physical position in terms of Cluster, Module, and Node id's (CMN). In CAN packets each of these id's is represented by a 6bit field; the hexadecimal representation of these three 6bit fields is a Node Id.

The module id is derived from the switch at the rear of the module. The numbering of the nodes within a module is shown in Figure C-1.

For example, the Node Id of processor with CMN 0:2:3 (processor 3, module 2, cluster 0) is 00083. The node's controlling H8 has the Node Id 00090. The node's module controller has the Node Id 0009d.

Figure C-1 CAN Addresses within a Module



Testing the CAN Device

Commands are provided to test the SPARC's CAN interface device, to test the board and module controllers, and to monitor activity on the CAN bus.

Testing the CAN Interface Device

The `test` command tests the SPARC's CAN device by writing various values into its test register. The test is repeated using the test registers on the board controller's H8 and the module controller's H8.

```
ok test /can
 Register test 0x00: OK
 Register test 0xff: OK
 Register test 0xaa: OK
 Register test 0x55: OK
 Checking on-board H8: OK.
 Checking module controller: OK.
```

Testing the CAN Bus

You can test the CAN bus connection between nodes by using the `rtest` command. In the following example data is transferred from the current node to node 4:

```
ok 4 rtest
Performing remote write/read test on node 4
Remote node type is MK405
..................................................
Time taken was 13630mSecs
```

Checking the Board and Module Controllers

You can check that both the board controller and module controller H8 processors are running by using the `ping-h8` and `ping-module` commands. Note that you need to change directory to `/can` before you use these commands.

```
ok cd /can
ok ping-h8
On-Board H8 is MK401.
ok cd ..
```

```
ok cd /can
ok ping-module
Module controller is MK515.
ok cd ..
```

Querying CAN Bus Usage

You can query the utilisation of the CAN bus by using the `perf` command. This command shows the number of CAN packets received since the machine was powered-up, and the number since the last query. You need to change to the `/can` directory before using this command:

```
ok cd /can
ok perf
Total number of messages received since power-up: 259380
No. of messages received per second since the last check: 4
ok cd ..
```

Monitoring CAN Bus Packets

You can *snoop* the CAN bus (monitor that packets on the bus) using the `snoop` command. Note that you cannot use this facility if you are connected to the Forth Monitor via a cancon connection. You need to change to the `can` directory before using this command.

```
ok cd can
ok snoop
Can't can-snoop if you are a cancon slave
ok cd ..
```

CAN Addresses

To determine the CAN address of this node use `.can-id`. This displays the node's address in terms of its CMN, Node Id, and Slot Id. The Slot Id is for Meiko engineering use[1].

```
ok .can-id
SlotId: 0090, CAN Node-id: 00088 [00:02:08]
```

Similarly the CAN address of the board's controlling H8 processor can be obtained with the `.h8-id` command:

```
ok .h8-id
The on-board H8 is node 00092.
```

1. The slot id is the node's physical position in the machine represented by a 5 bit cluster number, a 5 bit module number, a 2 bit slot number, 2 unused (always 0 bits), and a 2 bit processor number; the 2 bits that represent the slot number are transposed.

The CAN address of the H8 that controls the board's module can be determined by the `.module-id` command:

```
ok .module-id
The module controller is node 0009d.
```

You can convert from CAN node id's to Cluster, Module, Node addresses (and vice versa) by using the `canid>cmn` and `cmn>canid` commands respectively. Note that you need to change directory to `can` before you use these commands.

```
ok cd can
ok 4 canid>cmn
0 0 4
ok cd ..
```

```
ok cd can
ok 0 2 8 cmn>canid
Node Id is 00088
ok cd ..
```

Querying CAN Objects

Can packets include a 10 bit address space which, although not sufficient to map into the MBus/H8 physical address space, is adequate to map-in various status and control devices. These are referred to as CAN objects. Reading or writing to these objects allows you to query the status of a processor, board, or module, and to issue control instructions. See the header file `/opt/MEIKOcs2/include/canio/canobj.h` for a list of object addresses and their meanings.

Local CAN objects are those that relate directly to this node. Remote CAN objects maybe those of a board, module controller, or remote SPARC.

You use the `rlo` command to read a local object. You need to pass an object id on the Forth stack; in the following example we request the board type and are returned 191 (an MK401):

```
ok 0 rlo
Read: 191
```

To read a remote object you need to push onto the Forth stack a CAN node id and the object id. In the following example we request the board type of node `c1` (module 3, board 0, node 2), which is an MK405:

```
ok c1 0 rro
Read: 195
```

The following additional example fetches the board type of node `dd`, which is the controller for module 3, cluster 0:

```
ok dd 0 rro
Read: 203
```

Similar commands exist to write to CAN objects, but their direct use is not recommended (they can reconfigure and reset the machine).

Remote Console Connections

You can create a console connection to a remote node by using `cancon`. You need to pass on the Forth stack a CAN node id.

You cannot create a `cancon` connection from within an existing cancon connection. If you are remotely interacting with a node's Forth monitor via `cancon` (or Pandora) an attempt to create another `cancon` connection will fail.

```
ok 4 cancon
Connected to node 00004

cs2-4 console login:
```

If your node is currently serving a remote console connection to someone else you can force it to disconnect that connection by using `cancon-dis`. In the following example the current connection to node 8 is dropped:

```
ok 8 cancon-dis
Disconnecting node 00008 [00:00:08] ...
```

Elan Commands

The Elan device includes self test code that can be executed by the `test-all` command (which tests memory, SBus, CAN, Elan and all other devices with self test code) or explicitly by the `test /elan` command.

```
ok test /elan
Initialising Elan/Selftest software ... OK
Checking threads processor ... OK

Testing from level 1 to level 1.
Generating a route to level 1 ... OK
Ping ... OK
Check-Ping ... OK
Spraying data to top switch ... OK
Testing spray buffer ... OK

Closing down Elan/Selftest software ... OK
```