SPARCbook™

SPARCbook 1
Technical Reference Manual

TADPOLE

SPARCbook 1
Technical Reference Manual

Tadpole Technology Inc
12012 Technology Blvd.
Austin, TX 78727
USA

Tel: 512-219-2200
Fax: 512-219-2222

Tadpole Technology plc
Cambridge Science Park
Milton Road
Cambridge, CB4 4WQ
ENGLAND

Tel: 0223 250030
Fax: 0223 250036

FCC Class B Notice

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna
- Increase the separation between the equipment and receiver
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult your supplier, or an experienced radio or television technician for help

Modifications

This equipment contains no user-serviceable parts. You are advised that unauthorized changes or modifications to the equipment could cause it to exceed Class B limits and may void the authority granted by the FCC to operate the equipment.

Shielded Cables

Connections between the SPARCbook and any connected peripherals must be made using shielded cables to maintain compliance with FCC radio frequency emission limits.

FCC Part 68 Modem Information

This equipment complies with Part 68 of the FCC rules. On the underside of this equipment is a label that contains, among other information, the FCC registration number and ringer equivalence number (REN) for this equipment. If requested, this information must be provided to the telephone company.

This equipment uses the following USOC jacks: RJ11C

The REN is used to determine the quantity of devices which may be connected to the telephone line. Excessive RENs on the telephone line may result in the devices not ringing in response to an incoming call. In most, but not all areas, the sum of the RENs should not exceed five (5.0). To be certain of the number of devices that may be connected to the line, as determined by the total RENs, contact the telephone company to determine the maximum REN for the calling area.

If this equipment causes harm to the telephone network, the telephone company will notify you in advance that temporary discontinuance of service may be required. If advance notice is not practical, the telephone company will notify the customer as soon as possible. Also, you will be advised of your right to file a complaint with the FCC if you believe it is necessary.
The telephone company may make changes in its facilities, equipment, operations or procedures that could affect the operation of the equipment. If this happens, the telephone company will provide advance notice in order for you to make the necessary modifications in order to maintain uninterrupted service.

If trouble is experienced with this equipment, please contact Tadpole Technology Inc, 8310 Capital of Texas Highway North, Austin, Texas 78731 Tel: (512) 338 4221 for repair and/or warranty information. If the trouble is causing harm to the telephone network, the telephone company may request you remove the equipment from the network until the problem is resolved.

The following repairs can be done by the customer: None

This equipment cannot be used on telephone company-provided coin service. Connection to Party Line Service is subject to state tariffs.

Issue A of January 1992
Part Number: 980 011

CONTENTS

INTRODUCTION
TO THE
SPARCbook TECHNICAL REFERENCE MANUAL

The SPARCbook Technical Reference Manual is published as a supplement to the *SPARCbook User's Guide* to provide additional technical information. This document describes the hardware operation of the SPARCbook notebook computer and also provides details about the resident operating system supplied with each unit.

MANUAL STRUCTURE

The SPARCbook Technical Reference Manual is structured as follows:

- **Technical Overview**
  Chapter 1 discusses the principle features of the SPARCbook and introduces the reader to the main hardware devices that provide control over the SPARCbook's operations.

- **Architecture Overview**
  Chapter 2 describes the internal architecture of the SPARCbook, showing how the major devices are connected together.

- **Detailed Device Descriptions**
  Chapters 3 - 11 discuss in some depth the implementation of the major hardware devices in the SPARCbook.

- **Schematics Description**
  Chapter 12 provides a description of the schematic diagrams and discusses the theory of operation.

INTENDED AUDIENCE

The SPARCbook Technical Reference Manual is aimed at the hardware engineer wishing to carry out service or repairs, and at the software engineer wishing to implement hardware drivers.

It is assumed that the reader is familiar with the operation of SPARCbook, as detailed in the *SPARCbook User Guide*, and has a basic understanding of computer hardware.

CONVENTIONS USED IN THIS DOCUMENT

The reader is advised to note the following conventions employed in the SPARCbook Technical Reference Manual.

Logic states

The terms *Clear* or *low* indicate that the signal being discussed is at the logic level '0',

The terms Set or *high* indicate that the signal being discussed is at the logic level '1'.

The term *Asserted* indicates that a signal is in its true or active state regardless of whether that state is high or low.

The term *Negated* indicates that a signal is in its false or inactive state regardless of whether that state is high or low.

Data Length Definitions

A *Halfword* is taken to contain 16 bits.

A *word* is taken to contain 32 bits.

A *doubleword* is taken to contain 64 bits

CHAPTER ONE
OVERVIEW

The SPARCbook provides in a compact notebook format many of the advanced features and high performance normally associated with desktop workstations. Low power consumption and compactness have been achieved by the use of highly integrated components without compromising either the functionality or performance of the SPARCbook.

The SPARCbook is housed in a magnesium alloy case measuring 11.8" x 8.5" x 1.9", and weighs 6.8lbs, including batteries and peripherals.

The SPARCbook's standard features include a 640 x 480 pixel color or monochrome liquid crystal display, an 82-key keyboard, a 3.5" floppy disk drive, an IDE hard disk drive, and Modem, Ethernet, Centronics and serial interfaces. A second hard disk drive is available as an option, when it is fitted in substitution for the floppy disk drive.

All of the major integrated circuit components are carried on a single printed circuit board, using the latest surface mount technology and avoiding the need for space consuming and power hungry expansion cards. The use of components with low power demand and the SPARCbook's advanced power management facilities allow the SPARCbook to be used continuously for 2.5 hours or more when powered from the battery.

CENTRAL PROCESSOR

Processing power in the SPARCbook is provided by the Cypress CY7C600 implementation of the Scalable Processor Architecture (SPARC) RISC microprocessor. The Cypress CY7C600 is a chip set which comprises the following:

- **Integer Unit (IU)**
- **Cache Controller and Memory Management Unit (CMU)**
- **Optional Floating Point Unit (FPU)**
- **64Kbytes zero wait state Cache**

The IU is the basic processing engine which executes all of the instruction set except for floating point instructions which are performed by the FPU when it is fitted. The CMU provides memory management and address translation facilities between the SPARC virtual bus and the main system bus. Two CY7C157 SRAM devices provide 64Kbytes of fast cache memory resident on the SPARC virtual bus.

The SPARC CPU is a RISC (reduced instruction set computer) based processor. It uses a simplified range of commands to carry out operations, enabling most instructions to execute within a single clock cycle.

Conventional processors tend to use large and complex instruction sets, taking several clock cycles to perform each instruction. Many of the instructions available to this type of processor are rarely if ever used, a fact that has led to the development of RISC processor concepts.

The already high performance of the SPARC CPU is further enhanced by the ability of FPU to execute instructions simultaneously with the IU, and by the provision of cache memory. The Cache memory is specialized area of fast (zero wait state) memory which allows many instructions and operands to be fetched locally by the CPU without it having to access the comparatively slow main memory.

MAIN MEMORY

The SPARCbook is built using either 4Mbit or 16Mbit DRAM devices to provide 8 or 32Mbytes of DRAM. The main memory provides a 64-bit wide bus interface designed specifically to support burst cycles by the CPU.

The memory management unit maintains entries in cache memory called cache lines, each one a 256 bits wide copy of a similar sized area of main memory. When the IU requires data that is not present in the cache, the MMU copies the new data from main memory to cache memory using burst cycles of four 64-bit reads (see Chapter Three).

Using burst transfers, the memory interface is capable of a transfer rate of 72Mbyte/s. While using random access a transfer rate of 40Mbyte/s is achieved. Both rates are at a 25MHz processor clock speed.

The main memory also features software selectable parity protection.

DISPLAY

The SPARCbook is equipped with a 640 x 480 pixel color or monochrome liquid crystal flat panel display. Both types are side-lit by a cold fluorescent lamp, which can be controlled to vary the screen brightness. Reducing the brightness of the sidelight reduces the power consumption, prolonging battery life.

The display is controlled by a GD6410 VGA compatible display controller made by Cirrus Logic. This provides control for a monochrome LCD display and for an external color CRT display. Both displays may be used simultaneously. Advanced grey scale control by the GD6410 provides quality grey scale rendering of color images for clear display on the LCD.

In the color SPARCbook, a GD6340 color LCD controller provides the additional control required for the color LCD display. The color model is able to display 16 colors simultaneously from a palette of 262144.

The SPARCbook supports three display modes:

- **Mode 1 – Inbuilt LCD only**
- **Mode 2 – Simultaneous LCD and standard VGA 640 x 480 Color monitor**
- **Mode 3 – Extended VGA 800 x 600 external high resolution only**

Both color and monochrome models of the SPARCbook are able to display an image in 16 colors on the external monitor. They are both able to display the same image simultaneously on the LCD and external CRT when in mode 2.

Mode 3 allows the SPARCbook to display an image on an external high resolution monitor, but turns the inbuilt LCD screen off. This mode requires the use of a multisync monitor.

UNIVERSAL PERIPHERAL CONTROLLER

One of the highly integrated devices used in the SPARCbook to achieve its compact size is the 82C710 Universal Peripheral Controller (UPC) manufactured by Chips and Technologies Inc. This single device provides the following:

- **Hard Disk Control**
- **Floppy Disk Control**
- **Centronics Interfacing**
- **External Mouse/Keyboard Interfacing**
- **RS232 Serial Interfacing**

Floppy Disk Control
The UPC provides control for one or two IDE (Integrated Drive Electronics) PC/AT compatible hard disk drives. It provides a complete electronic interface, and control of a 16-bit data buffer.

Floppy Disk Control
Single hard disk versions of the SPARCbook contain a 3.5" floppy disk drive. The UPC provides an NEC765 compatible command set for control of the drive, and the SPARCbook's operating system allows 720Kbyte and 1.4Mbyte SUN and DOS format disks to be read and written.

Centronics Interface
The UPC provides control and data buffering for the parallel interface which appears on the rear panel of the SPARCbook. This interface supports connection to a printer with a Centronics interface.

External Mouse/Keyboard Interface
The UPC provides data buffering and control over a serial interface. This interface supports the connection of an external mouse or keyboard and appears via a 6-way mini DIN socket on the rear panel. This interface operates with TTL signal levels and does not allow the connection of RS232 devices.

RS232 Serial Interface
In addition to the TTL serial interface, the UPC provides data buffering and control over an RS232 serial interface.

MICROCONTROLLER SUBSYSTEM

An 87C51 microcontroller provides control over the in-built keyboard, the bit I/O interface and the power management functions.

The bit I/O interface enables aspects of the SPARCbook's functionality to be controlled in software using keyboard commands. The microcontroller interprets certain special characters from the keyboard as being commands, and carries out system configuration changes accordingly. This includes display screen brightness and contrast.

The bit I/O interface also controls the LEDs next to the display in order to convey machine status information to the user.

The power management function maintains a constant check on the condition of the supply voltages, and provides early warning when the battery requires recharging. Recharging is carried an in-built power supply unit and battery charger when the SPARCbook is connected to an external electricity supply.

The microcontroller also provides control over the Mousekey.

ETHERNET INTERFACE

The SPARCbook is equipped with an IEEE802.3 compliant Ethernet interface controlled by a 86960 Network Interface Controller with Encoder (NICE) from Fujitsu. This provides complete control over the network interface.

The NICE has direct control over a 64Kbyte area of SRAM in which it maintains receive and transmit buffers. The host is able write data into the transmit buffer and read data of the receive buffer via port in the NICE's address space. The NICE is able to carry out network operations with minimal demand on the CPU, and an integral driver and receiver interfaces directly to the 15-way D-type socket at the rear of the unit.

MODEM

The SPARCbook's modem interface is implemented with chipset comprising a Sierra SC11075 Modem Access Controller (MAC) and SC11054. The SC11075 provides a direct interface between the host system and SC11054 modem; it incorporates an Intel 8096 equivalent processor core, supporting an AT command set; it contains a built-in 16C450 equivalent UART; and it contains 16Kbytes of on-chip ROM and 320bytes of RAM.

The SC11054 is a complete 2400bps (bits per second) modem IC which, when combined with the SC11075, provides Sendfax capability up to 9600bps.

The modem interface can be controlled completely using Hayes compatible commands written to the UART section of the MAC. Commands are sent to the interface using character strings, and these are interpreted by the MAC's in-built processor core and acted on appropriately.

Command characters and data for transmission onto the telephone line are written into the UART's transmit buffer. Data received from the telephone line and status information from the MAC can be read from the UART's receive buffer.

REAL TIME CLOCK AND EPROM

Many of the background operations of the SPARCbook, such as system startup and battery condition monitoring, are controlled by the Tadpole resident firmware monitor. This is a body of software (most often referred to as firmware) that is permanently stored in an EPROM and is not erased when the machine is without power.

A 48T02 provides a battery backed real time clock and non-volatile RAM. This provides time of day clock and calendar functions and a 2Kbyte area of RAM which retains data when the board is without power.

A 1Mbit serially accessed EEPROM is used by the resident firmware to store the SPARCbook's Ethernet address and manufacturing information. The EEPROM is accessible via the microcontroller.

SPARCbook TECHNICAL SPECIFICATION

- **Processor and Memory**

Processor	25MHz CY601 Integer Unit
Cache Controller/MMU	25MHz CY604 CMU
Cache	64Kbyte direct mapped virtual cache
FPU (optional)	25MHz CY602 Floating Point Unit
DRAM (standard)	8Mbytes with byte parity
DRAM (-E models)	32Mbytes with byte parity

- **Components**

EPROM	128Kbytes - 1Mbyte	One 8-bit 32-pin JEDEC socket
SERIAL	UPC	One asynchronous RS232 port TxD, RxD, CTS, RTS, DCD, DTR, DSR, RING
MISC	MK48T02	RTC/2Kbyte DRAM with battery backup
	NMC93C46	1Kbit EEPROM
NETWORK	Fujitsu NICE Controller	IEEE802.3 interface with local buffer memory
DISK	82C710 UPC Controller	Two IDE disks supported Floppy disk controller

- **I/O Interfaces**

Ethernet	IEEE802.3 compliant 15-pin D-type AUI interface with slidelock
Centronics	IBM PS/2 compatible interface 36-pin miniature D-type with cable to 25-pin D-type socket
Mouse/Keyboard	IBM PS/2 compatible (TTL) 6-pin mini-DIN socket
Modem	2400baud data and 9600baud SendFax FCC Part 68 compliant, 0.3 REN V.29, V.27ter, V.22bis, V.22, V.21 Bell 212A, Bell 103 standards RJ11 jack socket

VGA	640x480 25.057MHz dot clock 800x600 36.242MHz dot clock (external monitor) 15-pin PS/2 compatible D-type connector
Serial	110 to 38400baud 9-pin mini-DIN connector

- **Peripherals**

Winchester disk	85Mbyte formatted IDE interface or 125Mbyte formatted IDE interface, depending upon model One or two drives per system Maximum transfer rate 8Mbyte/s buffer to memory Maximum sustained transfer rate 1.8Mbyte/s
Floppy disk	720Kbyte/1.44Mbyte formatted 3.5" floppy disk drive Fitted in single Winchester disk models

- **Display**

Monochrome	640x480 VGA paper-white display side-lit Displays 16 gray scales from 64
Color (-C models)	640x480 VGA color STN display side-lit Displays 16 colors from 256

- **Keyboard**

82-key notebook style with integral Mousekey based upon Force Sensitive Resistor (FSR) technology

- **LEDs**

External DC supply connected
Hard disk active
User controlled indicator
Battery low/exhausted indicator

- **Case and Emissions**

Magnesium alloy AZ91 castings
SPARCbook is certified to FCC Part 15 Class B

- **Power Supply**

Internal	Switched mode PSU with battery charge and charge detection facilities
External	Universal input 110V-240V AC Output 18V 0-3.1A DC

- **Physical**

Dimensions	11.8" x 8.5" x 1.95"
Weight (with battery)	6.8lb/7.0lb (monochrome/color)
(without battery)	5.2lb/5.4lb (monochrome/color)

- **Environmental**

Temperature (operational)	5-40 degrees Celsius
(storage)	-20 - 60 degrees Celsius
Vibration (operational)	5-100Hz 0.25G
(storage)	5-100Hz 1.2G
Shock (operating)	4G
(storage)	50G
Humidity (operating)	8-80% relative humidity

- **Operating System**

SunSoft Solaris 1.01	SPARCbook Version A
Comprising	SunOS 4.1.2 sun4m architecture OpenWindows V3 SPARCbook utilities Meets SPARC Compliance Definition (SCD) 1.1

Mono Backlight Power Control Board Hitachi TW-8594V-0

SPARCbook DETAILED ADDRESS MAP

This section provides a detailed address map for the SPARCbook. Please note the meanings of the following abbreviations used in the address map. All addresses are shown in hexadecimal.

R/W: R = read, W = Write
SIZE: b = byte, h= halfword (16-bits), w= word (32-bits), d = doubleword (64-bits)

DRAM

ADDRESS	DEVICE or REGISTER	R/W	SIZE
000000000 - 0007FFFFF	DRAM (8Mbyte)	R/W	b/h/w/d
000000000 - 001FFFFFF	DRAM (32Mbyte)	R/W	b/h/w/d

Display Controller

00C000000 - 00CCFFFFF	CL-GD6410/CL-GD6340† cirrus logic 510/623-8300		
00C0003B4	CRTC Index MP	R/W	b
00C0003B5	CRTC Data MP	R/W	b
00C0003C0	Attribute Controller Index	R/W	b
	Data	W	b
00C0003C1	Attribute Controller Data	R	b
00C0003C2	Misc Output	W	b
	Feature	R	b
00C0003C3	Motherboard Sleep Address	R/W	b
00C0003C4	Sequencer	R/W	b
00C0003C5	Sequencer	R/W	b
00C0003C6	RAMDAC Pixel Mask	R/W	b
00C0003C7	RAMDAC Address – Read Mode	W	b
	RAMDAC Status Register	R	b
00C0003C8	RAMDAC Address – Write Mode	W	b
00C0003C9	RAMDAC Data	R/W	b
00C0003CA	Feature Control	R	b
00C0003CC	Misc Output	R	b
00C0003CE	Graphics Controller and Extensions Index	R/W	b
00C0003CF	Graphics Controller and Extensions Data	R/W	b
00C0003D4	CRTC Index	R/W	b
00C0003D5	CRTC Data	R/W	b
00C0003DA	Feature Control	W	b
	Display Status	R	b
00C0046E8	AT Adapter Sleep Address	R/W	

Note † CL-GD6340 only present in color models. It contains RAMDAC registers shadow mapped over similar CL-GD6410 registers.

Ethernet Controller

ADDRESS	DEVICE or REGISTER	R/W	SIZE
00C200000 - 00CFFFFF	Fujitsu 86960 NICE		
00C200000	Transmit Status	R/W	b
00C200001	Receive Status	R/W	b
00C200002	Transmit Interrupt Mask	R/W	b
00C200003	Receive Interrupt Mask	R/W	b
00C200004	Transmit Mode	R/W	b
00C200005	Receive Mode	R/W	b
00C200006	Control Register 1	R/W	b
00C200007	Control Register 2	R/W	b
00C200008	Node ID (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b
	Buffer Memory Port (Register Bank 2)	R/W	h
00C200009	Node ID (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b
00C20000A	Node ID (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b
	Transmit packet Control 1 (Register Bank 2)	R/W	b
00C20000B	Node ID (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b
	Transmit packet Control 2 (Register Bank 2)	R/W	b
00C20000C	Node ID (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b
	DMA Enable (Register Bank 2)	R/W	b
00C20000D	Node ID (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b
	DMA Burst Control (Register Bank 2)	R/W	b
00C20000E	TDR (LSB) (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b
	Receive Buffer Pointer (Register Bank 2)	R/W	b
00C20000F	TDR (MSB) (Register Bank 0)	R/W	b
	Mask Address (Register Bank 1)	R/W	b

Note: Register Bank 0, 1 and 2 are selected using bits 2 and 3 of Control Register 2

Modem Interface

ADDRESS	DEVICE or REGISTER	R/W	SIZE
00C400000 - 00C4FFFFF	Sierra SC11075 – UART Registers		
00C400000	Receive Buffer	R	b
	Transmit Buffer	W	b
	Divisor Latch (LSB)†	R/W	b
00C400001	Interrupt Enable	R/W	b
	Divisor Latch (MSB) †	R/W	b
00C400002	Interrupt ID	R	b
00C400003	Line Control	R/W	b
00C400004	Modem Control	R/W	b
00C400005	Line Status	R/W	b
00C400006	Modem Status	R/W	b
00C400007	Scratch Pad	R/W	b

Note: † The divisor latch can be accessed only when bit 7 of the Line Control Register is set.

Real Time Clock / SRAM

00C600000 - 00C7FFFFF	MK48T02		
00C600000	SRAM	R/W	b/h/w/d
00C6007F8	Calibration	R/W	b
00C6007F9	Seconds	R/W	b
00C6007FA	Minutes	R/W	b
00C6007FB	Hours	R/W	b
00C6007FC	Day	R/W	b
00C6007FD	Date	R/W	b
00C6007FE	Month	R/W	b
00C6007FF	Year	R/W	b

16-Bit I/O Region – Hard Disk Data Interface

00C800000 - 00CBFFFFF			
00C8001F0	IDE – Hard Disk Data Interface	R/W	h

8-Bit I/O Region – UPC Associated Interfaces

ADDRESS	DEVICE or REGISTER	R/W	SIZE
00CC00000 - 00CEFFFFF	82C710 Universal Perpheral Controller		
00CC001F1	IDE – Error Register	R	b
00CC001F2	IDE – Sector Count	R/W	b
00CC001F3	IDE – Sector Number	R/W	b
00CC001F4	IDE – Cylinder Low	R/W	b
00CC001F5	IDE – Cylinder High	R/W	b
00CC001F6	IDE – Drive/Head Register	R/W	b
00CC001F7	IDE – Status Register	R	b
	IDE – Command Register	W	b
00CC003F6	IDE – Alternate Status	R	b
	IDE – Fixed Disk Register	W	b
00CC00310	Mouse Port – Data Register	R/W	b
00CC00311	Mouse Port – Status/Control Register	R/W	b
00CC00330	Parallel Port – Data Register	R/W	b
00CC00331	Parallel Port - Status Register	R	b
00CC00332	Parallel Port – Command Register	W	b
00CC00390	UPC Configuration Index	W	b
00CC00391	UPC Configuration data	R/W	b
00CC003F2	Floppy I/F – Digital O/P Register	W	b
00CC003F4	Floppy I/F – Main Satus Register	R	b
00CC003F5	Floppy I/F – Data Register	R/W	b
00CC003F7	Floppy I/F – Data Rate	W	b
	Floppy I/F – Digital Input Register	R	b
00CC003F8	UART – Receive Buffe	R	b
	UART – Transmit Buffer	W	b
	UART – Baud Rate Divisor (LSB)†	R/W	b
00CC003F9	UART – Interrupt Enable Register	R/W	b
	UART – Baud Rate Divisor (MSB)†	R/W	b
00CC003FA	UART – Interrupt Flag Register	R/W	b
00CC003FB	UART – Byte Format Register	R/W	b
00CC003FC	UART – Modem Control Register	R/W	b
00CC003FD	UART – Line Status Register	R/W	b
00CC003FE	UART – Modem Status Register	R/W	b
00CC003FF	UART – Scratch Pad	R/W	b

Note: † The Baud Rate Divisor Registers can only be accessed when bit 7 of the Byte Control Register is set.

MPI ASIC

ADDRESS	DEVICE or REGISTER	R/W	SIZE
00D00000 – 00DFFFFF	Mbus to Pbus Interface Controller		
00D000000	Mbus Error Status	R	b
00D000010	Set Count CT0	R/W	w
00D000018	Current Count CT0	R	w
00D000020	Control CT0	R/W	b
00D000028	Set Count CT1	R/W	w
00D000030	Current Count CT1	R	w
00D000038	Control CT1	R/W	b
00D000050	Interrupt Pending	R	w
00D000058	Interrupt Polarity	R/W	w
00D000060	Interrupt Enable	R/W	w
00D000068	Interrupt Acknowledge	W	w
00D000070	Serial Receive Port	R	b
00D000088	Serial Receive Status	R	b
00D000090	Serial Transmit Port	W	b
00D000098	Serial Transmit Data	W	b
00D0000A0	Reset	R/W	b
00D0000A8	Clock Control	R/W	b
00D0000B8	Power Down	W	b
00D0000C0	FIFO Start Address	R./W	w
00D0000C8	FIFO Control	R/W	w
00D0000F8	FIFO Data	R/W	d

R/W: R = read, W = Write
SIZE: b = byte, h = halfword (16-bits), w = word (32-bits), d = doubleword (64-bits)

CHAPTER TWO
ARCHITECTURE

One of the factors that contributes to the compactness of the SPARCbook is that at its heart is a single board of less than 12 square inches, the Base Board. This one board carries the processor module, provides 8 or 32Mbytes of memory and provides all of the control required for the display, keyboard, disks and interfaces.

This section provides a brief description of the architecture of the SPARCbook. Figure 2-1 shows how the main components connect together.

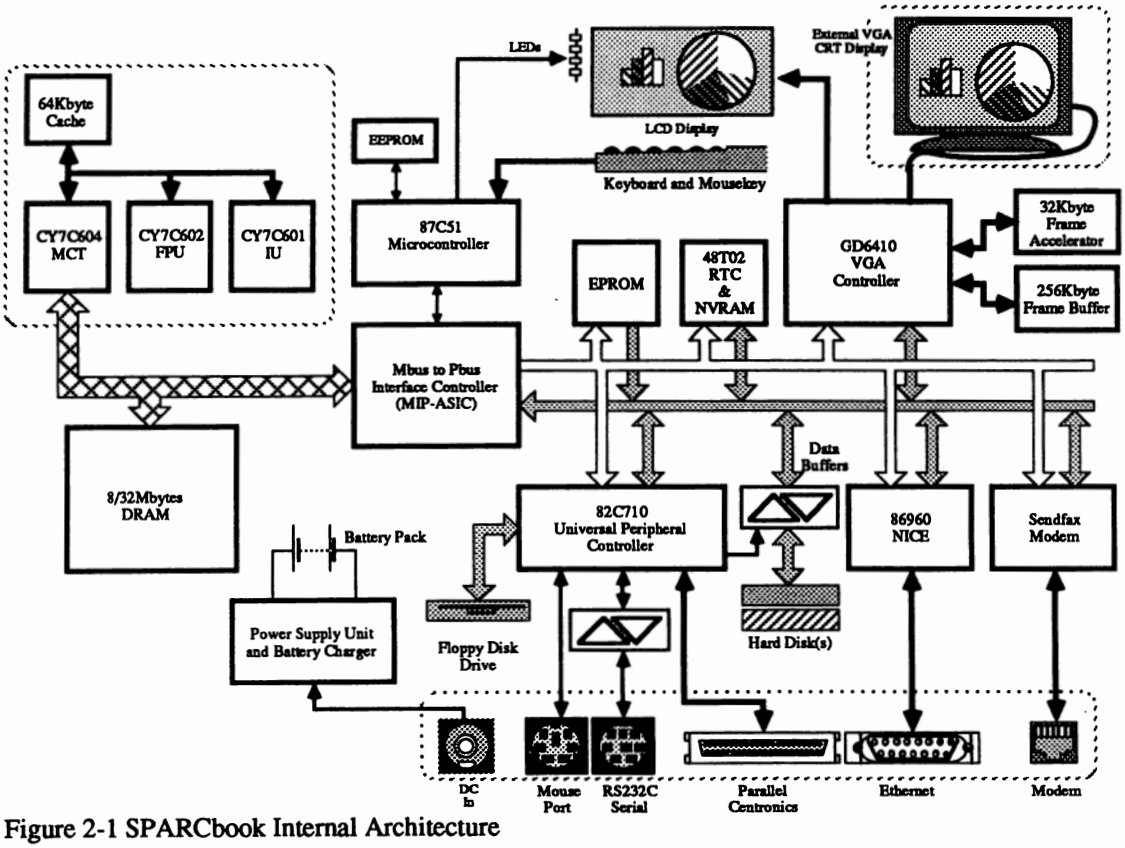


Figure 2-1 SPARCbook Internal Architecture

BUS STRUCTURE

It will be seen from Figure 2-1 that the SPARCbook is based on a dual bus architecture, with the SPARC CPU and DRAM connected to a fast 64-bit bus, called the Mbus, and the I/O devices connected to a 16-bit peripheral bus called the Pbus.

The SPARC processor is the only Mbus master device and is able to access all locations in DRAM, and locations on the Pbus via the MPI.

As described in the previous chapter, the SPARC CPU comprises the integer unit, floating point unit and the combined memory management unit and cache controller. These communicate via a private address bus which remains local to the SPARC chipset. This consists of a 32-bit address bus, an 8-bit address space identifier bus, and a 32-bit data bus. The IU uses logical (or virtual) addresses as labels to identify locations in its address space. The memory management unit translates the virtual addresses used on this bus into physical addresses used on the M-bus.

Mbus

The Mbus is an address and data multiplexed bus, which means that some of the wires are used to carry both address information and data, though at different times. The time when address information is on the bus is the address phase, and the time when data is on the bus is the data phase. In a SPARC based environment, a bus cycle consists of an address phase followed by one data phase for single access cycle, or an address phase and four data phases for a burst cycle.

The memory management unit produces a physical address on Mbus(35:0) and, during the address phase of the Mbus, provides bus control information on Mbus(45:36). The whole bus, Mbus(63:0), is used to transfer data during the data phase.

Pbus

The Pbus is a 16-bit PC-AT style bus which supports the on-board I/O devices. It consists of 16 data lines, PD(15:0), and 19 address lines, PA(18:0). The CPU addresses and exchanges data with the disk, Ethernet , modem and serial interfaces and the EPROM and RTC using the Pbus.

Mbus - Pbus Interface

The gateway between the Mbus and Pbus is provided by the Mbus to Pbus Interface controller (MPI), a Tadpole Technology designed custom ASIC (Applications Specific Integrated Circuit).

The MPI provides an internal 512byte bidirectional FIFO, address decode facilities for the Pbus, interrupt support and two internal counter/timers. It provides a byte packing interface which converts between 8-, 16-, 32- and 64-bit accesses used on the Mbus, and 8- and 16-bit accesses used on the peripheral bus. This allows the two buses to operate at their optimum speed.

The MPI provides two data paths between the Mbus and Pbus.. The first path provides direct access by the CPU to the Pbus devices, and the second path is via the MPI's internal 512byte FIFO. The MPI also provides DMA support for floppy disk operations.

In addition to providing an interface between the Mbus and Pbus, the MPI contains a fifteen level interrupt controller and two 16-bit counter/timers. The interrupt controller section is used to consolidate the large number of interrupt requests from devices within the SPARCbook, and from the FIFO controller and counter/timers within the MPI. Interrupt controller provides the CPU with a code which identifies a pending interrupt, and allows individual interrupts to be masked

CHAPTER THREE
THE SPARC CPU MODULE

The SPARC CPU module comprises the CY7C601A integer unit (IU), the CY7C602A floating point unit (FPU), and the CY7C604A cache controller and memory management unit (CMU). In the SPARCbook implementation, these are carried on a small module along with two CY7C157A cache storage units (CSU).

The devices on the SPARCbook CPU module are connected together via a 32-bit virtual address bus and a 32-bit data bus. The CMU provides an interface between this bus and the Mbus. The Mbus is a SPARC architecture standard 64-bit multiplexed address and data bus that provides the SPARCbook with a high bandwidth data path between the CMU and main memory. The CMU performs address translations between these buses and provides memory management and cache control functions. The two CY7C157A SRAMs provide 64Kbytes of zero wait-state cache memory.

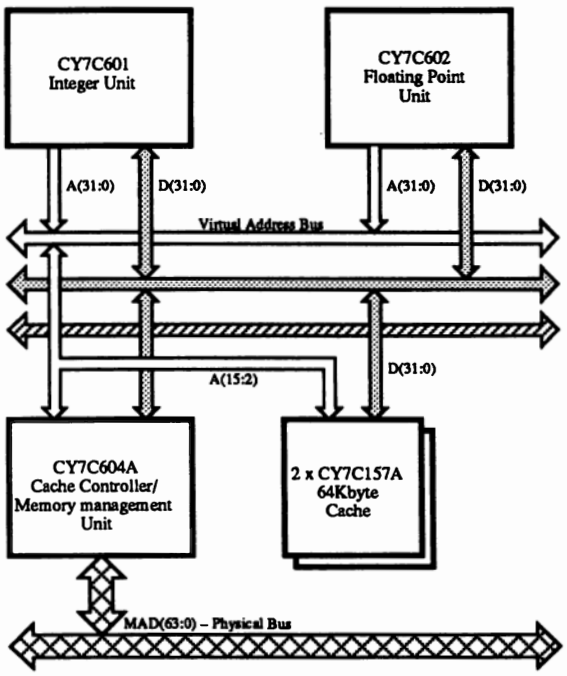


Figure 3-1 The SPARC CPU Module

CY7C601A INTEGER UNIT

The IU is the main processing engine that executes all of the instruction set except for floating point operations.

Instructions Overview

The CY7C600 architecture supports 62 integer instructions which fall into the following basic categories:

- **Load and Store Instructions**
- **Arithmetic, Logical and Shift Instructions**
- **Control Transfer Instructions**
- **Read/Write Control Registers Instructions**

The load and store instructions are the only instructions which cause the movement of data on the Mbus. They use two registers, or a register and a constant, to calculate the memory address involved. Halfword accesses must be aligned on 2-byte boundaries, word accesses on 4-byte boundaries, and doubleword accesses on 8-byte boundaries. These alignment restrictions greatly speed up memory access.

The arithmetic, logical and shift instructions compute a result that is a function of one or two source operands and then place the result non-destructively in a register. They perform arithmetic, logical, or shift operations.

The control transfer instruction category includes jumps, calls, traps, and branches. Control transfers are usually delayed until after execution of the next instructions so that the pipeline is not emptied every time a control transfer occurs, allowing compilers to be optimized for delayed branching.

The read/write control register instructions include instructions to read and write the contents of various control registers. Generally the source or destination is implied by the instructions.

In addition to the instruction types within the categories listed above, there are Floating Point instructions executed by the FPU.

Table 3-1 provides a summary of the IU instruction set.

INPUTS	OPERATION	
Load and Store Instructions		
LDSB(LDSBA*)	Load Signed Byte	(from Alternate Space)
LDSH(LDSHA*)	Load Signed Halfword	(from Alternate Space)
LDUB(LDUBA*)	Load Unsigned Byte	(from Alternate Space)
LDUH(LDUHA*)	Load Unsigned Halfword	(from Alternate Space)
LD(LDA*)	Load Word	(from Alternate Space)
LDD(LDDA*)	Load Doubleword	(from Alternate Space)
LDF	Load Floating Point	
LDDF	Load Double Floating Point	
LDDFSR	Load Floating Point Status Register	
LDC	Load Coprocessor	
LDDC	Load Double Coprocessor	
LDDCSR	Load Coprocessor Status Register	
STB(STBA*)	Store Byte	(In Alternate Space)
STH(STHA*)	Store Halfword	(In Alternate Space)
ST(STA*)	Store Word	(In Alternate Space)
STD(STDA*)	Store Doubleword	(In Alternate Space)
STF	Store Floating Point	
STDF	Store Double Floating Point	
STFSR	Store Floating Point Status Register	
STDFQ*	Store Double Floating Point Queue	
STC	Store Coprocessor	
STDC	Store Double Coprocessor	
STCSR	Store Coprocessor Status Register	
STDCQ*	Store Double Coprocessor Queue	
LDSTUB(LDSTUBA*)	Atomic Load/Store Unsigned Byte	(In Alternate Space)
SWAP(SWAPA*)	Swap r Register With Memory	(In Alternate Space)
Arithmetic, Logical and Shift		
ADD(ADDcc)	Add	(Modify icc)
ADDX(ADDXcc)	Add With Carry	(Modify icc)
TADD(TADDccTV)	Tagged Add and Modify icc	(and Trap on Overflow)
MULScc	Multiply Step and Modify icc	
AND(ANDcc)	AND	(Modify icc)
ANDN(ANDNcc)	AND NOT	(Modify icc)
OR(ORcc)	OR	(Modify icc)
ORN(ORNcc)	OR NOT	(Modify icc)
XOR(XORcc)	Exclusive OR	(Modify icc)
XNOR(XNORcc)	Exclusive NOR	(Modify icc)
SLL	Shift Left Logical	
SRL	Shift Right Logical	
SRA	Shift Right Arithmetic	
SETHI	Set High 22 Bits of r Register	
SAVE	Save Caller's Window	
RESTORE	Restore Caller's Window	

INPUTS	OPERATION
Control Transfer	
Bicc	Branch on Integer Condition Codes
FBicc	Branch on Floating Point Condition Codes
CBccc	Branch on Coprocessor Condidition Codes
CALL	Call
JMPL	Jump and Link
RETT	Return from Trap
Ticc	Trap on Integer Condition Codes
Read/Write Control Registers	
RDY	Read Y Register
RDPSR	Read Processor State Register
RDWIM	Read Window Invalid Mask
RDTBR	Read Trap Base Register
WRY	Write Y Register
WRPSR*	Write Processor State Register
WRWIM*	Write Window Invalid Mask
WRTBR*	Write Trap Base Register
UNIMP	Unimplemented Instruction
IFLUSH	Instruction Cache Flush
FP and Coprocessor Ops	
FPop	Floating Point Operations
CPop	Coprocessor Operations
* Privileged instuctions	

Table 3-1 Instruction Set Summary

CY7C601A Internal Registers

The IU contains two types of registers, working registers (or r registers) and control registers. The r registers are used for storage by processes and the control registers are used to track and control the state of the IU. The r registers are contained within a large register file containing one hundred and thirty-six 32-bit registers. Eight of these are global registers and are always accessible to a program, and the remaining registers are accessed via register windows.

Figure 3-2 shows the register window organization

The register file contains eight register windows, and each window contains twenty-four working registers. Each register window is divided into three sections called *ins*, *outs*, and *locals*. There are eight registers in each of these sections. A window shares its ins and outs with adjacent windows. The outs of the previous window are the ins of the current window, and the outs of the current window are the ins of the next window. The windows are joined together in a circular stack where the outs of the last window are the ins of the first window.

A current window pointer (CWP) in the processor state register keeps track of which window is currently active. The CWP is decremented when the program calls a subroutine that causes the processor to access to the next window, and is incremented when the processor returns to the previous window.

Register windows may be marked as invalid in the WIM register, and interrupts may be enabled to signal when movement into an invalid window is caused by an instruction.

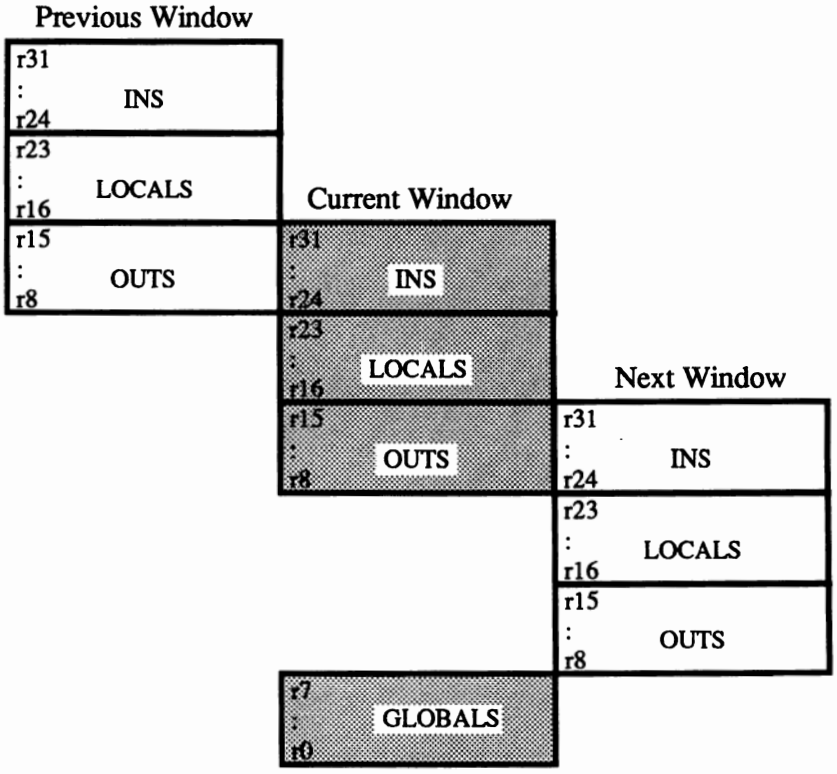


Figure 3-2 Window Register Organization

Control Registers

These include the Processor State Register, the Window Invalid Mask Register, the Trap Base Register, the Y Register, illustrated in Figure 3-3, and the Program Counter.

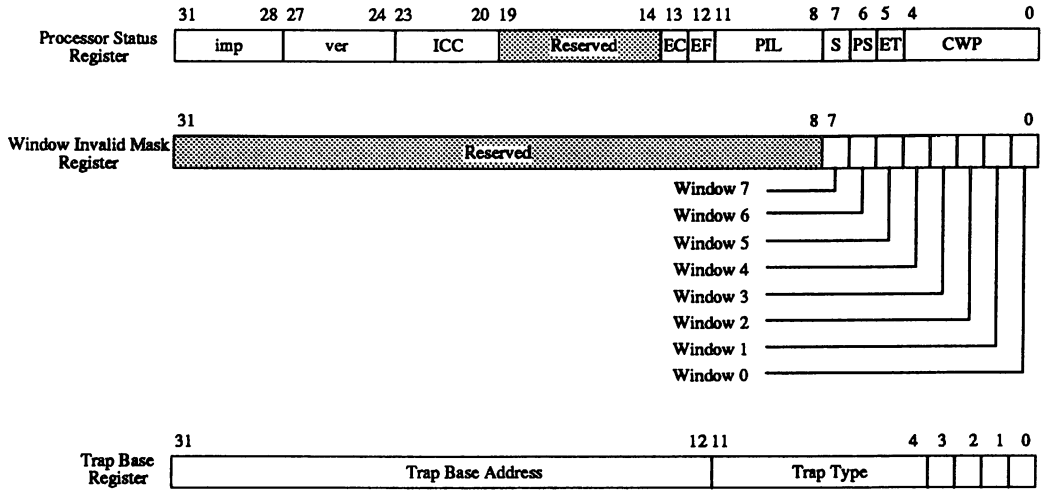


Figure 3-3 IU Control Registers

The Processor Status Register (PSR) contains a number of fields that control the processor and hold status information. The PSR is modified by SAVE, RESTORE, Ticc and RETT instructions, and also by all instructions that modify the condition codes. The privileged instructions RDPSR and WRPSR read and write the PSR directly.

Bits 31:28 impl – identifies the implementation of the IU

Bits 27:24 ver – identifies the version number of the IU

Bits 23:20 icc – contains the IU's condition code and are set by the conditions occurring during integer logic and arithmetic operations:
Bit 23 = Negative flag
Bit 22 = Zero flag
Bit 21 = Overflow flag
Bit 20 = Carry

Bits 19:14 Reserved

Bit 13 EC – Enable Coprocessor. This feature should remain disabled

Bit 12	EF – Enable Floating Point Unit. This bit enables the FPU. An FP instruction will trap if the EF = 0, or if an FPU is not fitted.
Bits 11:8	PIL – identifies the interrupt levels in operation. The IU will only accept interrupts whose level is greater than the contents of this field.
Bit 7	S - Supervisor. This bit determines whether the processor is in supervisor or user mode. S = 1 selects supervisor mode.
Bit 6	PS – Previous Supervisor. This bit indicates the value of the S bit at the time of the most recent trap.
Bit 5	ET – Enable Traps. This bit determines whether traps are enabled. A trap automatically resets ET. When ET = 0, an interrupt request is ignored and an execution trap causes the IU to halt execution, typically resulting in a reset trap.
Bits 4:0	CWP – Current Window Pointer. This field identifies the current window into the r registers. Hardware decrements the pointer when traps are encountered and on SAVE instructions. The pointer is incremented on RESTORE and RETT instructions

Window Invalid Mask Register

The Window Invalid Mask Register (WIM) allows windows to be marked as invalid in the WIM register. Movement into an invalid window caused by a SAVE, RESTORE or RETT instruction will cause a window overflow or underflow interrupt to be generated.

Trap Base Register

The Trap Base Register contains three fields that generate the address of the trap handler when a trap occurs.

Bits 31:12	Trap Base Address. This contains the most significant 20 bits of the trap table address.
Bits 11:4	Trap Type. This field is written by hardware at the time of a trap. It provides an offset into the trap table.
Bits 3:0	Reserved - Should always be written as zeros.

Y Register

The Y Register is used to store the partial product during multi-step instructions.

Program Counter

The Program Counter (PC) contains the address of the instruction being executed. The next Program Counter contains the address of the next instruction to be executed.

Traps and Interrupts

The CY7C600 design supports a full set of traps and interrupts. They are handled by a table that supports 128 hardware and 128 software traps. Even though floating-point instructions can execute concurrently with integer instructions, floating-point traps are precise because the FPU supplies (from the table) the address of the instruction that failed.

The IU supports both asynchronous traps (interrupts) and synchronous traps (error conditions and trap instructions). Traps transfer control to an offset within the trap table. The base address of the table is specified by the Trap Base Register and the offset is a function of the trap type. Traps are taken before the current instruction causes any changes visible to the programmer and can therefore be considered to occur between instructions.

Table 3-2 shows priority levels of the exceptions and traps.

Interrupts from the peripheral devices within the SPARCbook are controlled and prioritized by the Mbus to Pbus interface (MPI) ASIC (see Chapter 5).

Memory Protection

The CY7C600 design provides memory protection, which is essential for smooth multitasking operation. Memory protection makes it impossible for user programs to corrupt the system, other user programs, or themselves.

The IU supports a multitasking operation system by providing user and supervisor modes. Some instructions are privileged and can only be executed while the processor is in supervisor mode. Changing from user to supervisor mode requires taking a hardware interrupt or executing a trap instruction. This instruction execution protection ensures that user programs cannot accidentally alter the state of the machine with respect to its peripherals.

TRAP	PRIORITY	OFFSET
Reset	1	??
Instruction access exception	2	0x01
Illegal Instruction	3	0x02
Privileged Instruction	4	0x03
FP Disabled	5	0x04
CP Disabled	5	0x24
Window Overflow	6	0x05
Window Underflow	7	0x06
Unaligned Memory Address	8	0x07
FP Exception	9	0x08
CP Exception	9	0x28
Data Access Exception	10	0x09
Tag Overflow	11	0x0A
Trap Instruction	12	0x80 0xFF
Interrupt Level 15 – Microcontroller/Floppy Disk	13	0x1F
Interrupt Level 14 – Battery Power Low	14	0x1E
Interrupt Level 13 – MPI (Counter/timer 0)	15	0x1D
Interrupt Level 12 – MPI (Counter/timer 1)	16	0x1C
Interrupt Level 11 – MPI (Keyboard buffer)	17	0x1B
Interrupt Level 10 – Mouse	18	0x1A
Interrupt Level 9 – Serial Port	19	0x19
Interrupt Level 8 – Modem	20	0x18
Interrupt Level 7 – Ethernet Interface	21	0x17
Interrupt Level 6 – VGA Controller	22	0x16
Interrupt Level 5 – Hard Disk Interface	23	0x15
Interrupt Level 4 – MPI (FIFO Full/Empty)	24	0x14
Interrupt Level 3 – MPI (FIFO Op Complete)	25	0x13
Interrupt Level 2 – Microcontroller/Floppy Disk	26	0x12
Interrupt Level 1 – Centronics Interface	27	0x11
Implementation Dependent Exceptions	Dependent	0x60 - 7F

Table 3-2 Trap Table

FLOATING POINT UNIT

The FPU provides high-performance, IEEE STD-754 compatible single- and double-precision floating-point calculations for CY7C600 systems and is designed to operate concurrently with the IU. All address and control signals for memory accesses by the FPU are supplied by the IU.

Overview

The IU and FPU are able to execute instructions simultaneously. The FPU performs all floating-point calculations with its own set of registers and arithmetic and logic unit.

Floating-point instructions are addressed by the IU, and are simultaneously latched from the data bus by both it and the FPU. Floating-point instructions are concurrently decoded by the IU and the FPU, but do not begin executing in the FPU until after the instruction is enabled by a signal from the IU. Pending and currently executing FP instructions are placed in an on-chip queue while the IU continues to execute non-floating-point instructions.

Floating Point Registers

The FPU contains thirty-two 32-bit floating-point F registers. These form a 32 x 32-bit register file. The contents of these registers are transferred to and from external memory under control of the IU using floating-point load/store instructions. Addresses and control signals for data accesses during a floating-point load or store are supplied by the IU, while the FPU supplies or receives the data.

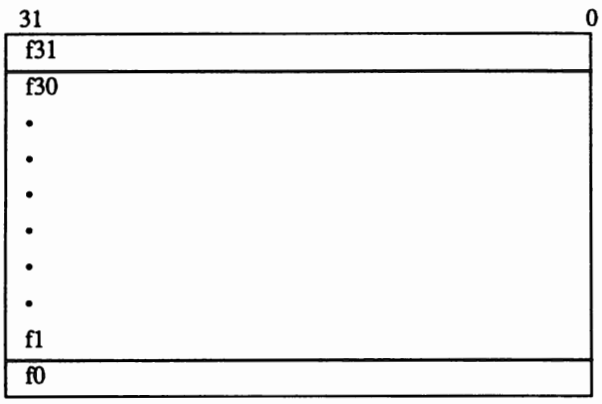


Figure 3-4 F Registers

Although the FPU operates concurrently with the IU, a program containing floating-point computations generates results as if the instructions were being executed sequentially.

Figure 3-4 illustrates the F registers.

A single F register is able to one single-precision operand, two are required to hold a double precision operand, and four are required to hold and extended precision operand.

Floating-Point State Register

The floating-point status register (FSR) contains mode and status information about the FPU. It is read and written using the STFSR and LDFSR. Figure 3-5 illustrates the FSR.

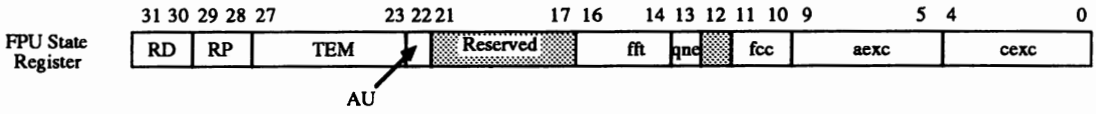


Figure 3-5 FSR

Bits 31:30 RD – Rounding Direction. These select the rounding direction for floating-point results:
00 = Nearest
01 = Zero
10 = +∞
11 = –∞

Bits 29:28 RP – Extended Rounding Precision
00 = Extended
01 = Single
10 = Double
11 = Unused

Bits 27:23 TEM – Trap Enable Mask. These are enable bits for each of the five floating-point exceptions:
Bit 27 = Invalid Operation Trap Mask
Bit 26 = Overflow Trap Mask
Bit 25 = Underflow Trap Mask
Bit 24 = Divide by Zero Trap Mask
Bit 23 = Inexact Trap Mask

Bit 22 AU – Abrupt Underflow. When set, this bit causes denormalized FP operands or results to be rounded to zero.

Bits 21:17 Reserved

Bits 16:14 ftt – Floating-Point Trap Type. These indicate the type of exception:
0 = None
1 = IEE E Exception
2 = Unfinished FP operation
3 = Unimplemented FP operation
4 = Sequence Error
5-F = reserved

Bit 13 qne – Queue Not Empty. When set, this bit indicates that the FP queue is not empty following an FP exception or STDFQ instruction was executed.

Bit 12 Reserved

Bits 11:10 fcc – FPU Condition Codes. These bits are updated by FP compare instructions:
00 = fsr1 = fsr2
01 = fsr1 < fsr2
10 = fsr1 > fsr2
11 = fsr1 ? fsr2 (unordered)

Bits 9:5 aexc – Accrued Exception Bits.
Bit 9 = Accrued Invalid Operation
Bit 8 = Accrued Overflow
Bit 7 = Accrued Underflow
Bit 6 = Accrued Divide by Zero
Bit 5 = Accrued Inexact

Bits 4:0 cexc – Current Exception Bits
Bit 4 = Current Invalid Operation
Bit 3 = Current Overflow
Bit 2 = Current Underflow
Bit 1 = Current Divide by Zero
Bit 0 = Current Inexact

CACHE CONTROLLER AND MEMORY MANAGEMENT UNIT

The interface between the bus, on which the IU, FPU and cache memory reside, and the Mbus, on which the main memory and peripheral interfaces reside, is provided by a CY7C604 Cache Controller and Memory Management Unit (CMU). This device provides translation between the 32-bit virtual addresses used on the CPU module and 36-bit physical addresses used on the Mbus. It provides hardware support for a demand-paged virtual memory environment for the IU. The Page size is fixed at 4Kbytes. The CMU also manages a 64Kbyte cache memory.

Memory Management Unit

The memory management unit (MMU) conforms to the standard SPARC architecture definition for memory management.

The MMU provides virtual to physical address translation using a translation lookaside buffer (TLB). This is an area of specialized on-chip memory containing sixty-four Page Table Entries (PTE) as shown in Figure 3-6.

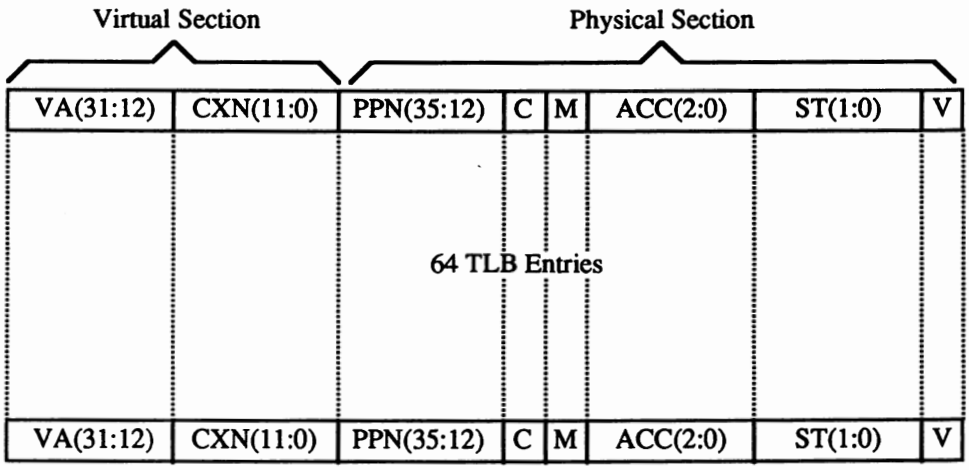


Figure 3-6 Translation Lookaside Buffer

Each entry within the TLB contains a virtual section and a physical section. The virtual section comprises a virtual address field, for comparison with VA(31:12), and 12-bit context field, for comparison with the contents of the context register.

The physical section contains the physical address field, PPN(35:12), a cache control bit, a modified bit, a 3-bit access protection field, a translation control field, and a valid bit.

Bits (31:12) of the virtual address are translated and expanded to physical address bits (35:12). During an access by the IU, the virtual address supplied by the IU and the contents of the context register are compared with the virtual section of all sixty-four entries simultaneously. When a match is found (or a "hit" occurs), the physical address field PPN(35:12) supplies the address of a 4Kbyte page in memory. Virtual address bits A(11:00) from the IU are passed through unchanged to supply a byte offset. This is illustrated in Figure 3-7. Each hit TLB entry is checked for memory protection attributes automatically and violations are reported to the IU as memory exceptions.

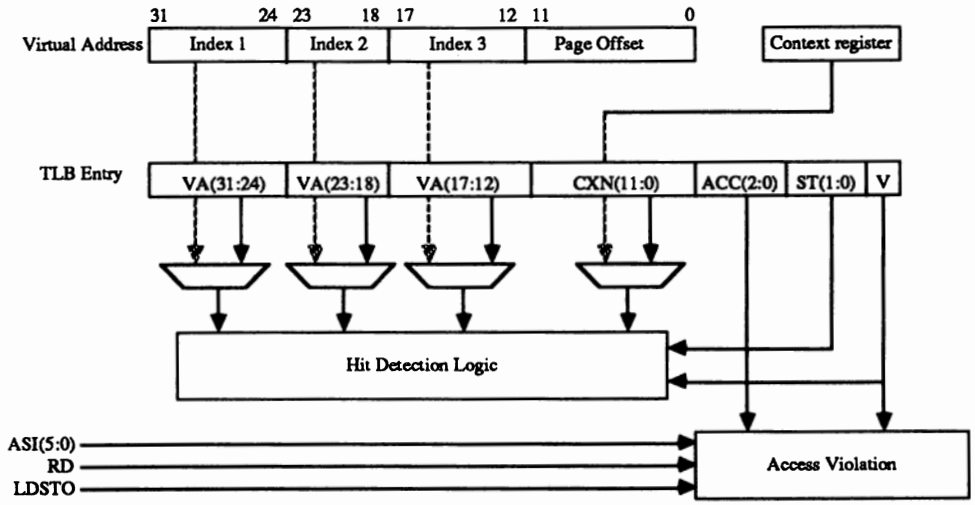


Figure 3-7 Address Translation

If the virtual address from the IU does not match an entry in the TLB, the CMU automatically performs a search (or table walk) through a translation table in main memory to obtain an address translation.

The translation table forms a tree structure in main memory as illustrated in Figure 3-8. The Context Pointer register provides a pointer to the context table, and the context register provides an index to the Root Pointer, which in turn points to a level 1 page table. Index 1 from the virtual address selects an entry within Level 1 pointing to a level 2 page table, where Index 2 selects a pointer to a level 3 table. Index 3 then selects one of the entries in the level 3 table which should point to a 4Kbyte memory page.

When a page table pointer (PTP) is encountered within the tables, the search continues to the next lower level. If a page table entry (PTE) is found, the search is terminated and the entry is stored in the TLB. If no PTE is found at all, a synchronous fault exception is signalled to the IU.

The PTE provides a pointer to a physical page while the lower 12 bits of the virtual address provides an offset.

The level at which a table walk terminates (i.e. a PTE is found) is related to the size of addressing region associated with the entry. A table walk which finds a PTE in the context table corresponds to a region of 4Gbytes. A PTE in level 1 corresponds to a 16Mbyte region, or 256Kbyte in level 2, or 4Kbyte in level 3. The virtual address bits not used to index table entries are used to supply the page offset.

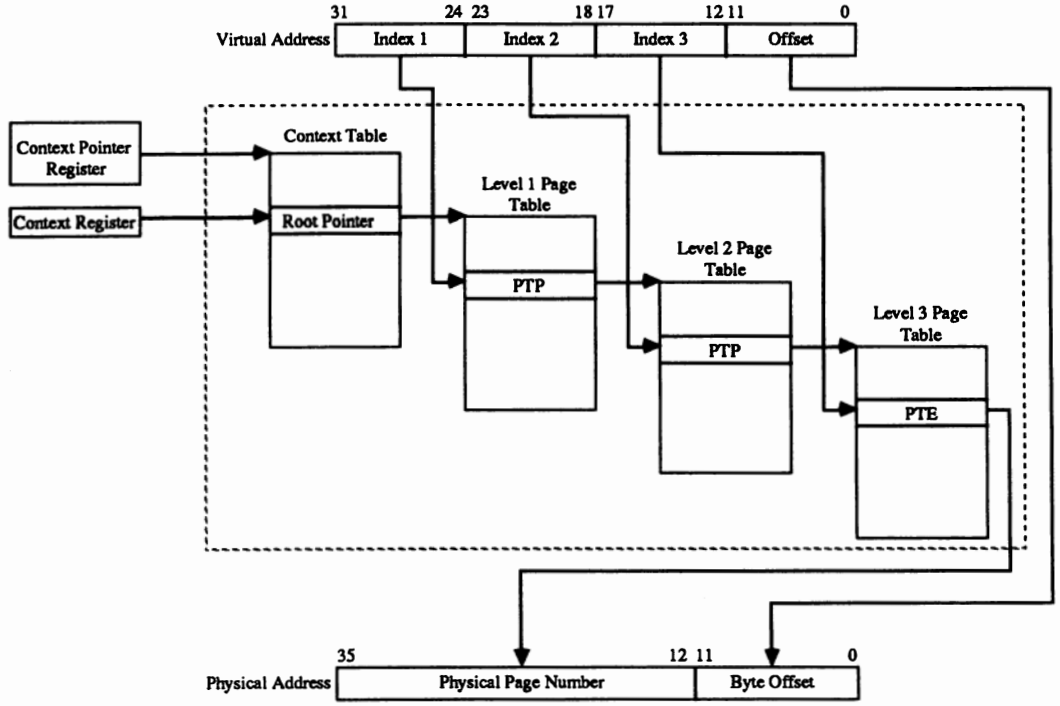


Figure 3-8 A Three Level Table Walk in Memory

Figure 3-8 shows a table walk which uses three page table levels. In this example A(31:12) from the virtual address are used to index the page tables, and A(11:0) supply an offset address into the selected memory page

Figure 3-9 shows a table walk which terminates at level 2. In this case A(31:18) are used as index bits, and A(17:0) provide an offset address into the selected page.

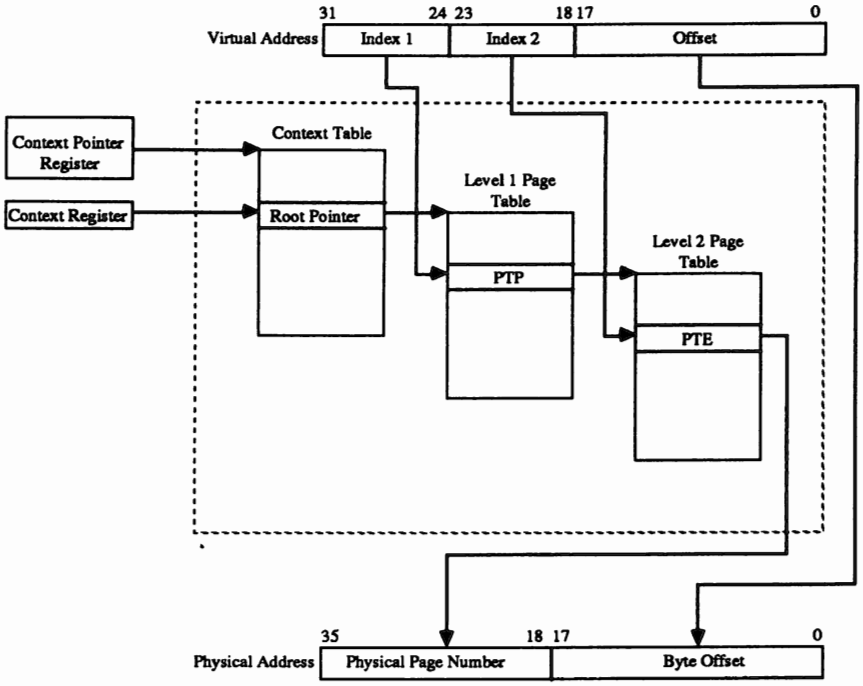


Figure 3-9 A TwoLevel Table Walk in Memory

Cache Controller

The cache controller provides access control for the 64Kbyte cache memory implemented on the CPU module.

The cache is organized with 2048 cache lines of 32bytes each, as illustrated in Figure 3-10. The CMU provides on-chip storage for 2048 cache address tags, one for each line. The tag entries can be directly written or read by the processor.

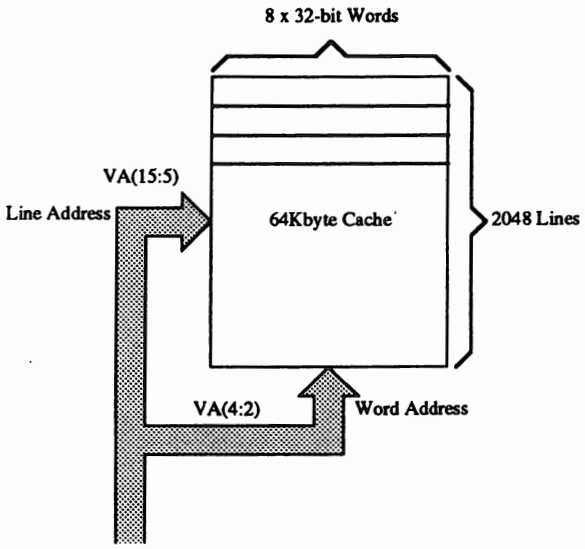


Figure 3-10 Cache Memory Organization

During an IU read operation, virtual address bits VA(15:5) select one of the 2048 lines and, at the same time, one of the cache tags maintained by the CMU. If the virtual address within a cache tag matches with VA(15:5), and the context field matches with context register, a hit occurs. The cache line represented by the valid tag is selected by the CMU and one or more of the 32bytes in the cache line are selected by VA(4:0) from the IU. If a miss occurs the CMU reads in a new cache line from main memory. Burst accesses are used by the CMU to fill cache lines. Burst accesses are always 64-bits wide and always consist of an address phase followed by four data transfer phases (32bytes in total). The IU is held until the CMU has loaded the new cache line, but this is completely transparent to the programmer.

Figure 3-11 illustrates cache tag comparison by the CMU.

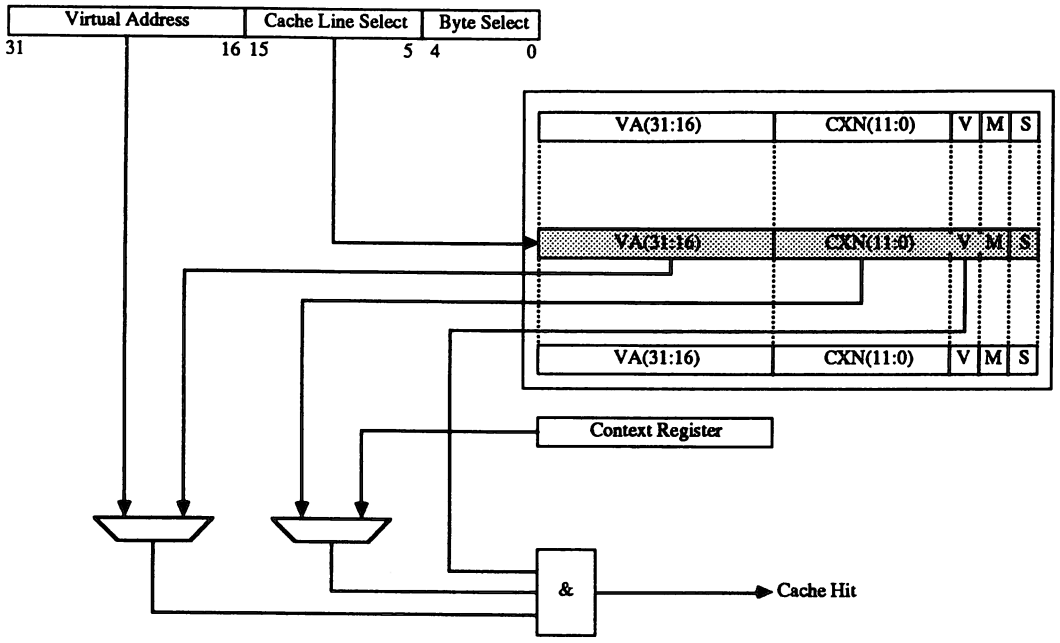


Figure 3-11 Cache Tag Comparison

CY7C157A CACHE STORAGE UNIT

The CY7C157A cache storage unit is a 16Kbyte x 16 SRAM CSU designed to interface easily to a CY7C600 processor and provide maximum performance. The CSU is a specialized device which has registered address inputs, and latched data inputs and outputs as well as a self-timed write pulse.

The device has a single clock that controls loading of the address registers, data input latches, data output latches, pipeline control latch, and chip enable register. The chip enable is clocked into a register and pipeline through a control register to condition the output enable. This pipeline design allows a cache that works as an extension of the internal instruction pipeline of the IU.

Figure 3-12 shows the internal architecture of a CY7C157A CSU chip.

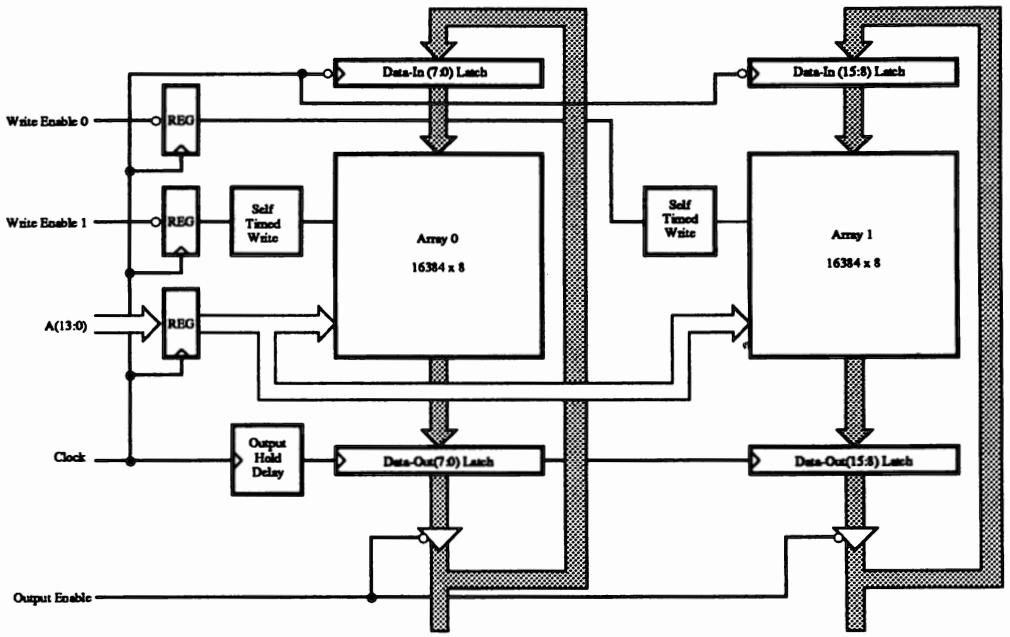


Figure 3-12 CSU Internal Architecture

CHAPTER FOUR
MAIN MEMORY

The SPARCbook may be manufactured with either 8Mbytes or 32Mbytes of main memory which provides a 64-bit wide storage for the SPARC CPU. The SPARCbook's Base board is always fully populated with DRAM chips. It is fitted with an array of sixteen 1Mbit x 4 DRAMs, if equipped with 8Mbytes of memory, or the same number of 4Mbit x 4 DRAMs if equipped with 32Mbytes. Field upgrades are not possible. Two additional DRAMs, of the same size as those used in the main memory array, provide storage for parity bits generated for each byte location.

Figure 4-1 shows the architecture of the main memory.

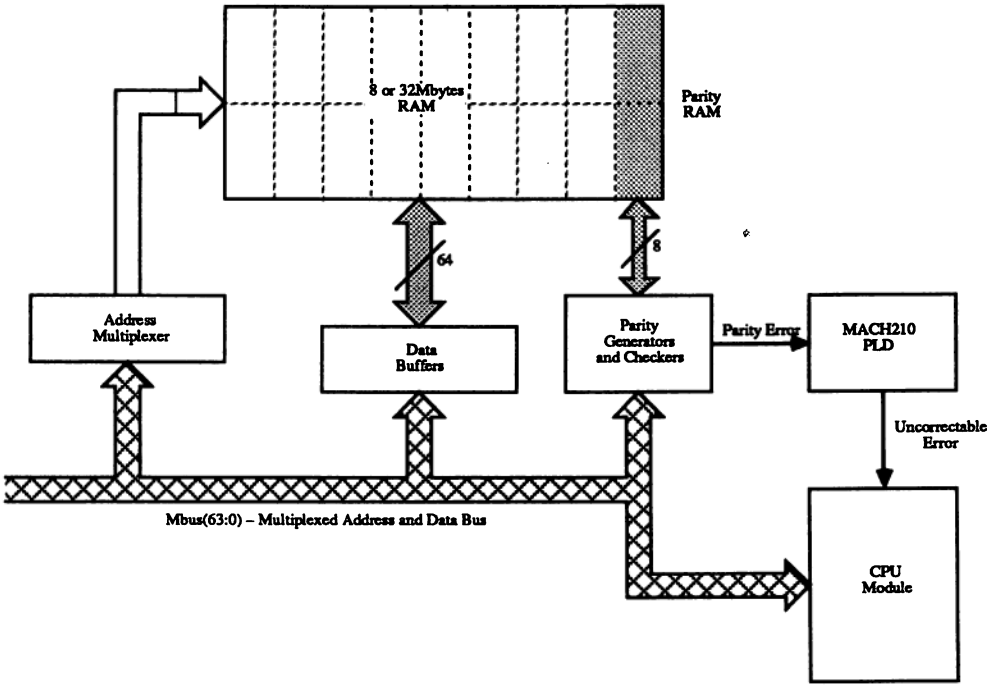


Figure 4-1 SPARCbook Main Memory

FUNCTIONAL DESCRIPTION

Memory Accesses
All main memory accesses in the SPARCbook are initiated by the SPARC CPU. It supports single cycle reads and writes of 8, 16, 32 and 64bits, and also burst cycles.

Access Control and memory refresh for the main memory array is provided by a powerful programmable logic device (PLD) containing the equivalent of 1800 logic gates. This, in conjunction with a custom 32-bit buffer chip, provides full addressing support for all of the SPARC CPU's memory access cycles.

Parity Checking
During writes to DRAM, parity bits are generated for each byte location written to. When the data is read out again, the parity bits and the stored data are checked. Errors in any byte lane are signalled to the PLC which in turn signals an uncorrectable error to the SPARC CPU. Parity checking is enabled via the microcontroller (see Chapter 6), and parity error interrupts are enabled via the MPI (see Chapter 5).

CHAPTER 5
MBUS TO PBUS INTERFACE

The SPARCbook baseboard is based on a dual bus architecture (see Chapter 2). The SPARC CPU and main memory are located on the SPARC architecture standard 64-bit Mbus, and the peripheral controllers are located on the 16-bit PC-AT style Pbus.

In the SPARCbook, transactions between these two buses are controlled by the Mbus to Pbus Interface controller (MPI). This is an application specific integrated circuit (ASIC) designed by Tadpole Technology.

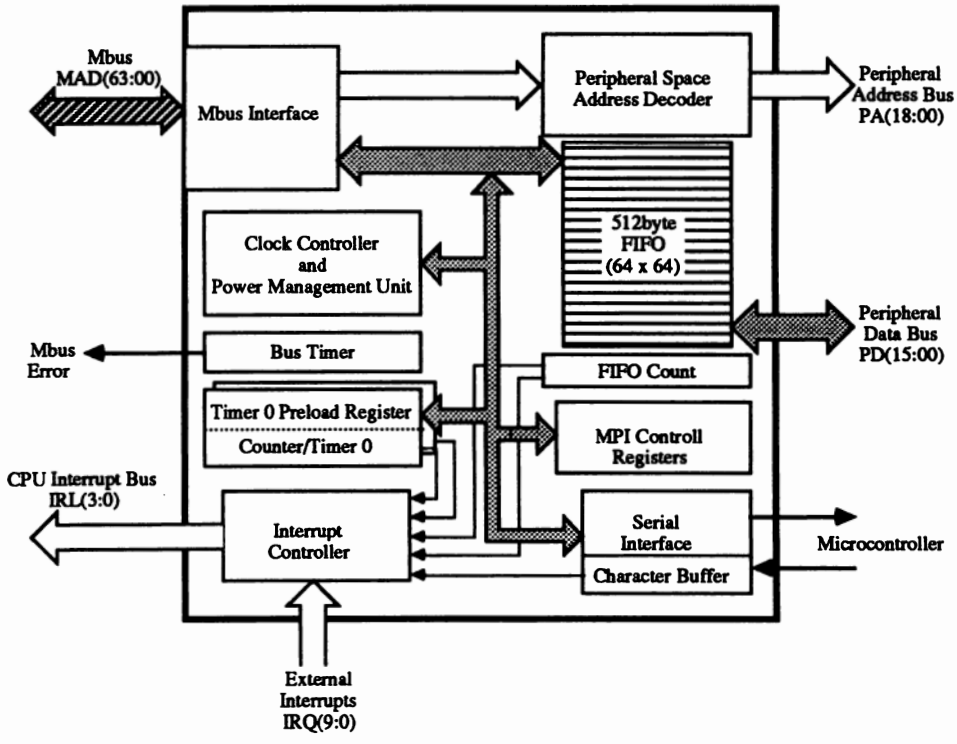


Figure 5-1 MPI Architecture

INTRODUCTION TO THE MPI

The Mbus and Pbus have very diverse characteristics so that interactions between them are complex. The Mbus is a 64-bit multiplexed data and address bus which operates at 25MHz and is capable of sustaining a data transfer rate of 40Mbyte/s in and out of main memory. The SPARC CPU also uses part of the Mbus to supply control signals during the address phase of a transfer operation.

The Pbus uses separate address and data buses, and separate control lines to control transfers. It is also very much slower than the Mbus.

The MPI contains data packing registers which allow direct 64-bit accesses by the SPARC CPU to the 8- and 16-bit peripheral controllers on the Pbus by converting 8-, 16-, 32- and 64-bit accesses on the Mbus into the appropriate number of single cycle accesses on the Pbus. It also contains a 512byte FIFO buffer which extends this access conversion to support Mbus burst cycles.

The MPI provides complete address decode facilities for the Pbus (the peripheral address space) and converts the control information from the Mbus into the Pbus control signals necessary to support accesses to the AT style peripheral controllers employed in the SPARCbook. In addition, the MPI provides the SPARCbook with a 15 level interrupt controller and two 16-bit counter/timers

PERIPHERAL BUS INTERFACE

The MPI is the only Pbus master device in the SPARCbook, no other device is able to initiate a data transfer cycle on the Pbus. Data may be transferred to and from the Pbus using 8- or 16-bit accesses by one of two independent paths. The first is a direct access path into the peripheral address space, and the second is via the internal FIFO.

Direct Operation

The direct access path into the peripheral address space takes a single transfer of 8, 16, 32 or 64bits from the host bus and performs the corresponding number of transfers required on the Peripheral bus. The Mbus transfers and corresponding Pbus transfers supported are shown in Table 5-1.

On a write cycle, the MPI allows write-posting from the Mbus. It latches the data presented into internal buffers and completes the Mbus transaction immediately, allowing the SPARC CPU to use the Mbus for main memory accesses. In the mean time, the MPI performs the appropriate number of write cycles to the addressed peripheral. The only possible errors on write-posted cycles are accesses by the CPU to non-responding or illegal locations, which are immediately decoded by the MPI and signalled as an uncorrectable Mbus error.

When the SPARC CPU executes a read operation to the peripheral address space, the MPI holds the Mbus until the requested data has been read from the Pbus into the internal buffers and then completes the Mbus cycle. Up to eight Pbus read operations (for a single 64-bit Mbus operation) are carried out for each Mbus read operation.

MBUS TRANSFER	PORT SIZE	PBUS TRANSFER
Byte on byte boundary	Byte	1 Byte transfer
Byte on byte boundary	Halfword	1 Byte transfer (lower or upper)
Halfword on halfword boundary	Byte	2 Byte transfers
Halfword on halfword boundary	Halfword	1 Halfword transfer
Word on word boundary	Byte	4 Byte transfers
Word on word boundary	Halfword	2 Halfword transfers
Doubleword on doubleword boundary	Byte	8 Byte transfers
Doubleword on doubleword boundary	Halfword	4 Halfword transfers
Note: Halfword = 16bits, Word = 32bits, Doubleword = 64bits		

Table 5-1 Conversion of Mbus Transfers to Pbus Transfers

FIFO Operations

The second means of data transfer to and from the host bus is by using the MPI's internal FIFO. Mbus transfers to and from the FIFO port are only permitted using 64-bit transfers.

FIFO operations are initiated by the SPARC CPU. To execute a FIFO operation, the SPARC CPU must set up control registers in the MPI to specify the following parameters:

- **The Peripheral address space start address**
- **The number of bytes to be transferred**
- **The direction of transfer (read or write)**
- **Incrementing or static Pbus addressing**
- **Pbus DMA (floppy disk operations only) or polled operation**

A FIFO operation is started by setting the FSTART bit in the FIFO control register.

Read Transactions
After the FIFO control registers have been used to initiate a read operation, the MPI executes a series of read operations on the Pbus, transferring the data into the FIFO. The operation is complete when either the FIFO becomes full or the byte count in the FCOUNT register reaches zero, signifying that the specified number of bytes have been transferred. The FIFO full and FCOUNT equals zero conditions are signalled to the SPARC CPU with interrupts.

In polled mode operations, the data transfers proceeds at the fastest rate permitted by the peripherals. In DMA mode, the DRQ/DACK lines are used to control the peripheral bus transfers. This latter mode is intended for floppy disk transfers.

Once in the FIFO, the data may be read out by the SPARC CPU using 64-bit single cycle or burst transfers. While data may be read from the FIFO as soon as it is available, the SPARC CPU should not read more data than has been read from the Pbus (this can be checked by using the FCOUNT status register). Reading more data than is available in the FIFO will result in a FIFO Empty interrupt (if it is enabled), and undefined data being returned.

Once the required number of bytes have been read by the CPU, Pbus transfers are stopped and the host is interrupted (under software control).

Write Transactions
For a write operation the host may write up to 512bytes into the FIFO data register using 64-bit single or burst transfers. Bursts of up to 32bytes (four Mbus data transfer cycles) are supported without wait states.

The FIFO Control register contains full and empty flags which may be programmed to generate interrupts. The MPI may be programmed to interrupt the CPU when the FIFO Full flag or FIFO Empty flag in the FIFO Control register become set.

The FIFO Start bit in the FIFO Control register must be set in order to allow a FIFO operation to proceed. Then, once the CPU writes data into the FIFO the corresponding Pbus transactions commence, beginning at the address specified in the FIFO Start Address register.

A special operating mode allows data in the FIFO to be written directly to the peripheral bus (a write operation with no data supplied by the SPARC CPU).

NOTES:

1 FIFO transfers may not be carried out over a 1Kbyte peripheral address boundary.

2 At any time the FIFO may be reset by asserting a bit in the System Reset Register. This causes the FIFO pointers and counters to be reset and creates an empty FIFO.

3 Mbus transfers other than those in Table 5-1 (eg misaligned transfers) are not supported.

Peripheral Space Address Decoder

All addresses for the Pbus are decoded by the MPI. The MPI divides the Pbus address space into six 2Mbyte regions, resulting in the address map shown in Figure 5-2.

In four regions the MPI asserts the chip select for one device, while the remaining two are assigned to provide one 8-bit region and one 16-bit Pbus I/O region. Accesses to non-responding locations will return undefined data on reads and write to undefined locations on writes.

The MPI does not provide a chip select signal for the 82C710 UPC, which is used to provide control of the disk and external mouse interface. This device is accessible at various locations within the 8- and 16-bit access regions (see Chapter 7).

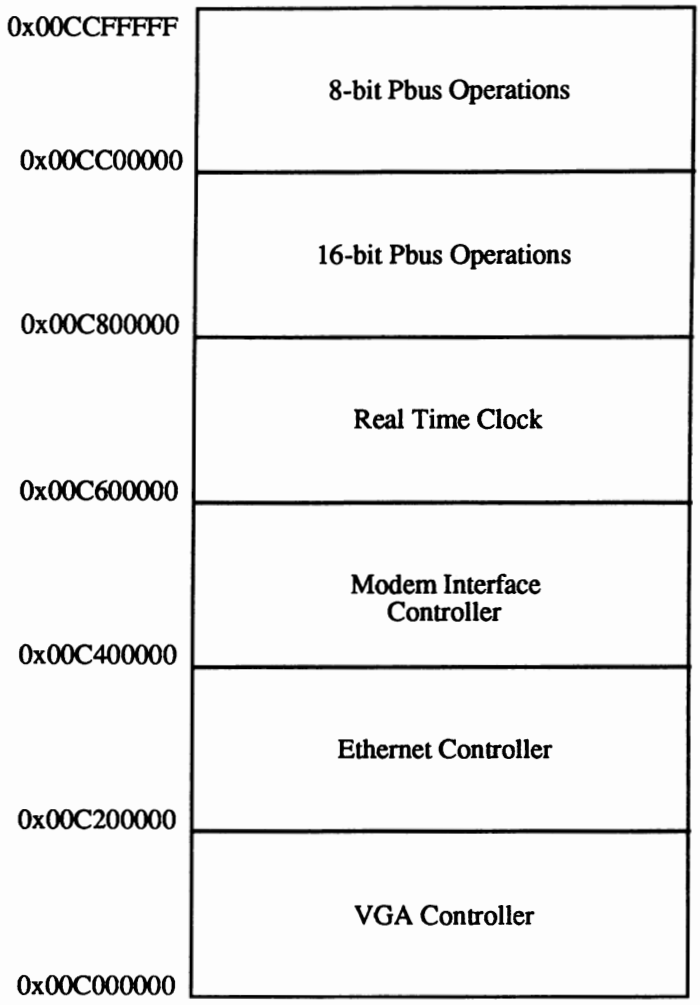


Figure 5-2 Pbus Memory Map

COUNTER-TIMERS

The MPI incorporates two general purpose 32-bit counter/timers, CT0 and CT1 for software use, and a bus timeout timer which prevents the SPARCbook from stalling due to an access to a non-responding location. These are illustrated in Figure 5-3.

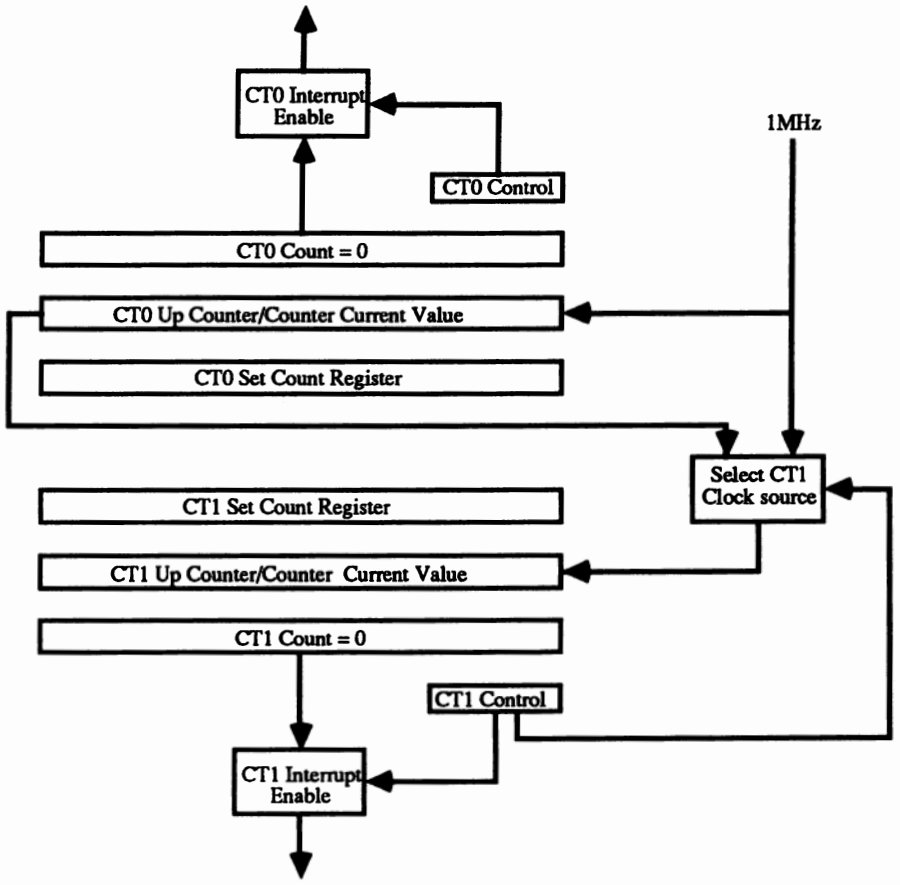


Figure 5-3 MPI Counter Timers

System Timers

The two general purpose counter/timers each consist of a 32-bit up counter, a 32-bit time constant register and a counter control register. These are clocked from a 1MHz clock, internally generated by the MPI, giving a timing resolution of 1µs. The timers count up and may be used to generate an interrupt after a programmable delay.

The timers can be chained to create a single 64-bit counter/timer. In this mode of operation, Timer 0 supplies the timing for Timer 1.

The timers may be programmed to operate in either one-shot mode or continuous mode. In one-shot mode the timer counts up from a preloaded value and completes the count when it reaches one after passing zero.

In continuous mode, the timer is reloaded with the preset count value each time it reaches zero and continues counting, generating a periodic interrupt each time zero is reached.

Further information is provided in the register descriptions in this chapter.

Bus Timeout Timer

The MPI also contains a bus timeout timer (see Interrupts below). This timer monitors all Mbus transactions. Any Mbus transaction that is not completed after detection of four CLK32K clocks causes termination of the cycle and an Mbus Error. This corresponds to a timeout of approximately 125µs. The bus timeout timer has a fixed timeout period and signals an Mbus error if this period expires.

This facility is used for all Mbus cycles and is not limited to accesses to the MPI control registers or to the peripheral address space.

SERIAL I/O PORT

The MPI contains a single channel serial interface which operates in conjunction with the microcontroller to provide an interface to the keyboard.

Serial Input

A single serial input line receives asynchronous data at 976.6 baud (1.024ms period) from the keyboard microcontroller. Data is formatted as 8 data bits with one start bit (0) and one stop bit (1), with the least significant bit transferred first. No other data formats are permitted. Up to three characters are buffered by the MPI. An interrupt may be generated if there are any received characters in the buffer.

Serial Output

A single serial output line transmits asynchronous data at 976.6 baud to the keyboard microcontroller. Data is formatted as 8 data bits with one start bit (0) and one stop bit (1). No other data formats are permitted. A status register contains a busy bit. If the busy bit is not asserted the host may write a single byte to the transmit register. The busy bit will be asserted until the transmitter buffer becomes available again.

POWER MANAGEMENT FUNCTIONS

The MPI incorporates a power management unit (PMU) which controls the SPARCbook internal clocks and power supply to the peripheral controller devices.

PM1 is used to switch off the main power to the system.

Clock Control

The MPI controls the main Mbus clock, MCLK0, to which all Mbus transactions are synchronized. The MPI performs the following functions using the input reference 25MHz.

Normal Operation
MCLK0 & MCLK1 are generated from CLK25M and operate at 25MHz. A 1MHz timing reference for the counter/timers is derived from CLK25M.

Slow Operation
MCLK0 & MCLK1 are generated from CLK25M but are reduced to 12.5MHz. The timing reference for the counter/timers remains unchanged.

Stop Operation
MCLK0 is stopped in the high state at the end of the cycle after the deassertion of /MRDY. MCLK1 is output at 12.5MHz. The timing reference for the counter/timers is unchanged.

Pause Operation
MCLK1 is not affected (remains at 12.5 or 25MHz). MCLK0 can be stopped temporarily in one of two software selectable (via the relevant CLK_CTL register bits) modes. The first is the INTPAUSE mode. In this mode the MCLK0 signal is held high (stopped) from the clock after the MAS strobe until the clock before the MRDY or MERR cycle termination for accesses which are to the MPI.

In the second, EXTPAUSE mode, the MCLK0 is held high (stopped) for a programmable number of clocks from 0 (default) to 7 after the MAS cycle when the MPI is not being addressed. This is used to conserve power during initial memory accesses. Note that in the INTPAUSE mode the Mbus timeout will still operate to terminate the cycle if required.

INTERRUPT CONTROLLER

The MPI incorporates an interrupt controller. This consists of a priority encoder and Interrupt Mask registers. The interrupt controller is illustrated in Figure 5-4.

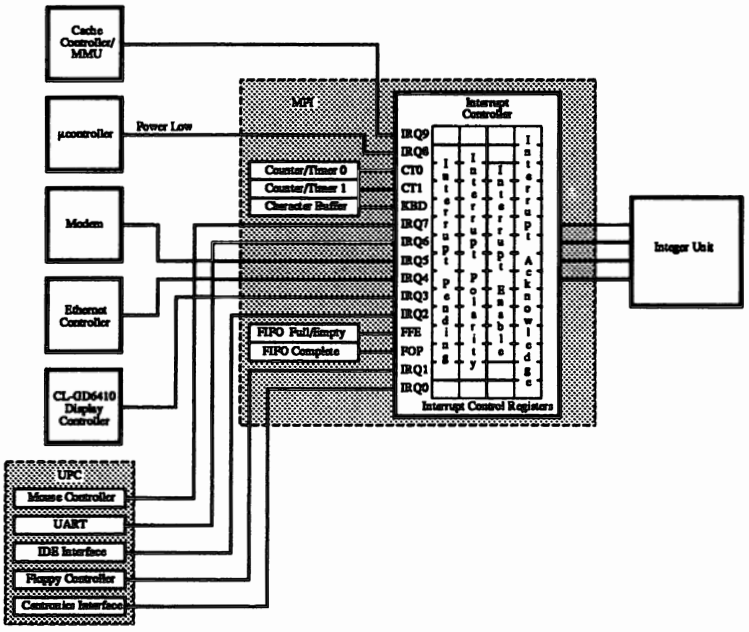


Figure 5-4 SPARCbook Interrupt Architecture

Table 5-2 shows the interrupt sources from the SPARCbook's devices and from within the MPI itself.

The MPI performs a simple priority encoding of the sixteen interrupt sources to four interrupt request lines to the SPARC CPU, IRL(3:0). The MPI external interrupts are all level sensitive, and may be individually programmed to be active high or active low. Each interrupt may be individually enabled or masked. At power up all interrupts are masked.

After an Mbus clock stop operation (under CPU control), the activation of any non-maskable interrupts will automatically restart the MCLK clock in the same mode (25MHz or 12.5MHz) as when it was halted.

IRQ9 is a special case. It is latched internally within the MPI when it has been asserted for one clock period. The latched signal is fed into the interrupt priority circuit. The latch (and the interrupt) is cleared by writing to the appropriate Interrupt Acknowledge bit.

LABEL	SOURCE	IRL(3:0)	DEVICE
IRQ9	External	1111	CMU
IRQ8	External	1110	Power Low/Floppy Disk
CT0	Internal	1101	Counter Timer 0
CT1	Internal	1100	Counter Timer 1
KBD	Internal	1011	Keyboard Buffer
IRQ7	External	1010	External Mouse
IRQ6	External	1001	Serial
IRQ5	External	1000	Modem
IRQ4	External	0111	Ethernet
IRQ3	External	0110	Display
IRQ2	External	0101	Hard Disk
FFE	Internal	0100	FIFO Full/Empty
FOP	Internal	0011	FIFO Operation complete (FCOUNT=0)
IRQ1	External	0010	Power Low/Floppy Disk
IRQ0	External	0001	Centronics
		0000	No Interrupt

Table 5-2 SPARCbook Interrupt Sources

INTERNAL REGISTERS

The MPI contains twenty control registers which affect the activity of the elements that the MPI contains. In addition to the control registers, the MPI provides a twenty first location which is used to move data into (for writes), and out of (for reads), the FIFO.

Table 5-3 summarizes the internal address map of the MPI. It provides the address, label, name and size of each register. The size refers to the data width that must be used when accessing the register.

ADDRESS	LABEL	NAME	ACCESS	SIZE
0x00D000000	MERR_STATUS	Mbus Error Status	RO	8-bit
0x00D000010	CT0_TIME	Set Count CT0	R/W	32-bit
0x00D000018	CT0_VALUE	Current Count CT0	RO	32-bit
0x00D000020	CT0_CONTROL	Control CT0	R/W	8-bit
0x00D000028	CT1_TIME	Set Count CT1	R/W	32-bit
0x00D000030	CT1_VALUE	Current Count CT1	RO	32-bit
0x00D000038	CT1_CONTROL	Control CT1	R/W	8-bit
0x00D000050	INT_PEND	Interrupt Pending	RO	32-bit
0x00D000058	INT_POLARITY	Interrupt Polarity	R/W	32-bit
0x00D000060	INT_ENABLE	Interrupt Enable	R/W	32-bit
0x00D000068	INT_ACK	Interrupt Acknowledge	WO	32-bit
0x00D000070	KBDDATA	Serial Receive Port	RO	8-bit
0x00D000088	KBDSTATUS	Serial Receive Status	RO	8-bit
0x00D000090	TXDATA	Serial Transmit Port	WO	8-bit
0x00D000098	TXSTATUS	Serial Transmit Status	RO	8-bit
0x00D0000A0	RESET	Reset	R/W	8-bit
0x00D0000A8	CLK_CTL	Clock Control	R/W	8-bit
0x00D0000B8	PM_PORT	Power Down	WO	8-bit
0x00D0000C0	FIFO_START	FIFO Start Address	R/W	32-bit
0x00D0000C8	FIFO_CONTROL	FIFO Control	R/W	32-bit
0x00D0000F8	FIFO_DATA	FIFO Data	R/W	64-bit only

Table 5-3 MPI Internal Registers

FIFO Control Registers

FIFO control registers are used to program the FIFO.

FIFO Start Address Register

This register is used to specify the Peripheral Space start address for a FIFO transaction. The first Pbus transaction accesses this address. Subsequent transactions will either automatically increment the Pbus address by the size of the port (by 1 for byte, by 2 for halfword), or retain the same address if the FNOINC bit is set in the FIFO_CONTROL register.

Note that although only 24 bits are valid the full address may be written into this register, the upper bits are ignored. On a read, address bit A(20) is set (signifying FIFO address). A(19) is hardwired to 0.

Bits 31:24	Reserved
Bits 23:0	Address Bits 23:0

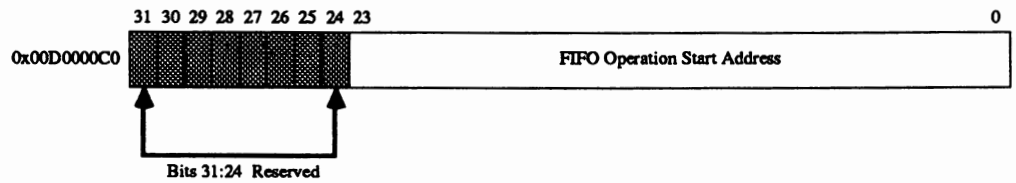


Figure 5-5 FIFO Start Address Register

FIFO Control Register

This register is used to to control and initiate FIFO operations. It also contains status bits to reflect the state of the FIFO.

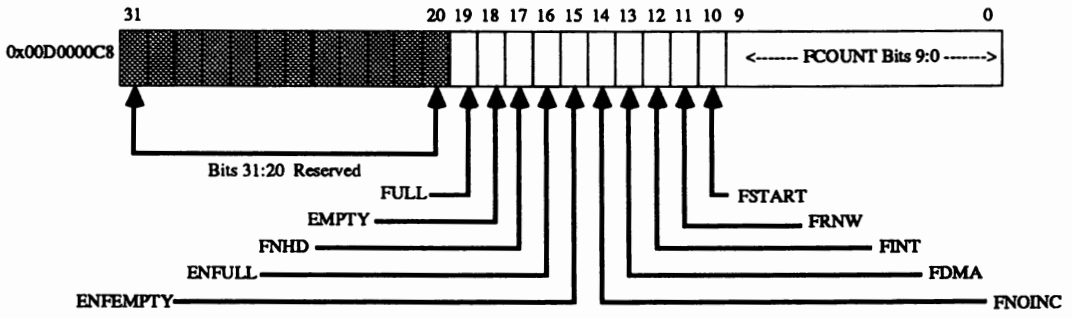


Figure 5-6 FIFO Control Register

Bits 31:20	Reserved
Bit 19	FULL 1 = FIFO is full
Bit 18	EMPTY 1 = FIFO is empty
Bit 17	FNHD 0 = 1 =
Bit 16	ENFULL 0 = Disable FIFO FUll interrupts 1 = Enable FIFO FULL interrupts
Bit 15	ENFEMPTY 0 = Disable FIFO EMPTY interrupts 1 = Enable FIFO EMPTY interrupts
Bit 14	FSTART 0 = Disable FIFO operations 1 = Enable FIFO operations
Bit 13	FRNW 0 = CPU write to Pbus 1 = CPU read from Pbus
Bit 12	FINT 0 = Disable FCOUNT equals interrupts 1 = Enable FCOUNT equals zero interrupt
Bit 11	FDMA 0 = Polled FIFO operations 1 = DMA FIFO operations (floppy disk only)
Bit 10	FNOINC 0 = Use incrementing Pbus address 1 = Use same Pbus address
Bit 9:0	FCOUNT Bits 9:0 – Number of bytes to be transferred.

A FIFO Full/Empty interrupt will cause the FEMPTY and FFULL bits to be frozen until the register is read by the CPU. These bits are set on a transition to the EMPTY or FULL state. They will not cause a new interrupt if the associated state is still true after being acknowledged, or if enabled when the FIFO is already in the Full/Empty state.

An operation is initiated by programming the FIFO Start Address register, programming the FIFO Control register parameters and setting the FSTART bit. The FSTART bit may be set in the same CPU as the programming of the rest of the FIFO Control register.

Initially, FCOUNT must be a multiple of 8. FCOUNT is decremented by 1 (for byte operations) or 2 (for halfword operations) for each peripheral bus read or write. The FINT bit determines whether a transition on FCOUNT to zero causes a FOP interrupt.

FIFO Data Register
The FIFO_DATA register is used for host FIFO data transfer operations. Only doubleword (64-bit) or 32byte burst operations are allowed to this register. FIFO data will be transferred into or out of the FIFO. Data is aligned to SPARC conventions. Full byte swapping and packing/unpacking between the Peripheral bus and the Mbus is performed within the MPI. Note that byte swapping cannot deal with addresses or pointers passed between the Mbus and Pbus.

Timer Control Registers

Set Count CT0 (CT1)
These are read-write registers used to set the count value for the associated counter. The value is specified in microseconds.

Counter Current Value
These registers provide the current Counter Value for the associated counter. They are read only registers, a write has no effect.

Counter 0 Control
This register provides control for Counter 0. The C0START bit may be set in the same operation as the setting of the rest of the control register.

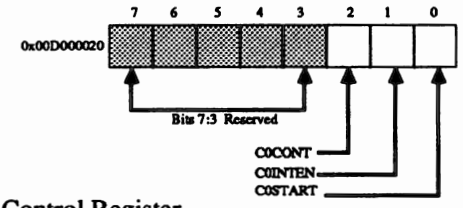


Figure 5-7 Counter 0 Control Register

Bits 7:3 Reserved

Bit 2 C0CONT
0 = Counter continues on zero to 1 (one shot)
1 = Counter reloads from CT0_TIME and continues running

Bit 1 C0INTEN
0 = Disable interrupt on zero
1 = Enable interrupt on zero

Bit 0 C0START
0 = Stop counter
1 = Start counter

Counter 1 Control

This register provides control for Counter 1. The C1START bit may be set in the same operation as the setting of the rest of the control register.

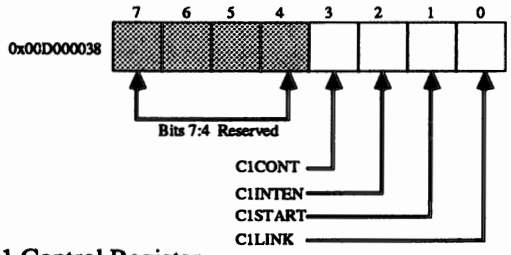


Figure 5-8 Counter 1 Control Register

Bits 7:3 Reserved

Bit 3 C1CONT
0 = Counter continues on zero to 1 (one shot)
1 = Counter reloads from CT0_TIME and continues running

Bit 2 C1INTEN
0 = Disable interrupt on zero
1 = Enable interrupt on zero

Bit 1 C1START
0 = Stop counter
1 = Start counter

Bit 0 C1LINK
0 = Count from 1MHz Clock
1 = Count from CT0

POWER MANAGEMENT CONTROL REGISTERS

Power Management Port
This register is used to control the state of the power management line provided by the MPI. It is an 8-bit read/write register containing one valid bit in D0. If this bit is set the PM1 line is asserted (high) and the system is powered down.

Clock Control Register
This register performs the MCLK management for the main CPU clock. Note that the only way of restarting MCLK0 after MCLKSTOP has been asserted is by the receipt of a non-masked interrupt. The programmer should therefore be careful to ensure that this will happen or the system will hang.

When an interrupt is received MCLK0 is automatically restarted and the MCLKSTOP bit is cleared. Note that when the clock restarts both MCLK0-1 will be restarted at 25MHz if MCLK12 is reset, and at 12.5MHz if MCLK12 is set.

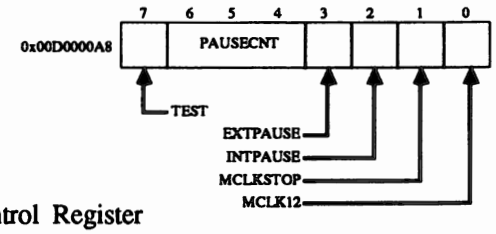


Figure 5-9 Clock Control Register

Bit 7	TEST Must be set to 0
Bits 6:4	PAUSECNT 001 - 111 = Number of MCLK0 clock cycles to stop during EXTPAUSE. 0 = do not pause clock
Bit 3	EXTPAUSE 1 = Enable EXTPAUSE clock stopping during other Mbus accesses
Bit D2	INTPAUSE 1 = Enable INTPAUSE clock stopping during MPI wait states
Bit D1	MCLKSTOP 1 = MCLK0 stopped (MCLK1 operates at 12.5MHz)
Bit D0	MCLK12 0 = MCLK0 and MCLK1 are 25MHz 1 = MCLK0 and MCLK1 are 12.5MHz

Interrupt Control Registers

Interrupts Pending
This register may be read to determine which of the interrupts are pending at any given time.

Interrupt Polarity
This register is used to specify the polarity level for the external interrupts. Bits corresponding to the internally generated interrupts are ignored and read as zero. A bit cleared to zero specifies an active low interrupt, a bit set to one specifies an active high interrupt. The register powers up as zero.

Interrupt Enable
This register is used to specify the interrupts which may interrupt the host. Clearing a bit to zero masks the corresponding interrupt level. A bit set to a one enables the corresponding level. All bits power up as zero.

Interrupt Acknowledge
This register is used to acknowledge the internally generated interrupt sources. It has no effect on the external interrupt sources except for IRQ9. It is write-only and a bit is set to acknowledge the corresponding interrupt. For the counter timers the interrupt will not occur again until the count transitions to zero. For the FIFOs the empty, full or FCOUNT=0 interrupt will not occur again until the transition to that condition happens again.

The interrupt control registers are organized as shown in Figure 5-10. Each of the interrupt request sources is assigned to the same bit in each of the control registers.

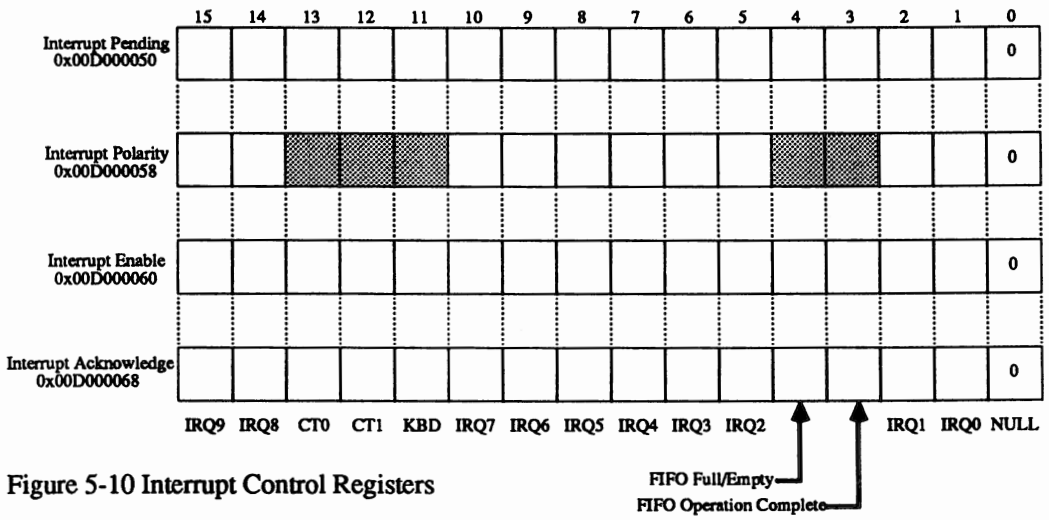


Figure 5-10 Interrupt Control Registers

Bit 15	IRQ9 – CMU
Bit 14	IRQ8 – Low Power Warning
Bit 13	CT0 – Counter Timer 0
Bit 12	CT1 – Counter Timer 1
Bit 11	KBD – Keyboard Character Buffer
Bit10	IRQ7 – Mouse
Bit 9	IRQ6 – External Serial Interface
Bit 8	IRQ5 – Modem
Bit 7	IRQ4 – Ethernet
Bit 6	IRQ3 – Display Controller
Bit 5	IRQ2 – Hard Disk Interface
Bit 4	FIFO FULL/EMPTY
Bit 3	FIFO Operation Complete (FCOUNT = 0)
Bit 2	IRQ1 – Floppy Disk Interface
Bit 1	IRQ0 – Centronics Interface
Bit 0	NULL – No Interrupt, hardwired to 0

Serial Input/Output Registers

Keyboard Byte
This register is read-only, it contains data from the serial input port which is used to communicate with the microcontroller (see Chapter 6). Up to three bytes are buffered in the MPI and the programmer is responsible for ensuring there is no overflow. A write to this register has no effect.

Note that Mousekey information passes via the Keyboard byte.

Keyboard Status
This register provides a single bit (Bit D0) that is asserted when there is valid data in the data register. It is cleared on a data read operation to the KBDSTATUS register. If there is further data to be read it is reasserted when that data is in the KBDDATA register.

Transmit Byte
This register is a write only 8-bit register; a read will return undefined data. Data may be written to this register when the TXBUSY status bit is not asserted.

Transmit Status
This register provides a single bit (Bit D0) that is asserted when the transmitter is busy.

Software Reset Register

Bit 4 RESET
Reinitializes the MPI, clears FIFO and sets all I/O bits to default levels – same effect as Power On Reset

Bit 3 IORESET
Asserts PRST line for 2µs

Bit 2 FIFOSTOP
Terminates any FIFO activity, resets FCOUNT and FSTART

Bit 1 TST1
Test vectors - must be left at 0

Bit 0 TST2
Test vectors - must be left at 0

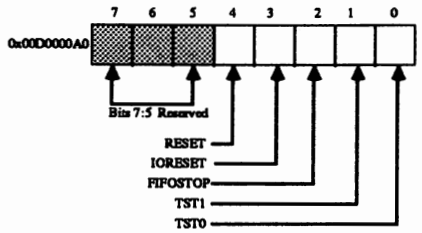


Figure 5-11 Software Reset Register

Error Status Register

This register provides the cause of an Mbus error generated by an access to the MPI ASIC. A read to this register clears the register bit latches.

Bit 5	BTO	MBus timeout
Bit 4	TYPE	Invalid MBus Type Field
Bit 3	SIZE	Invalid MBus Size Field
Bit 2	FNHD	FIFO No Host Data Error
Bit 1	PFERR	Pbus FIFO Setup Error
Bit 0	MFERR	Mbus FIFO Setup Error

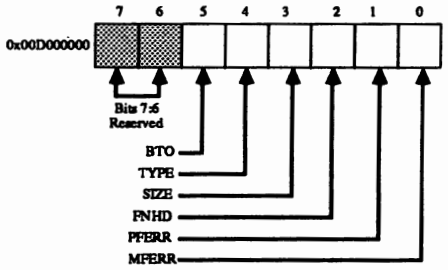
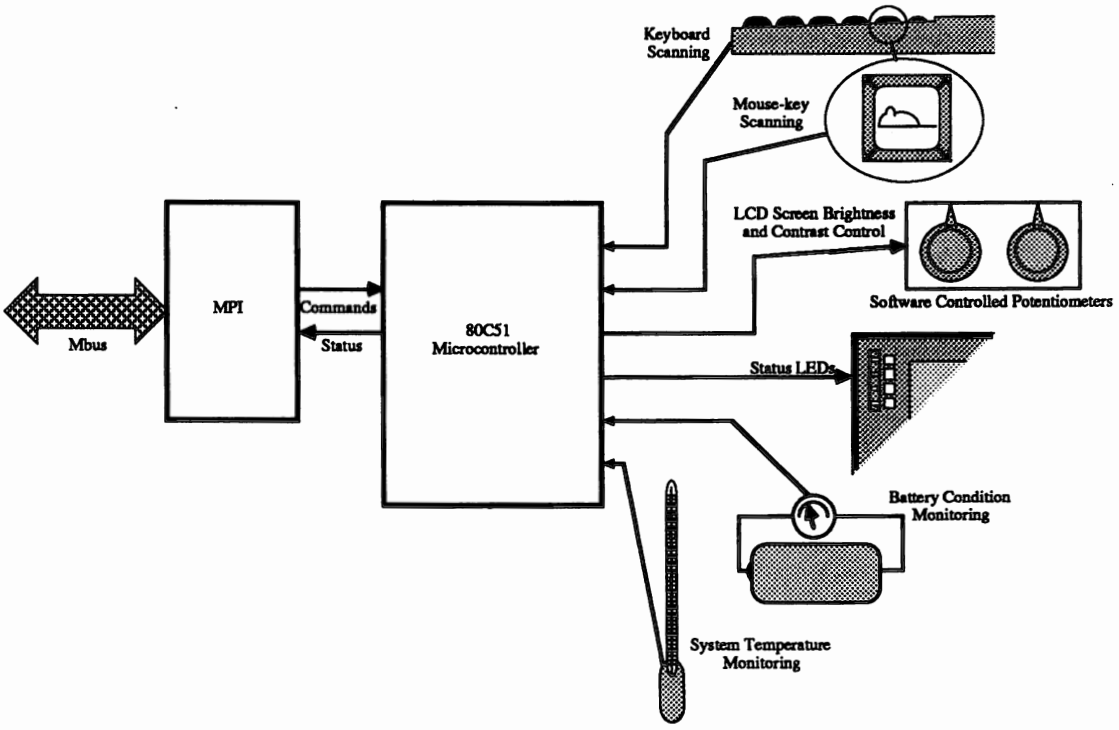


Figure 5-12 Error Status Register

CHAPTER SIX
MICROCONTROLLER SUBSYSTEM

Software monitoring of battery condition, keyboard input, and control of screen brightness and contrast are controlled by a dedicated microcontroller. This offloads these tasks from the main CPU, allowing its processing power to be concentrated on running applications.



The microcontroller performs the following functions:

- **Keyboard interfacing**
- **Mousekey interfacing**
- **Power supply monitoring**
- **Controlling the Status LEDs**
- **System temperature monitoring**
- **Screen brightness and contrast control**
- **System EEPROM interfacing**
- **Hardware reset control**

The microcontroller interfaces with the CPU (or host) via the serial interface of the MPI (see Chapter 5). During normal operation, the microcontroller performs keyboard and Mousekey scanning and monitors the power supply. On receipt of a command from the host or user command from the keyboard it performs a number of additional functions.

Note: the host and microcontroller exchange characters which represent hexadecimal numbers. In this manual, hexadecimal numbers are shown with the prefix 0x. The characters 0x do not form part of the command code or the data returned.

KEYBOARD INTERFACE

The microcontroller interfaces to the keyboard via a Honeywell 380105 keyboard scanner. This periodically scans the keyboard to detect key press events, and passes the serialized character codes for any depressed keys to the microcontroller. The microcontroller in turn interprets characters received and acts upon any that represent microcontroller commands, and conveys all others to the host via the serial port of the MPI (see Chapter 5).

Key Repeat

The microcontroller provides a software programmable key repeat function which produces multiple characters while a key remains depressed. Key depression and release are registered. A key press causes an a character code to be returned to the host, and key release causes the same code character to be returned but with the top bit (bit 8) set. Thus, if the code 0x33 is returned when a key is pressed, 0xB3 is returned when it is released.

The key repeat rate is specified by two parameters: the time until repeat and the time between repeats. These are programmable via a host command and are both measured in units of 10ms. The time until repeat is the time from when a key is pressed until it starts to repeat. Once repeat has started, the key code of the last key pressed will be presented to the host every time-between-repeats x 10ms.

Special Characters

Certain character combinations are intercepted by the microcontroller and used to activate keyboard enetered user commands. These are as follow:

- **Display Brightness Control**
  ALT and UP or DOWN together

- **Display Contrast Control**
  ALT and LEFT or RIGHT together

- **Reset Control**
  ALT and ESC and R together

Only the ALT depression and release codes are sent to the host during these sequences. The operations performed by these sequences are repeated at a rate determined by the current key repeat parameters.

Special Routines

Brightness Control
The brightness of the in-built LCD display is controlled with a digital potentiometer. This is a device which contains a software controllable variable resistance, providing 99 (dec) steps.

The UP cursor key pressed simultaneously with the ALT key increases the brightness (decreases the Brightness digital potentiometer value). If the key continues to be depressed the brightness continues to be changed until the end of track is reached.

The DOWN cursor key decreases the brightness (increases the pot value)

Contrast Control
A similar digital potentiometer controls the brightness of the in-built LCD display. This facility functions in a similar manner to the Brightness Control.

Reset Control
If a reset command from the keyboard is detected, the Power Supply Status bit is set to a 1 (indicating a user initiated reset) and the System Reset Line is asserted Low for 100ms. This causes a system reset.

MOUSEKEY INTERFACING

The Mousekey contains four force sensitive resistors (FSR); two are used to control vertical cursor movement, and two are to used to control horizontal movement. The Mousekey is isolated from the rest of the keyoard and is scanned separately via four channels of an LTC1093 combined data aquisition and analogue to digital converter (ADC) chip. Each of the four channels is assigned one FSR. The pressures exerted on a each resistor is measured in terms of a voltage and converted to a binary value by the ADC.

When the Mousekey left hand edge is pressed, for example, pressure is increased on the left FSR and reduced on the right FSR. The microcontroller calculates the difference between the voltages measured across the two FSRs and uses the result to determine magnitude and direction of any cursor movement.

Mousekey Initialization

At system initialization, the Mousekey is calibrated by reading FSR voltages and storing them in variables within the microcontroller memory. When the Mousekey is subsequently scanned, any voltages that differ from these calibration values are deemed to be an indication of Mousekey pressure. The algorithm for converting left, right, up and down voltages into mouse movements is shown below.

```
Read voltage on left FSR
Read voltage on right FSR
Calculate the difference (Vright - Vleft)
If result is 0 - no horizontal motion requested
If result is positive a net right motion has been requested.
        Return this as a positive X motion.
If result is negative a net left motion has been requested
        Return this as a negative X motion.
```

Vertical movements are calculated in a similar way and returned as positive or negative Y motions respectively. The algorithm evaluates relative voltages to counteract drift in the steady state FSR values due to thermal effects.

Mousekey Events

The microcontroller passes mouse events to the host as a sequence of four bytes, 0xFD 0xSS 0xXX 0xYY. The first byte, 0xFD, indicates to the host that three mouse event bytes are to follow. The next, 0xSS, provides the following status information:

Bit 7	Y data overflow (always 0) 1 = overflow
Bit 6	X data overflow (always 0) 1 = overflow
Bit 5	Y data sign 1 = negative Y motion
Bit 4	X data sign 1 = negative X motion
Bit 3	Event Flag
Bit 2	Middle button status 1 = depressed
Bit 1	Right button status 1 = depressed
Bit 0	Left button status 1 = depressed

The 0xXX and 0xYY bytes represent the magnitude of the X and Y motions recorded in a particular mouse event. The force on the Mousekey is represented by 3 bits, so the values of 0xXX and 0xYY ranges from 0x00 to 0x07. These values cannot overflow, so the X and Y data overflow bits will always be 0.

Mouse Buttons

The buttons found on a conventional three button mouse are simulated using special key combinations on the keyboard. Mouse button events are detected by the keyboard scanning routine, which is always executed immediately before the Mousekey scanning routine. During system initialization, the Event Flag is cleared to 0. If the keyboard scanning routine detects a mouse button event, it sets the associated button bits, and sets the Event Flag to indicate that the status has changed and a mouse event packet should be sent to the host.

Cursor Movement Requests
After the keyboard scanning routine is complete, the mouse scan routine is called. It determines if a net motion has been detected and if so, fills in the sign bits, X and Y data fields and sets the Event Flag (it may already have been set by the keyboard scanning routine) to indicate a changed status.

At the end of the Mousekey scanning routine, the Event Flag is checked to see if it has been set. If it has, a mouse event packet is sent to the host and the reserved bit is cleared.

Mouse Event Parameters
Mouse operation is controlled by two parameters: the force on the Mousekey, and the time between successive calls to the mouse scan routine. Key pressure is quantized into eight levels, so that a light key pressure moves the cursor in steps of one unit every time the mouse scan routine is called, and the heaviest measureable key pressure causes the cursor to move in steps of 8 units every time the mouse scan routine is called. The result of this is that increasing the pressure on the Mousekey accelerates cursor motion.

Mouse scan rate is variable in units of 10ms and determines how fast the mouse moves when a constant pressure is applied to the Mousekey. This rate is software programmable and is stored in EEPROM to maintain its value during system power down.

HARDWARE INTERFACES

The microcontroller communcates with the host via serial port of the MPI. It provides control, data aquisition and status aquisition via four ports, each with eight 1-bit I/O ports. These are assigned as shown in Table 6-1

Host Instruction Protocol

The microcontroller incorporates a mask programmable read-only memory which contains the complete microcontroller firmware. The host may make various requests to the microcontroller via the serial port of the MPI. The microcontroller receives and interprets these requests and takes the appropriate action. It returns values (if required) to the host as characters.

The microcontroller issues the code 0xFE to acknowledge all host instructions, followed by one or more bytes as required to fulfill the request before resuming normal keyboard and Mousekey scanning.

Table 6-2 outlines the request codes available to the host and the responses from the microcontroller .

BIT-PORT	DIRECTION	FUNCTION
Port0(0)	Output	Keyboard Scanner Reset
Port0(1)	Output	Keyboard Scanner Clock
Port0(2)	Input	Keyboard Data
Port0(3)	Input	DC Level OK
Port0(4)	Input	Battery Level 1
Port0(5)	Input	Battery Level 2
Port0(6)	Input	Battery On
Port0(7)	Input	Battery Full
Port1(0)	Output	Parity Detect Enable
Port1(1)	Output	Parity Error Inject
Port1(2)	-	Reserved
Port1(3)	-	Not Used
Port1(4)	-	Not Used
Port1(5)	-	Not Used
Port1(6)	Input	Floppy Disk Density Status
Port1(7)	Input	Floppy Disk Ready Status
Port2(0)	Output	Bit I/O Device Control 1
Port2(1)	Output	Bit I/O Device Control 2
Port2(2)	Output	Brightness Chip Select
Port2(3)	Output	Contrast Chip Select
Port2(4)	Output	ADC Chip Select
Port2(5)	Output	EEPROM Chip Select
Port2(6)	Output	Temperature Sense Read Data
Port2(7)	Output	EEPROM Read Data
Port3(0)	Input	Serial Data From MPI
Port3(1)	Output	Serial Data to MPI
Port3(2)	Output	System Reset
Port3(3)	Output	Sound Beeper
Port3(4)	Output	Battery low LED
Port3(5)	Output	User LED
Port3(6)	Output	External LED
Port3(7)	Output	Interrupt CPU on Level 14

Table 6-1 Bit-Port Assignment

REQUEST	MEANING	VALUE RETURNED (after 0xFE)
System Information		
0x10	Read Ethernet Address	3bytes with Net Address. For example, for address 0x0044FA the bytes returned would be 0x00, 0x44, 0xFA
0x11	Read Hardware Revision	2bytes with Hardware Revision Major and Minor. For example Rev 1.3 is returned as 0x01, 0x03
0x12	Read Controller Firmware Rev	Format as for Hardware Revision This data is hardwired into the microcontroller firmware
0x13	Read System MAXTEMP	Value in degrees centigrade hexadecimal Eg 25 degrees is returned as 0x19.
0x14	Read System MINTEMP	As for MAXTEMP
0x15	Read Temperature Sensor ADC	As for MINTEMP
0x16	Read POWERCOUNT	2bytes, most significant first, giving the number of times the SPARCbook has been powered.
0x17	Read POWERONSECONDS	4bytes seconds SPARCbook has been powered
0x18	Return Microcontroller Status	Returned byte Bit 7 Bad Checksum Bit 6 Diagnostics Fail Bit 5 Watchdog Reset (1=Watchdog Reset) Bit 4 User Reset bit (0=POR) Bit 3 Battery Full bit Bit 2 Battery On bit Bit 1 Battery Level 2 Bit 0 Battery Level 1
0x19 0xNN	Read ADC channel 0xNN	2 bytes containing 10-bit value. First byte contains most significant 8 bits, second byte contains least significant 2 bits i.e. bbbbbbbb 000000bb CH 0xNN FUNCTION 0 0x00 Force sensitive resistor 0 1 0x01 Force sensitive resistor 1 2 0x04 Force sensitive resistor 2 3 0x05 Force sensitive resistor 3 4 0x02 Battery Voltage sensor 5 0x03 Temperature sensor

REQUEST	MEANING	VALUE RETURNED
Commands		
0x20	Turn USER LED off	None
0x21	Turn USER LED on	None
0x22 0xNN	Sound Beeper for 0xNN increments of 10ms	None
0x23 0xNN	Increment Brightness 0xNN times (0 - 63 hex)	None
0x24 0xNN	Decrement Brightness 0xNN times (0 - 63 hex)	None
0x25 0xNN	Increment Contrast 0xNN times (0 - 63 hex)	None
0x26 0xNN	Decrement Contrast 0xNN times (0 - 63 hex)	None
0x27 0xNN	Read EEPROM register 0xNN	16-bit register Value
0x28 0xNN 0xXX 0xYY	Write EEPROM register 0xNN with data 0xXXYY	None
0x29 0xMM 0xNN	Rollover parameters MM is initial delay in 10ms increments, 0xNN is the interval between characters in 10ms increments. Default is 400ms and 10 chars/sec.	None
0x2A 0xNN	Mouse scan period in units of 10ms	None
0x2B 0xNN	Write byte 0xNN to microcontroller port 1	None
0x2C 0xNN	Read pin values from microcontroller port 1	1-byte port value
Miscellaneous		
0x30	Acknowledge Interrupt	None
0x40	Enable Watchdog (if watchdog commands are not received at least 1 per second the system will be reset)	None
0x41	Disable Watchdog (default)	None
0x42	Watchdog command	None

Table 6-2 Host Instructions and Replies

Host Interrupts

If either of the battery low signals is detected low, or the external power supply is plugged in or removed, the host is interrupted on Level 14 by setting the Interrupt Host port bit. The port bit remains set until an interrupt acknowledge is received from the host. It is returned to its default zero condition. The host is able to read the Microcontroller Status Byte to determine the cause of the interrupt and then take the appropriate action.

EEPROM Data

The following data is defined in the EEPROM. This is a 1024bit device treated as 128 byte locations:

Note that bytes 0-31 may be not written, they are reserved for manufacturing information.

BYTE	FUNCTION	COMMENT
0	Ethernet Address (serial no) low byte	
1	Ethernet Address (serial no) middle byte	
2	Ethernet Address (serial no) high byte	
3	HW Revision – Minor Level	
4	HW Revision – Major Level	
5-14	RESERVED for hardware use	
15	Byte Checksum for bytes 0-14	
16	MAXTEMP in degrees C	
17	MINTEMP in degrees C	
18	POWERCOUNT byte 0	No of power up cycles
19	POWERCOUNT byte 1	
20	POWERONSECONDS byte 0	No of power on secs
21	POWERONSECONDS byte 1	
22	POWERONSECONDS byte 2	
23	POWERONSECONDS byte 3	
24	Keyboard time until repeat	10ms units
25	Keyboard time between repeats	10ms units
26	Time between Mouse scans	10ms units
27-31	RESERVED for system use	Access code protected
31-63	RESERVED for system use	No Access code needed
64-127	RESERVED for application use	No Access code needed

Table 6-3 EEPROM Address Map

CHAPTER SEVEN
UNIVERSAL PERIPHERAL CONTROLLER

One of the devices implemented in the SPARCbook design to achieve its high level of integration is the 82C710 Universal Peripheral Controller (UPC) from Chips and Technologies Inc. This single chip provides the SPARCbook with the following facilities:

- **Floppy disk controller**
- **IDE Hard disk interface**
- **Parallel Centronics interface**
- **Serial Port**
- **Mouse Interface**

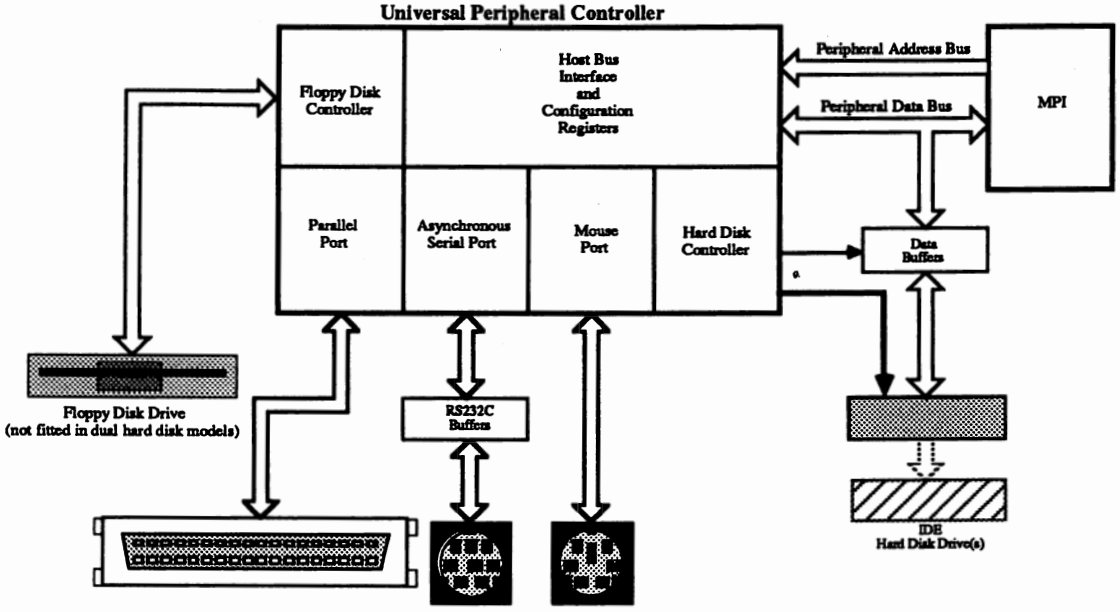


Figure 7-1 UPC Peripheral Control Architecture

FLOPPY DISK CONTROLLER

The UPC contains a floppy disk controller (FDC) fully compatible with the NEC μPD72065B. An integral 48mA floppy interface buffer allows the disk drive to connect directly to the UPC. The SPARCbook is equipped with one 3.5" floppy disk drive in single hard disk models, but does not contain a floppy disk drive in dual hard disk models.

Host Interface

The activities of the FDC are controlled via a set of four register locations, shown in Table 7-1.

Digital Output Register
This is a write-only register which provides drive select (bits 1:0 = 00bin), motor enable (bit 4 = 1), DMA enable (bit 3 = 1) and reset control (bit 2 = 1).

Main Status Register
This is an 8-bit register which contains information about current activity by the FDC.

Data Register
The data register is used by the host to issue instructions to the FDC, and by the FDC to return status information to the host. The FDC automatically directs information to or from the appropriate internal location.

Data Rate Register
The lower two bits in this register are used to select the data rate for the FDC. 00bin selects a 500Kbyte/s data rate.

Digital Input Register
Only bit 7 in this register is used for FDC operations. It provides the complement of the Disk Change input pin.

ADDRESS	REGISTER	ACCESS
0x00CC003F0	Unused	
0x00CC003F1	Unused	
0x00CC003F2	Digital Output Register	W
0x00CC003F3	Unused	
0x00CC003F4	Main Status Register	R
0x00CC003F5	Data Register	R/W
0x00CC003F6	Unused	
0x00CC003F7	Data Rate Select Register	W
	Digital Input Register	R

Table 7-1 FDC Register Map

Floppy Disk Commands

The host controls the FDC with commands written via the Data Register at 0x00CC003F5. There are seventeen commands available which are initiated by a multi-byte write to the data register. The first byte contains an operation code, and this is followed by a number of bytes containing parameter information (the number of parameter bytes depends upon the instruction). Most commands to the FDC entail a three part sequence consisting of the Command Phase, the Execution Phase and the Result Phase.

Command Phase
During the command phase, the host writes an instruction byte together with the parameters required to carry out the instruction via the Data Register. The FDC automatically determines number of parameters required with the command byte and the order in which they should be supplied by the host.

Execution Phase
During the execution phase, the FDC acts upon the command supplied during the command phase.

Result Phase
After execution of an instruction, the FDC automatically presents status information for the host to read from the Data Register. The status information supplied varies according to the action carried out.

Figure 7-2 summarizes the command set of the FDC. The bytes written during the command phase are shown within shaded areas, next to the bytes read by the host during the associated result phase.

Key to abbreviations used in Figure 7-2:

C	Cylinder	HD	Selected head	NCN	New cylinder number
D	Data pattern	HLT	Head load time	R	Record
DTL	Data length	HUT	Head unload time	R/W	Read/Write
EOT	End of track	MF	FM/MFM mode	SC	Sectors per cylinder
GPL	Gap 3 length	MT	Multi-track	SK	Skip
H	Head Number	N	Bytes per sector	SRT	Step rate time
STP	Scan step			ST0, 1, 2, 3	Status registers 0, 1, 2, 3
US0, 1	Selected drive number				

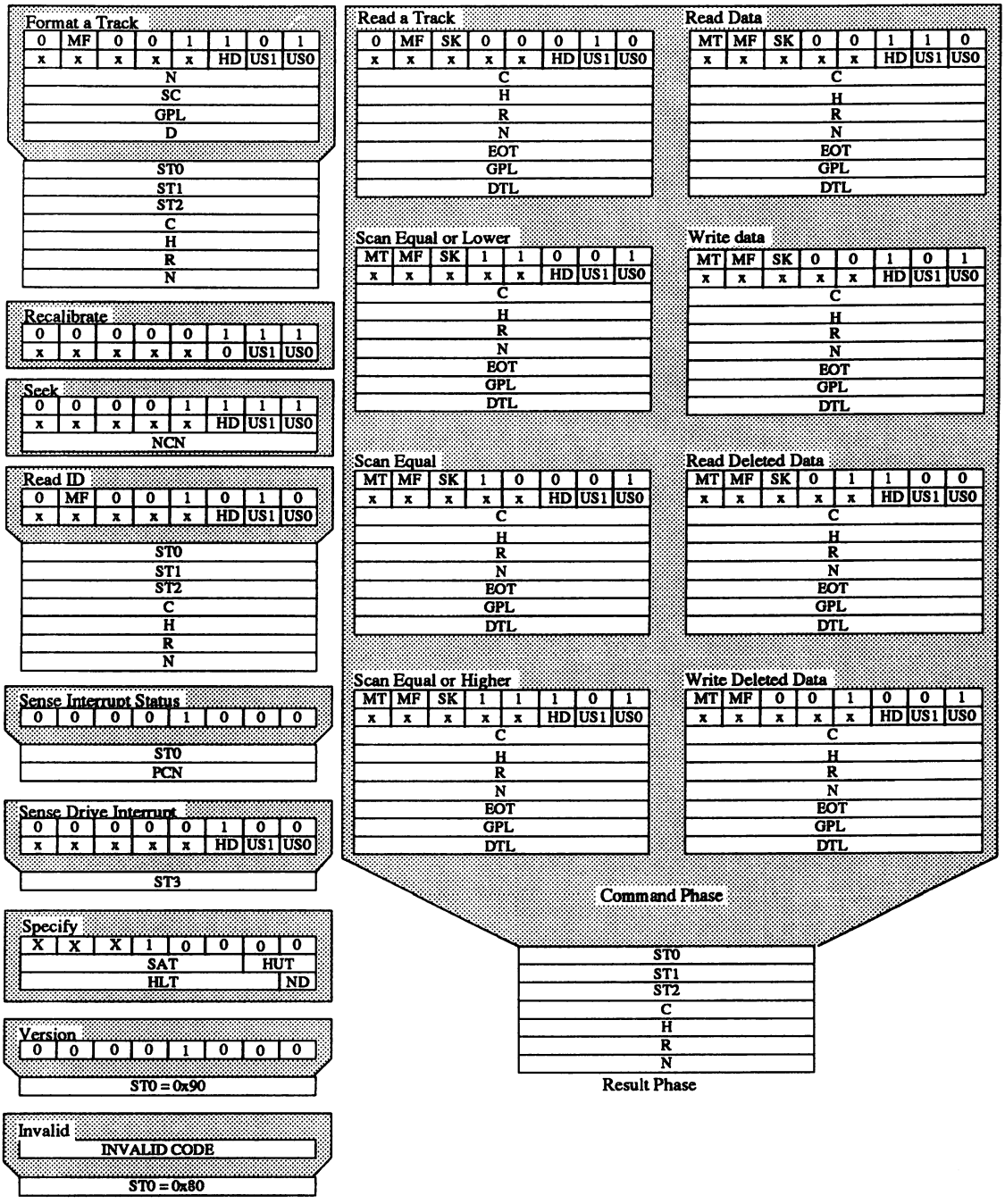


Figure 7-1 FDC Instruction Summary

FDC Operating Modes

The FDC in the SPARCbook application only ever controls a single 3.5" floppy disk drive. The FDC reads and write SunOS format disks, which can be either high density 1.4Mbyte disks or low density 720Kbyte disks.

Table 7-2 summarizes the disk types supported by the SPARCbook and the parameters required to configure the FDC to correctly carry out commands.

PARAMETER	HIGH DENSITY	LOW DENSITY	UNIT
Step Rate	3	3	ms
Head Unload Time	64	64	ms
Head Load Time	12	12	ms
Gap 3 Length†	0x24	0x2A	bytes
Gap 3 Length††	0x6D	0x50	bytes
Filler (format pattern)	0xE5	0xE5	-
Number of Cylinders	80	80	
Sectors/track	18	9	-
Sector Size	512	512	bytes
Surfaces	2	2	
Data Encoding Mode	MFM	MFM	

† Gap 3 for read and write commands
†† Gap3 for format command

Table 7-2 Floppy Disk Parameters

IDE HARD DISK INTERFACE

The UPC provides an interface controller capable of supporting two hard disk drives with Integrated Drive Electronics (IDE) interfaces. The UPC also provides control of external 16-bit data buffers.

IDE Overview

IDE hard disk drives incorporate an integral controller which interprets control signals and commands from the host, and reads data from and writes data to the disk surfaces.

IDE drives are able to support applications in PC/XT and PC/AT environments. The SPARCbook only supports drives configured for PC/AT compatibility. One or two drives may be present, but in a two drive system, the drives must be electrically configured as primary and secondary drives. Reference should be made to the technical literature supplied with the disk drive for information about drive configuration.

IDE Interface Operating Modes

The UPC provides two IDE interface modes: the AT mode, used in the SPARCbook and the XT mode. The mode is configured via bit 6 of the UPC Configuration Register at 0x0C. When this bit is at '1', the AT mode is selected.

The UPC IDE interface control registers are accessible in the 8-bit I/O region of the SPARCbook memory map, and the data register is accessible within the 16-bit I/O region of the MPI's Pbus address map. This results in the address loactions of the IDE Task File Registers shown in Table 7-3. Transfers between hard disk sectors and the SPARCbook main memory take place via the MPI (see Chapter 5). The hard disk interface uses polled operations only, DMA operations are not supported.

Task File Registers

The activities of an IDE hard disk drive are controlled via a set of register locations within the integral drive controller called Task File Registers. The UPC decodes host accesses to these registers. Table 7-3 lists the task file registers

ADDRESS	REGISTER	ACCESS
0x00C8001F0	Data Register	R/W
0x00CC001F1	Error Register	R
0x00CC001F2	Sector Count	R/W
0x00CC001F3	Sector Number	R/W
0x00CC001F4	Cylinder Low	R/W
0x00CC001F5	Cylinder High	R/W
0x00CC001F6	Drive/Head	R/W
0x00CC001F7	Status Register	R
	Command Register	W
0x00CC003F6	Alt. Status Register	R
	Fixed Disk Register	W
0x00CC003F7	Digital Input Register	R

Table 7-3 Task File Registers

Data Register
The data register is the register through which all data is passed on read and write commands. All data transfers are 16-bits with the exception of the ECC bytes transferred during Read or Write Long commands. Data is stored on the disk least significant byte first, then most significant byte.

Error Register
This read-only register contains the status of the last executed command.

Sector Count Register
This register contains the number of sectors to be read or written during a Verify, Read, Write or Format command. Note that a value of 0 means a 256 sector transfer.

Sector Number Register
This register contains the starting sector number for Read, Write and Verify commands.

Cylinder Number Registers
These registers contain the LSB and MSB of the first cylinder number where the disk is to be accessed for Read, Write, Seek and Verify commands.

Drive/Head Register
This register contains the drive and head select bits:

Bit 4	Drive select 0 = Primary 1 = Secondary
Bit 3:0	Head number

Status Register
This register contains status infromation for the drive and controller. The contents of this register are updated at the completion of each command. Pending interrupts are cleared whenever this register is read by the host.

Bit 7	Busy
Bit 6	Ready for command
Bit 5	Fault
Bit 4	Seek command completed
Bit 3	Ready to transfer data
Bit 2	Data correction sucessful
Bit 1	Index mark detected
Bit 0	Error from last command

Command Register
This register is used to pass commands to the hard disk. The commands consist of an 8-bit command code. Execution begins immediately after this register is written. Table 7-4 shows a summary of the executable commands and the parameters used.

Alternate Status Register
This register contains the same information as the Status Register. The difference is that reading the alternate status register will not clear pending interrupts.

Fixed Disk Register
This register contains interrupt enable and reset control bits. Bit 3 when set holds the drive in a reset state. If two drives are present, the second drive will be reset as well. Bit 2 when clear enables interrupts from the hard disk drives.

Digital Input Register
This register loops back the drive select and head select of the currently selected drive. Bit 7 of this register is used by the FDC and has no relevence to the IDE interface.

COMMAND NAME	CODE (hex)	PARAMETERS USED SC	SN	CY	SDH
Recalibrate	1x	n	n	n	D
Read Sector (retry)	20	y	y	y	y
Read Sector (no retry)	21	y	y	y	y
Read Sector Long (retry)	22	y	y	y	y
Read Sector Long (no retry)	23	y	y	y	y
Write Sector (retry)	30	y	y	y	y
Write Sector (no retry)	31	y	y	y	y
Write Sector Long (retry)	32	y	y	y	y
Write Sector Long (no retry)	33	y	y	y	y
Read Verify Sector (retry)	40	y	y	y	y
Read Verify Sector (no retry)	41	y	y	y	y
Format Track	50	n	n	y	y
Seek	7x	n	n	y	y
Drive Diagnostics	90	n	n	n	n
Initialize Drive Parameters	91	y	n	n	y
Read Sector Buffer	E4	n	n	n	d
Write Sector Buffer	E8	n	n	n	d
Identify Drive	EC	n	n	n	d
Power Management	F8 - FD				

Key:
SC = Sector Count Register; SN = sector number Register; CY = Cylinder Register; SDH = Drive Head Register; y = Parameter is used; n = Parameter not used; D = drive parameter only; x = Dont care

Table 7-4 AT Command Summary

IDE Command Summary

Recalibrate

This command moves the heads to cylinder 0. On receipt of this command, the drive sets BSY and moves the heads to cylinder 0. The drive waits for the seek to complete before updating its status, resetting BSY, and generating an interrupt. If the drive cannot reach cylinder 0, it sets both the "ERR" bit in the Status register and the "TKO" bit in the Error register. An aborted command response will be given if the drive is not ready. When the command completes successfully, the Task File registers will be affected as follows:

Cylinder High	00
Cylinder Low	00
Error	00
SDH	Unchanged
Sector Count	Unchanged
Sector Number	Unchanged

Read Sector
This command reads from 1 to 256 sectors, as specified in the Task File, beginning at the specified sector. A sector count of 0 requests 256 sectors. As soon as the Command register is written, the drive sets BSY and begins execution of the command. An ID not found error is returned if incorrect task file parameters are passed. If the drive is not already on the desired track, a seek is initiated. When the sector ID is located, the DRQ bit is set, and an interrupt is generated.

The DRQ bit is always set, regardless of the presence or absence of an error condition at the end of a sector. When the command completes successfully, the Task File register contains the cyclinder, head, and sector number of the last sector read. The sector count is zero after successful execution of the command.

Multiple sector reads set DRQ and generate an interrupt at the completion of each sector. DRQ is reset and BSY is set immediately when the Host completes reading the sector. If an error occurs during a multiple sector read, it will terminate after the sector in error is transferred to the host. The Task File indicates the location of the sector where the error occurred. The Host may then read the Task File to determine what error has occured, and on which sector. If no error is detected, the cylinder, head, and sector registers are updated to point to the next sequential sector.

Read Long
The Read Long command performs similarly to the Read Sectors command, except that it returns the data and the ECC bytes contained in the data field of the desired sector. During a Read Long command, the drive does not check the ECC bytes to determine if there has been a data error. Only single sector Read Long operations are supported. Data transfers are 16bits wide and ECC byte transfers are 8bits wide.

Write Sector
This command writes up to 256 sectors, as specified in the Task File, beginning at the specified sector. As soon as the Command register is written, the drive waits for the Host to fill the sector buffer with the data. There is no interrupt generated to start the first buffer fill operation. Once the buffer is full, the drive sets BSY and begins command execution.

An ID not found error is returned if incorrect task file parameters are passed. If the drive is not already on the desired track, an implied seek is performed. Once at the desired track, the drive locates the appropriate ID field and writes data from the buffer, plus 7 bytes of ECC. Upon command completion, the Task File registers contain the cylinder, head, and sector number of the last sector written. The sector count is zero after successful execution of the command.

Multiple sector writes set DRQ and generate an interrupt each time the buffer requires filling. DRQ is cleared and BSY set immediately when the Host fills the sector buffer. If an error occurs during a multiple sector write, it will terminate at the sector where the error occurs. The Task File indicates the location of the sector where the error occurred. The Host may then read the Task File to determine which error occured, and on which sector. If an error is not detected, the cylinder, head, and sector registers are updated to point to the next sequential sector.

Write Long
This command is identical to the Write Sectors command, except that it writes the data and ECC bytes directly from the sector buffer. The drive does not generate ECC for this command. Only single sector operations are supported.

Read Verify
This command functions identically to the Read Sectors command, except that no data is transferred to the Host. Up to 256 sectors will be read into the sector buffer and the ECC bytes are verified.

When each sector has been verified, the Task File is updated, but no DRQ or interrupt is set until all sectors have been verified. A value of 0 in the sector count register indicates that 256 sectors are to be verified.

Format Track
This command formats the track specified in the Task File. Once command registers have been written, the drive waits for the host to fill the buffer with 512 bytes of format data. After the data is written to the data register, the drive analyzes the information for each sector and performs the requested action for each sector. Upon command completion, the drive places status information in the Task File and signal an IRQ to the host.

The 512 bytes of data in the sector buffer must consist of 2 bytes for each sector to be formatted. The most significant byte of each word designates the sector to be formatted. The least significant byte contains a descriptor indicating which action should be carried out with that sector as follows:

0x00	Format Sector Good
0x80	Format Sector Bad
0x40	Assign Sector to Alternate
0x20	Unassign Alternate Location for this Sector

Seek
This command initiates a seek to the track and selects the head specified in the Task File. The drive need not be formatted for a seek to execute properly. When the command is issued, the drive sets BSY, initiates the seek, clears BSY, and generates an interrupt. Only the Cylinder register is valid for this command. The drive does not wait for the seek to complete before returning the interrupt. Seek complete (DSC) will be set upon completion of the command. If a new command is issued while a seek is in progress, the drive will wait, with BSY active, until the seek is complete before starting the new command. No checks are made on the validity of the sector number. The ERR bit in the Status register and the IDNF bit in the Error register will be set if an illegal cylinder number is specified.

Initialize Drive Parameters
This command enables the host to set the head switch and cylinder increment points for multiple sector operations. In translate mode, the logical head, sector, and cylinder numbers in the Task File will be translated to their native physical values in the Task File. They are not checked for validity by this command, therefore if they are invalid, no error will be reported until an illegal access is made by some other command. Upon receipt of the command, the drive sets BSY, saves the parameters, resets BSY, and generates an interrupt.

Read Buffer
This command allows the host to read the current contents of the drive's sector buffer. When this command is issued, the drive sets BSY, sets up the sector buffer for a read operation, sets DRQ, clears BSY, and generates an interrupt. The host then reads up to 512bytes of data from the buffer.

Write Buffer
This command allows the host to overwrite the contents of the drive's sector buffer with any data pattern desired. Only the command register is valid for this command. When this command is issued, the drive will set BSY, set up the sector buffer for a write operation, set DRQ, reset BSY, and generate on interrupt. The host may then write up to 512 bytes of data to the buffer.

Identify Drive
The Identify Drive command allows the host to obtain parameter information from the drive. When the command is issued, the drive sets BSY, stores the required parameter information in the Sector buffer, sets the DRQ bit, and generates an interrupt. The parameter words in the buffer are in hexadecimal format right justified as shown Table 7-5.

WORD	DESCRIPTION
00	A constant – 0A5A
01	Number of fixed cylinders
02	Number of removable cylinders
03	Number of heads
04	Number of unformatted bytes/track
05	Number of unformatted bytes/sector
06	Number of physical sectors/track
07	Number of bytes in the inter-sector gaps
08	Number of bytes in the sync fields
09	Vendor Unique
10-19	Serial number
20	Buffer type
21	Buffer size (x 512 bytes)
22	Number of ECC bytes passed during read or write long commands
23-26	Controller firmware revision
27-46	Model Number
47	Number of sectors/interrupts
48	Double word transfer flag
49-255	Reserved

Table 7-5 Drive Parameters

Power Save Commands
In order to conserve power, these commands allow the drive to operate in modes other than fully operational. These are: Idle Mode, in which the driveis up to speed is and ready to accept a command; Sleep Mode in which the drive is spun down with only the interface chip powered up; and Standby Mode in which the drive is spun down with all normal electronics on.

The Power Save Commands are as follows:

0xEO	Enter Standby Mode immediately.
0xE1	Enter Idle Mode immediately.
0xE2	Enter Standby Mode immediately.
0xE3	Enter Idle Mode immediately.
0xE5	This command allows the host to check the drive power status. If it is in Active or Power Save mode, the Sector Count register will be set to 0xFF. If the drive is in, going to, or recovering from STANDBY MODE, the Sector Count register will be set to 00.
0xE6	Enter Sleep Mode

PARALLEL PORT

The Parallel Port is compatible to the IBM XT-AT Parallel Port, with a PS/2 like extended mode for bi-directional transfers. When the parallel port is disabled via the configuration register, all outputs are disabled, and register contents are preserved. Upon power up, the control signals are inactive. The status registers reflects the status signals.

Printer Interface Registers

Table 7-6 shows the registers associated with the parallel printer port. These are compatible with the IBM PC parallel port. All addresses for the parallel port are offset from the base address specified during the UPC configuration process. The table shows the value contained in the parallel port registers after a hardware reset.

The parallel port registers are accessible in the SPARCbook's 8-bit I/O region at 0x00CC00000. The address offset of the parallel port registers is programmable via one of the UPC configuration registers. Table 7-6 shows the addresses assigned by the resident firmware.

Data Register
Data written to this register is transmitted to the printer. Data read from this port is identical to that which was last written by the host.

ADDRESS	REGISTER	ACCESS	RESET
0x00CC00330	Data Register	R/W	Dont Care
0x00CC00331	Status Register	R	0x80
0x00CC00332	Control Register	W	0x00

Table 7-6 Parallel Port Registers

Printer Status Registers
This register contains the following status information:

Bit 7 BUSY
0 = Printer busy
1 = Printer ready to accept data

Bit 6 ACK.
0 = Printer has received a character and is ready for another.
1 = Printer not ready

Bit 5 PE-Paper Empty
0 = Paper OK
1 = Paper end

Bit 4 SLCT
0 = Printer not selected
1 = Printer is online

Bit 3 ERROR
0 = Printer Error
1 = No errors.

Bits 2:0 Reserved

Printer Control Register
The bit definitions for this register are:

Bit 7:6 Reserved

Bit 5 DIR – Parallel Control Direction
In printer mode, the direction is always out, regardless of the state of this bit.
In the extended mode:
0 = output
1 = input

BIt 4 IRQEN.
0 = IRQ disabled
1 = Enable IRQ when ACK changes from active to inactive

Bit 3 SLCTIN
0 = Printer not selected
1 = Printer selected

Bit 2 INIT
0 = Start the printer (50µs pulse minimum)

Bit 1: AUTOFD
0 = No autofeed.
1 = Printer to generate a line feed after each line

Bit 0: STROBE
0 = No strobe
1 = Assert STROBE

MOUSE PORT

The SPARCbook features an inbuilt Mousekey on the keyboard which is controlled by the microcontroller (see Chapter 6), and not the UPC.

In addition, the UPC provides a port which appears via a 6-pin mini-DIN connector located on the connector panel, to which an external pointing device (such as a mouse or graphics tablet) or keyboard may be connected.

Pointing devices provide the computer to which they are connected with position and button press information, and external keyboards provides character information. The UPC provides a serial data input channel and shift register to allow the host to read this information from an 8-bit data port. The UPC also provides a clock output used by the external device to synchronize its communications.

Devices connected to the mouse port must use TTL level signals and not RS232.

Connecting RS232 devices to the mouse port may cause damage to the SPARCbook

Mouse Port Registers

The UPC contains two registers associated with mouse controller operations. These are accessible in the 8-bit I/O region at 0x00CC00000. The offset address for the mouse registers is programmable via one of the UPC configuration registers. The address used by the resident firmware is 0x310. Table 7-7 shows the mouse port registers.

ADDRESS	REGISTER	ACCESS
0x00CC00310	Mouse Port Data	R
0x00CC00310	Mouse Port Status/Control	R/W

Table 7-7 Mouse Port Registers

Mouse Data Port
This location is read by the host to obtain data input by an external mouse.

Mouse Status and Control Register
This register contains a number of status bits and a number of control bits:

Control Bits

Bit 7	Mouse Clock Enable 0 = Clock disabled 1 = Clock enabled
Bit 4	Pointing Device Interrupt 0 = Disable interrupt 1 = Enable interrupt
Bit 3	Pointing Device Reset 0 = Normal operation 1 = Reset device

Status Bits

Bit 6	Pointing Device Clear
Bit 5	Pointing Device Error
Bit 2	Pointing Device Transmit Idle
Bit 1	Pointing Device Character Received
Bit 0	Pointing Device Idle

SERIAL PORT

The UPC incorporates a universal asynchronous receiver and transmitter (UART) which provides a serial interface. This appears on the connector panel via an 8-pin mini DIN connector. Buffers external to the UPC convert the serial channel signals to RS232C levels. The UART allows data rates from 50 to 38400 Baud; a character size of 5 to 8 bits, with one start bit and 1, 1.5 or 2 stop bits; and the use of even, odd or no parity.

Serial Port Registers

The UPC contains eleven registers associated with UART operations. These are accessible in the Pbus 8-bit I/O region at 0x00CC00000. The offset address for the UART registers is programmable via one of the UPC configuration registers. The address used by the resident firmware is 0x3F8. Table 7-8 shows the UART port registers.

Access to the Transmit and Receive buffers, to the Interrupt Enable register and Divisor register is controlled by bit 7 of the Byte Format register.

ADDRESS	REGISTER
Byte Format Register Bit 7 = 0	
0x00CC003F8	Receive Buffer/Transmit Buffer
0x00CC003F9	Interrupt Enable Register
Byte Format Register Bit 7 = 1	
0x00CC003F8	Baud Rate Divisor LSB
0x00CC003F9	Baud Rate Divisor MSB
Byte Format Register Bit 7 = x	
0x00CC003FA	Interrupt Flag Register
0x00CC003FB	Byte Format Register
0x00CC003FC	Modem Control Register
0x00CC003FD	Line Status Register
0x00CC003FE	Modem Status Register
0x00CC003FF	Scratch Pad Register

Table 7-8 UPC Serial Port Registers

Transmit and Receive Buffers
A read from this location returns data received from the serial port. A write is used to load a byte for transmission.

Interrupt Enable Register
This register controls the enabling of four UART interrupt sources.

Bits 7:4 Reserved

Bit 3 Modem Status Interrupts
0 = Interrupts disabled
1 = Interrupt when Modem Register bits change state

Bit 2 Error Status Interrupts
0 = Interrupts disabled
1 = Interrupt on error

Bit 1 Transmit Buffer Interrupt
0 = Interrupt disabled
1 = Interrupt on Transmit Buffer empty

Bit 0 Receive Buffer Interrupt
0 = Interrupt disabled
1 = Interrupt when Receive Buffer contains valid data

Interrupt Flag Register
This register identifies the highest pending interrupt. It can be read to determine the source of a UART interrupt.

Bits 7:0	Reserved
Bits 2:1	Interrupt Source 11 = Error Interrupt – highest priority 10 = Receive Buffer Full 01 = Transmit Buffer Empty 00 = Modem Status
Bit 0	Interrupt Flag 0 = Valid interrupt pending 1 = No interrupts pending

Byte Format Register
This register is used to select the character length, number of stop bits, parity control and break control. It also contains a bit which controls host access to the Divisor registers.

Bit 7	Divisor register address bit 0 = No host access to divisor registers 1 = host access to divisor registers
Bit 6	Break control 1 = Force TxD pin to logic 0 (break condition) 0 = TxD pin normal operation
Bit 5	Force Parity 0 = Force Parity Disabled 1 = Force Parity Enabled
Bit 4	Parity Sense 0 = Odd Parity 1 = Even Parity
Bit 3	Parity Enable 0 = no parity generation or checking 1 = Parity generation and checking enabled
Bit 2	Stop Bits 0 = 1 Stop bit 1 = 1.5 if character length is 5 bits 2 bits if word length is 6, 7, or 8 bits
Bits 1:0	Character Length 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits

Modem Control
This register controls the handshake lines.

Bit 7:2	Reserved in SPARCbook Should be 0's
Bit 1	RTS Control 0 = Assert RTS
Bit 0	DTR control 0 = Assert DTR

Line Status Register
This register contains information about error conditions. The following conditions are signified when the corresponding bit is at 1.

Bit 7	Always 0
Bit 6	Transmitter Empty
Bit 5	Transmit Buffer Empty
Bit 4	Break Interrupt
Bit 3	Framing Error
Bit 2	Parity Error
Bit 1	Overrun Error
Bit 0	Receive Buffer Full

Modem Status Register
This register contains the information about modem lines.

Bit 7	Current DCD state
Bit 6	Current Ring Indicator state
Bit 5	Current DSR state
Bit 4	Current CTS state
Bit 3	DCD changed since this register was last read
Bit 2	Ring Indicator changed since this register was last read
Bit 1	DCR changed since this register was last read
Bit 0	CTS changed since this register was last read

UPC CONFIGURATION

A significant portion of the 82C710 circuitry is used for configuration, all of which can be performed under software control. This permits user friendly (menu-driven) UPC configuration. DIP switches and jumpers are eliminated meaning that it is not necessary to open the chassis to change the configuration of a peripheral interface.

Since the UPC is software configured, a setup program must be run whenever the configuration is changed. This process entails placing the UPC in configuration mode and programming the on-chip configuration registers.

It should be noted that although the SPARCbook provides a PC-AT style hardware environment for the UPC, it does not use a standard BIOS implementation. The SPARCbook operating system manages all device configurations and it is therefore not normally necessary for the user to access the UPC's internal registers.

UPC Configuration Sequence

In order to setup or change the configuration of the UPC, two consecutive addresses are used to select and access the internal configuration registers. These are occupied by the Configuration Register Index (CRI) which is located at an even address and is used to point to the configuration register; and the Configuration Access Port (CAP) which is located at the next address, and is used to write data to the selected register.

Before these two locations can be used, however, the UPC must be placed in Configuration Mode, and in the process an address for the CRI supplied.

The UPC is located within the 8-bit I/O region of the SPARCbook's address map. Any address chosen for the CRI will have to lie within this region.

The configuration sequence comprises three steps:

1 **Entering the configuration mode**
2 **Configuring the 82C710**
3 **Escaping the configuration mode**

Entering Configuration Mode
Assuming an address of 0x00CC00390 is selected for CRI, the following sequence would be used to enter the configuration mode:

Write 0x55 to 0x00CC002FA
Write 0xAA to 0x00CC003FA
Write 0x36H to 0x00CC003FA
Write 0xE4 to 0x00CC003FA
(Where 0xE4 is 0x390 divided by 4)
Write 0x1B to 0x00CC002FA
(Where 0x1B is the complement of 0xE4)

Following this sequence, the UPC is in configuration mode. If there is any departure from this sequence, the UPC will revert to its idle state, and the sequence will need to be started again.

Configuring the UPC
To access and write data to any of the configuration registers, two writes are needed.

First a pointer is written to the CRI; bit 3:0 provide the pointer to the configuration register, and bits 7:4 should be set to 0.

Following this, the data may be written to the selected register via the CAP.

Escaping Configuration Mode
To escape configuration mode, write any value into the configuration register F, as follows:

Write 0xOF to 0x00CC00390
Write 0x?? to 0x00CC00391 (?? = any value)

Configuration Register Description

There are sixteen configuration registers in the UPC. Settings are retained as long as standby power is maintained.

These registers are not affected by the RESET signal and are set to their default state only upon power up. Table 7-9 shows the configuration registers in the UPC with the default values upon power up, and values loaded by the SPARCbook resident firmware required for normal operation.

OFFSET	FUNCTION	POWER UP DEFAULT	SPARCbook DEFAULT
0x00	Configuration 0	0x0C	0x0C
0x01	Configuration 1	0x00	0x40
0x02	Configuration 2	0x0x	0x0x
0x03	Reserved	-	-
0x04	UART Port Address - Unused	0xFE	0xFE
0x05	Reserved	-	-
0x06	Parallel Port Address	0x9E	0xCC
0x07	Reserved	-	-
0x08	Reserved	-	-
0x09	General Port Chip Select - Unused	0xB0	0xB0
0x0A	Configuration A	0x00	0x00
0x0B	Configuration B	0x00	0xF0
0x0C	Configuration C	0xA0	0xA0
0x0D	Mouse Port Address	0x00	0xC4
0x0E	Configuration E	0x00	0x00
0x0F	Configuration Index	-	-

Table 7-9 UPC Configuration Registers

The only registers changed by the firmware are:

Configuration Register 1
Parallel Port Address
Configuration Register B
Mouse Port Address

Configuration Register 1

Only bit 6 in this register is changed in order to configure the parallel port for printer operation. All other bits retain their default settings.

Bit 7	Reset Control This bit determines the manner in which the Reset pin affects the serial port. The default is 0, normal Reset.
Bit 6	Parallel Port Mode 0 = Output, printer only 1 = Bi-directional
Bit 5	UART CTS Control – Not used
Bit 4	UART DSR Control – Not used
Bit 3	UART DCD Control – Not used
Bits 2:0	Reserved

Parallel base address

This register holds the base address of the parallel port. The contents of this register are multiplied by 4 to give the base address of the parallel port. A setting of 0xCC identifies an address of 0x330.

Configuration B

This register is used to set the polarity of the interrupts issued by the UPC. All interrupts in the SPARCbook are prioritized by the MPI (see Chapter 5) which requires requests to be configured as active low.

Bit 7	Mouse Interrupt Polarity 0 = Active High 1 = Active Low†
Bit 6	Floppy Interrupt Polarity 0 = Active High 1 = Active Low†
Bit 5	UART Interrupt Polarity Not used, should be 0

Bit 4	Parallel Port Interrupt Polarity 0 = Active High 1 = Active Low†
Bit 3:0	Not Used

†– Required by SPARCbook

Mouse Port Address

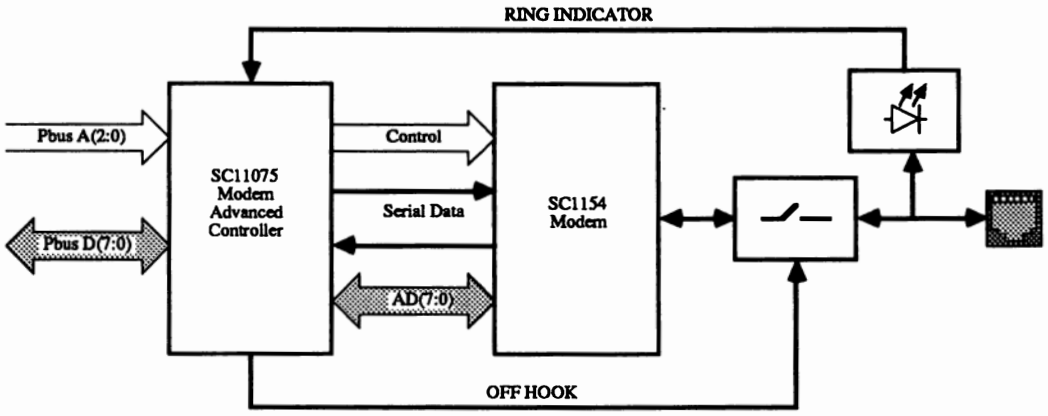
This register holds the base address of the mouse port. The contents of this register are multiplied by 4 to give the base address of the parallel port. A setting of 0xC4 identifies an address of 0x310.

CHAPTER EIGHT
MODEM

The SPARCbook is equipped with a modem interface which allows its connection via a public telephone system (in the US only) for data communications or facsimile transmission.

The modem interface is implemented on the base board with a two-chip set comprising the SC11075 modem access controller (MAC) and the SC11054 modem, both from Sierra Semiconductor Corporation. These provide an extremely compact and low power interface with power management facilities to reduce power consumption when the modem is not in use.

Figure 8-1 shows the implementation of the modem interface in the SPARCbook.



MODEM OVERVIEW

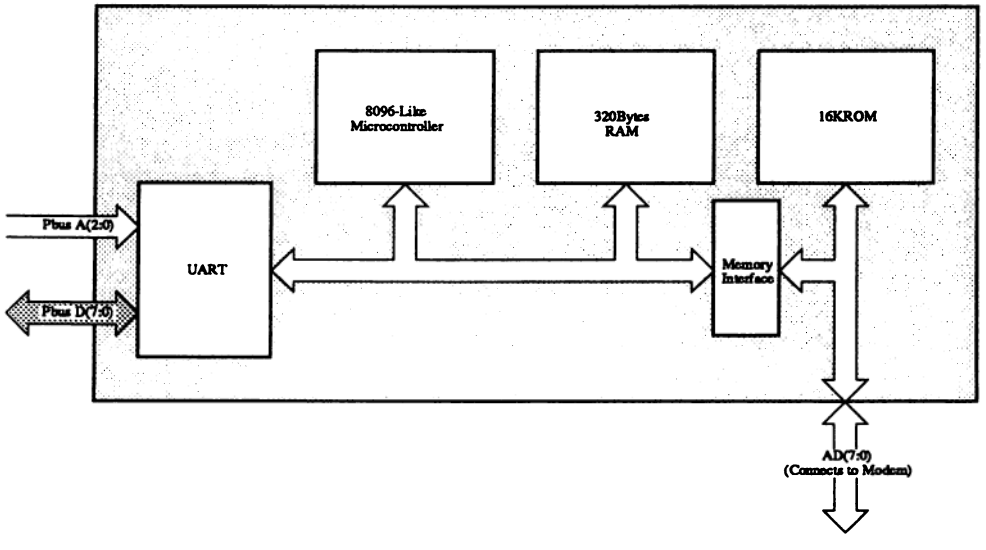
The SC11075 provides a direct interface between the host system and SC11054 modem; it incorporates an Intel 8096 equivalent processor core, supporting an AT command set; contains a built-in 16C450 equivalent UART; and contains 16Kbyte of on-chip ROM and 320bytes of RAM.

The SC11054 is a complete 2400bps (bits per second) modem IC which, when combined with the SC11075, provides Sendfax capability of up to 9600bps.

The modem interface can be controlled using Hayes compatible commands written to the UART section of the MAC. Commands are sent to the interface using character strings, and these are interpreted by the MAC's processor core and acted on appropriately.

Command characters and data for transmission onto the telephone line are written into the UART's transmit buffer. Data received from the telephone line and status information from the MAC can be read from the UART's receive buffer.

The modem interface operates in two modes: Command Mode or On-Line Mode. In Command Mode, characters written to the transmit buffer are regarded as being command characters. In the On-line mode, data written to this location (apart from the escape sequence +++) are transmitted onto the telephone line.



INTERFACE CONTROL

The host may access the UART registers within the MAC; the MAC contains a number additional registers but these are accessible only by the internal microcontroller. The UART registers are accessible at base address 0x00C400000 and are shown in Table 8-1.

Bit 7 in the Line Control register is used as a pointer to either the transmit and receive buffers and the IRQ Enable register, or to the Divisor Latch registers. Bit 7 must contain '0' to gain access to the transmit and receive Buffers.

The host is responsible for managing the transmit and receive buffers. During transmit operations, the host must supply data at a sufficient rate to ensure that there is always a new character for the modem to transmit; the modem is unable to wait and will regard an empty buffer as an error condition and disconnect the telephone line. During receive operations, the host must read the data in the buffer in time for the next incoming character from the telephone line.

The Modem generates interrupts on IRQ5 when the receive buffer contains a character, when the transmit buffer is empty, and also to signal error conditions. All interrupt requests within the SPARCbook are maskable and are prioritized by the MPI, encoding IRQ5 from the modem on the CPU level 8 interrupt request (see Chapter 5).

ADDRESS	REGISTER	ACCESS
00C400000	Receive Buffer	R
	Transmit Buffer	W
	Divisor Latch (LSB)†	R/W
00C400001	Interrupt Enable	R/W
	Divisor Latch (MSB) †	R/W
00C400002	Interrupt ID	R
00C400003	Line Control	R/W
00C400004	Modem Control	R/W
00C400005	Line Status	R/W
00C400006	Modem Status	R/W
00C400007	Scratch Pad	R/W

Table 8-1 UART Registers

Note: † The divisor latch can be accessed only when bit 7 of the Line Control Register is set.

MAC Integral UART Registers

This section describes the more significant registers in the UART, from the point of view of managing the flow of commands and data between main memory and the UART.

Interrupt Enable Register

This register contains interrupt control bits. Setting one of the interrupt enable bits has the effect of enabling the associated interrupt request.

Bits 7:4	Reserved
Bit 3	Enable Modem Status Interrupt
Bit 2	Enable Receiver Line Status Interrupt
Bit 1	Enable Transmitter Holding Register Empty Interrupt
Bit 0	Enable Received Data Available Interrupt

Interrupt Identification Register

This register provides the identity of the highest priority pending interrupt condition, and a flag which indicates whether or not there is an interrupt pending. The UART prioritizes the internal interrupt requests as shown below.

Bit 7:3	Reserved
Bit 2:1	Interrupt ID
	11 = Receiver Line Status – highest priority
	10 = Receiver Data Available
	01 = Transmit Buffer Empty
	00 = Modem Status – lowest priority
Bit 0	Interrupt Pending Flag
	1 = No Interrupt Pending
	0 = Interrupt Pending

Line Status Register

The Line Status register contains infromation which allows the condition of the transmit and receive buffers to be monitored.

Bit 7	Reserved
Bit 6	Transmitter Empty
	This bit when set indicates that both the transmit buffer and the transmit shift register are empty
Bit 5	Transmitter Buffer Empty
	This bit when set indicates that the transmit holding register is empty. It is cleared automatically when the host writes data into the transmit holding register
Bit 4	Break Interrupt Flag

Bit 3	Framing Error Flag
Bit 2	Parity Error
Bit 1	Overrun Error
Bit 0	Receiver Data Ready

THE AT COMMAND SET

This section describes the basic AT command set supported by the SC11075/1154 chipset, and the Sendfax extended command set. Table 8-2 provides a summary of the commands supported by the SPARCbook.

CODE	DESCRIPTION
Basic AT Commands	
A	Go Off-hook in Answer Mode
A/	Re-execute Previous Command
AT	Attention Characters
B	Bell/CCITT Protocol
D	Dial Telephone Number
E	Command Echo
Hn	Switch Hook Control
In	Identification
L	Speaker Volume
M	Speaker Control
O	Return to Online
Qn	Quiet Command Reset Code
Sn=	Writing to S-Register
Sn?	Reading From S-Register
V	Enable Short-form Result Code
X	Enable Extended Result Code Set
&F	Fetch Factory Configuration
+++	Switch to Command Mode, but retain line connection
Sendfax Commands	
#Bn	Speed Control
#En	Received Frame Display Format Selection
#Fn	Mode Control
#Pn	Number of Pages to be Transmitted
#Rn	Resolution Control

Table 8-2 Hayes AT Command Set Summary

The modem enters the Command Mode when it is reset, when it loses contact with a remote modem, or when it is in Data Mode and receives the escape sequence +++.

When the modem is in Command Mode, it will accept instructions in the form of command lines, and in many instances will return responses. The Command Mode allows the modem to be instructed to perform functions such as originating or answering a call. When the modem makes a connection with a remote modem, it sends a connect response to the CPU.

Commands lines are written to the UART location at 0x00C400000 and responses are read from the same location.

All commands lines begin with the characters AT (with the exception of A/), may contain one or more commands and are terminated by a RETURN. Command lines may contain up to 40 characters, not including spaces. All characters before AT are ignored.

Basic Command Set

A	**Answer Incoming Call**	This forces the modem to go off-hook in answer mode.
A/	**Re-execute Previous Command**	The A/command repeats the last command. It is not preceded by the AT characters or terminated by pressing RETURN.
AT	**ATention Characters**	These characters must appear at the beginning of all command lines.
Bn	**Bell/CCITT Protocol**	This command selects the communication standard: n = 0 CCITT V.21 n = 1 V.22/V.22bis n = 2 V.23
D	**Dial Telephone Number**	This command causes the modem to dial up a remote modem. The following modifiers may be added: P Pulse Dial T Touch-tone Dial R Originate Call in Answer Mode W Wait for Dial Tone , Delay a Dial Sequence

		@ Wait for Quiet ! Go On-hook ; Return to Command Mode S=n Dial a Stored Number
En	**Echo Command Characters**	This controls whether the modem echoes command characters back to the host: n = 0 Characters Echoed n = 1 Characters Echoed
Hn	**Switch Hook Control**	n = 0 Modem goes on-hook (hangs up) n = 1 Modem goes off-hook to access the telephone line
In	**Identification**	This command causes the modem to respond to the host by providing identification codes: n = 0 Request Product Code n = 1 ROM Checksum n = 2 Return OK Response n = 3 Manufacturers ID n = 4 Configuration Mode n = 33 Sierra ID
O	**Return to Online**	This command is used to return to the MAC to the Data Mode following an escape sequence used to enter Command Mode.
Qn	**Command Response Control**	This command controls whether the modem provides responses to commands: n = 0 Return Response to host n = 1 Do Not Send Response
Sr=n	**Change Register Value**	This command selects an S register and changes its contents: r = 0 - 27 n = 0-255
Sn?	**Read S Register**	This command returns the contents of an S register: n = 0 - 27

Vn	**Response Format**	This command is used to select the format of response made by the modem to the host: n = 0 Single Digit Response n = 1 Extended Response
Xn	**Select Extended Response Set**	
&F	**Fetch Factory Configuration**	This command recalls the factory settings of the modem chipset.

Sendfax Command Set

#Bn	**Speed Control**	This command selects the initial fax transmission speed: n = 4 2400 bps n = 5 4800 bps n = 6 7200 bps n = 7 9600 bps
#En	**Received Frame Display Format**	This command selects the display format for HDLC frames: n = 0 Disable Display of HDLC Frames n = 1 Display Frame in Binary Format n = 2 Display Frame in 2 Digit ASCII Hex Format
#Fn	**Mode Control**	This Command selects the modem operating mode n = 0 Return to Normal Mode n = 1 Enter Fax Mode
#Pn	**Number of Pages to be Transmitted**	n = 0 to 255
#Rn	**Resolution Control**	This command selects the resolution used for document transmission: n = 0 Send Document with Normal Resolution n = 1 Send Document with Fine Resolution

S-REGISTERS

The MAC maintains a set of registers which can be written to and read using the S-register commands described previously. There are twenty-eight S-Registers summarized in Table 8-3.

REGISTER	FUNCTION
S0	Rings Until Auto-answer Calls
S1	Count Number of Incoming Rings
S2	Escape Character
S3	Carriage Return Character
S4	Line Feed Character
S5	Backspace Character
S6	Dial Tone Wait Time
S7	Wait Time for Remote Carrier
S8	Comma Pause Time
S9	Carrier Detect Response Time
S10	Delay Time Between Loss of remote Carrier and Hang-up
S11	DTMF Dialing Speed
S12	Escape Guard Time
S13	Reserved
S14	E, Q, V Commands; Accept or Ignore Commands; TPDA Commands
S15	Reserved
S16	Modem Loopback Tests
S17	Reserved
S18	Modem test Timer
S19	Reserved
S20	Reserved
S21	J, &R, &D, &C, &S, Y Commands
S22	L, M, X, &P, &T4, &T5 Commands, DTE Speed, Parity
S23	&T4 and &T5 Commands, DTE Speed, Parity
S24	Reserved
S25	DTR Delay for Synchronous Operation
S26	RTS/CTS Delay
S27	&M, &L, &X and B Commands

Table 8-3 S-Register Summary

Reading an S-Register
The current value of an S-Register can be read using the Read S-Register command. For example, the command line:

```
ATS0?
```

will cause the modem to respond by returning the contents of the S-Register 0 in three-digit form followed by OK.

Changing an S-Register
The contents of an S-Register can be changed using the Change S-Register Command. For example, the command line:

```
ATS0=2
```

will cause the modem to change the contents of S-Register 0 to 2. The modem responds by returning OK.

FAX RESPONSES

When a command is executed by the MAC, the hardware may report the result to the host. Status information is returned to the host in the form of responses. The MAC may return short response codes consisting of a single character (or digit), or verbose responses consisting of a string of characters. The responses produced as a result of a fax session are summarized in Table 8-4.

VERBOSE	DIGIT	MEANING
CED	a	Answertone Detected
CFR	g	Remote Machine confirmation to receive
CONNECT2400/FAX	w	Connection Speed 2400bps
CONNECT4800/FAX	x	Connection Speed 4800bps
CONNECT7200/FAX	y	Connection Speed 7200bps
CONNECT9600/FAX	z	Connection Speed 9600bps
CRC ERROR	e	Error in received frame
CPR	c	Repeat Request
CSI	-	Remote Machine Identification
DCN	d	Disconnect
DIS	b	Remote Machine Capabilities Frame
FTT	f	Failure to Train
INVALID FRAME	i	Received Frame Invalid
MCF	m	Message Received OK
RTN	h	Message Not Received OK
RTP	j	Retrain Positive

Table 8-4 Response Summary

CHAPTER NINE
ETHERNET INTERFACE

The SPARCbook is equipped with an Ethernet interface which allows its connection to a network for fast communications with other workstations, fileservers, printservers, and so on.

This section of the SPARCbook Technical Reference describes how the Ethernet interface in the SPARCbook is implemented. An in-depth discussion about networking is beyond the scope of this manual, but a brief overview is provided.

Figure 9-1 SPARCbook Network Connection

NETWORKING OVERVIEW

Ethernet is a send and receive half duplex serial communications system capable of data transfers at up to 10 million bits per second.

A networking standard has been formalized by the Institute of Electrical and Electronic Engineers in the IEEE802.3 standard. This defines the type of cables and connections used to form the network, the topology of the network and the protocols which control access to the network.

Coupling the SPARCbook to a network cable requires an external network transceiver and transceiver cable (sometimes called a 'drop cable'). The network transceiver is attached to the network cable, and the transceiver cable is connected between the 15-way locking D-type connector on the rear of the SPARCbook and the network transceiver, as illustrated in Figure 9-1.

The transceiver transmits and receives the data carrier signal on the network cable. In addition, it provides isolation between the SPARCbook and network so that its addition to or removal from the network does not affect the network. The transceiver is also responsible for detecting 'collisions', which occur when two or more nodes attempt to transmit simultaneously.

Ethernet Protocol

The IEEE802.3 standard specifies a Carrier Sense Multiple Access with Collision Detection protocol (CSMA/CD) as a simple and efficient means of determining how a node may gain access to and transmit information over the network.

Carrier Sense
A node wishing to transmit data over the network must first listen to the cable to detect whether another station is already using the cable. Transmission is deferred until the cable is clear.

Multiple Access
Any station wishing to transmit may do so if the cable is clear. There is no central controller to allocate use of the network.

Collision Detection
This provides a mechanism for resolving the situation which may occur when more than one station starts to transmit at precisely the same moment. Each transmitting station listens to the signal on the cable. If more than one station is transmitting, then the signal on the cable will no longer be the same the signal being transmitted, but will be garbled; i.e. a collision will have occurred. In this situation

the competing stations continue to transmit for a time in order to 'jam' the network, which signals the collision to other stations on the network. All competing stations then suspend tarnsmission for a random delay before attempting again.

Basic Operations

At any given time, stations attached to the network function as either a transmitter or receiver. Data is passed between stations in bursts (or packets) at up to 10 million bits per second.

An Ethernet packet contains the destination address of the intended recipient, the source address identifying the sender, the data length and then the data itself. The format of an Ethernet packet is shown in Figure 9-2.

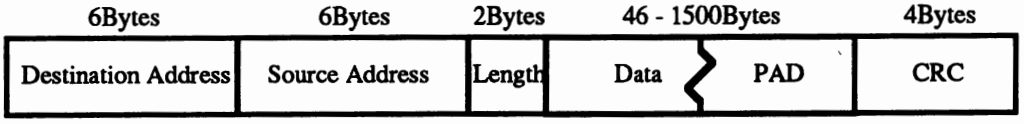


Figure 9-2 Ethernet Packet

Transmission
Packets are placed into an area of buffer memory by the CPU ready for transmission by the Ethernet controller. The Ethernet controller automatically performs a cyclic redundancy check (CRC) during transmission and appends a 4byte result to the end of the packet as a frame check. The overall length of the packet must be between 64 to 1518bytes, including the checksum; padding may be added by the controller to achieve the minimum required length if required. The controller also adds a preamble and synchronization pattern to the start of the packet to enable detection and synchronization by the receiver. A Manchester encoder produces a differential signal which carries the data and an embedded clock signal onto the network cable.

Reception
When the receiver node detects a carrier on the network, it passes the incoming packet to the Manchester decoder within the controller. A phase locked loop synchronizes with the preamble to allow recovery of the clock signal and data. The controller examines the destination address in the header and stores packets containing a valid address in a buffer memory where it can be read by the CPU. During reception, the receiver performs a CRC and checks the result against the frame check. Discrepancies are reported as CRC errors.

Errors

The Ethernet controller reports any errors, such as CRC errors or missed packets, to the host using interrupts and error bits in its status registers.

Addressing

There are three addressing modes. The first mode is physical addressing which uses the 48-bit (6byte) destination address in the packet header to uniquely identify one receiver. The physical address requires comparison by a receiver with its own unique 'Node ID'. The issuing of Ethernet addresses is overseen by the IEEE, ensuring globally that no two stations share the same physical address. Each SPARCbook has its own unique Ethernet address programmed into it during manufacture.

The second mode is Multicast (or Logical) Addressing which may be used to send packets to several receivers at once. The third mode is Broadcast Addressing, where packets are sent to every node attached to the network.

THE SPARCBOOK ETHERNET IMPLEMENTATION

The Ethernet interface in the SPARCbook is implemented with a Fujitsu 86960 Network Interface Controller with Encoder/Decoder (NICE). NICE is a highly integrated device which incorporates a Manchester encoder and decoder, a data link controller, and a buffer memory management unit in a single chip.

Figure 9-3 illustrates the architecture of the Ethernet interface in the SPARCbook.

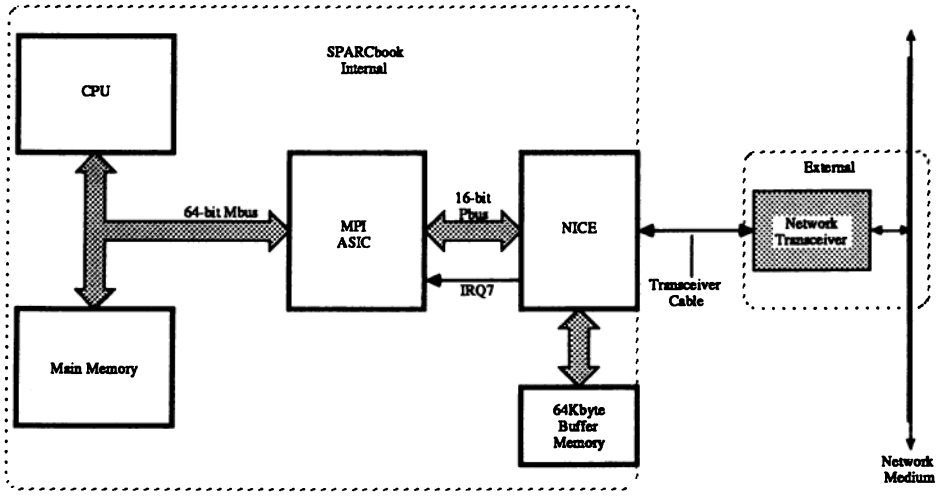


Figure 9-3 Ethernet Interface Architecture

The NICE is provided with a dedicated 64Kbyte buffer memory accessible to the SPARC CPU (or host) via a 16-bit buffer port. The NICE allows multiple data frames to be transmitted directly from the buffer to the network with a single transmit command.

The interface between the NICE and the SPARC CPU is provided by the MPI (see Chapter 5). The MPI transfers data between the buffer memory and the Mbus via internal FIFOs and provides data packing between the Mbus and Pbus, allowing the CPU to access the buffer memory using 64-bit single cycle or burst accesses.

Fabricated in low power CMOS technology, the NICE features a stand-by mode which can be used to reduce its power consumption when the network interface is not in use. The SPARCbook's power management facility takes full advantage of this feature in order to maximize battery life.

FUNCTIONAL DESCRIPTION OF THE NICE

The MB86960 incorporates four major functional elements, as shown in Figure 9-4, these are:

- **Host System Interface**
- **Buffer Management Unit**
- **Data Link Controller**
- **Manchester Encoder and Decoder**

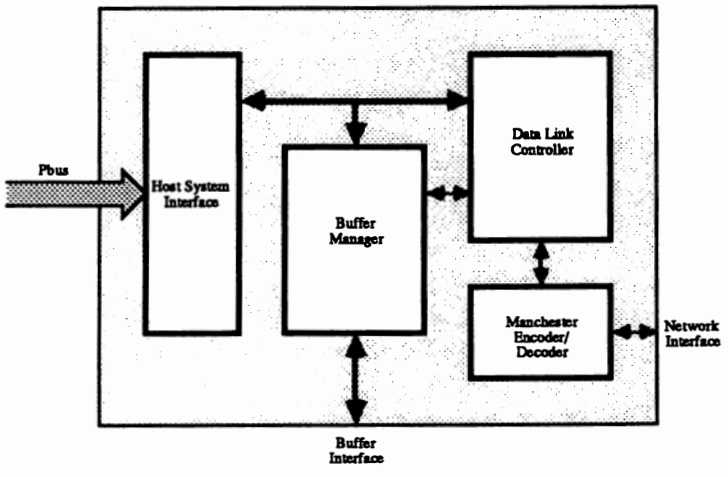


Figure 9-4 NICE Simplified Block Diagram

NICE Host System Interface

The host system interface provides the connection between the NICE's internal registers and buffers and the Pbus and hence the SPARC CPU. The NICE contains three register sets. These are the Data Link Controller Register set, the Multicast Address Register set, and the Buffer Port Memory Register set. These are shown in Figure 9-5.

Data Link Controller Register Set

The Data Link Controller register set contains sixteen registers which are used by the host to initialize and control the activity of the NICE, and to provide the host with error status information. The lower eight registers of this group occupy the lower eight address locations within the NICE's address space, and are always accessible. The upper eight registers in this group share the NICE's remaining address space with the two other register groups.

Bits 2 and 3 in Control Register 2 are used to select between the register groups sharing the upper half of the NICE's address space, as illustrated in Figure 9-5.

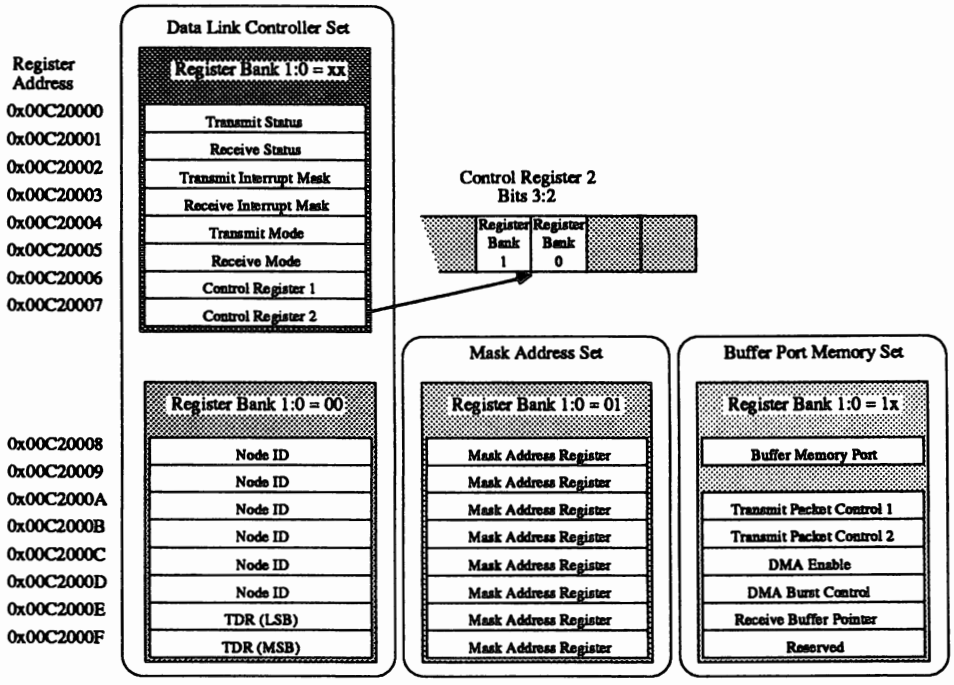


Figure 9-5 NICE Internal Registers

After initialization, it is normally only necessary to access the lower eight registers in the Data Link Controller registers, to obtain status information, and the Buffer Memory Port to read and write data packets and to issue transmit commands.

The NICE supports 8- or 16-bit accesses, little or big-endian byte ordering and can provide DMA support. In the SPARCbook implementation the NICE should be set up with a 16-bit interface and little endian addressing. The NICE's own DMA facilities should be disabled; these are not used because host accesses to the NICE are controlled by the MPI.

When the NICE receives a data packet from the network, it checks the header information for a valid destination address.

Physical addresses, those with a '0' in the least significant bit, are compared with the contents of the Node ID registers in the Data Link Controller Register set. These combine to form a single 48-bit node ID. When an address match occurs, the receiver transfers the incoming packet into the receive buffer memory.

Mask Address Register Set
All eight registers in this set combine to provide a 64-bit logical address filter (or hash filter). A multicast (or logical) address is identified by a '1' in the least significant bit of the destination address contained in the packet header.

When the NICE detects a '1' in the first bit of an incoming address, it passes the address through its internal CRC logic. This produces a 32-bit result. The least significant 6bits of this result are used to provide an address for one of the 64bits contained in the eight multicast address registers. If the bit selected is a '1', then the packet is accepted into the receive buffer.

This mechanism provides the NICE with the ability to recognize up to 64 destination addresses from the network as being possibly valid for the SPARCbook; the software has to examine the validity of the address, but has to do so far less often than if every address had to be checked in software.

Multicast address filtering is illustrated in Figure 9-6

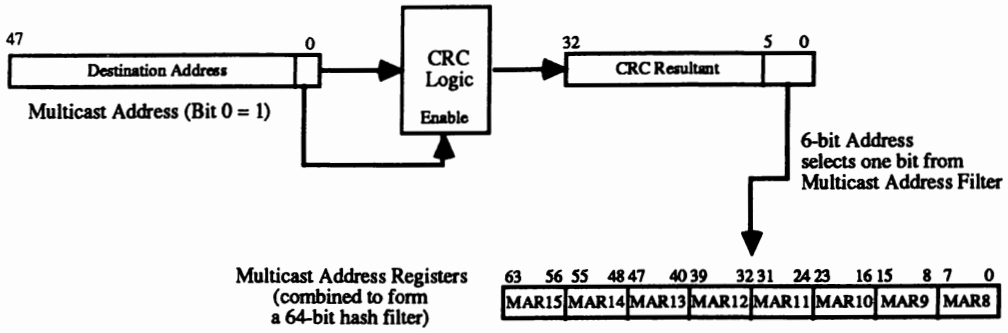


Figure 9-6 Multicast Address Filterering

Buffer Port Memory Register Set
The Buffer Port Register set provides the host with access to the transmit and receive buffer memory and also contains packet control and DMA control registers.

The Buffer Memory Port allows the host to write data into the transmit buffer using 16-bit transfers. The NICE maintains internal pointers so that repeated writes to this location are automatically directed to the next 16-bit location. Similarly, the internal pointers allow successive reads of the Buffer Memory Port to access successive 16-bit locations in the receive buffer memory.

Bit 7 in Packet Control Register 1 is used by the host to command a start of transmission, and bits 6:0 are used to inform the buffer manager of the number of packets to be transmitted, allowing back to back transmission of up to 128 packets.

The DMA control registers should be configured to disable DMA operations by the NICE.

Buffer Manager

Two SRAM devices provide the NICE with a 64Kbyte buffer memory. The Buffer Manager within the NICE automatically prioritizes and services requests for access to the buffer from the host and Data Link Controller. It updates all buffer pointers and allocates memory space for incoming packets, and provides status information via the status registers.

In the SPARCbook, the CPU is able to write to the Buffer Memory Port using 64-bit single cycle accesses or burst accesses via the MPI (see Chapter 5). The MPI automatically converts a 64-bit access into a sequence of four 16-bit cycles on the Pbus.

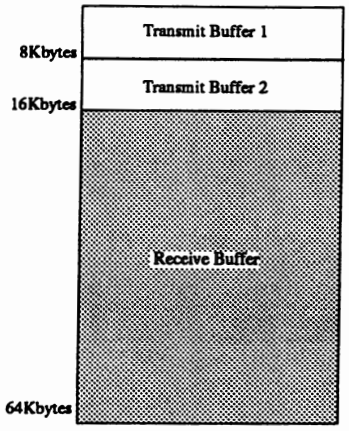


Figure 9-7 Transmit and Receive Buffer Memory Organization

The buffer contains two areas: one for transmit data and one for receive data. These are shown in Figure 9-7.

Transmit Buffer

The transmit buffer may be configured in software, via Control Register 1, as a single banked 16Kbyte buffer or a dual banked 4, 8 or 16Kbyte buffer. With a dual banked configuration, the host is able to write to one bank while the Data Link Controller transmits from or receives packets into the other bank. The internal pointer controls bank and location selection within the buffer transparently to the programmer. The NICE switches banks each time a Start Transmission command is issued.

When all packets to be transmitted have been loaded into the buffer, transmission is initiated by programming the number of packets in Packet Control Register 1 bits and asserting the transmit start bit in the same register.

Receive Buffer

The receive buffer occupies the remaining space in the buffer memory after the allocation of the transmit buffer size.

Each received packet is preceded by a 4byte header. The first byte provides status information, the second contains irrelevant data, and the remaining two contain the packet length. Reading more than the indicated number of bytes will advance the internal pointers beyond one packet and into the beginning of the next or may cause a bus read error. Reading fewer bytes will leave garbage at the start of the next packet.

INTERRUPTS

All interrupts in the SPARCbook are handled via the MPI (see Chapter 5). The NICE produces an active low interrupt on IRQ level 7.

Interrupts are generated when a packet is transmitted or received, or if a transmit or receive error occurs and the corresponding mask is set.

CHAPTER TEN
DISPLAY CONTROLLER

The SPARCbook uses a CL-GD6410 to provide control of the LCD flat panel display built in to the SPARCbook and an external CRT monitor. It provides a VGA subsystem for SPARCbook allowing the user to operate an externally connected analogue CRT VGA monitor without disabling the LCD display, and without using a VGA add-on card. The CL-GD6410 supports the generic VGA monitor concept advanced by the Video Electronics Standards Association (VESA) for 350, 400 and 480 scan line monitors.

Models fitted with a color LCD display also contain a CL-GD6340 Color LCD Controller. This provides additional interfacing between the display controller and color LCD display. To software, the display controller and color LCD controller appear to be one device. Each one decodes addresses for accesses to its internal registers and color palette.

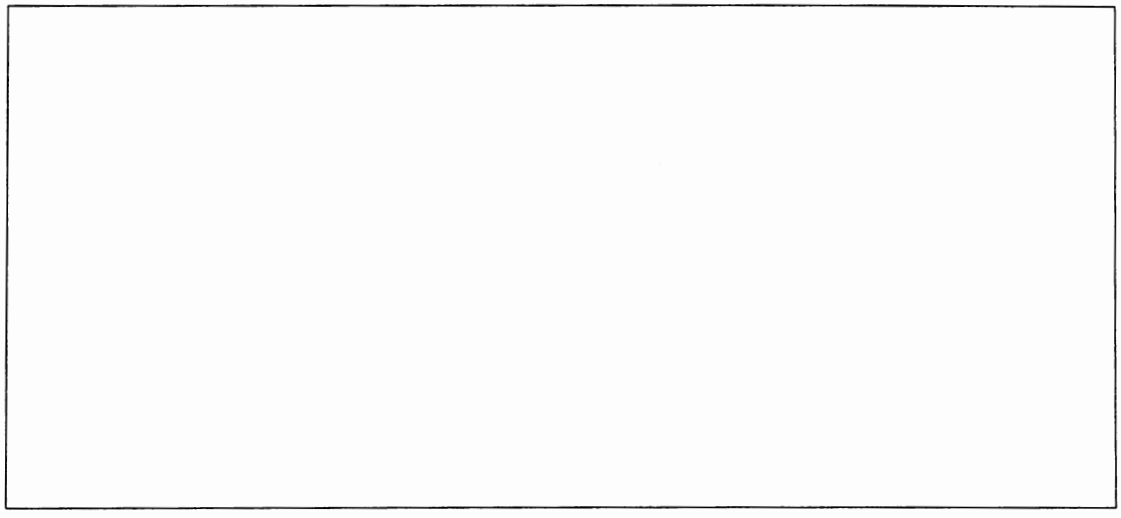


Figure 10-1

DISPLAY INTERFACE OVERVIEW

The CL-GD6410 interfaces directly to the CPU via the 16-bit Pbus and MPI, to 256 Kbytes of video memory and to the display devices.

The SPARCbook is equipped with a 640 x 480 pixel monochrome or color LCD panels. The color version is able to display up to 16 colors simultaneously from a palette of 262144, while the monochrome version provides color to grayscale rendering to display 16 gray levels. Color to grayscale rendering allows color images to be displayed to good effect on the monochrome LCD display.

The CL6410 provides a host interfacewhich allows access to its internal control register, color palette and display memory. The CL6410 also incorporates a digital to analogue converter which transforms color information from the color palette into the RGB signals required to directly drive a color CRT display.

HOST INTERFACE

In the SPARCbook, the MPI provides the SPARC CPU with direct access to the CL-GD6410. Memory reads and writes use 16-bit bus cycles, while I/O commands use 8-bit bus cycles.

Some of the internal registers within the CL-GD6410 and CL-GD6340 can be accessed directly by the host, but many others are accessed via index registers. To access an extension register, a pointer to the required register is written to the index registers and then, in a second access cycle, the data is written to or read from the selected register. During register accesses by the host, memory accesses by the CL6410 and screen refresh activities continue to occur uninterrupted.

Host accesses to video memory are channelled via the CL-GD6410. The SPARCbook resident firmware establishes the proper address/data/timing parameters in the CL-GD6410 registers in order to transfer to and from video memory. The CL-GD6410 also contains an intelligent address sequencer that allocates video memory cycles to the host, to the VRAM refresh and the display CRT controllers.

Although the SPARCbook provides a PC-AT style hardware environment for the display controllers, it does not provide a standard BIOS. Configuration of the display interface is normally carried out via the operating system, as described in the *SPARCbook User's Guide*.

ADDRESS	VGA PORT
0x00C0003B4	CRTC Index MP (r/w)
0x00C0003B5	CRTC Data MP (r/w)
0x00C0003C0	Attribute Controller Index (r/w) Data (w)
0x00C0003C1	Attribute Controller Data (r) in VGA Attribute Controller Data (r/w) in EGA
0x00C0003C2	Misc Output (w), Feature (r)
0x00C0003C3	Motherboard Sleep Address (r/w) In address space only if switched enable.
0x00C0003C4	Sequencer (r/w)
0x00C0003C5	Sequencer (r/w)
0x00C0003C6	RAMDAC Pixel Mask (CPRd and CPWr)
0x00C0003C7	RAMDAC Address Reg. Read Mode(w) (affects CPWr on writes) RAMDAC Status Register
0x00C0003C8	RAMDAC Address Register Write Mode (CPRd and CPWr)
0x00C0003C9	RAMDAC Data (CPRd and CPWr)
0x00C0003CA	Feature Control (r)
0x00C0003CC	Misc Output (r)
0x00C0003CE	Graphics Controller and Extensions Index (r/w)
0x00C0003CF	Graphics Controller and Extensions Data (r/w)
0x00C0003D4	CRTC Index (r/w)
0x00C0003D5	CRTC Data (r/w)
0x00C0003DA	Feature Control (w), Display Status (r)
0x00C0046E8	AT Adapter Sleep Address (r/w) (if switch enabled)

Table 10-1 Display Controller Memory Map Summary

Host Accesses to the Color Lookup Table

To write a color definition to the lookup table, a value specifying the address location in the lookup table is first written to the RAMDAC Write Mode address register. Following this, the color values for the red, green and blue intensities are written in succession to the RAMDAC Pixel Data register. After the blue data is latched, the new color data is transferred into the lookup table at the defined address, and the address register is incremented automatically to point to the next palette location.

Read operations are similar but use the RAMDAC Read Mode Address register to point to locations in the color palette.

If the address register is loaded with a new starting address while an unfinished sequence is in progress, the system resets and starts the new sequence.

Display Access to Video Memory

The GD6410 contains video shift registers which interface with the display. It performs the operations necessary to fetch scan line data from the font bitmaps, while seperately controlling the video memory address buses.

The display interface in the SPARCbook is operated in a bit-mapped graphics mode, with 4bits/pixel. The pixel data is used to address entries in the color palette, selecting red, green and blue color value for that pixel to be output to the DAC. The GD6410 monitors the active and inactive areas of the screen and cursor position, and supplies the HSYNC, VSYNC, BLANK and Display Enable screen control signals.

The GD6410 is connected directly to a monochrome dual-scan LCD display. An additional 64K x 4 DRAM is provided to act as a frame accelerator for split-panel data formating. Data from the frame accelerator and the video memory is supplied in parallel to the 4-bit upper and lower panel data buses of the LCD flat panel. This technique maintains normal display contrast but reduces the power consumption of the video circuitry becuase the rate at which data is fetched from video memory is effectively half the panel frame rate.

In the color version, the CL-GD6340 takes the pixel information for the LCD and provides color mapping for the color LCD display.

INTERNAL RAMDAC OPERATION

The GD6410 and GD6340 both include an internal RAMDAC. A RAMDAC consists of an area of RAM containing a color lookup table (or color palette) and a digital to analogue converter.

The color palettes contain 256 x 18 color lookup table which is addressed by pixel information from the VRAM to provide three 6-bit color values for each pixel. The digital to analogue converter converts the three 6-bit values from the palette into the analogue red, green and blue video signals required to drive a CRT. The RAMDAC also provides a pixel mask register and a border color register.

The two color palettes occupy the same address in the SPARCbook's address map so that the color mappings contained by the two display controllers are always identical. Since the external CRT is controlled by the GD6410 in both the monochrome and color models of the SPARCbook, identical color palettes in the two devices ensures color consistency between the color LCD and CRT displays

SIMULTANEOUS DISPLAY ENABLING

Simultaneous display operations are controlled by the CL-GD6410 using the Display Mode Register, the Write Control Register, and the CRTC Test Register

The CL-GD6410 includes all registers and data paths required for VGA compatibilty. VGA enhancements include sixteen simultaneously loadable text fonts , Write Mode 3 and readable registers. The CL-GD6410 supports extended-resolution display modes with CRT displays.

Extended graphics resolution of 800 x 600 mode with a 4:3 aspect ratio can be displayed using multifrequency analog CRT monitors.

High-resolution text modes of between 100 columns by 30 rows and 132 columns by 60 rows can be supported on analog CRT monitors by the CL-GD6410. Text modes of up to 100 columns by 30 rows are supported for 640 x 480 LCD flat panel displays.

Grayscaling and color rendering for the LCD displays is accomplished by modulating the ON-to-OFF time of individual pixels in the panel and allowing the eye to integrate the superimposed pixels to 16 perceptible grays levels on a monochrome display or 256 colors on a color display. Flicker is eliminated by proprietary Cirrus Logic techniques involving distribution of time between ON and OFF pixels during frame modulation.

INTELLIGENT POWER MANAGEMENT AND SEQUENCING

Notebook and laptop PCs have stringent power limitations due to battery operation and heat dissipation. To meet these needs, the CL-GD6410 is manufactured using low-power CMOS technology. In addition, the CL-GD6410 has programmable output pins as well as other intelligent power management features

Normal Mode Conditions

- **Power to LCD panel and full screen refresh**
- **CPU access to video memory**
- **Refresh to video memory**
- **CPU access to the I/O registers and the RAMDAC**

Standby Mode Conditions

- **No power to LCD panel; no screen refresh**
- **Panel power Sequencing is observed**
- **CPU access to video memory, I/O registers and the RAMDAC**
- **Refresh to video memory**
- **Frequency Synthesizer remains powered-up**

The primary power savings in this mode comes from shutting down power to the LCD panel only. Since there is no screen refresh, normal clock rates are not required and may be replaced by slower clock rates to further reduce power consumption. Any RAMDAC I/O can be executed.

The system will recover from standby mode after detecting either video memory read or write accesses or the presence of the SSCLK signal. If power sequencing is in progress, then the CL-GD6410 will allow the sequencing to complete before exiting from standby mode.

Power-save timers that allows it to be programmable in increments of 1 minute up to 63 minutes. If the power-save timer is enabled, the programmed time-out time from the last stimuli is used to automatically switch to standby mode. The timer can be activated by either the SSCLK signal or by reads or writes to CPU memory access.

Suspend Mode

- **No power to LCD panel; no screen refresh**
- **Panel power sequencing is observed**
- **Refresh to video memory continues**
- **No CPU access to RAMDAC, video memory, or registers**
- **Frequency Synthesizer is powered-down**

The power savings in this mode occurs because host access to video memory is denied, and a slower clock is in use. This slow clock refreshes video memory by keeping CAS low and pulsing on RAS throughout the Suspend Mode. If slow-refresh DRAM is used, a clock running as slow as 32 kHz can be used, clocking both edges. Other than this refresh logic, the rest of the CL-GD6410 has no clocks, reducing power consumption even further.

Suspend Mode can be activated or deactivated, under program control.

Shut Down

Prior to initiating a system-wide shutdown, the system can save the state of the video subsystem so it can be restored later. The CL-GD6410 allows the system to save or restore the status of all controller registers.

A pixel word mask is incorporated to allow the incoming pixel address to be altered or masked, permitting changes to the color lookup table contents to be made immediately. This feature allows special display operations such as flashing objects and overlays to be created.

The color lookup table contents are accessed via its 8-bit-wide host interface. An internal synchronizing circuit allows the color value accesses to be completely asynchronous to the pixel video operation.

CHAPTER ELEVEN
MK48T02 BATTERY BACKED REAL TIME CLOCK AND SRAM

The MK48T02 is a 2Kbyte zero power timekeeper and RAM which combines a CMOS SRAM, a bytewide accessible real-time clock, a crystal oscillator and a long life lithium-carbon-monofluoride battery in a single package. The device appears as an ordinary 2K x 8 RAM array with the added advantages that data is retained during power-down periods and that the uppermost 8bytes provide an accurately updated real-time clock.

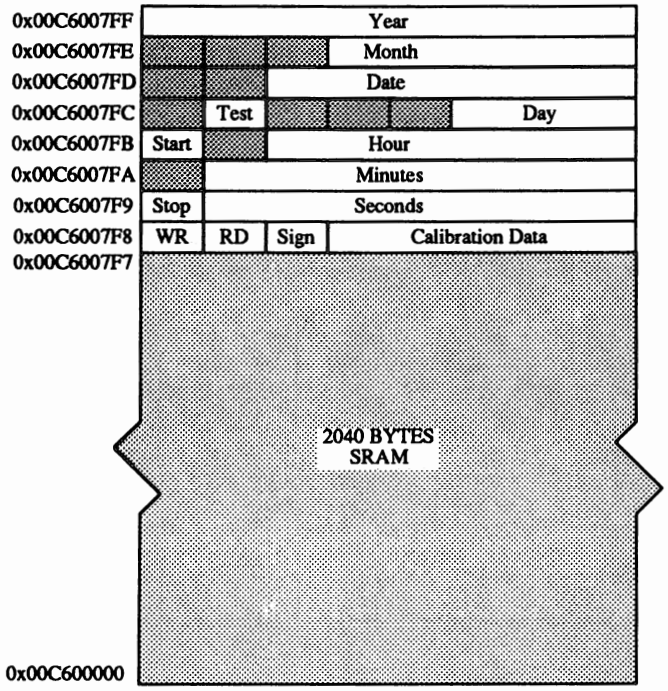
The RTC/RAM appears in the SPARCbook address map at address 0x00C600000. Within the 2Kbyte space that the device occupies, the RTC is contained in the uppermost 8bytes, and the RAM the remaining 2040bytes.

BATTERY BACKED SRAM

The battery-backed RAM of the MK48T02 is used in the conventional manner. The power-up and power-down transitions to and from the battery-backed mode are handled automatically by the hardware and do not require any special intervention from the user.

REAL TIME CLOCK

The Real Time clock area of the MK48T02 consists of 8 registers which contain the clock data and control information. The register organisation of the MK48T02 is shown in Figure 11-1.



RTC Operation

Within the RTC are a number of Binary Coded Decimal (BCD) counters configured to operate as a seconds, minutes, hour, day/date, month and year counter chain. A set of read-write registers are used to communicate with the counter chain. The counter chain is timed by a quartz crystal oscillator and programmable calibrator which generate accurate master signals to clock the entire RTC. Finally, a set of control and status bits are used to control of the oscillator, calibrator and read and write registers.

The control bits which govern the read and write operations to the RTC are bits 6 and 7 of the control register. These are positive polarity enable bits which control the transfer of data between the RTC counter registers and the external bus.

When the Read bit is set, the data at the RTC counter output is latched into the RTC read registers. In this mode the unchanging RTC data can be read accurately . Although the RTC data is stationary when the Read Bit is asserted, the RTC counter stages continue to be updated in the normal manner. When the RTC data has been read, the Read bit should be cleared. Conventional read operations of the RTC counter registers are possible when the Read Bit is '0', but in this mode the changing data cannot be guaranteed to be read correctly.

When the write bit is set, values may be written into the RTC. In this mode a complete set of time and date information may be assembled at leisure within the relevant RTC inputs. When the Write Bit is subsequently cleared the assembled time information is transferred in one operation into the RTC counters. The RTC then recommences operation from this point with the new information.

When the RTC is not required for very long periods of time (many months), it can be placed in a low-power mode to extend the battery life. In this case the oscillator is halted and normal RTC operation ceases. The low power mode is initiated by setting the ST bit, the most significant bit (bit 7) of the Seconds register. The typical lifetime of the RTC/SRAM is expected to be in excess of 7 years for most operating conditions.

In order to start the oscillator again, not only must the Stop bit be reset back to a '0', but also the Start bit must be taken slowly through a 0, 1, 0 cycle. In this case the complete start-up sequence is follows:

1. Set the Write bit to a 1 – to enable write operations to the RTC
2. Reset the Start bit to a 0 – to enable oscillations
3. Set the Start bit to a 1 – to set up the Start Oscillator command
4. Reset the Write bit to a 0 – to issue the Start Oscillator command
5. Wait 2 seconds – to allow the Oscillator to settle
6. Set the Write bit to a 1 – to enable write operations to the RTC
7. Reset the Start bit to a 0 – to turn off kick-start mode
8. Set the correct time and date
9. Reset the Write bit to a 0 – to transfer new data into the RTC

Real Time Clock Calibration

Although the RTC has good time-accuracy, the final operational accuracy will depend on the temperature and temperature-stability of the operating environment.

Recalibration is performed by writing a calibration value into the control register. This figure is used to periodically add or subtract pulses from the RTC chain. The value that is required for compensation can be calculated most easily from a knowledge of the parts-per-million (ppm) error of the RTC. This can be derived from the quantity of error which accumulates over a known time period.

The calibration value that is written into the control register is the ppm error divided by -2.034; that is, the RTC can be calibrated in increments of 2 ppm (approximately). The calibration data occupies the lowest 6 bits of the control register. Of this data, the lowest 5 bits constitute a calibration value and the most significant bit is a sign bit. When the sign bit is a zero, the calibration figure is negative and controls how much slower the clock will run; when the sign bit is a one, the calibration value is positive and controls how much faster the clock will run.

CHAPTER TWELVE
S1-BASE SCHEMATICS DESCRIPTION

Sheet 1 DRAM CONTROL

The MACH210 device (U39) is a progammable device similar to a PAL but considerably larger. Full details of the mode of operation of this device are available in Chapter 5. It is used as the DRAM State Machine, controlling the operation of the DRAM array.

The TS32A (U43) is used as the DRAM address multiplexer, providing the Row and Column addresses for the DRAM array on the BMA lines. The multiplexing is controlled by the signals / ROWEN and /COLEN generated by U39. The address is latched from the MAD lines into the 'B' side of the TS32A by the signal MCLK qualified by the signal /MAS, which is the synchronized cycle start signal. /MAS is generated by U1 on Sheet 1 of the S1-CPU board schematics. Three of the multiplexed address lines (BMA0, 1 and 10) are driven by the TS32A during the Row Address phase and by U39 during the Column Address phase.

U43 also latches MAD(3:0, 41:40) onto the permanently enabled DLA lines and MAD(36) onto the permanently enabled READ line. These are used as control signals by U39.

U19 and U51 are the Parity Storage DRAMs, holding one parity bit per byte for the entire memory array. Write-Per-Bit DRAMs are used, which allow any combination of the x4 active bits at each address to be altered in a single cycle (masking off the remaining bits). The masking is carried out by writing a code to the device on the data lines during the Row Address phase of the Write cycle. U12 is used to buffer the code supplied by U39 onto the parity data bus, PARD(7:0), and is enabled during the Row Address phase by the signal /ROWEN from U39.

SHEET 2 DRAM

The DRAM is organized into a 64-bit wide array, using sixteen x4 devices, either 4Mbit or 16Mbit fast page mode chips. The array is subdivided into two 32-bit wide blocks, the Upper and Lower blocks, which are identified by the use of the /RASU and /RASL signals. These, combined with the CAS(3:0) signals, allow valid combinations of byte, halfword, word and doubleword to be made active. Note that accesses that are unaligned across the upper and lower block boundaries are not catered for – the Parity Storage system will not cater for these types of (illegal) access. The DRAMs take the row and column addresses from the BMA(11:0) lines and data is on the MD(63:0) bus.

Sheet 3 PARITY GENERATION

On the right-hand side of the sheet are eight ACT11286 devices (U7, U11, U14, U21, U40, U41, U49 and U50). The ACT11286 is a 9-bit Parity Generator/Checker chip. For Writes to the DRAM array, the /PARXMIT signal is driven low, causing the Parity Error lines to be driven high (ie inactive) and the PARD lines are driven to the correct parity levels as generated by the 11286 devices. When the /PARXMIT signal is high, the PARD lines are used as inputs into the 11286 devices and the stored parity is checked against the data on the MD bus. Errors are flagged by one or more of the PERR lines being driven low. The individual PERR lines are then combined by U47 to produce a common parity error signal PARERR. If an even number of the data lines to a single device are High, the associated PARD line should be high; if an odd number of data lines are high, then the PARD line should be low.

The ACT16245 devices also shown on Sheet 3 are the DRAM data buffers. They connect the memory data bus to the multiplexed MAD bus. The direction of data flow is controlled by the /DBUFDIR signal and the devices are enabled by the /DBUFEN signal. Both of these signals are generated by U39 on Sheet 1. When /DBUFDIR is Low, the data flow is from the MAD bus to the MD bus. When /DBUFDIR is High, the data flow is from the MD bus to the MAD bus.

Sheet 4 MBUS INTERFACE & MISCELLANEOUS I/O

U38 is the MPI ASIC (Mbus to Peripheral Interface Application-Specific Integrated Circuit) that forms the interface between the Mbus (MAD0:63) and the on-board PC-AT style buses PA(18:0) and PD(15:0). Several other functional units are integrated into U38, including a programmable interrupt encoder (encoding the indiviual interrupt lines /IRQ(9:0) onto the four IRL(3:0) lines that are connected to the 7C600 Integer Unit), an address decoder (decoding the Mbus address map and providing several Chip Select signals – see Chapter 5) and two connections to U28, the 87C51 programmable microcontroller. The MPI ASIC is discussed fully in Chapter 5.

The MPI ASIC operates by changing Mbus cycles into PC-AT bus cycles. The device includes a DMA controller for floppy disk data transfers, and an internal 512byte FIFO to improve efficiency. The Mbus cycles can be 8, 16, 32 or 64bit transfers. The MPI performs byte packing and unpacking as required on the 16-bit Pbus, running the necessary number of data cycles on the Pbus until the required amount of data has been transferred. The MPI also handles write-posting to the Pbus from the Mbus.

U28 features four 8bit I/O ports and is used to control the following facilities:

Keyboard scanning
Mouse key interface with host conversion
Intercepting specific keyboard commands (eg brightness, contrast adjustment)
Monitoring power supply status
Interfacing to power supply and user system LED indicators
System temperature measurement
Reading/writing the system EEPROM
Adjusting brightness and contrast
Hardware reset

U35, the LTC1093 from Linear Technology, is a six-channel 10bit serial analogue to digital converter. A serial command word can be clocked into the device and the A/D conversion result, from the channel selected by the command, clocked out and into U28. This provides the mechanism by which the system temperature and mouse key are monitored. Four channels are dedicated to the force-sensitive resistors that make up the mouse key, enabling the 87C51 to return a mouse motion value to the IU (the 87C51 calculates the mouse motion using the values returned from the force-sensitive resistors and an algorithm programmed into its internal memory).

U58 and U59 are digitally controlled potentiometers, with a minimum resistance of 40Ω, an increment size of 505Ω and a maximum resistance of 50kΩ. U58 controls the display brightness and U59 the contrast. The X9C103 from Xicor contains a small area of nonvolatile memory (EEPROM) so that settings are not lost when the system is powered off. In response to user commands, the 87C51 can increment or decrement the setting of the potentiometer, thereby adjusting the LCD brightness and contrast.

U31, the 380105 device, is a keyboard scanner. At the start of a sequence, U28 asserts then releases the Reset line to U31. The 87C51 then issues a string of CLK pulses to the keyboard scanner, continuing until the cycle is finished when Reset is driven again. U31 responds by driving DR(1) on the first clock after Reset, DR(2) on the second, and so on. U31 also monitors the Sense line, SL(x) that corresponds to the currently active Drive line, DR(x). If U31 detects a returned signal, it asserts the KEYOUT line. The program within U28 keeps track of which Drive lines produce a Sense signal within each cycle. If a keyboard character is detected, an interrupt is issued to the IU, causing the IU to read the character from a register within U28.

U32 is a battery-backed 2Kbyte nonvolatile RAM chip, with an integral real time clock. The RTC control and time registers overlay the top eight bytes of the memory. U33 is the system EPROM, containing the operating firmware for the board. The 8bit data from the EPROM is assembled into 32-bit instructions and fixed data for the CPU by the MPI ASIC.

Sheet 5 PERIPHERAL I/O

U3 is the 82C710 Universal Peripheral Controller, which features a 16450-compatible USART (Universal Serial Asynchronous Receiver/Transmitter), an IDE interface for connection to one or two hard disk drives, a µPD72065B-compatible Floppy Disk Controller, a mouse port and a 16-bit PC-AT compatible bidirectional parallel port. The 82C710 contains a large number of registers and many features are software-selectable. To change any of the configuration register values, a complicated sequence of write operations has to be undertaken before the changes can be made. This configuration sequence is intentionally kept complicated to minimize the chance of changes being made as a result of a programming bug.

The USART section of U3 is connected to U45 which converts TTL-level signals to and from RS232 levels for the serial interface.

Sheet 6 GRAPHICS

U46 is the VGA display controller for the monochrome flat panel display, with a supplementary CRT display capability. The two 256K x4 DRAMs, U16 and U17, form the video memory block, with U16 containing memory planes 0 and 1, U17 planes 2 and 3. The 64K x4 DRAM, U8, is the Frame Accelerator memory, used to buffer the top and bottom half of the display alternately. This procedure is used to reduce the display flicker without drastically increasing the current consumption of the display controller chip. Note that this device is not required in a color system.

U23 is a Western Digital WD90C61 frequency synthesizer chip. The internal voltage-controlled oscillators and phase-locked loop circuits derive the clock signals OSC and SQCLK from the 14.312MHz oscillator (XL4) signal. The frequency of OSC is controlled by the three CLKSEL lines. SQCLK is fixed at 37.585MHz by the settings of the MCLK pins.

The LT1017 (U34) is a dual comparator/op amp device and is used to pass the LCD contrast and drive signals to the LCD. The signals out of U34 can be turned off by the signal PPCTL, which is derived from the /DISPOFF signal using TR2 (also on Sheet 6). This circuitry provides a software-controlled mechanism for turning off the LCD. The signal BLOFF from TR1 turns the LCD backlight off.

Sheet 7 MODEM AND ETHERNET

U15 is the Fujitsu 86960 NICE Ethernet Controller, a 16-bit Ethernet controller which uses a 64Kbyte block of buffer memory (U53, U56) for storing receive/transmit data. The CPU accesses the buffer memory via the Buffer Memory Port registers within the 86960. Reads from this register retrieve data from the receive buffer, writes are directed to the transmit buffer. Read and write pointers are maintained by the 86960 itself, simplifying the operation of loading and retrieving network packets. The buffer memory is addressed using the BA(14:0) bus, and data is transferred over the SD(15:0) bus.

U22 and U57, together with associated components, make up the Modem interface. U22, the Modem Advanced Controller (MAC), communicates with U57 over the 8bit AD bus and the /SIN, /SOUT lines (Serial In, Serial Out). The serial lines are for transmit and receive data. The parallel bus is a multiplexed address and data bus and is used for setting up, and passing commands to, U57. U22 communicates with the IU over the lower 8 bits of the PD bus, with 3 address bits selecting the register within the MAC to be accessed by the IU. U22 contains an area of ROM which is programmed to interact with U57 and provide a 2400bps full duplex intelligent modem. The two device chip-set performs all the modem functions as well as automatic control features compatible with the Hayes "AT" command set.

Sheet 8 CONNECTORS

Sheet 8 shows all the connectors present on the S1 Base card.

PH1 is the 100-way miniature interboard connector that carries the signal between the base board and the CPU module.

SK1 carries the signals between the base board and the internal floppy disk drive unit, SK2 connects to the LEDs.

SK3, 4 and 5 make connections with the S1-I/O board, carrying signals for several of the external interfaces.

SK6 passes the control, timing and data sugnals to the LCD.

SK7, a 5-pin flexible circuit connector, is used to carry signals to the flat panel display backlighting system.

SK8 connects to the force sensitive resistors that make up the mouse key and the On/Off switch.

SK9, an 8-way connector marked 'KEYBD 1' on the schematics, is used in conjunction with SK10, an 11-way connector marked 'KEYBD 2' on the schematics, to make connection with the built-in keyboard. SK9 carries the Sense lines, SK10 the Drive lines.

PL1 is used to make the connection to the internal hard disk unit or units.

(Note the presence of two Chip Select signals, /HDCS0 and 1, to allow up to two IDE devices to be connected to the cable from PL1.)

PL2 is used to carry signals and power to and from the power supply unit, with TR3 being used to hold the supply ON when the Power On switch is activated.

Color Systems only - Sheet 9

On the color version of the SPARCbook, the S1-BASE-C schematics vary slightly from the mono-chrome version. The changes are confined to some minor changes on Sheets 6 and 8 and the addition of a Sheet 9.

Sheet 8 has extra connectors to carry the signals to the color display.

Sheet 9 shows the CL-GD6340 color LCD interface controller (U60) which works in conjunction with U46 on Sheet 6, plus the color LCD power supply generator circuitry (U63 and associated components).

S1-I/O SCHEMATIC DESCRIPTION

The S1-I/O board is used to mount the connectors that interface to the external devices via the backpanel of the SPARCbook. Numerous filtering and screening components are present to reduce electromagnetic radiation, allowing the SPARCbook to meet FCC standards.

The circuitry in the lower half of the sheet is used to connect the SC11054 modem device (U57, Sheet 7 of the S1-Base schematics) to the telecommunications network. The signal OH is generated by the SC11075 Modem Advanced Controller or MAC (U22, Sheet 7 of the S1-Base schematics) and is the "off-hook" signal. /MRI is the Ring signal from the same device and when active indicates that the modem is receiving a ringing signal. RXTA is the analogue receive/transmit line which is referenced to MVREF.

S1-CPU SCHEMATIC DESCRIPTION

Sheet 1 CY7C601 INTEGER UNIT

Sheet 1 shows the CY7C601 Integer Unit (or IU, the Central Processor Unit, U2), the CY7C602 Floating Point Unit (U3), the CY7C604 Cache Controller and Memory Management Unit (U1, the MCT), and two 16Kbyte Cache Storage Units (U4 and U5) – a specialized form of SRAM. All these devices are linked by the IU address and data buses, A(31:0) and D(31:0) respectively. U1 provides the interface between these buses and the 64-bit multiplexed Mbus, MAD(63:0). It also keeps track of the information stored within the cache memory and controls the cache memory (U4 and U5). For example, if the Integer Unit carries out a read operation from a main memory location marked as cacheable, U1 will cause the data to be written into the cache memory at the same time as it is read by U2. If U2 later attempts to carry out another operation to that same address, U1 will divert the cycle to the cache instead. If U2 addresses a location that is not held in cache, U1 will translate the processor bus cycle into an Mbus cycle. U1 will also handle any logical to physical address translation that may be required.

The IU is the SPARC processor itself. The two SIZE lines define the size of transfer currently being executed. 8, 16, 32 and 64-bit operations are identified, with 64-bit operations (Load/Store Double) being a special category of data transfer using two 32-bit bus cycles. The ASI lines are the "address space identifier" lines, used by multitasking operating systems. MHOLDA is an active LOW signal, used to freeze the clock to both the IU and the FPU in the event of a cache miss or an access to slow memory via the MCT. DOE is also an active LOW signal and is deasserted to tristate the bus drivers on the IU and FPU, allowing the MCT to control the address and data buses.

The optional FPU (U3) operates concurrently with the IU. The IU identifies floating point instructions by means of the FINS lines and allows the FPU, when fitted, to pick up instructions and data or to supply data. A floating point instructions is concurrently decoded by both the IU and FPU, but does not begin executing in the FPU until after the instruction is enabled by a signal from the IU. Several signal lines pass between the FPU and IU to provide the control and handshaking required for these operations.

Sheet 2 100-WAY MINIATURE CONNECTOR

Sheet 2 shows the 100-way miniature connector that carries the signals between the CPU board and the base board.

CHAPTER THIRTEEN
SYSTEM COMPONENT SPECIFICATIONS

LCD COLOR DISPLAY

Mechanical Characteristics

Item	Specification	Unit
Dimensions	224(w) x174(h) x10.5 max	mm
Structure	640 x R.G.B x 480 (h)	dot
Dot size	0.07 (w) x 0.25(h)	mm
Dot pitch	0.09 (w) x 0.27 (h)	mm
Effective area	178 (w) x 134 (h)	mm

Temperature Range

Item	Min	Max	Unit
Operating temperature	10	40	°C
Storage temperature	-10	50	°C
Ambient temperature (Ta)	Ta ≤ 40°C		85% Relative Humidity max
(no dew)	Ta > 40°C		Below maximum absolute humidity of 40°C 85% RH

Absolute Maximum Ratings

Item	Symbol	Min	Max	Unit
Power supply for logic	V_{DD}-V_{SS}	0	6.0	V
Power supply for LC				
Driving	V_{EE}-V_{SS}	0	42	V
Input voltage	V_I	V_{SS}	V_{DD}	V

Electrical Characteristics

Ta = 25°C, V_{DD} = 5.0V ±5%, V_{EE}-V_{SS} = +38V ± 1V

Item	Symbol	Condition	Min	Typ	Max	Unit
Input high voltage	V_{IH}		0.8 V_{DD}		V_{DD}	V
Input low voltage	V_{IL}		V_{SS}		0.2V_{DD}	V
Clock frequency	f_{CL2}		5.38	5.61	6.14	MHz
Current consumption	I_{DD}	f_{CL2} = 5.61MHz			20	mA
	I_{EE}	UD,~UD, =low			60	mA
		LD,~LD, =low				

Operating voltage for LC driving

Item	Symbol	Condition	Min	Typ	Max	Unit
Operating voltage		Ta = 10°C			38	V
(1/480 duty)	V_o-V_{ss}	Ta = 25°C		(33)		V
		Ta = 40°C	26			V

Optical Characteristics

Item	Symbol	Condition	Min	Typ	Max	Unit
Viewing angle		Ta = 25°C	20			deg
Contrast ratio		Ta = 25°C	2			-

MONOCHROME DISPLAY

Mechanical Characteristics

Item	Specification	Unit
Dimensions	255(w) x172(h) x9.0 max	mm
Structure	640 x 480 (h)	dot
Dot size	0.27 (w) x 0.27(h)	mm
Dot pitch	0.30 (w) x 0.30 (h)	mm

Temperature Range

Item	Min	Max	Unit
Operating temperature	10	40	°C
Storage temperature	-20	60	°C
Ambient temperature (Ta)	Ta ≤ 40°C		85% Relative Humidity max
(no dew)	Ta > 40°C		Below maximum absolute humidity of 40°C 85% RH

Absolute Maximum Ratings

Item	Symbol	Min	Max	Unit
Power supply for logic	V_{DD}-V_{SS}	0	6.5	V
Power supply for LC				
Driving	V_{DD}-V_O	0	27.5	V
Input voltage	V_I		-0.3	V_{DD}+0.3 V
Input current	I_I		0	1 A

Electrical Characteristics

Item	Symbol	Condition	Min	Typ	Max	Unit
Power supply for logic	V_{DD}-V_{SS}		4.75	5.0	5.25	V
Power supply for LC	V_{EE}-V_{SS}			-22.0		V
Input high voltage	V_{IH}		0.8 V_{DD}		V_{DD}	V
Input low voltage	V_{IL}		0		0.2V_{DD}	V
Clock frequency	f_{CFL}			(70)	(85)	KHz
Current consumption	I_{DD}	V_{DD}-V_{SS} = 5.0V		(15)		mA
	I_{EE}	V_{EE}-V_{SS} = -22.0V		(11.0)		mA

Operating voltage for LC driving

Item	Symbol	Condition	Min	Typ	Max	Unit
Operating voltage		Ta = 10°C		(23.9)		V
	V_{DD}-V_O	Ta = 25°C		(23.6)		V
		Ta = 40°C		(23.0)		V

Optical Characteristics

Item	Symbol	Condition	Min	Typ	Max	Unit
Viewing angle		Ta = 25°C		(40)		deg
Contrast ratio		Ta = 25°C		(5)		-

80MBYTE HARD DISK

Mechanical Dimensions

Item	Specification
Height (max)	19.0mm
Width	70.0mm
Depth	101.6mm
Weight	176g

Environmental limits

	Storage	Operating
Ambient temperature	-40°C to 70°C	5°C to 55°C
	20°C/hr gradient	10°C/hr gradient
Ambient Relative Humidity	5 to 95% non-condensing	8 to 85% non-condensing

Power Requirements (at nominal environmental conditions)

Item	Specification
Voltage	+5V ±5%
Ripple and Noise	50mV
Current:	
Sleep/Standby	0.04A
Parked	0.16A
Idle	0.34A
Active (R/W)	0.40A
Active (Seeking)	0.44A
Start up (Peak)	1.10A

Timing Specifications

Item	Maximum under nominal conditions	Unit
Single-Track Seek	5	ms
Average Seek	19	ms
Full Stroke Seek	30	ms
Average Rotational Latency	8.3	ms
Sequential Head Switch	4	ms
Command Overhead	1	ms
Power on to Drive Ready	<4.0	s
Sleep/Standby to Drive Ready	<2.5	s

Notes: Seek time includes head settling time but not command overhead or rotational-latency delays.

Power On to Drive Ready is the time elapsed between the supply voltage reaching the operating range and the drive being ready to accept all commands.

An ambient temperature of 25°C, nominal supply voltages and no applied shock or vibration constitute nominal conditions.

Vibration and Shock

Item	Nonoperating	Operating
Vibration:		
5 to 300Hz sine wave (peak to peak)	2G	1G
1 oct/min sine sweep		
Shock:		
1/2 sine wave of 10ms duration	100G	10G
(10 hits max; >2 seconds between hits)		

120MBYTE HARD DISK

Mechanical Dimensions

Item	Specification	Units
Height (max)	19.1	mm
Width (max)	70.1	mm
Depth (max)	101.9	mm
Weight (max)	185	g

Environmental limits

	Storage	Operating
Ambient temperature	-40°C to 70°C	5°C to 55°C
	30°C/hr gradient	30°C/hr gradient
Ambient Relative Humidity	8 to 90% non-condensing	8 to 80% non-condensing

Power Requirements (at nominal environmental conditions)

Item	
Voltage	+5V ±5%
Current:	

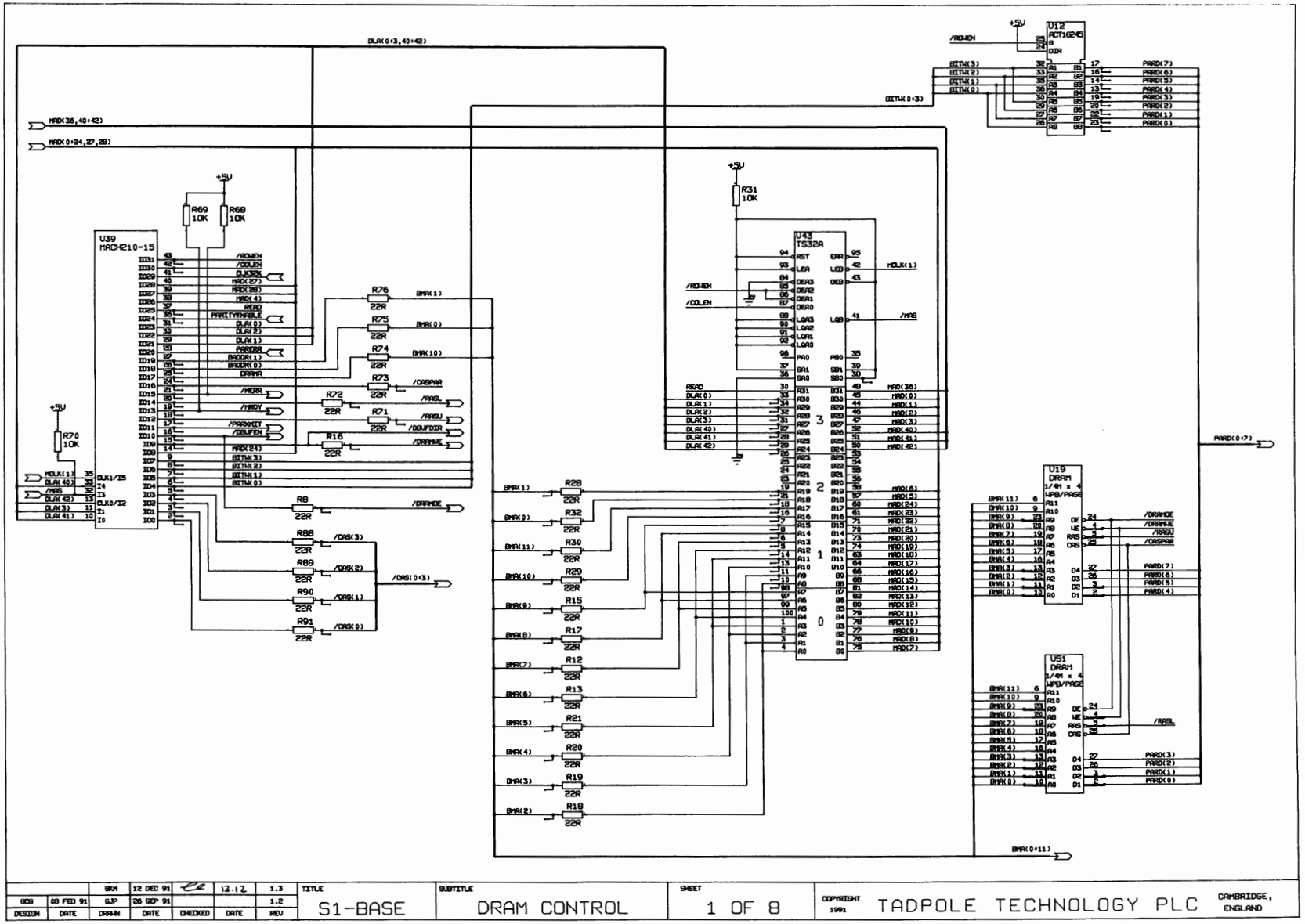
Sleep/Standby	0.10A
Idle	0.23A
Active (R/W)	0.42A
Active (Seeking)	0.42A
Start up (Peak)	1.00A

Timing Parameters

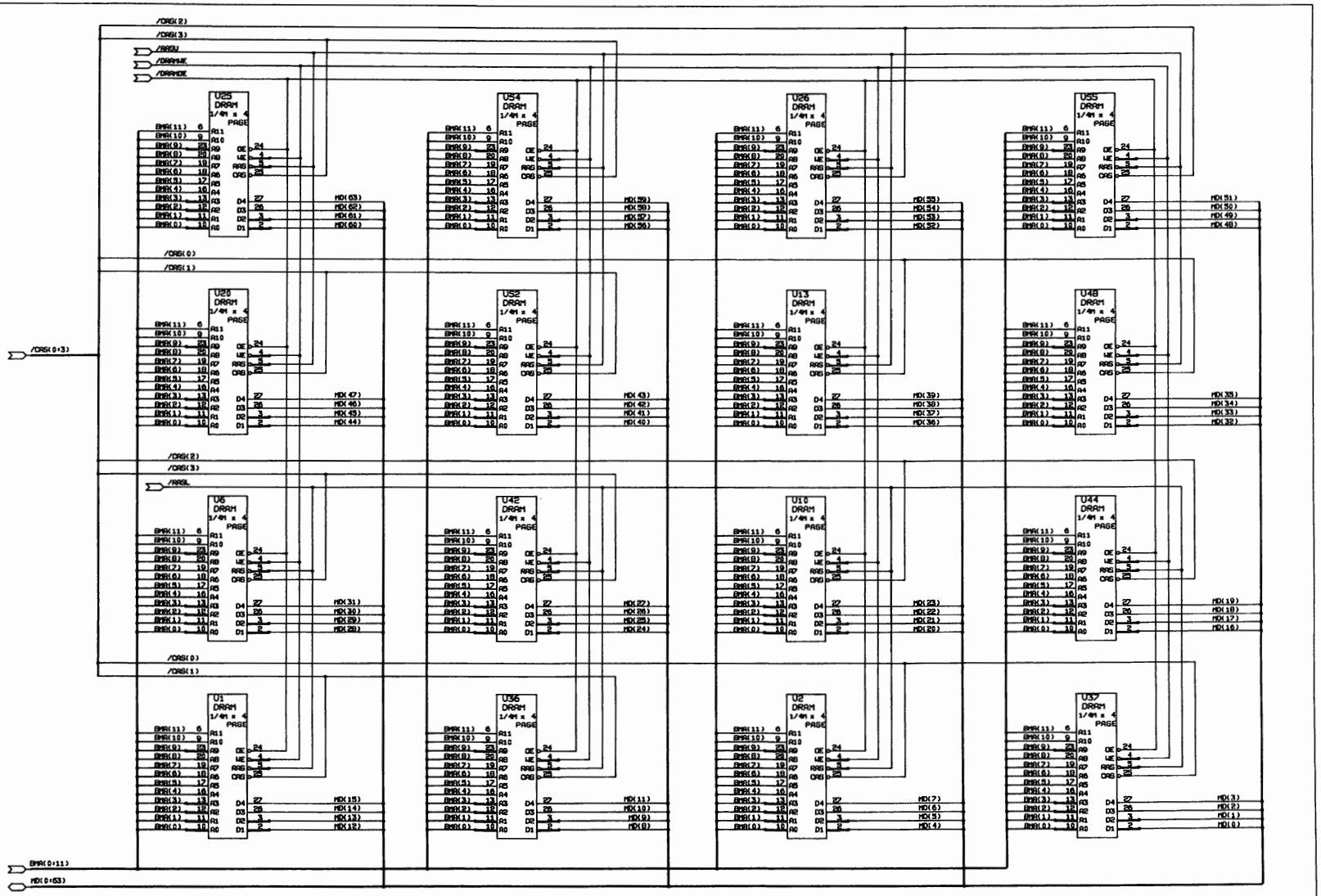
Item	Maximum under nominal conditions	Unit
Single-Track Seek	8	ms
Average Seek	19	ms
Full Stroke Seek	30	ms
Average Rotational Latency	8.7	ms

Vibration and Shock

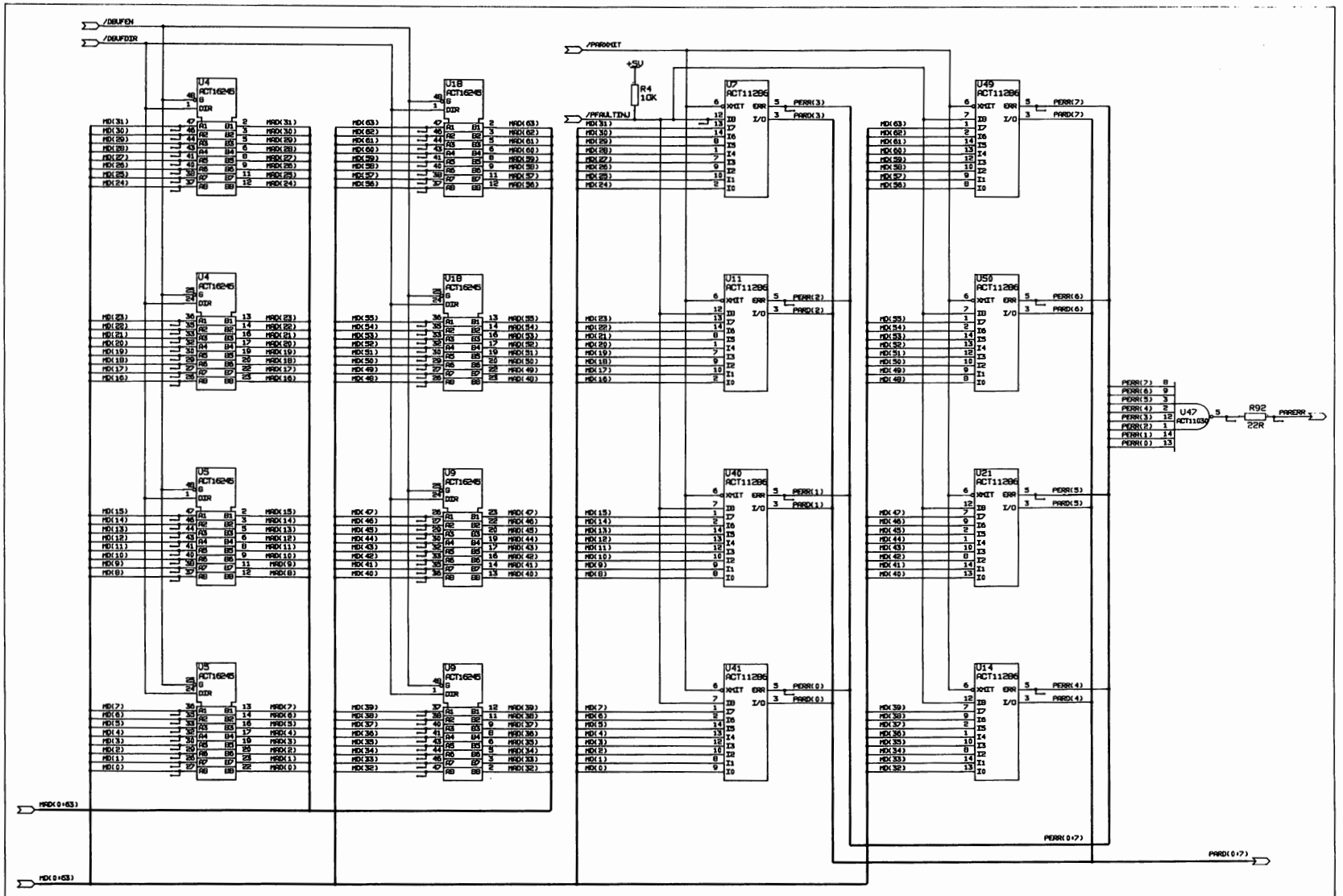
Item	Nonoperating	Operating
Vibration:		
22 to 500Hz sine wave (peak to peak)	4G	0.5G
1 oct/min sine sweep		
Shock:		
1/2 sine wave of 10ms duration	100G	10G
(10 hits max; >2 seconds between hits)		



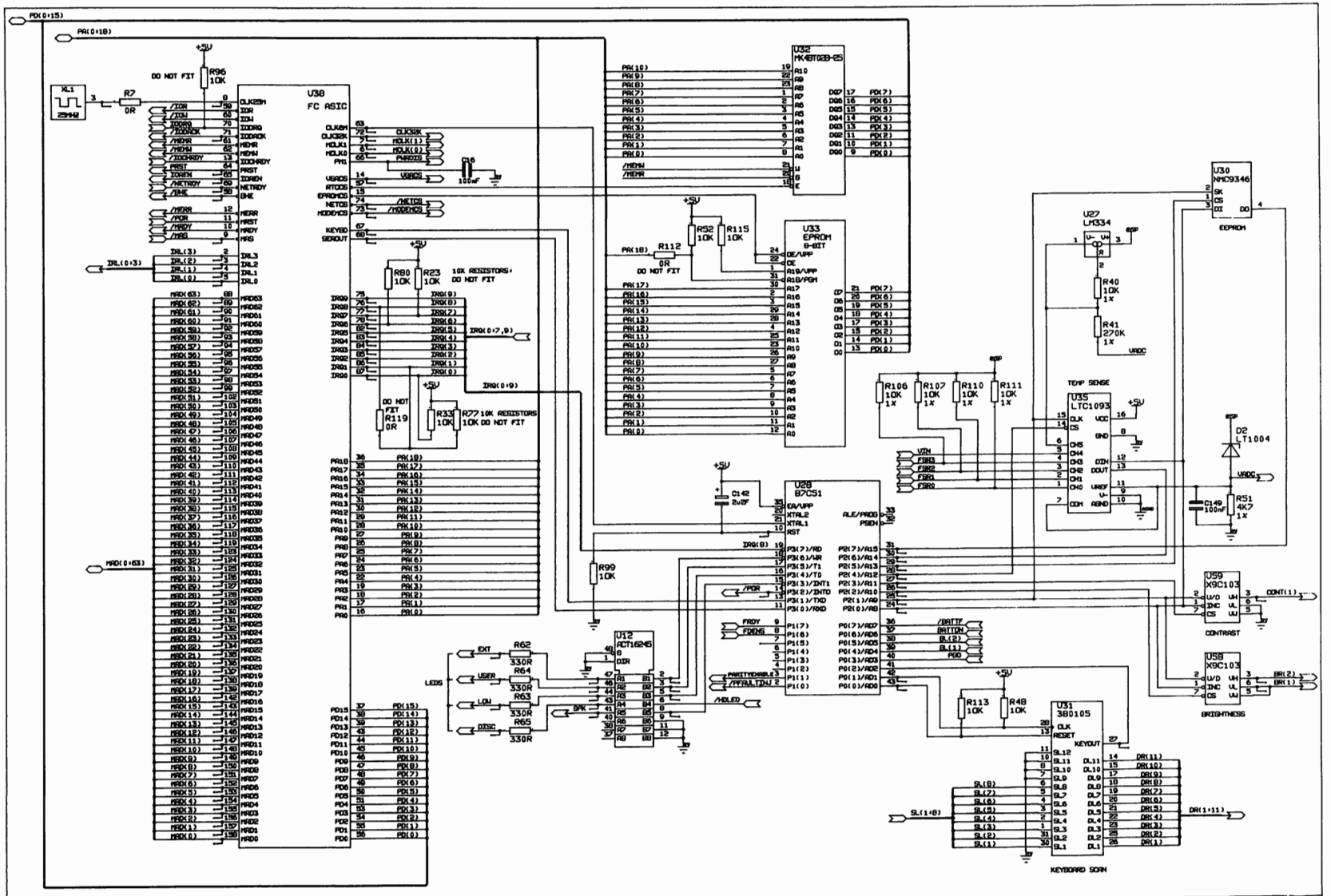
		SKM	12 DEC 91		12.12	1.3	TITLE	SUBTITLE	SHEET
003	09 FEB 91	GJP	26 SEP 91			1.2	S1-BASE	DRAM CONTROL	1 OF 8
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV			



		BKH	12 DEC 91		IJ.12	1.3	TITLE: S1-BASE	SUBTITLE: DRAM	SHEET: 2 OF 8	COPYRIGHT 1991 TADPOLE TECHNOLOGY PLC CAMBRIDGE, ENGLAND
GDB	08 FEB 91	SJP	26 SEP 91			1.2				
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV				

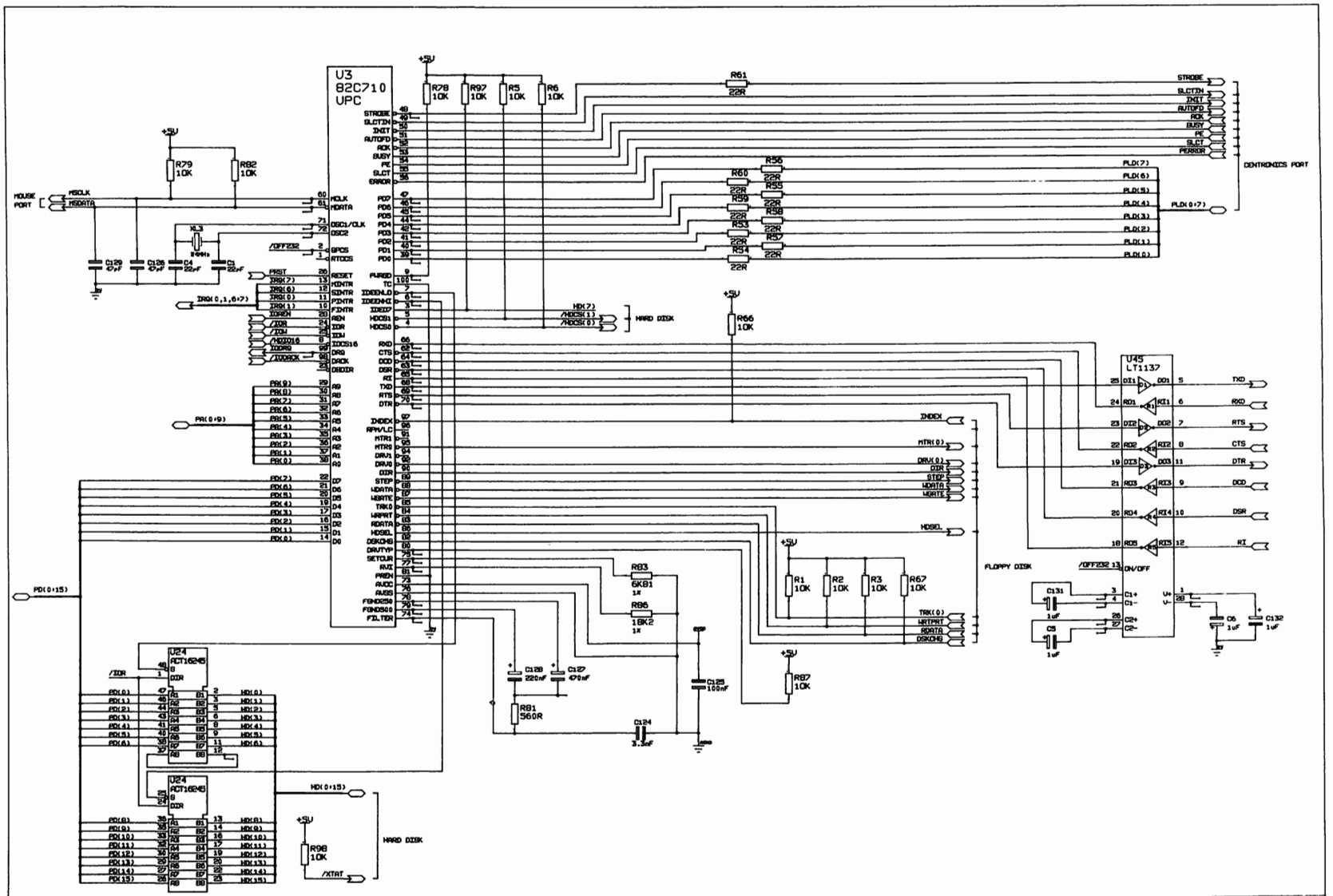


		GKH	12 DEC 91				1.3	TITLE	SUBTITLE	SHEET	COPYRIGHT	
BDS	DB FEB 91	GJP	26 SEP 91				1.2	S1-BASE	PARITY GENERATION	3 OF 8	1991	TADPOLE TECHNOLOGY PLC CAMBRIDGE, ENGLAND
DESIGN	DATE	DRAWN	DATE	CHECKED		DATE	REV					

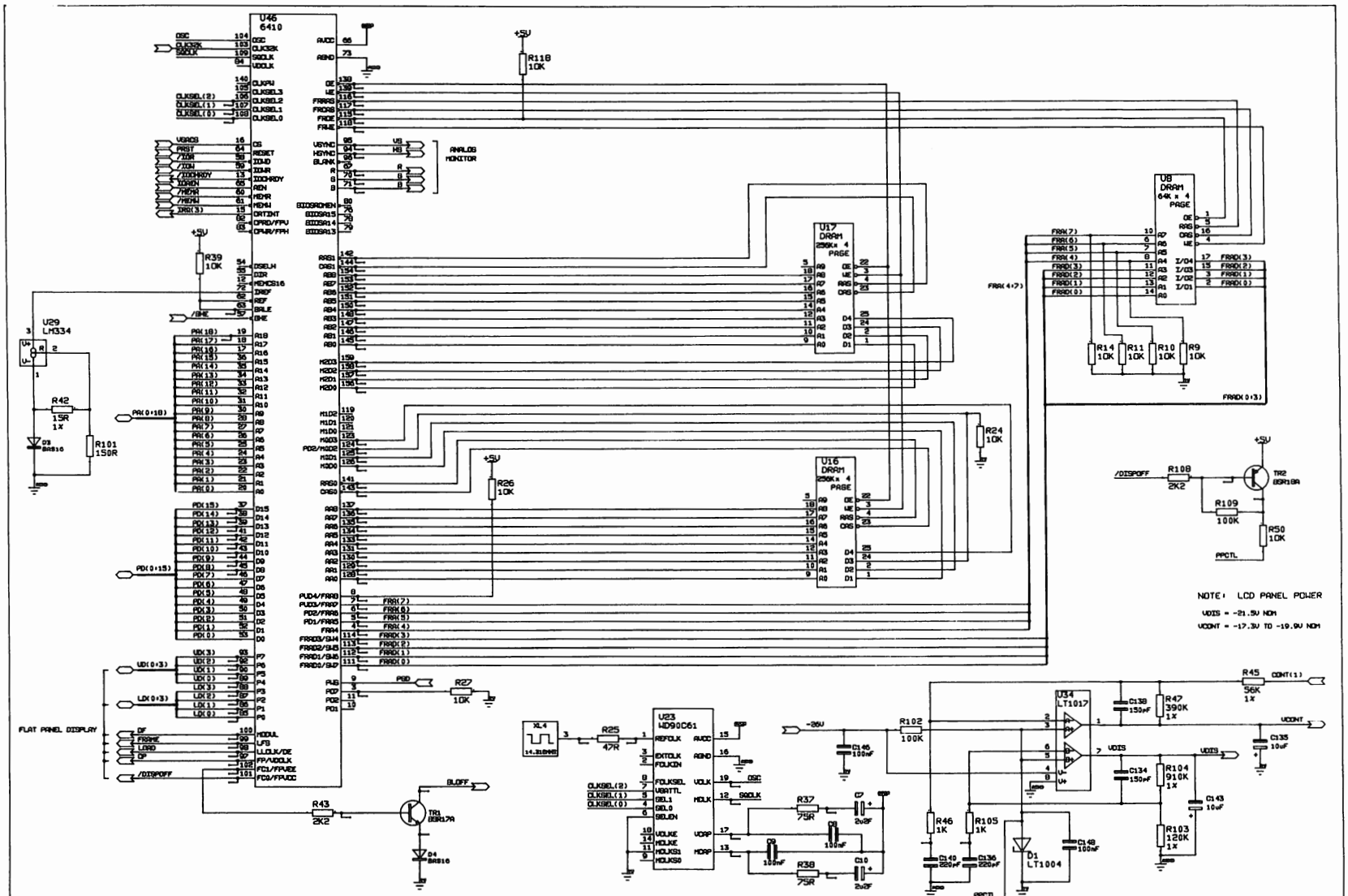


GDS	08 FEB 91	GJP	26 SEP 91			1.2	
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV	

		BKH	12 DEC 91		13.12	1.3	
TITLE	S1-BASE	SUBTITLE	MBUS INTERFACE & MISCELLANEOUS I/O	SHEET	4 OF 8	COPYRIGHT 1991	TADPOLE TECHNOLOGY PLC CAMBRIDGE, ENGLAND

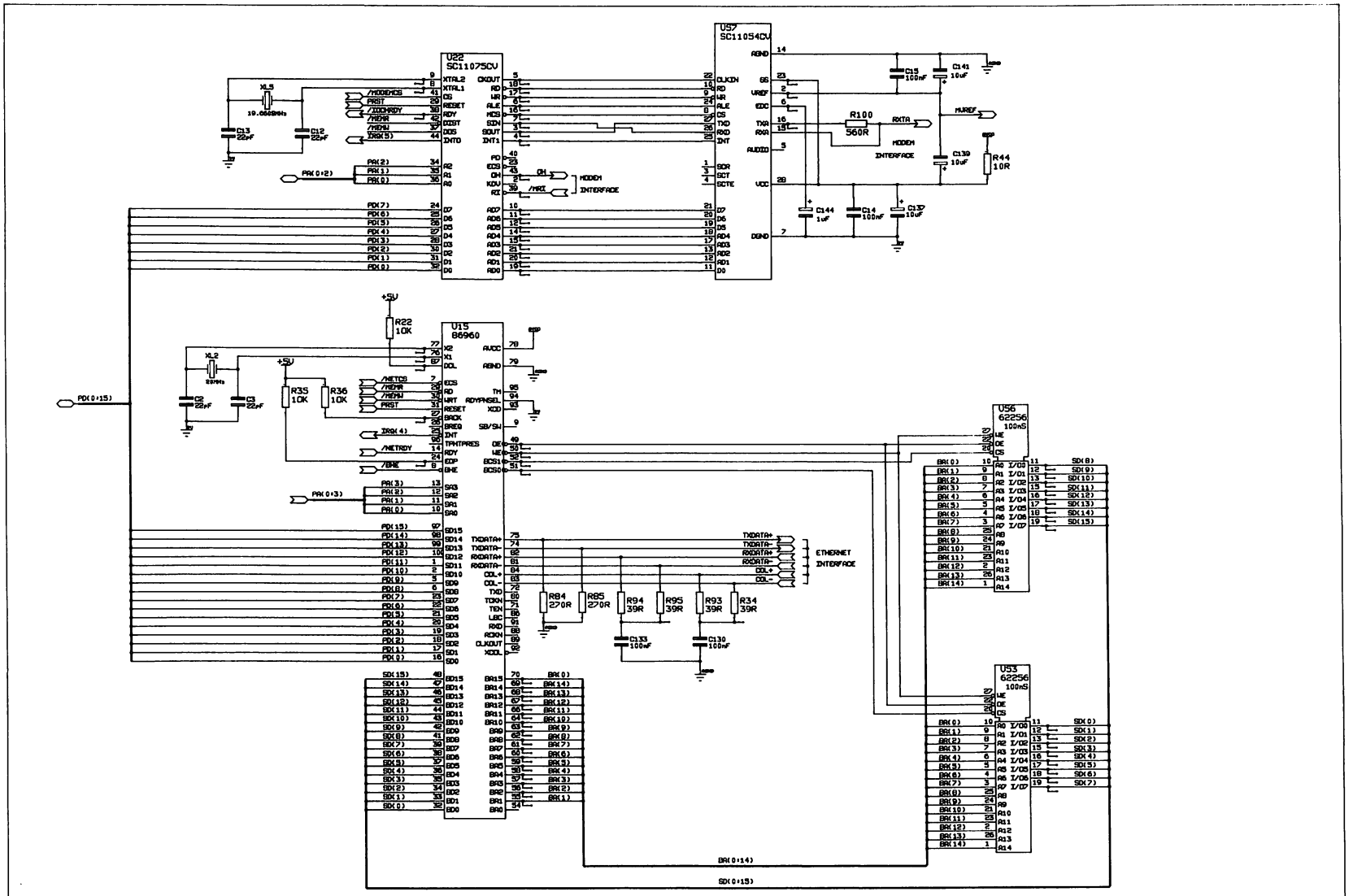


		GM	12 DEC 91			1.3	S1-BASE	PERIPHERAL I/O	5 OF 8
GCD	08 FEB 91	GJP	26 SEP 91			1.2			
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV	TITLE	SUBTITLE	SHEET



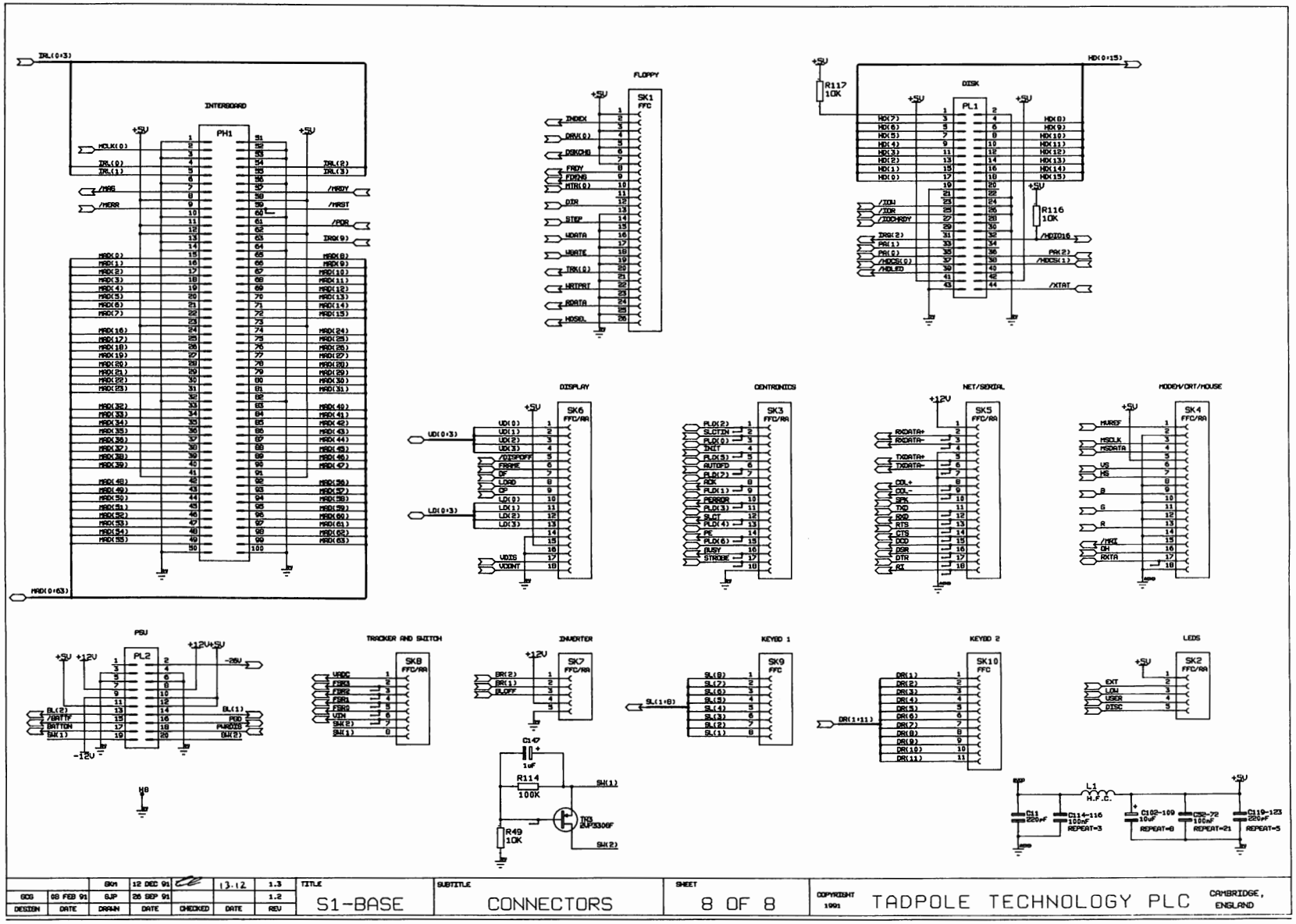
		GKH	12 DEC 91		1.3	TITLE	SUBTITLE	SHEET	
BDB	28 FEB 91	SJP	26 SEP 91		1.2	S1-BASE	GRAPHICS	6 OF 8	
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV			

 TADPOLE TECHNOLOGY PLC CAMBRIDGE, ENGLAND

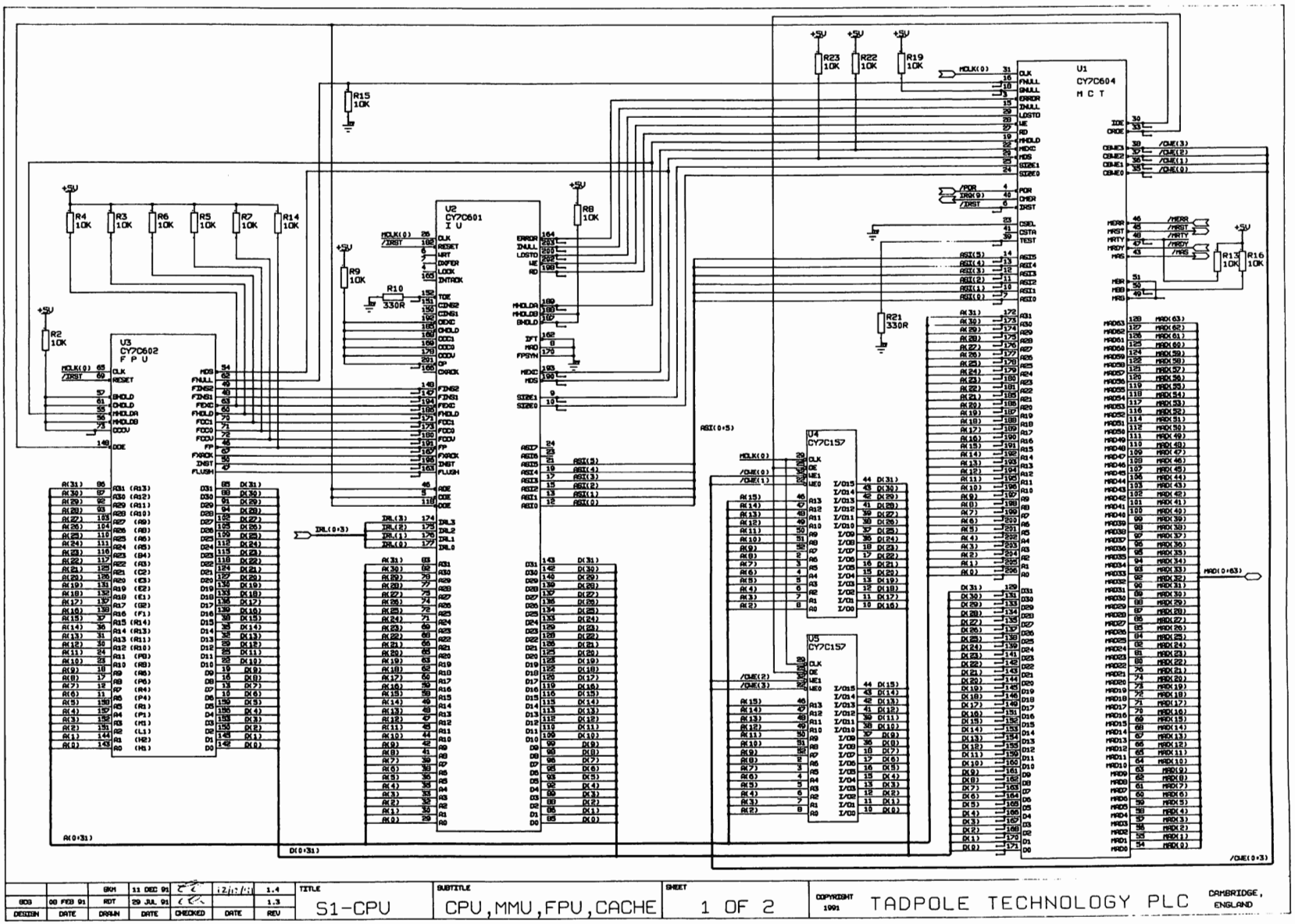


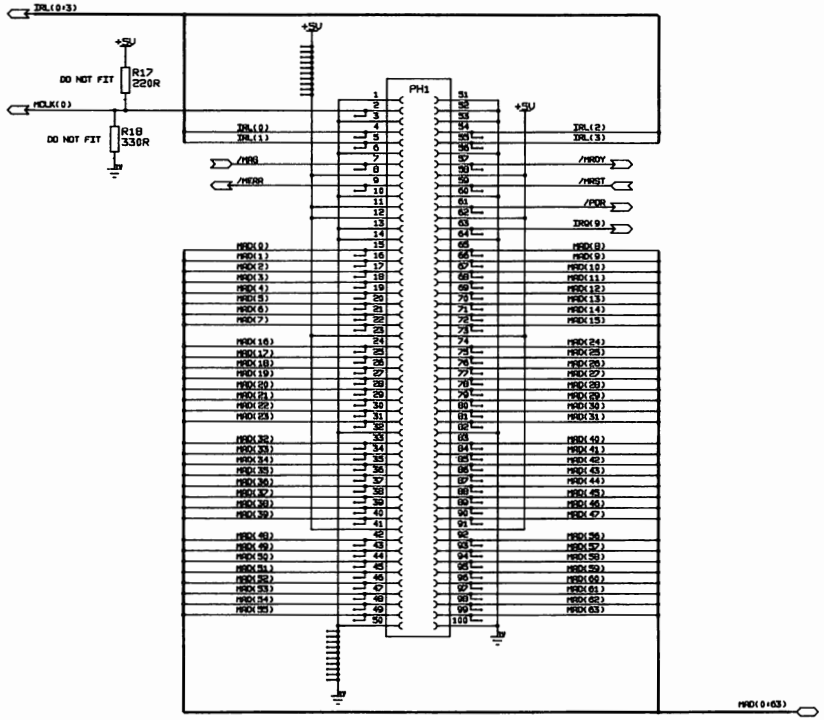
		BKH	12 DEC 91		1 \ . 1 2	1.3	TITLE	SUBTITLE	SHEET
GDB	08 FEB 91	GJP	26 SEP 91			1.2	S1-BASE	MODEM AND ETHERNET	7 OF 8
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV			

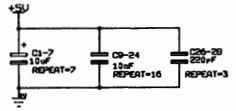
TADPOLE TECHNOLOGY PLC CAMBRIDGE, ENGLAND



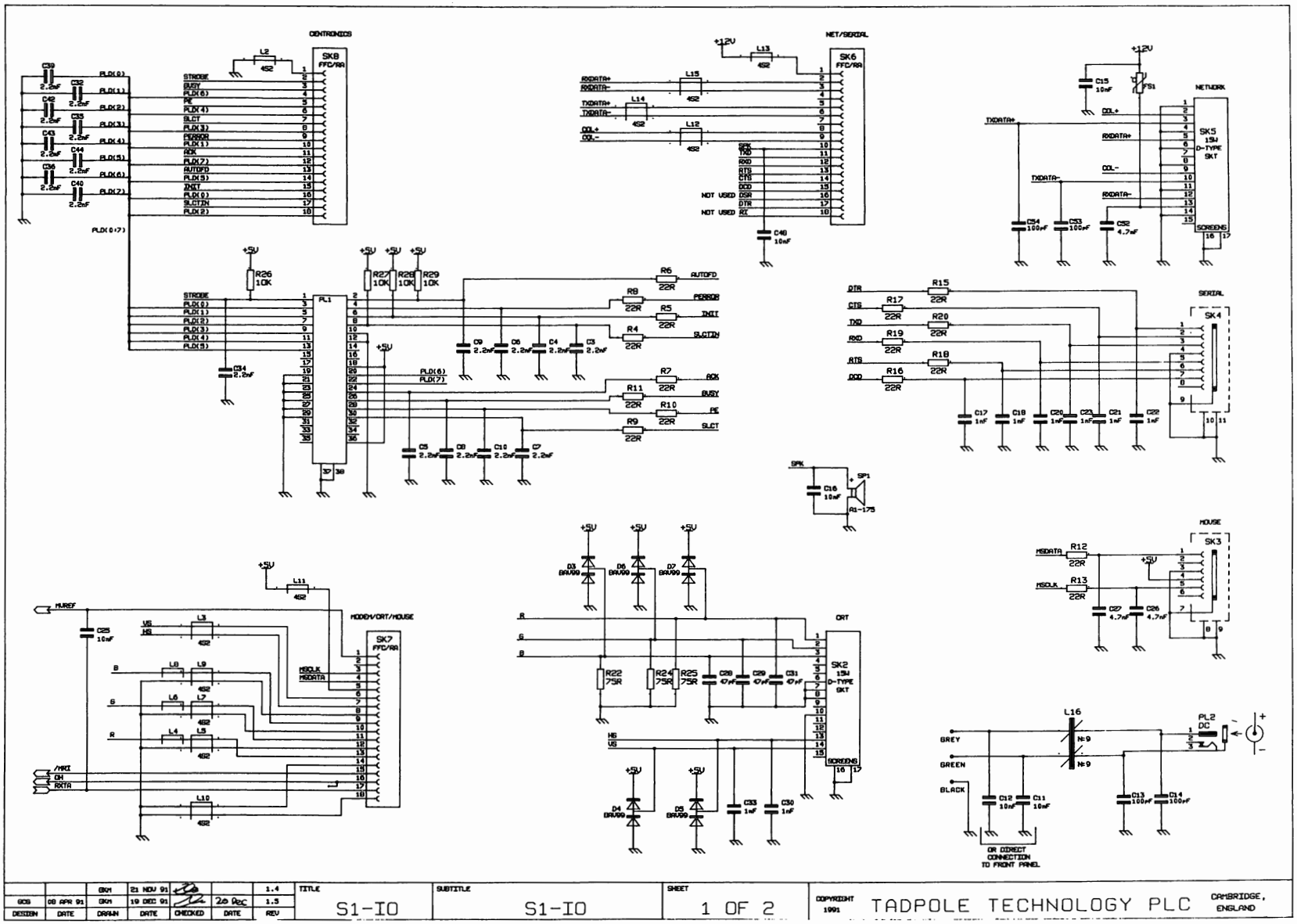
		SKM	12 DEC 91		13.12	1.3	TITLE	SUBTITLE	SHEET
GDS	08 FEB 91	GJP	26 SEP 91			1.2	S1-BASE	CONNECTORS	8 OF 8
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV			


U1
CY7C604
M C T
U2
CY7C601
I U
U3
CY7C602
F P U
U4
CY7C157
U5
CY7C157
R2 10K
R3 10K
R4 10K
R5 10K
R6 10K
R7 10K
R8 10K
R9 10K
R10 330R
R13 10K
R14 10K
R15 10K
R16 10K
R19 10K
R21 330R
R22 10K
R23 10K
TITLE
S1-CPU
SUBTITLE
CPU,MMU,FPU,CACHE
SHEET
1 OF 2
COPYRIGHT 1991
TADPOLE TECHNOLOGY PLC
CAMBRIDGE, ENGLAND
BDI 08 FEB 91 RDT 29 JUL 91 1.3
DESIGN DATE DRAWN DATE CHECKED DATE REV
BM1 11 DEC 91 1.4



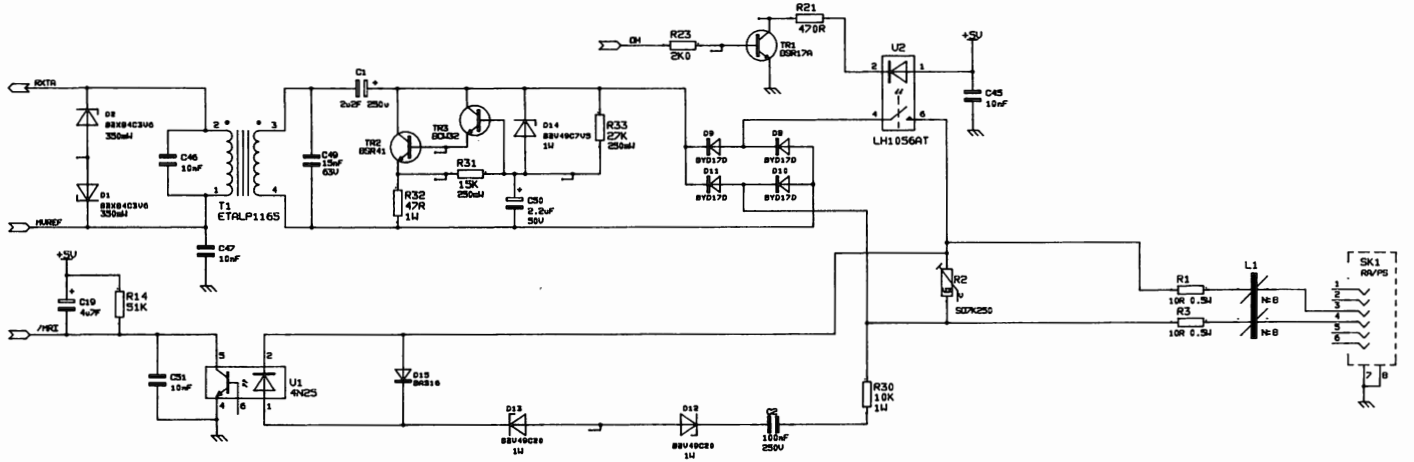


		BDI	11 DEC 91	e. c.		1.4	TITLE S1-CPU	SUBTITLE HYPERMODULE CONNECTOR	SHEET 2 OF 2
BDS	08 FEB 91	RDT	29 JUL 91			1.3			
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV			



		GKH	21 NOV 91			1.4	TITLE	SUBTITLE	SHEET
GDS	08 APR 91	GKH	19 DEC 91		20 Dec	1.5	S1-IO	S1-IO	1 OF 2
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV			

TADPOLE TECHNOLOGY PLC CAMBRIDGE, ENGLAND



		BKH	21 NOV 91			1.4	TITLE	SUBTITLE	SHEET	COPYRIGHT 1991 TADPOLE TECHNOLOGY PLC CAMBRIDGE, ENGLAND
BDS	08 APR 91	BKH	19 DEC 91			1.5	S1-IO	MODEM	2 OF 2	
DESIGN	DATE	DRAWN	DATE	CHECKED	DATE	REV				