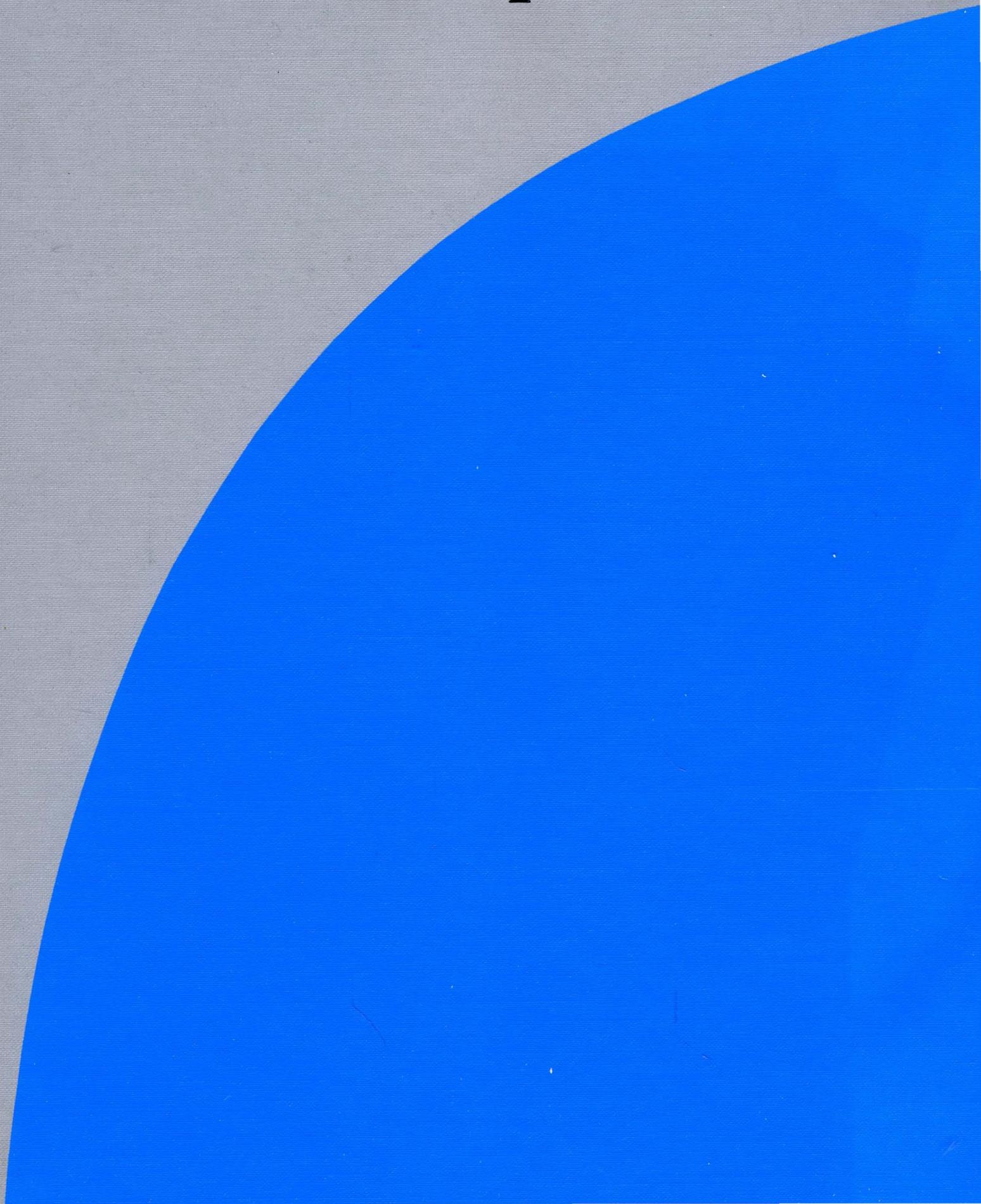


# Solbourne Computer



---



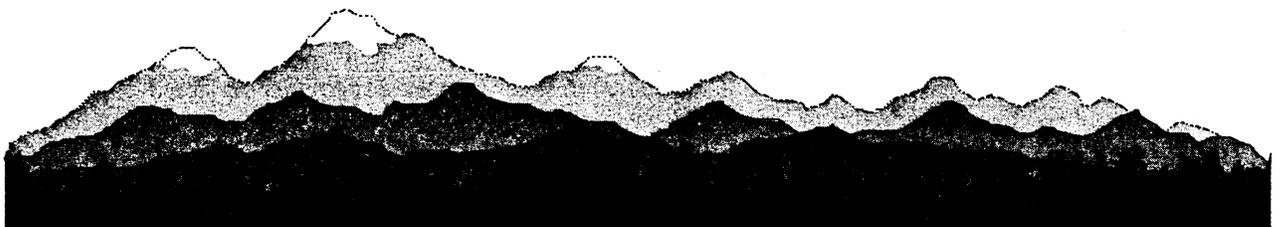
# Solbourne Computer

---

---

**Series4 and Series5 Theory of Operations**

**Solbourne Confidential - Do Not Reproduce**



Solbourne Computer, Inc.    303-772-3400  
1900 Pike Road  
Longmont, Colorado 80501

*For Solbourne support call: 1-800-447-2861*

**Solbourne, Series4, Series5, Virtual Desktop, Kbus, OS/MP, Object Interface Library (OI), and the Solbourne logo are trademarks of Solbourne Computer, Inc.**  
**SPARC, SunOS, and Sun-4 are trademarks of Sun Microsystems.**  
**UNIX is a trademark of the AT&T Bell Laboratories.**  
**Zilog is a trademark of Zilog Corporation.**  
**Exabyte is a trademark of Exabyte Corporation.**  
**Ciprico is a trademark of Ciprico Corporation.**  
**Xylogics is a trademark of Xylogics Corporation.**  
**VMEbus is a trademark of VMEbus Manufacturers' Association.**

**This document contains highly-sensitive confidential information  
that may only be viewed by employees of Solbourne Computer, Inc.**

**DO NOT COPY OR DISTRIBUTE THIS MANUAL.**

Part Number: 101250-AB

February 1990

Copyright © 1990 by Solbourne Computer, Inc. All rights reserved. No part of this publication may be reproduced, stored in any media or in any type of retrieval system, transmitted in any form (e.g., electronic, mechanical, photocopying, recording) or translated into any language or computer language without the prior written permission of Solbourne Computer, Inc., 1900 Pike Road, Longmont, Colorado 80501. There is no right to reverse engineer, decompile, or disassemble the information contained herein or in the accompanying software.

Solbourne Computer, Inc. reserves the right to revise this publication and to make changes from time to time without obligation to notify any person of such revisions or changes.

## Preface

This manual covers the theory of operation of the Solbourne Series4 and Series5 systems. Theory of operations deals with the function of the major hardware components of the Series4 and Series5 systems. The difference between Series4 and Series5 is the design and performance of the CPU Boards in each system; otherwise the systems are identical in hardware functions. The theory of operation of each of these CPU Boards is detailed in this document.

The Series4 and 5 theory of operations applies to two implementations of each series—the Series4 and Series5/500 (or desktop) model and the Series4 or Series5/600 (or deskside) model. The major difference between the desktop and deskside model is the inclusion of the VMEbus backplane in the deskside, an interface not provided in the desktop model. The material dealing with the VMEbus interface in the System Board section applies only to Series4 and Series5/600 systems. The System Board of the Series4 and Series5/500 is identical to that used in the Series4 and Series5/600, but the VMEbus connections are not used.

Theory of operation covers the functional blocks on each board, how they interface to the larger system and what their inputs and outputs are.

This manual is divided into nine sections, as follows:

### Section 1 - System Overview

This section introduces each of the components that is described in this manual.

### Section 2 - Kbus Operation

This section describes the Kbus, which is the main communication link among the components detailed in this manual. It provides the name and description of each bus signal.

### Section 3 - Series4 CPU Operation

This section describes the Series4 CPU Board, giving diagrams and descriptions of major internal buses, functional blocks, and external interfaces.

### Section 4 - Series5 CPU Operation

This section describes the Series5 CPU Board, giving diagrams and descriptions of major internal buses, functional blocks, and external interfaces.

### Section 5 - System Board Operation

This section describes the System Board, giving diagrams and descriptions of major internal buses, functional blocks and external interfaces. The System Board is the communications hub for I/O in the system.

### Section 6 - 16 Mbyte Memory Board Operation

This section describes the 16 Mbyte Memory Board, giving diagrams and descriptions of major internal buses, functional blocks, and external interfaces.

### Section 7 - 32 Mbyte Memory Board Operation

This section describes the 32 Mbyte Memory Board, giving diagrams and descriptions of major internal buses, functional blocks, and external interfaces.

### Section 8 - CG 40 Color Board Operation

This section describes the CG 40 Color Board, giving diagrams and descriptions of

major internal buses, functional blocks, and external interfaces. The CG 40 was the original color frame buffer designed by Solbourne.

**Section 9 - CG 30 Color Board Operation**

- This section describes the CG 30 Color Board, giving diagrams and descriptions of major internal buses, functional blocks, and external interfaces. The CG 30 provides a performance improvement over the CG 40.

## Table of Contents

Section 1: System Overview .....	1-1
1.1 Introduction .....	1-1
1.1.2 Series4 and Series5 CPU Boards .....	1-1
1.1.3 Series4 and Series5 Implementations .....	1-1
1.2 Major System Functions .....	1-2
1.3 Kbus .....	1-5
1.4 VMEbus .....	1-5
1.5 CPU Boards .....	1-6
1.6 System Board .....	1-6
1.7 CG 30 Color Board .....	1-7
1.8 Memory Boards .....	1-8
1.9 Peripherals .....	1-8
Section 2: Kbus Operation .....	2-1
2.1 Introduction .....	2-1
2.2 Kbus Organization .....	2-2
2.3 Bus Arbitration .....	2-5
2.4 Transactions .....	2-6
2.4.1 Cache Memory Transactions .....	2-6
2.4.2 RIO Transactions .....	2-7
2.4.3 Other Transactions .....	2-7
2.5 Cache Protocol .....	2-7
2.6 System Interrupts .....	2-8
Section 3: Series4 CPU Board Operation .....	3-1
3.1 Introduction .....	3-1
3.2 CPU Board Address, Data, and Control Buses .....	3-2
3.3 CPU Section .....	3-8
3.3.1 CPU Subsection .....	3-9
3.3.2 MMU Subsection .....	3-11
3.3.3 Cache Tag Subsection .....	3-12
3.3.4 Cache Subsection .....	3-13
3.4 Kbus Interface Section .....	3-13
3.4.1 ROM Subsection .....	3-14
3.4.2 Bus Launcher Subsection .....	3-15
3.4.3 Bus Watcher Subsection .....	3-17
3.4.4 Error Check Circuit Logic Subsection .....	3-18
3.5 Interrupt Handler Section .....	3-18
Section 4: Series5 CPU Board Theory of Operation .....	4-1
4.1 Introduction .....	4-1
4.2 Series5 CPU Board Features .....	4-1
4.3 Overview .....	4-2
4.4 CPU Section .....	4-4
4.4.1 Address Pipeline .....	4-5
4.4.2 Pipeline Operations .....	4-5
4.4.3 Cache Organization .....	4-6

4.5 Launcher Section .....	4-7
4.6 Kdat Section .....	4-9
4.7 Watcher Section .....	4-10
<b>Section 5: System Board Operation .....</b>	<b>5-1</b>
5.1 Introduction .....	5-1
5.2 Kbus Interface Subsection .....	5-2
5.3 VMEbus Interface Subsection .....	5-3
5.4 SCSI/Ethernet Interface Subsection .....	5-4
5.5 Slow Bus Subsection .....	5-6
5.6 I/O ASIC Subsection .....	5-8
5.7 Monochrome Video Subsection .....	5-8
5.7.1 Kbus Interface for Monochrome Graphics .....	5-9
5.7.2 Video Control Subsection .....	5-10
5.7.3 Video RAM Subsection .....	5-12
5.7.4 Video Drivers Subsection .....	5-15
<b>Section 6: 16 Mbyte Memory Board Operation .....</b>	<b>6-1</b>
6.1 Introduction .....	6-1
6.2 Address Register and Latch Subsection .....	6-2
6.3 Memory Control Subsection .....	6-3
6.4 Kbus Data Interface Subsection .....	6-5
6.5 Address Multiplexers Subsection .....	6-6
6.6 Memory Array Subsection .....	6-7
6.7 I/O Devices Subsection .....	6-9
6.8 Kbus Transactions .....	6-10
<b>Section 7: 32 Mbyte Memory Board Operation .....</b>	<b>7-1</b>
7.1 Introduction .....	7-1
7.2 Address Register and Latch Subsection .....	7-3
7.3 Memory Control Subsection .....	7-4
7.4 Kbus Data Interface Subsection .....	7-5
7.5 Address Multiplexers Subsection .....	7-6
7.6 Memory Array Subsection .....	7-7
7.7 I/O Devices Subsection .....	7-9
<b>Section 8: CG 40 Color Graphics Board .....</b>	<b>8-1</b>
8.1 Introduction .....	8-1
8.2 Kbus Address Interface .....	8-2
8.3 Kbus Interface Control Logic .....	8-2
8.4 Kbus Data Interface .....	8-2
8.5 Overlay Planes .....	8-2
8.6 Image Plane .....	8-2
8.7 Brooktree RAMDAC .....	8-2
8.7.1 Bt458 Registers .....	8-3
8.7.2 Kbus Reads and Writes to the Bt458 .....	8-3
8.8 Video Control Logic .....	8-5
8.9 Control Registers .....	8-5
8.9.1 CG 40 Board Control Register .....	8-6
8.9.2 Space Register .....	8-6
8.9.3 Video Control Registers .....	8-6
8.10 Interrupt Information and Interrupt Vector Registers .....	8-7

8.11 ID PROM .....	8-7
Section 9: CG 30 Color Graphics Board .....	9-1
9.1 Introduction .....	9-1
9.2 Major Design Goals .....	9-1
9.3 CG 30 Color Board System Connection .....	9-1
9.4 Functional Block Diagram .....	9-3
9.5 Kbus Interface .....	9-4
9.6 CG 3 Compatibility Section .....	9-4
9.6.1 The Frame Buffer .....	9-5
9.6.2 Color Map .....	9-6
9.6.3 Shadow RAM .....	9-6
9.6.4 Video Output Signals .....	9-6
9.7 The ID Space Section .....	9-7
9.7.1 Cursors .....	9-7
9.7.2 CG 30 I/O Ports .....	9-7



## List of Figures

Figure 1-1.	Major Series4 and Series5/600 Components and Connections .....	1-2
Figure 1-2.	Relation Between Video Image and Frame Buffer RAM .....	1-6
Figure 2-1.	Kbus Backplane Connectors .....	2-2
Figure 3-1.	CPU Board Block Diagram .....	3-2
Figure 3-2.	CPU Section Buses .....	3-3
Figure 3-3.	Kbus Interface Section Buses .....	3-6
Figure 3-4.	CPU Subsection Block Diagram .....	3-9
Figure 3-5.	MMU Subsection Block Diagram .....	3-11
Figure 3-6.	Cache Tag Subsection Block Diagram .....	3-12
Figure 3-7.	Cache Subsection Block Diagram .....	3-13
Figure 3-8.	ROM Subsection Block Diagram .....	3-14
Figure 3-9.	Bus Launcher Subsection Block Diagram .....	3-15
Figure 3-10.	Bus Watcher Subsection Block Diagram .....	3-17
Figure 3-11.	ECC Logic Block Diagram .....	3-18
Figure 4-1.	Block Diagram of the Series5 CPU Board .....	4-2
Figure 4-2.	Block Diagram of the CPU Section .....	4-4
Figure 4-3.	Block Diagram of the Launcher Section .....	4-7
Figure 4-4.	Block Diagram of the Kdat Section .....	4-9
Figure 4-5.	Block Diagram of the Watcher Section .....	4-10
Figure 5-1.	System Board Block Diagram .....	5-1
Figure 5-2.	Kbus Interface Subsection Block Diagram .....	5-2
Figure 5-3.	VMEbus Interface Subsection Block Diagram .....	5-4
Figure 5-4.	SCSI/Ethernet Interface Subsection Block Diagram .....	5-5
Figure 5-5.	Slow Bus Subsection Block Diagram .....	5-6

Figure 5-6.	Monochrome Graphics Subsection Block Diagram .....	5-8
Figure 5-7.	Kbus Interface Subsection Block Diagram .....	5-9
Figure 5-8.	Video Control Subsection Block Diagram .....	5-10
Figure 5-9.	Video RAM Subsection Block Diagram .....	5-12
Figure 5-10.	Video Drivers Subsection Block Diagram .....	5-15
Figure 6-1.	Memory Board Block Diagram .....	6-1
Figure 6-2.	Address Register and Latch Subsection Block Diagram .....	6-2
Figure 6-3.	Memory Control Subsection Block Diagram .....	6-3
Figure 6-4.	Kbus Data Interface Subsection Block Diagram .....	6-5
Figure 6-5.	Address Multiplexers Subsection Block Diagram .....	6-6
Figure 6-6.	Memory Array Subsection Block Diagram .....	6-8
Figure 6-7.	I/O Devices Subsection Block Diagram .....	6-9
Figure 7-1.	32 Mbyte Memory Board Block Diagram .....	7-2
Figure 7-2.	Address Register and Latch Subsection Block Diagram .....	7-3
Figure 7-3.	Memory Control Subsection Block Diagram .....	7-4
Figure 7-4.	Kbus Data Interface Subsection Block Diagram .....	7-5
Figure 7-5.	Address Multiplexers Subsection Block Diagram .....	7-6
Figure 7-6.	Memory Array Subsection Block Diagram .....	7-8
Figure 7-7.	I/O Devices Subsection Block Diagram .....	7-9
Figure 8-1.	CG 40 Block Diagram .....	8-1
Figure 9-1.	CG 30 System Connections .....	9-1
Figure 9-2.	CG 30 Functional Block Diagram .....	9-3
Figure 9-3.	Relation Between Video Image and Frame Buffer RAM .....	9-5

## List of Tables

Table 5-1.	System Board Interrupts .....	5-7
Table 8-1.	Writing to Overlay Registers .....	8-4
Table 8-2.	Reading Overlay Registers .....	8-4
Table 8-3.	Writing to the Other Registers .....	8-5
Table 8-4.	Reading Other Registers .....	8-5
Table 9-1.	Video Timing .....	9-7



## Section 1: System Overview

### 1.1 Introduction

The Series4 and Series5 family is a high-speed computing platform designed to provide optimal performance at a reasonable price. It is designed specifically as a workstation for the engineering, scientific, and technical marketplace.

The Series4 and Series5 family is designed around industry standard technologies such as the UNIX operating system, the SPARC Reduced Instruction Set Computer (RISC) processor architecture, VMEbus, Ethernet local area network, and RS-423-A (compatible with RS-232-C) asynchronous serial interface, and Small Computer System Interface (SCSI). Through adoption of the SPARC architecture, and through licensing of the SunOS, the Series4 and Series5 systems provide binary compatibility with Sun Microsystems' SPARC (Sun-4 and SPARCstation) compatible software.

In addition to the industry-standard technologies, the Series4 and Series5 family includes some proprietary hardware enhancements which improve its overall operation. These proprietary improvements include: the high-speed Kbus backplane, the high-speed Central Processing Unit (CPU) with full virtual memory management, and a high-speed Application Specific Integrated Circuit (ASIC) devoted to controlling the flow of information to and from peripheral Input/Output (I/O) devices.

#### 1.1.1 Series4 and Series5 CPU Boards

The distinction between Series4 and Series5 is in the CPU Board or boards that are installed in a system. When a system is upgraded from Series4 to Series5, an upgrade to OS/MP release 4.0C or greater is needed, as well. However, the Series4 can also operate with the 4.0C or greater release. Other than that, the Series4 and Series5 versions are virtually identical. That is, no change need be made to the other Kbus boards, to the mass storage devices, monitor, and so on. The only additional change is to the system identification EAROM on the System Board. Therefore, in the rest of this manual, virtually no distinction need be made between the Series4 and Series5 systems, except in Sections 3 and 4 where the CPU Boards themselves are covered.

☆ ☆ ☆ NOTE ☆ ☆ ☆

Changes to the EAROM are described in the *Series4 and Series5/600 Service Manual* (101249).

#### 1.1.2 Series4 and Series5 Implementations

Two basic implementations of the Series4 and Series5 architecture are available: a desktop and a deskside version. These are designated by the "500" (desktop) and "600" (deskside) extensions to the model numbers. The variations of "500" and "600" (e.g., 501, 502, 602, 603, etc.) represent the number of CPU Boards installed in the system. Thus, a typical model number is Series5/502 — signifying which series CPU Board is present (Series5), which chassis type (desktop), and the number of CPU Boards (two).

The main functional differences between the desktop and deskside implementations are as follows:

- The VMEbus is not present in the desktop version
- The SCSI bus has fewer internal connection points in the desktop version
- Fewer Kbus backplane slots are available in the desktop version

Otherwise, the two versions are functionally equivalent. The same CPU, Memory, Color Graphics, and System Boards operate in them. Similar SCSI devices (but not the same ones) are available internally, and identical ones are available externally. Therefore, in the rest of this manual, virtually no distinction is made between the desktide and desktop implementations.

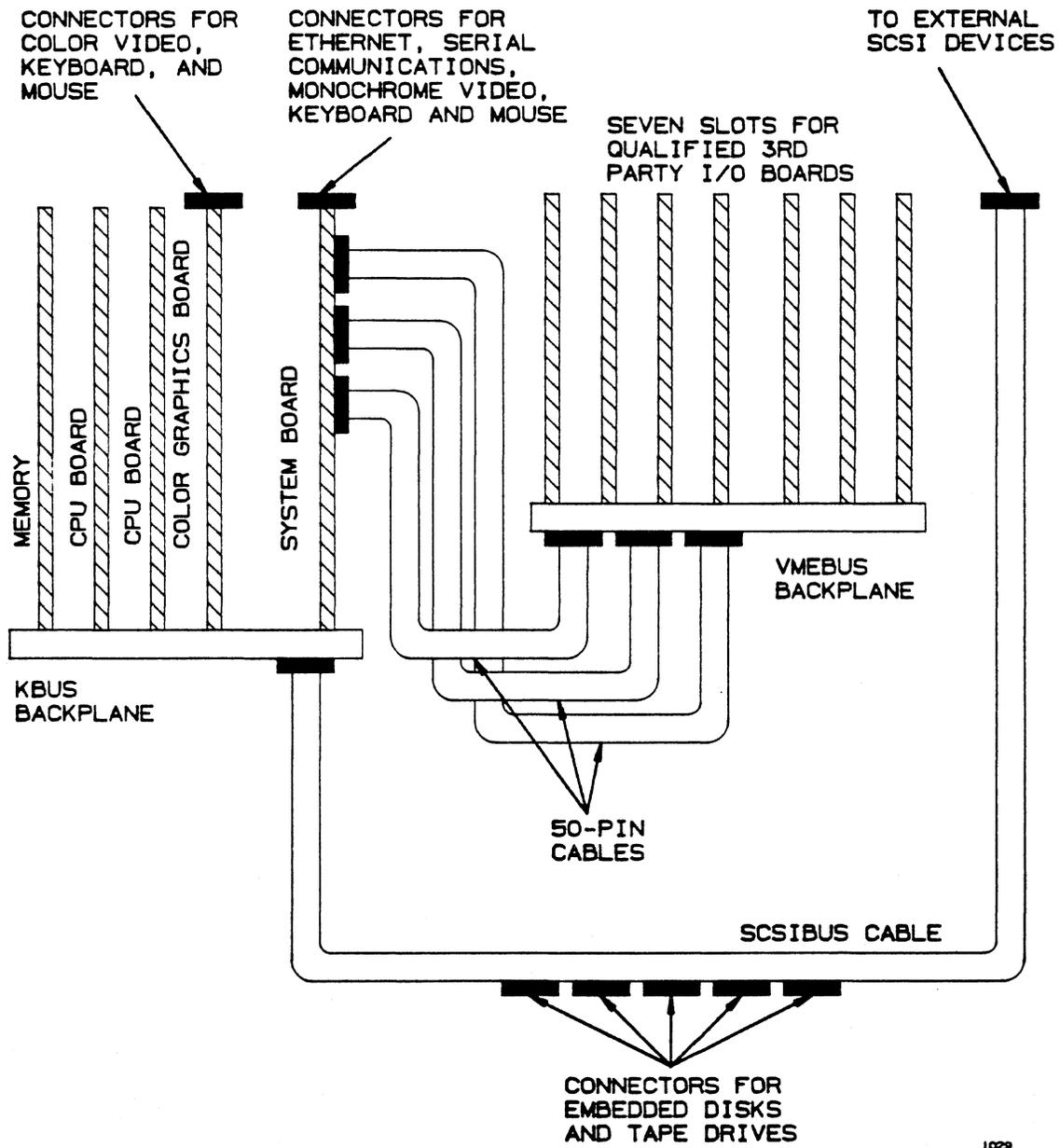
## **1.2 Major System Functions**

The major components of the Series4 and Series5 include: the display monitor, the keyboard, the mouse, and the processor unit. The processor unit is the heart of the system since it contains the system CPU, system memory, some of the mass storage, and (in systems with VMEbus) interface boards. The keyboard and mouse are I/O devices which provide input to the CPU in the processor unit. The display monitor is the primary device for providing feedback to the operator. A speaker in the keyboard provides audible feedback to the operator in the form of clicks and beeps.

There are several printed circuit boards in the Series4 and Series5 which are dedicated to specific functions within the system.

- Kbus backplane
- CPU Board(s)
- System Board
- Graphics Board
- Memory Board(s)
- VMEbus backplane (not in desktop version)

Figure 1-1 shows these components and the major system interconnections.



1029

Figure 1-1. Major Series4 and Series5 Components and Connections

☆ ☆ ☆ NOTE ☆ ☆ ☆

Figure 1-1 shows a fully configured Series4 or Series5 system. A desktop model would not implement the VMEbus interface.

The Kbus backplane is the medium through which all Kbus boards communicate. The Kbus backplane is a proprietary backplane that is at the center of the system's operation. The Kbus backplane can interconnect up to seven (five in desktop) Kbus boards. All signals that pass to and from the processor unit must pass over the Kbus backplane.

The CPU Board controls most system operations such as operating system interface, memory management, interrupt handling, floating point calculations, etc. There can be up to five CPU Boards in the system (three in the desktop) although only one is needed for basic system operation.

The System Board plays a pivotal role in the operation of the processor unit. It is the main point of interconnection for all I/O devices in the system. Input from the keyboard enters the processor system via the System Board. Input from the mouse passes through the System Board. Data transferring to and from any of the mass storage devices must transfer through the System Board. Signals transferring to and from the processor unit via the Ethernet are received and transmitted through the System Board. The only I/O signals that don't pass through the System Board are signals that are sent to the color display monitor (monochrome video is also on the System Board).

The Color Graphics Board takes care of all signals that pass to the display monitor. The Color Graphics Board contains on board Random Access Memory (RAM) which serves as a frame buffer for the operating system. The frame buffer RAM serves as a digital representation of the graphics being displayed on the monitor screen. To change any element on the screen, the CPU only need change the value of the element's representation in the Graphics Board frame buffer RAM. The Graphics Board then converts the data in the frame buffer RAM into vertical and horizontal synchronization signals for the Cathode Ray Tube (CRT) driver circuit.

Up to five Memory Boards can be installed in the processor unit. Each Memory Board supplies 16 or 32 Mbytes of RAM to the system. The operating system uses the RAM on the Memory Boards to store information such as operating system code, application code, data files, etc. The Series4 and Series5 architectures are based on a virtual memory system in which the operating system takes responsibility for keeping the virtual address access currently needed by the system supplied with translations to physical memory locations. As in all UNIX systems, the Solbourne Series4 and Series5 memory management is demand paged. This means that memory is divided into 8 Kbyte segments called pages. Each process views the memory as though memory belonged to it exclusively. This illusion is maintained by the operating system, which provides the pages needed by each process currently operating. The "demand" in demand paged, means that the operating system will bring pages into memory only as they are needed rather than using some algorithm to bring all possible needed pages in at the beginning of a process. The process runs until it tries to access a page that is not in memory and finds it absent. At this point an interrupt is sent to the processor to initiate a page-in sequence. As processes move in and out of the CPU, some pages are no longer needed, and they are moved out to disk. When needed again, they are swapped back in. This entire process is transparent to the process and invisible to the hardware, except that the hardware knows if a page is not present in memory. When it detects this, it notifies the operating system to execute a paged-out procedure which results in the needed page being returned to main memory.

The VMEbus backplane is an industry standard backplane in which qualified third-party VMEbus boards can connect to the Series4 or Series5 processor units. The VMEbus backplane connects to the Kbus backplane through three logic cables that connect to the System Board. The VMEbus backplane can support up to seven boards that conform to the VMEbus standard. Boards that can plug into the VMEbus backplane include:

- Solbourne Serial Multiplexer Board (Xylogics 781)

- Xylogics 753 SMD Controller
- Interphase Eagle 4207 Ethernet Controller

Additional third-party boards can be installed with the user performing the qualification procedure prior to operating. See Solbourne's *Solutions* catalog for more detail about third-party product availability and support. All compatible boards, whether user- or Solbourne-qualified, must be 6u form factor.

### 1.3 Kbus

The Kbus backplane is a seven slot, 21-connector printed circuit board located at the back of the Kbus card cage. The Kbus card cage holds up to seven Kbus boards. Each slot of the backplane has three connectors which interconnect each Kbus board with the backplane. The connectors are J1, J2, and J3. The signals carried on all three connectors get used during normal operation.

J1 has the address bus and most interface signals. J2 provides all DC power and bus clock signals. J3 contains all data lines and a few miscellaneous interface signals. The Kbus is comprised of several multi-line busses:

- a 32-bit address bus, KADDR [31:0]
- a four-bit I/O space bus, KSP[3:0]
- a 64-bit data bus, KDATA[63:00]
- an eight-bit check byte bus, KCB[7:0]
- two eight-bit I/O request and acknowledge buses, KIOREQ[15:8] and KIOACK[15:8]
- an eight-bit transaction identification bus, KTID[7:0]
- a five-bit transaction type bus, KTTYPE[4:0]

The Kbus also contains numerous clock and control signals which are used for synchronizing the transfer of information among all System Boards.

Kbus boards are Kbus masters or Kbus slaves during the transfer of information. Some of the boards are always Kbus masters and some of the boards are always Kbus slaves. Only Kbus masters can start Kbus transactions. Slaves can only respond to the Kbus masters. The System Board can be a master or a slave, depending on the type of transaction being performed. Kbus masters include the CPU Boards and the System Board. Kbus slaves include the Memory Boards, the Graphics Board, and the System Board when receiving instructions from the Kbus CPU Boards.

In order for one of the Kbus masters to transfer data on the Kbus, it must first become the master of the bus. Each Kbus master gains ownership of the bus through an arbitration process. There is an arbitration Programmable Array Logic (PAL) on the Kbus backplane that determines which Kbus master has control of the bus. Each Kbus master wanting to perform a transfer over the bus sends a board request signal to the arbitration PAL. The PAL uses a round-robin approach to granting ownership to the requesting Kbus masters. Only one Kbus master owns the bus during any one transfer.

### 1.4 VMEbus

The VMEbus connects to the Kbus through the System Board with three 50-pin ribbon cables. The VMEbus backplane is a seven slot, 14-connector printed circuit board located at the back of the VMEbus card cage. The VMEbus card cage holds up to seven VMEbus boards. Each slot of

the backplane has two connectors which interconnect each VMEbus board with the backplane. These connectors are J1 and J2. Most signals that are essential to VMEbus data transfers are located on J1. J2 is an auxiliary connector used by some VMEbus boards for specific functions. On the Series4 and Series5 the J2 connector carries the upper 16 bits of the 32-bit data bus. These are the only signals of the J2 connector that are used. The VMEbus is comprised of several multi-line busses:

- A 24-bit address bus A00-A23
- A 16-bit address bus A00-A15
- A 32-bit data bus D00-D31
- Three four-bit bus request/grant buses BR0-BR3, BG0IN-BG3IN, and BG0OUT-BG3OUT
- A six-bit address modifier bus AM0-AM5
- Seven levels of interrupt requests IRQ1-IRQ7

As on the Kbus, VMEbus boards are either masters or slaves. The SMD controller, for example, is a slave to the System Board when data is being written to an SMD disk device. When the SMD device is reading from a mass storage device, though, it becomes the master and the System Board is the slave. There are no Memory Boards in the VMEbus card cage, so when a VMEbus master is writing to RAM, it writes to a Memory Board in the Kbus card cage.

VMEbus master boards gain ownership of the VMEbus through the VMEbus convention of fixed priority. The slot number in which the VMEbus resides determines its priority level during bus arbitration. The highest priority slot is the first slot in the card cage. The lowest priority slot is the last slot in the cage. If a board in VMEbus Slot 1 and a board in VMEbus Slot 5 request use of the bus at the same time, the board in Slot 1 always gets first use of the bus because it has the higher priority location. Because board placement determines VMEbus arbitration level, VMEbus boards that must respond to events are placed in higher level slots.

## 1.5 CPU Boards

The CPU Boards used in the Series4 and Series5 is designed around the Scalable Processor Architecture (SPARC) Reduced Instruction Set Computer (RISC) processor. Currently two implementations of the SPARC architecture are used in Series4 and Series5 family CPU Boards — the 33 MHz Cypress SPARC microprocessor and the 16 MHz Fujitsu SPARC microprocessor, in the Series5 and Series4 respectively. Each CPU implements the SPARC instruction set and register file design. Both are used with a Floating Point Unit as a coprocessor, in the case of the Cypress the Weitek 3171; in the case of the Fujitsu, the Weitek 1164/1174.

Along with the two processors available in the family, there are two CPU Board designs. The CPU Board based on the Fujitsu part is called the Series4 CPU; the CPU Board based on the Cypress part is called the Series5 CPU. Multiple CPU Boards can be installed in Series4 and Series5 family systems; however, Series4 and Series5 CPUs cannot be mixed in the same unit.

Both boards include the TLB, Cache, Bus Watcher, Boot ROM and ECC functional blocks. Both board designs implement the Solbourne patented Cache Consistency Protocol that underlies the Multiple Processor enhancement Solbourne has made in its OS/MP, its version of the SunOS.

## 1.6 System Board

The System Board handles all I/O processes for the Series4 and Series5 systems. It interconnects the Kbus and the VMEbus. It also contains the transceivers for the keyboard, mouse, and two

RS-423-A ports. It contains the monochrome video frame buffer for communicating with a monochrome graphics monitor. The System Board controls all RAM refresh cycles for the Memory Boards in the processor unit. It also provides Kbus maintenance functions by providing Kbus clocking signals to which all Kbus transactions are synchronized.

The I/O ASIC is the major component of the System Board. The I/O ASIC enables the System Board to interface directly with Ethernet and mass storage devices without the use of VMEbus-based boards.

The System Board is sectioned into five basic functional areas: the Kbus interface area, the SCSI/Ethernet interface section, the VMEbus interface section, the I/O port functional area and the I/O ASIC functional area. The Kbus interface area synchronizes all signals that pass between the other areas of the System Board and the Kbus. The SCSI/Ethernet area interfaces all mass storage devices and the Ethernet to the I/O ASIC. The VMEbus interface area of the System Board controls all data transfers between the VMEbus and the I/O ASIC or the Kbus interface area. The VMEbus area is inactive in the desktop version of the Series4 and Series5 family. The I/O port section of the system is dedicated to receiving and transmitting data between the Kbus interface area and all the I/O ports (keyboard, mouse, RS-423-A, etc.) The I/O ASIC section of the board includes all support buffers and transceivers that aid the ASIC in its operation of all System Board functions.

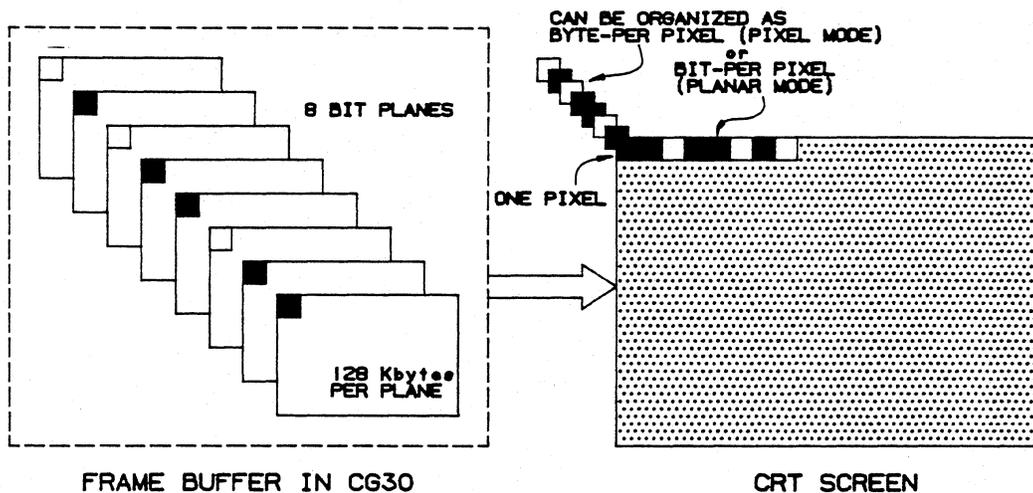
## **1.7 CG 30 Color Board**

The CG 30 connects to the Kbus backplane of Series4 and Series5 family products. Although the CG 30 Color Board resides on the Kbus, it exists in I/O Space. Accesses to the frame buffer are I/O cycles on Kbus rather than memory accesses.

The board contains a frame buffer RAM that is compatible with a Sun CG3 and some hardware enhancements that make the manipulation of images more efficient. A frame buffer is memory directly mapped to the picture elements (pixels) of a bit-mapped graphics display monitor so that if the first bit in the memory corresponds to the first pixel on the monitor screen, and so on.

The CG 30's frame buffer consists of a total of 1 Mbyte of RAM.

Figure 1-2 shows conceptually two ways that a video image can be organized with the CG 30.



7014

Figure 1-2. Relation Between Video Image and Frame Buffer RAM

The video RAM in the CG 30 is organized in eight 128-Kbyte bit planes. The 1152 x 900 pixel video screen contains approximately 1 million pixels. Thus, there are approximately eight times as many bits in the frame buffer as there are pixels in the video image, as  $128 \text{ Kbytes} \times 8 \times 8 = 8 \text{ Mbit}$ .

## 1.8 Memory Boards

Each Kbus Memory Board used in the Series4 and Series5 family has either 16 or 32 Mbytes of dynamic RAM contained in four separate banks. The RAM chips used in the Memory Board are 256 Kbyte by 4 bit chips. So, two chips make up a full byte. Each bank of RAM has high and low sections for the high and low nibbles of the data bytes.

The Memory Board is sectioned into two basic functional areas: memory control and the RAM arrays. Since the Memory Board is a Kbus slave, there is no Kbus interface functional area like those areas found on the other Kbus boards. The memory control area loads address and data information from the Kbus. The memory control area then deciphers the Kbus information and, if commanded, prepares the RAM arrays for a memory access.

The RAM arrays perform the function of multiplexing and channeling the data to and from the Kbus as commanded from the memory control area. The RAM arrays transfer data out to the Kbus during a read cycle and collect the data from the Kbus during a write cycle.

## 1.9 Peripherals

Peripheral devices supported in the Series4 and Series5 architecture are generally developed by third parties and adapted for use in the Series4 and Series5 system by Solbourne. As such, their theory of operations is not within the scope of this manual. The following is a brief summary of the peripheral interfaces available in the Series4 and Series5 families.

Interfaces in the Series4 and Series5 systems are:

- Small Computer System Interface (SCSI)
- Ethernet (IEEE 802.3)
- RS-423-A Asynchronous Serial Interface
- VMEbus
- Keyboard/mouse
- Monitor

Each interface except for the color graphics monitor is connected to the System Board. Internal connections are provided within the Series4 and Series5 system for VMEbus and SCSI. That is SCSI-compliant devices and VMEbus-compliant devices are locally connected inside the unit chassis. Note, however, that the desktop model does not implement the VMEbus interface. SCSI also has an external connection on all models. Ethernet and RS-423-A are limited to optional external connections, as are keyboard and monitor.

The SCSI interface implemented by Solbourne is the single-ended version of the ANSI standard, and has been slightly modified from the Sun Microsystems' implementation. Synchronous block mode transfers are supported. The VMEbus standard is implemented by Solbourne with certain interpretations. For details of both implementations, see the *Series4 and Series5/600 Service Manual* (102249).

The device drivers available in the Solbourne OS/MP include:

- **xp(4s)**—Xylogics serial interface for multiplexer board
- **sd(4s)**—SCSI disk
- **st(4s)**—SCSI tape
- **si(4s)**—SCSI I/O ASIC
- **sr(4s)**—SCSI Rimfire (Ciprico)
- **ei(4s)**—Ethernet I/O ASIC
- **eg(4s)**—Ethernet Eagle (Interphase)
- **xd(4s)**—Xylogics disk (SMD) controller
- **xs(4s)**—Zilog 8530 serial controller chip (keyboard, mouse)

Note that, from a hardware point of view, Rimfire Ciprico is no longer supported.



## Section 2: Kbus Operation

### 2.1 Introduction

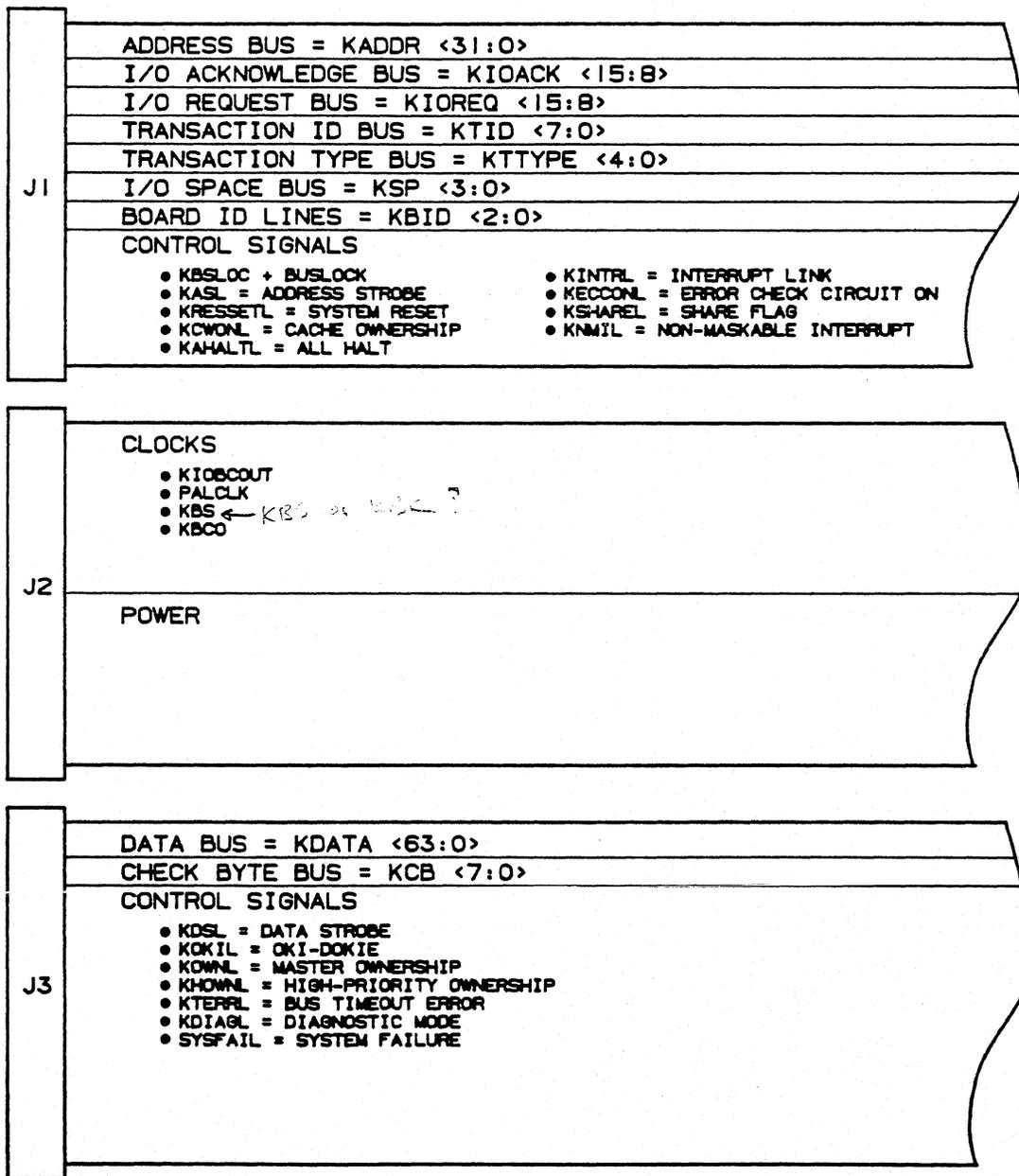
The Kbus backplane interconnects all the logic boards in the Series4 and Series5 systems. All information transfers transactions that occur between system logic boards must pass over the Kbus and conform to the Kbus transaction schemes. Logic voltage for the Kbus boards is provided through the Kbus backplane.

Kbus boards are either Kbus masters or Kbus slaves during the transfer of information. Some of the boards are always Kbus masters and some of the boards are always Kbus slaves. Only Kbus masters can start Kbus transactions. Slaves can only respond to the Kbus masters. The System Board can be a master or a slave, depending on the type of transaction being performed.

In order for one of the Kbus masters to transfer data on the Kbus, it must first become the master of the bus. Each Kbus master gains ownership of the bus through an arbitration process. There is an arbitration Programmable Array Logic (PAL) on the Kbus backplane that determines which Kbus master has control of the bus. Each Kbus master wanting to perform a transfer over the bus sends a board request signal to the arbitration PAL. The PAL uses a round-robin approach to granting ownership to the requesting Kbus masters. Only one Kbus master owns the bus during any one transfer.

## 2.2 Kbus Organization

Physically, the Kbus backplane is comprised of seven slots with three connectors on each slot into which the Kbus boards plug. The connectors are J1, J2, and J3. J1, the top connector in each slot, carries all address, interface, and control buses and signals used between the boards. J2, the middle connector, carries clocking signals and power to the boards. J3, the bottom connector of each slot, carries signals and buses dealing with the actual transfer of data across the bus (see Figure 2-1).



1031

Figure 2-1. Kbus Backplane Connectors

Some of the signals of the Kbus are general signals used for system and Kbus operation. Signals used for general Kbus maintenance are:

- **KRESETL** - The Kbus reset signal is used to indicate a power up reset condition and is synchronously asserted and deasserted by a voltage sensing circuit on the System Board. KRESETL can be asserted by any device.
- **KNMIL** - The Kbus non-maskable interrupt signal causes a non-maskable exception to be encountered by all CPUs in the system.
- **KIOBCOUT** - The System Board bus clock out signal is the source clock used by the Kbus backplane clock drivers to create the KBC/KBCO clocks. KIOBCOUT is originated on the System Board and feeds only the Kbus clock drivers.
- **PALCLK** - The Programmable Array Logic clock timer provides clocking states to the Kbus PAL for its bus arbitration duties. PALCLK originates on the System Board and feeds only the on-board PAL. PALCLK runs at a nominal speed of 20MHz.
- **KBC and KBCO** - These identical bus clocks are the clock sources for each board on the Kbus. KBC and KBCO are used on each board for generating the board's RBC (Reconstructed Bus Clock) which is used as the timing reference for all signal transitions. KBC and KBCO clocks are driven by the System Board, and has a nominal frequency of 20 MHz.
- **KDIAGL** - The Kbus diagnostics line is used to take the system out of normal operational mode and place it into diagnostic mode. This signal is asserted by setting a switch mounted to the front of the system to the "Diagnostic" position and rebooting the system.

Some of the buses and signals of the Kbus backplane are primarily concerned with acquisition of the Kbus. Before any board can implement a transaction, it must first arbitrate for and gain control over the Kbus. Arbitration is performed by an arbitration PAL mounted on the Kbus backplane. Several control signals are involved with Kbus acquisition:

- **KBDRL** - Each slot's Kbus board request signal connects between the board and the arbitration PAL on the backplane. One KBDRL line is connected to each card slot. Each board asserts KBDRL when it wishes to become master over the Kbus. Once it has become bus master and has received an KOWNL signal, it continues to hold KOWNL until it releases KBDRL for at least one state. Releasing KBDRL unlocks the arbitration PAL to select another master.
- **KOWNL** - Each slot's KOWNL signal is generated by the arbitration PAL on the backplane in response to KBDRL. Each of the seven KOWNL signals is individually connected to a low priority bus master backplane slot. Only one KOWNL asserts at a given time and indicates that that slot is the next bus master. The KOWNL must be qualified at each board by KHOWNL, KBSLOC and KAHALTL which can negate ownership.
- **KAHALTL** - The Kbus all halt signal is gated with KOWNL by all bus masters. When KAHALTL is asserted, masters are prevented from initiating a new transaction. The I/O ASIC can ignore KAHALTL for running high priority I/O transactions and for keeping memory refreshed.
- **KINTRL** - The Kbus interrupt signal carries serial interrupt link messages between boards in the system. This signal is driven and sensed by all boards which have interrupt capability.
- **KHOWNL** - The Kbus high-priority ownership signal is asserted by the System Board when it needs to take control of the bus and run a transaction. KHOWNL is gated with KOWNL on each master to cause temporary loss of ownership even though BDRL may be asserted.

The KHOWNL signal is not an input to the arbitration PAL so that the state of the KOWNL signals do not change as a direct result of assertion of KHOWNL. When KHOWNL deasserts, ownership is transferred to the low priority master whose KOWNL input is asserted.

- **KIOREQ[15:8]** - The eight Kbus I/O request lines are used to make an I/O resource ready before running an I/O transaction. Eight KIOREQ channels are supported on the Series4 and Series5 family systems. Each KIOREQ is mapped into a physical I/O space of each board. I/O devices are configured by software through ID space transactions to respond to certain channels. For some I/O devices, KIOREQ is interpreted as a request to lock down an I/O resource such as the VMEbus.
- **KIOACK[15:8]** - The eight Kbus I/O acknowledge lines are actively driven signals asserted by an I/O device to indicate that the device is available or ready. A device may choose to assert KIOACK in response to an KIOREQ to indicate that a resource has been locked down, or it may simply leave IOACK asserted whenever it is not busy. I/O devices are configured through ID space to drive certain KIOACK lines.

Once any Kbus master has gained control of the Kbus, it can begin a transaction. Transactions begin with the Kbus master sending an address onto the Kbus. Addressing must conform with the standards of the Kbus specification. Signals that deal with Kbus addressing and board/device identification are:

- **KADDR** - The address bus is 32 bits wide and can address four Gbytes of cacheable memory. For transactions between Kbus board devices and I/O devices, the upper four bits of address are driven to 0 and these four bits are replaced with a value on the Kbus space bus. The Kbus address is valid when the KASL strobe is asserted.
- **KSP[3:0]** - The four Kbus space lines have the same timing as the address lines and the KTTYPER lines. The four bits of the KSP bus indicate the I/O space to which an I/O-related transaction is directed. Space 0 is reserved for boot space and space 1 is reserved for ID space. The state of the space lines is undefined during Kbus memory transactions.
- **KASL** - The Kbus address strobe is used to signal that a valid transaction, consisting of an address, a transaction type, and space, has been placed on the Kbus. Once KASL is asserted, it is held until either KTERRL or KDSSL and KOKIL, from the previous transaction have asserted. KASL deasserts for four states before it can reassert.
- **KTTYPER[4:0]** - The transaction type bits specify the type of transaction being initiated. KTTYPER has the same timing as the Kbus address lines.
- **KBSLOC** - The Kbus bus lock signal is used to indicate that an atomic transaction is in progress. KBSLOC is asserted at the same time as KASL and deasserts two state after KASL deasserts. In the case of an atomic transaction, KBSLOC is held until two states after the last KASL of the transaction deasserts. When the KBSLOC signal is asserted, the direction of the address, transaction type and space buffers is frozen until KBSLOC deasserts.
- **KCOWNL** - The Kbus cache ownership signal is asserted on cache transactions by the owner of the referenced cache block. If a bus watcher detects that it is the owner, then it indicates so by asserting KCOWNL on the second state after KAS has asserted. KCOWNL is used to prevent memory from supplying data. For Memory Boards, the assertion of KCOWNL turns a read access into a write access for the RAM array. The device asserting KCOWNL is the device that supplies data for the transaction.
- **KSHAREL** - The Kbus share signal is asserted by each bus watcher to indicate that it is holding a valid copy (any state of ownership other than invalid) of cache block addressed in

the current transaction. State timing for KSHAREL is the same as KCOWNL.

- **KBID[2:0]** - The board identification pins of each slot are hardwired with the number of the slot. This allows a board to read its slot number for use in ID space address decoding and KTID generation.
- **KTID[7:0]** - The transaction identification bus is an eight-bit wide multiplexed bus which indicates the source of Kbus transactions and Kbus data. KTID7:4 indicate the slot number of the sourcing device while KTID3:0 indicate which device on that board is the originator. The KTID bus is also used identify the source of data.
- **KOKIL** - Kbus okie-dokie signal is a handshaking signal which is asserted by a Kbus slave device to acknowledge a transaction. KOKIL is asserted for one state. The Memory Board is always the source of KOKIL for cacheable memory transactions. Caches that are owners of blocks receive KOKIL as an input signal. KOKIL is asserted by the slave I/O device during I/O transactions to acknowledge the cycle.
- **KTERRL** - The Kbus timeout error signal is an actively driven signal generated by the System Board when more than the specified amount of time has elapsed between the initiation of a transaction and the response of both KOKIL and KDSL. Assertion of KTERRL takes the place of KOKIL and KDSL, and terminates the transaction request. KTERRL is asserted for one state.

Once the device on the Kbus has been addressed, data can transfer over the bus. Signals that deal with the transfer of data over the Kbus include:

- **KDATA[63:0]** - The Kbus data bus is 64 bits (one cache line) wide. Data is always transferred four lines at a time in a block. For the transfer of data that is less than four lines wide, the four-line block transfer is still done, but the location of the data within the block is indicated by the address and transaction type. KCB is also transmitted with each line.
- **KCB[7:0]** - The eight-bit Kbus check byte field contains an ECC check field for the current eight-byte cache line data transfer on the Kbus. The KCB field is valid whenever the KECCONL signal is asserted. Kbus masters are responsible for generating the ECC field when sourcing a cacheable transaction and checking for ECC errors when receiving data.
- **KECCONL** - The Kbus error checking circuit on signal indicates that the data in the KCB field should be used with the current cache line as an ECC check byte.
- **KDSL** - The Kbus data strobe asserts with the transmission of the first cache line of a four-line block. KDSL is asserted by the device supplying data: the Kbus master during writes, the Kbus slave during reads, and the data owner during I/O data fetches.

## 2.3 Bus Arbitration

Any device on Kbus must become Kbus master before it can initiate a Kbus transaction. The arbitration PAL state machine on the Kbus backplane board controls the arbitration process. All the Kbus masters in the Kbus card cage have the same arbitration priority level except the System Board. **The System Board is a higher priority Kbus master.** CPU's are lower priority Kbus masters.

The arbitration PAL grants ownership of the Kbus in a round-robin fashion. Each Kbus board has its own unique board request and Kbus ownership signal, KBDRL and KOWNL. Each Kbus master wanting control of the Kbus asserts its KBDRL signal. Ownership is granted by the arbitration PAL by asserting the board's KOWNL signal. When the current owner of a the Kbus is finished with its transaction, it clears its KBDRL signal. This allows the arbitration PAL to

grant control of the Kbus to the next requesting Kbus master. Ownership cannot change as long as KBSLOC is asserted by the present Kbus owner.

The System Board can take control of the Kbus at any time by asserting the KHOWNL. The state of the lower priority KOWNL ownership signal does not change as the System Board takes control of the bus. Once the System Board is finished, Kbus ownership returns to the previous owner. The System Board cannot take control of the bus if the present owner is in the process of transferring data. If the previous master has asserted AS and BSLOC, the System Board waits.

## 2.4 Transactions

Transactions on Kbus are used by Kbus masters to communicate with slave devices and other Kbus masters. A device capable of initiating transactions must become Kbus master of the system before it can initiate a transaction. Once the Kbus master has control of the Kbus, it initiates a transaction by placing a physical address, transaction type code and, if necessary, an I/O space on the Kbus. The transaction begins when the Kbus master asserts the KASL address strobe.

All transactions on the Kbus are completed by the transfer of a 32 byte block of data. Each block contains four eight-byte cachelines. During transactions to the I/O devices, data transferred can be less than 32 bytes long. In this case, the data is transferred within the block transfer with the size of the data indicated by the transaction type code and the location of the data indicated by address bits KA2:0. There are three classes of transactions on Kbus:

1. Cacheable memory transactions which include full block transfers to and from system memory and selected I/O locations known as I/O Broadcast (IOB) transactions
2. Register to/from I/O (RIO) transactions which do not include full block transfers to and from selected I/O locations
3. Other types of transactions

Cacheable memory transactions transfer cache blocks between Kbus masters or between Kbus masters and Kbus memory slaves. Cacheable transactions address the full four Gbytes of physical address space (32 address bits) in the Series4 and 256 Mbytes of physical address space (28 address bits) in the Series5. Since the 32 byte cache block can be addressed with physical address lines KADDR[32:5], the physical address lines KADDR[4:0] are undefined during a cacheable memory transactions. During IOB transactions, KADDR[31:28] are driven low and are replaced by space bits KSP[3:0] to indicate the I/O space.

RIO transactions support byte addressability in a 28 bit physical address space defined by KADDR [27:0]. During an I/O transaction, KADDR[34:28] is driven to zero by the master and the value of SP[3:0] indicates which I/O space the target device resides in.

Other transactions that can occur on the Kbus include transactions such as RAM refreshes.

### 2.4.1 Cache Memory Transactions

Cache blocks are moved over Kbus KDATA bus in four successive transfers of eight bytes (64 bits) each. The address of the block is identified by address lines KADDR[31:5]. This address indicates the location of the most significant byte of the first of the four transfers. Once the data is ready for transfer, the data strobe KDSL asserts indicating valid data is on the data bus. Each byte of data transfers on each clock edge of the Kbus clock.

During all cache transactions, the memory system either supplies data or writes the data on Kbus into its RAM. When a cache block is transferred during a cacheable transaction, memory

is updated. Four types of cache memory transactions can be used to move cache blocks between Kbus masters, Kbus RAM, and I/O.

1. Read and Invalidate cache transactions cause the owner of a cache block to transmit the data from RAM then all owners of the data invalidate their copies.
2. Write and Invalidate cache transactions cause a cache block to be written to memory and all caches that have a copy of the block invalidate their copies.
3. Cacheable Read cache transactions instruct the owner of a block to transmit a read-only copy of the data.
4. I/O Broadcast (IOB) Kbus transactions allow cache block transfers between owners of cacheable memory and a specified I/O space. This type of cache transaction is also used to flush cache blocks out to memory.

### 2.4.2 RIO Transactions

RIO transactions transfer data between CPU registers and I/O devices when the data size of the transfer is less than the size of a full cache block. Each RIO transfer falls into one of four RIO transfer categories:

1. Byte Size transfer which reads or writes a single byte to a physical space.
2. Doublebyte Size transfer which reads or writes two bytes to a physical space.
3. Quadbyte Size transfer which reads or writes four bytes to a physical space.

All data transfers on the Kbus are 32 bytes in size, regardless of the type of transaction being performed. RIO transactions of all sizes are still performed within the 32-byte transfer process. For RIO data transfers, the data is placed within the block being transferred. Kbus address bits KADDR[2:0] specify where the data being transferred resides within the block. The transaction type bits KTYPE[4:0] indicate the size of the data. Data sizes of byte, doublebyte, quadbyte and hachibyte are supported. Data within each of the four cachelines is the same and data other than that specified by the address and transfer size is undefined. RIO transactions do not affect data held in various caches.

There are 16 physical I/O address spaces which are each specified by space bits KSP[3:0] during an I/O transaction. Each space is 256 Mbytes and is indexed by Kbus address bits KADDR[27:0]. Kbus address bits KADDR[31:28] are always zero during RIO transactions and replaced by the four KSP bits.

### 2.4.3 Other Transactions

Two types of transactions do not fall in the cache block and RIO categories: RAM refresh cycles and byte RMC transfers. RAM refresh transactions are periodically generated signaling all Memory Boards to enter into a refresh state. A byte RMC transfer is a test and set operation supported by some I/O devices. This transaction causes an I/O device to atomically read a byte at an indicated I/O address and return the value which was read to the Kbus master. The I/O device then writes the byte back to its source with bit 7 overwritten with a high value.

## 2.5 Cache Protocol

For more efficient operation, each CPU Board has its own on-board cache RAM. With each board potentially having its own copy of the data held in system memory, it is very important

that each board follow a cache protocol for transferring data between its cache and system memory or other caches. The Kbus cache protocol guarantees that the data being transferred about the system gets updated correctly and that no data gets overwritten without alerting other Kbus masters. In addition, each Kbus Memory Board monitors the transactions of the Kbus and updates its RAM with the data that passes from one CPU to another.

All cache transactions transfer a 32-byte block across the Kbus. Each cache block according to the protocol is classified as being one of four states:

1. Invalid                      A block is invalid if the entry is not in the cache. An invalid block cannot be written or read and can be replaced by a CPU Board's cache controller at will.
2. Unowned                    A block is unowned if it is valid and read only access is permitted. The block can be replaced at will.
3. Exclusively Owned Clean    A block is this state if it is the only valid copy of the block in the system aside from memory. The block can be replaced at will.
4. Exclusively Owned Dirty    A block is this state if it is the only valid copy of the block in the system. The block can be replaced at will.

A number possible events can cause the state of a block of data to change. A CPU can read or write the block. The MMU controller on a CPU Board can flush the block. A cache read, IOB read and invalidate or IOB write and invalidate transaction can occur on the Kbus.

Ownership of a block of data during a Kbus transaction is determined by a Kbus watcher on each CPU Board. If the watcher is the owner of a block, then it asserts its KCOWNL signal. If no CPU is the owner, the memory, by default, is the owner. During each block transfer, if a Kbus watcher has a valid copy of the block, it indicates so by asserting the KSHAREL signal. If a Kbus master indicates that it is the owner of the block by asserting the KCOWNL signal, it also asserts KSHARE.

A Kbus IOB transaction causes the owner of a block to supply data. A cache read causes the owner to supply data and note that there is a cacheable copy in the system. A read and invalidate causes the owner to supply data and then causes all caches in the system to invalidate their copies of that block. A write and invalidate causes memory to be updated and all caches to invalidate that block.

## 2.6 System Interrupts

System interrupts on the Kbus are implemented through a serial interrupt link control signal using a bit-by-bit comparison algorithm. Any device or processor in the system is able to interrupt another processor in the system in two ways:

1. By directed interrupt through which an interrupt can be specifically directed to a specific processor
2. By undirected interrupt through which an interrupt can be issued to every board on the Kbus. The lowest priority processor must then acknowledge and service the interrupt.

An interrupt can have one of 256 vectors which imply an interrupt priority level. Each processor in the system has its one of the 256 software controlled interrupt priority levels. Vector 255 is the highest priority level and Vector 0 is the lowest priority level. Each processor in the system has a unique device identification number (DID) with which to be interrupted by a directed interrupt.

With an undirected interrupt, the highest priority pending interrupt is transmitted on the Kbus and is acknowledged by the lowest operating priority processor. If more than one processor is available to service the undirected interrupt, the processor with the lowest DID services the interrupt. In order to be serviced, an incoming undirected interrupt must have an interrupt level that is greater than or equal to the operating priority level of any one of processors in the system. For an incoming directed interrupt to be serviced, it must have an interrupt level that is greater than or equal to the current processor priority level.

The serial interrupt control signal is referenced to the Kbus clock in order to broadcast and prioritize interrupt information in the system. Each CPU Board in the system has an interrupt handling area which monitors and decodes the interrupt signals. Each transaction on the serial interrupt signal consists of a frame containing 22 bits of information. A transaction can begin once the interrupt handler PAL observing link activity indicates that no transaction is in progress on the Kbus.

Since the System Board is the main interface between the Kbus and all I/O devices, it is capable of issuing both directed and undirected interrupts. The interrupt controller on the System Board takes care of 16 interruptible devices connected directly to it or through the VMEbus. Interrupt vectors 128 through 143 are allocated to the 16 I/O devices.



## Section 3: Series4 Central Processing Unit (CPU) Board Operation

### 3.1 Introduction

The CPU Board for the Series4 uses the Fujitsu SPARC and Weitek 1164/1165 chip sets to create a single-CPU high performance RISC processor. The board uses a dual mode virtual/physical cache. This allows the RISC processor to execute one instruction every clock cycle and allows multiple CPU Boards to maintain cache consistency.

The Series4/600 can contain up to four CPU Boards. The Series4/500 can contain up to two CPU Boards. Each CPU runs at 16.67 MHz, the maximum speed for the Fujitsu SPARC CPU. The estimated performance of each CPU is approximately 10 Sun MIPS. Floating point performance is about 0.8 million floating point operations per second (MFLOPS). In a multiprocessor configuration, four CPU Boards can provide about 33 Sun MIPS.

The main hardware features of the CPU Board are:

- 16.67 MHz Fujitsu SPARC CPU
- 64 Kbyte direct mapped virtual/physical cache
- Hardware assisted memory management unit (MMU)
- 64-bit data bus with error check and correction (ECC)
- Weitek 1164/1165 floating point chip set
- Transparent multiprocessor cache consistency

The Series4 design has been optimized to support multiple general purpose CPUs for a UNIX operating system environment. Each CPU Board implements the Kbus cache protocol to ensure data consistency between each CPU Board's cache in the system. The protocol allows each CPU to access data in any other CPU's cache and only allows one CPU to modify a piece of data at a given time. This protocol is transparent from the view point of multiple software programs interacting and modifying common data.

Data is referenced and exchanged between CPUs by accessing the physical address of a piece of data in memory. The SPARC CPU references data using a virtual address which is then mapped into a physical address by the MMU. Direct high speed access to the cache by the CPU is provided by setting up the cache as a virtual cache and keeping a set of physical tags which mirror those in the bus watcher.

The CPU Board is made up of three major functional sections: Kbus interface, CPU, and an interrupt handler. The following figure shows further the subfunctions within the Kbus interface and CPU functional blocks (see Figure 3-1).

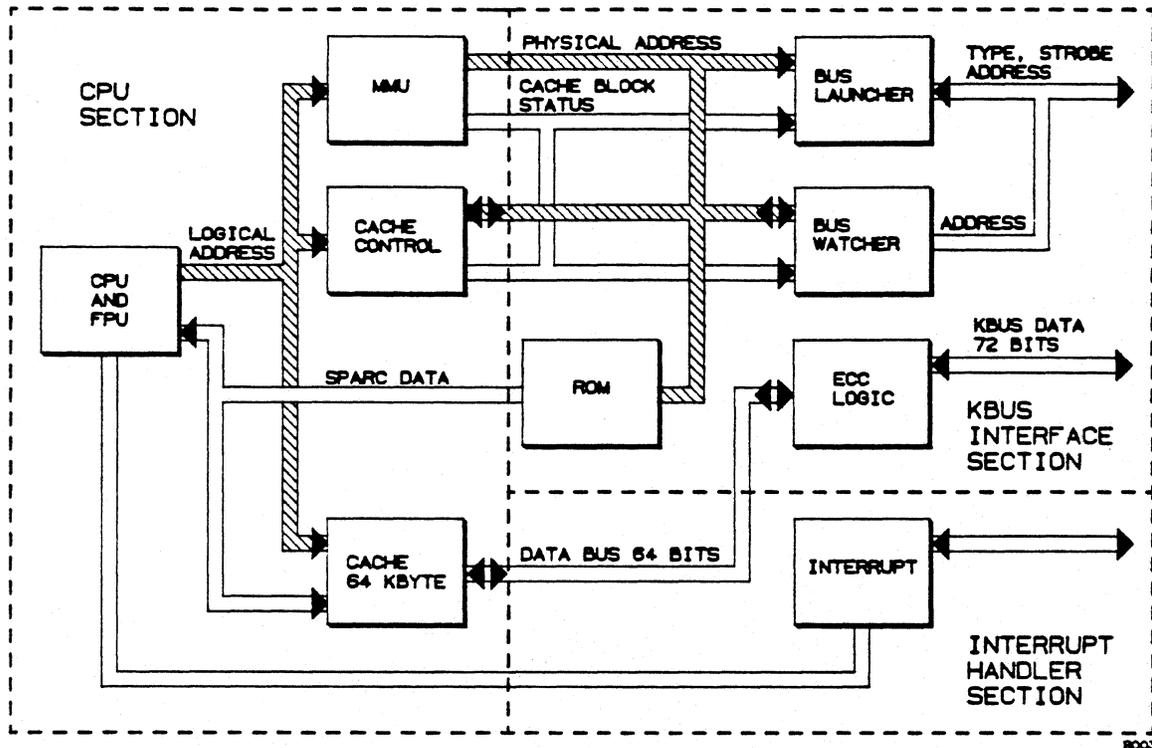


Figure 3-1. CPU Board Block Diagram

The address and data buses shown in Figure 3-1 are symbolic of many buses that interconnect the various functional blocks of the CPU Board. Prior to describing the functional blocks in more detail, the next section describes the buses that make up the CPU.

### 3.2 CPU Board Address, Data, and Control Buses

The data, address, and control buses of the CPU Board are many. They can, however, be grouped into four distinct subsets:

1. Buses used in the CPU section
2. Buses that are common to all CPU Board sections
3. Buses used in the Kbus interface section
4. Buses from the Kbus

The buses used in the CPU section are highly involved in the memory management process. These CPU section buses are shown in Figure 3-2.

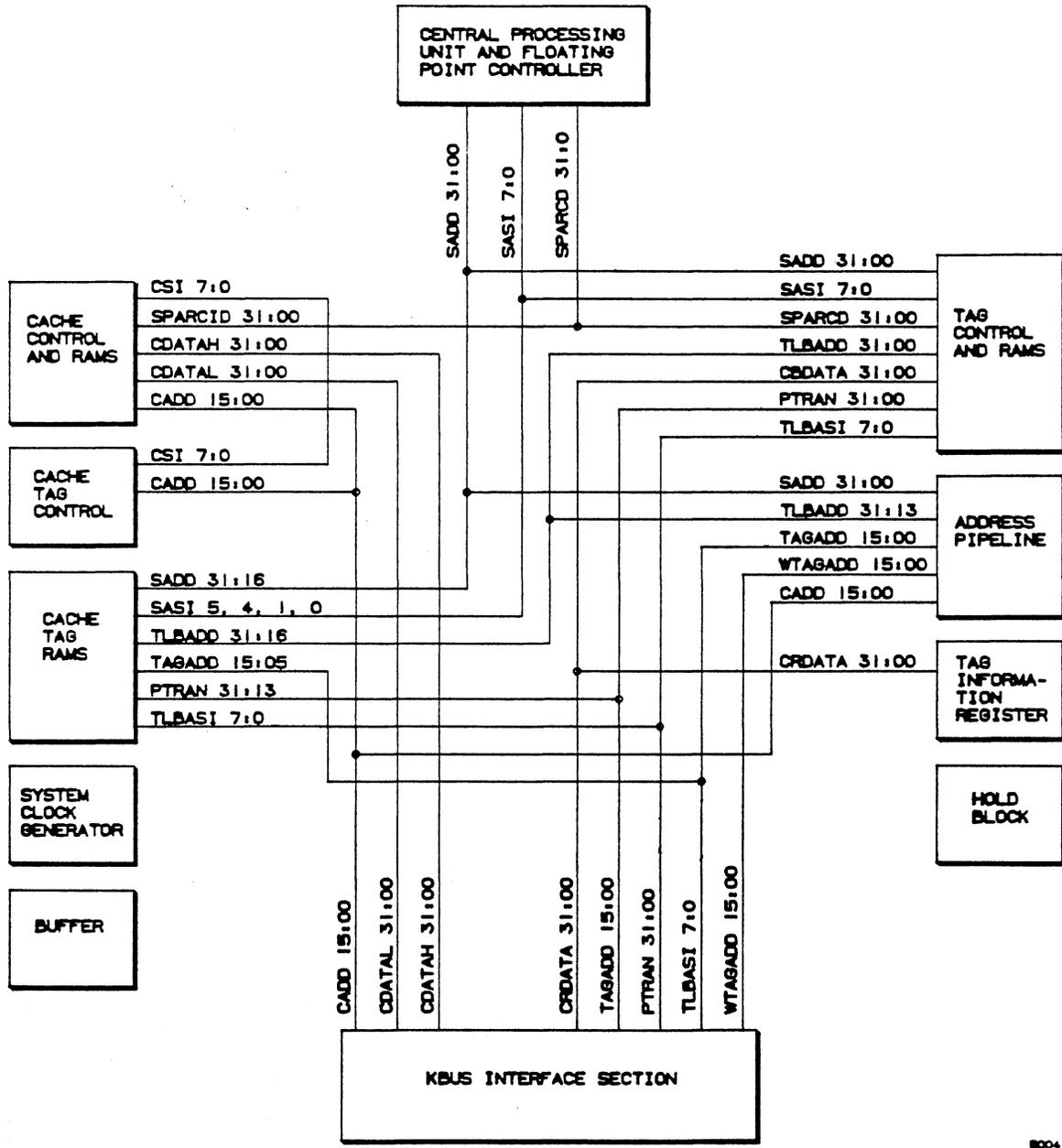


Figure 3-2. CPU Section Buses

All address, data, and control buses that originate at the SPARC CPU are converted within the CPU block and do not exit this block. The SPARC CPU issues a 32-bit SADD address bus, a 32-bit SPARCD data bus, and the eight-bit SASI control bus. The logical address placed on the SADD address bus gets divided among the TLBADD address bus, the TAGADD address bus,

and the CADD address bus in the address pipeline block. The upper two bytes of the SADD address are compared with the TAGADD address in the cache tag block to determine whether the virtual tags of the logical address match. Only the lower 13 bits of the SADD bus are sent to other CPU Board sections via the PTRAN address bus.

The SASI bus is used directly in the TLB block and the cache tag block of the CPU section. In the TLB block, the SASI bus lines are used as inputs to an octal D-type flip-flop which drives the TLBASI bus. The TLBASI bus then issues the ASI value to the rest of the CPU subsections. In the cache tag block, SASI bits 0, 1, 4, and 5 are used in the virtual hit determination PAL when evaluating whether a virtual hit has been made.

The SPARCD data bus terminates in the TLB block and the cache block. In the TLB block, the SPARCD data bus is connected to the CRDAT data bus at four octal transceivers which can transfer information between the two buses. Also, the upper three bytes of the SPARCD data bus interconnect to the PTRAN address bus at three octal transceivers. In the cache block, the 32 bits of the SPARCD data bus are used to assemble and disassemble the 64-bit CDAT data bus (represented by two 32-bit buses, CDATH and CDATL). These buses interconnect in the cache RAM block at eight octal transceivers.

Two other buses shown in Figure 3-2 are used only within the CPU section, the eight-bit CS1 control bus and the 32-bit TLBADD address bus. The CS1 control bus is issued by the chip select control PAL in the cache control block. These eight signals are used for selecting each of the eight cache RAM chips. The TLBADD address bus originates from the SADD bus in the address pipe and provides the TLB index address to the cachetag block and the TLB block.

The remaining buses used in the CPU section are common to the the Kbus interface section too. These buses include:

- 16-bit CADD cache address bus
- 16-bit TAGADD tag address bus
- 16-bit WTAGADD watcher tag address bus
- 32-bit PTRAN physical translated address bus
- Eight-bit TLBASI address space identifier bus
- Lower and upper halves of the 64-bit CDAT cache data bus, CDATH and CDATL
- 32-bit CRDAT control data bus

The CADD cache address bus carries the index showing the location of the data held in the CPU's cache to various blocks in the CPU section. The CADD bus can originate in the Kbus address receiver block of the Kbus interface section or the address pipe of the CPU section. In the address pipe block, the CADD bus is taken from or becomes the lower two bytes of the SADD address bus. In the cache control block, three bits of the CADD bus are used in selecting cache RAM chips. In the cache block, the CADD lines actually address the cache RAM locations.

The TAGADD tag address bus carries the index showing the location of the virtual and physical tags that reside in the cache tag RAMs. The TAGADD address consists of bits 15 through 5 and it originates at the address pipe block. The TAGADD either comes from SADD 15:05 or WTAGADD 15:05, depending on which block is writing to or reading from the tag address RAMs. In the cache tag block, the TAGADD lines address the virtual and physical RAM locations. The upper three bits of the TAGADD address bus are used in the Kbus interface section of the board to generate bits 15:13 of the CADD and WTAGADD address buses.

**Solbourne Confidential Information – Do Not Distribute**

The WTAGADD is used in the address pipe to generate the tag address (TAGADD) pipe when the watcher section is reading or writing the cache tags. The WTAGADD bus originates in the Kbus address receiver block of the Kbus interface bus launcher subsection.

The WTAGADD lines also address an EEROM in the ECC generator subsection identification block when the system is reading from or writing identification information into the EEROM (see Figure 3-3).

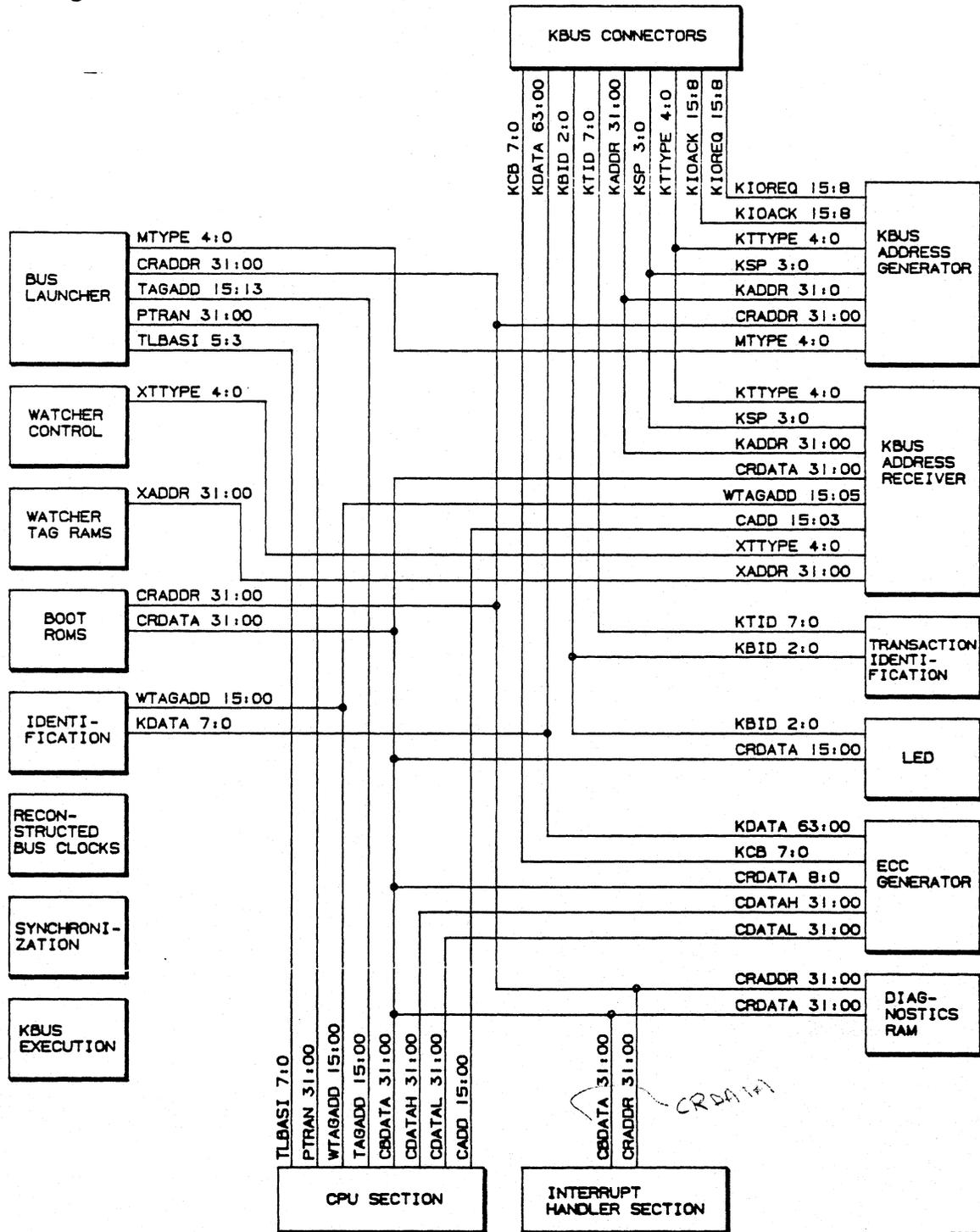


Figure 3-3. Kbus Interface Section Buses

The physical translated address bus, PTRAN, contains the actual physical address of the data being transferred to or from system memory. The PTRAN address originates in the TLB block. Bits 13 to 31 of the PTRAN address come from the TLB RAMs and bits 0 to 12 are taken directly from the SADD address bus. The TLB RAM portion of the PTRAN bus (Bits 31:31) gets compared in the cache tag block with the physical tag address held in the cache tag RAMs. The cache tag block comparators asserts match signals if the two values match. In the Kbus interface section of the CPU Board, the PTRAN address bus enters the bus launcher block of the Kbus launcher subsection. In this block, the PTRAN lines become inputs to octal D-type flip-flops which drive the CRADDR, control address, bus. The CRADDR address bus then carries the physical address to the Kbus.

31:31

The TLBASI address space identifier bus originates in the TLB block and is the latched version of the SASI identifier byte issued by the SPARC CPU. The TLBASI bus is used in two other places on the board, the cache tag block and the bus launcher. In the cache tag block, the TLBASI bits are used to determine whether to assert two virtual hit enable lines. In the bus launcher block of the Kbus interface section, the bits 3, 4, and 5 of the TLBASI are used as status inputs for two control PALs.

The two halves of the cache data bus, CDATH and CDATL, each carry 32 bits of the 64-bit data lines. These two parallel buses carry the data between the RAMs in the CPU section cache block and the latches and PALs of the ECC generator block of the Kbus interface section.

The CRDAT control data bus transfers control and address information between the SPARC CPU and many blocks on all sections of the CPU Board. The SPARC CPU can read from and write to the CRDAT bus through octal transceivers in the MMU section's TLB block. The CRDAT bus can also be loaded with board operation information held in the test information register latches. In the Kbus interface section of the board, the CRDAT bus is used by several blocks:

- Addresses received from the Kbus address bus are latched in the Kbus address receiver block of the bus launcher subsection and loaded onto the CRDAT bus for transfer to the CPU
- When the CPU is reading the boot ROMs, the data is placed on the CRDAT bus
- Information to be displayed on the CPU LEDs gets transferred to the LED block on the lower two bytes of the CRDAT bus
- Information held in the diagnostic RAM block of the ECC generation subsection is issued on the lowest byte of the CRDAT bus
- When the CPU is loading the Kbus with a data check byte, check byte data enters the ECC generator block through the lower nine bits of the CRDAT bus

CRDATA in fig 3-3

Finally, the lower two bytes of the CRDAT bus are used in the interrupt handler section of the CPU to feed serial interrupt information out to the Kbus.

A 32-bit CRADD control address bus accompanies CRDAT bus in some Kbus interface section blocks and in the interrupt handler section. Similar to the CRDAT bus, the CRADD bus is used in the following instances:

- Whenever the CPU addresses the boot ROMs, the Kbus address generator, or the diagnostics RAM block of the Kbus interface section, it places the address on the PTRAN address bus which is then transferred to the CRADDR address bus by the bus launcher control block
- When the CPU reads the boot ROMs, the CRADDR address is used to address the ROMs
- Addresses destined for the Kbus address bus enter the Kbus address generator block via the Kbus interface section bus launcher subsection

- Information held in the diagnostic RAM block of the ECC generation subsection is addressed via the Kbus interface section bus launcher subsection
- The lower eight bits of the CRADDR bus are used by two control PALs in the Kbus interface section to assert control signals within the interrupt block

The remaining buses on the CPU Board are used only within the Kbus interface section of the board. Some transfer information between Kbus interface section blocks namely: an internal address bus XADDR which addresses the watcher tag RAMs, and two internal transaction type buses XTTYPE and MTTYPE. All of the other buses are standard Kbus signals. These buses are not discussed here: data bus KDATA [63:00], check byte bus KCB [7:0], address bus KADDR [31:00], space identifier bits KSP [3:0], transaction type bus KTTYPE [4:0], board identification bus KBID [2:0], transaction identifier bus KTID 7:0 and the two request/acknowledge buses, KIOREQ 15:08 and KIOACK 15:08.

### 3.3 CPU Section

The CPU section of the CPU Board consists of the following subsections:

- CPU
- MMU
- Cache tags and cache

The CPU section is composed of a SPARC based CPU from Fujitsu. The Fujitsu MB86900 CPU provides approximately 10 SPARC MIPS, which is equivalent to 6.5 VAX MIPS. In addition to the integer processor, the CPU block contains a Fujitsu MB86910 floating point controller and two floating point processors from Weitek (1164/1165). These processors provide 0.8 MFLOPS of double precision linpack.

The MMU can map a total of 16 Mbytes which is divided equally between data and instruction space. If an access is not present in the MMU, the operating system executes a software table walk. To increase the efficiency of this operation, the MMU contains an auto control register increment and a page directory base address register.

The cache tags block supports two sets of cache tags for each cache entry; a virtual and a physical cache tag. The virtual tags are used by the CPU for zero wait state access, while the physical tags are used to relate the virtual cache address to a physical memory location. In addition to the tag addresses, the cache control detects write protection and user protect faults.

The Series4 cache is a 64 Kbytes logical cache. The memory is divided into two banks of 32 Kbytes, an even and an odd word bank. By making the cache 64 bytes wide and two banks, the cache can accept 64 bits of data from Kbus every 50 ns and provide 32 bits of data to the processor every 60 ns.

### 3.3.1 CPU Subsection

The CPU subsection of the CPU Board contains three major functional areas: CPU block, clock generation, and an address pipeline circuitry. Figure 3-2 shows the blocks that make up the CPU subsection of the board.

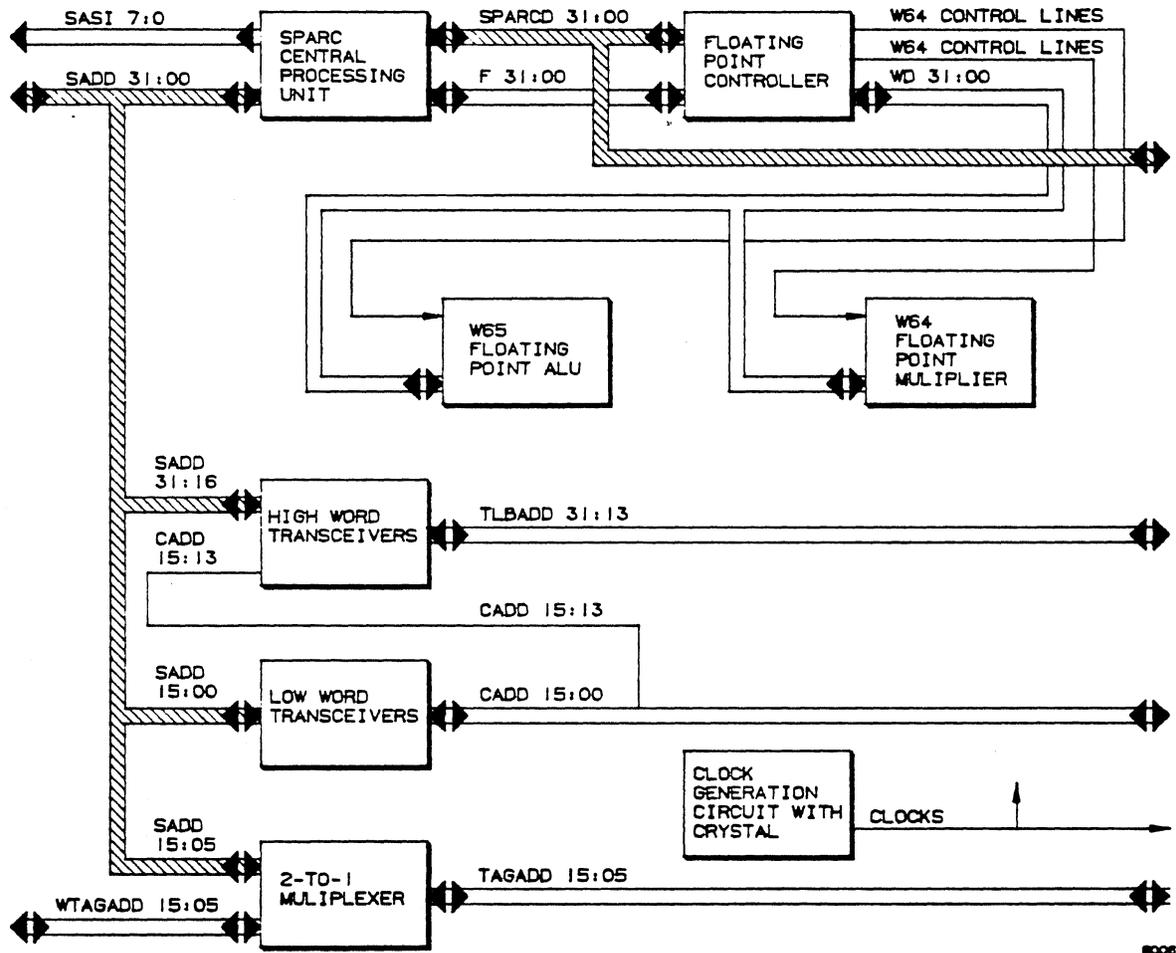


Figure 3-4. CPU Subsection Block Diagram

The CPU subsection block contains:

- Clock generation circuit
- Fujitsu SPARC MB86900 central processing unit
- Fujitsu MB86910 floating point controller
- Weitek 1164 and 1165 floating point processors
- Address pipeline circuitry with high word and low word latches and a multiplexer

The clock generation circuit produces all the clocking signals used to synchronize activity on the CPU Board. This circuit is comprised of a 66 MHz crystal, NAND gates, and octal latches. The

circuit outputs clocking signals which are phased at 30 degree intervals.

The CPU has a 32-bit address bus, SADD [31:00], a 32-bit data bus, SPARCD [31:00] and a 32-bit floating point data bus, F [31:00]. In addition, the SPARC CPU expands the system capabilities with an eight-bit address space identifier bus, SASI [7:0]. The floating point data bus interconnects solely the CPU and the floating point controller. The other buses interface with many other functional circuits of the CPU Board

The floating point controller interfaces with the two floating point processors with a 32-bit data bus, WD [31:00], and two sets of control lines for the 1164/1165 chip set. The data bus delivers data between the floating point controller and the two processing chips. The two sets of control lines provide command signals to the floating point processors such as: exception status, load/unload direction and enable commands, functional control, etc.

The address pipeline circuitry uses octal bus transceivers to channel bits between the 32-bit SADD bus and the cache address CADD bus. Upon commands from the MMU hold block, data transfers between the lower 16-bits of the SADD bus and the cache address bus, CADD [15:00]. The address pipeline circuit also transfers, on command from the MMU hold block circuit, the upper two bytes of the SADD address bus to the upper two bytes of the translation look aside address bus TLBADD [31:15]. The upper three bits of the CADD bus are added to the TLBADD bus at bits 13, 14, and 15.

### 3.3.2 MMU Subsection

The memory management unit subsection of the CPU performs the function of translating the logical addresses issued by the CPU into physical addresses of RAM. The MMU is subdivided into the following sections as shown in Figure 3-5:

- Hold block
- Test Information Register (TIR)
- TLB block which contains MMU control and TLB RAMs

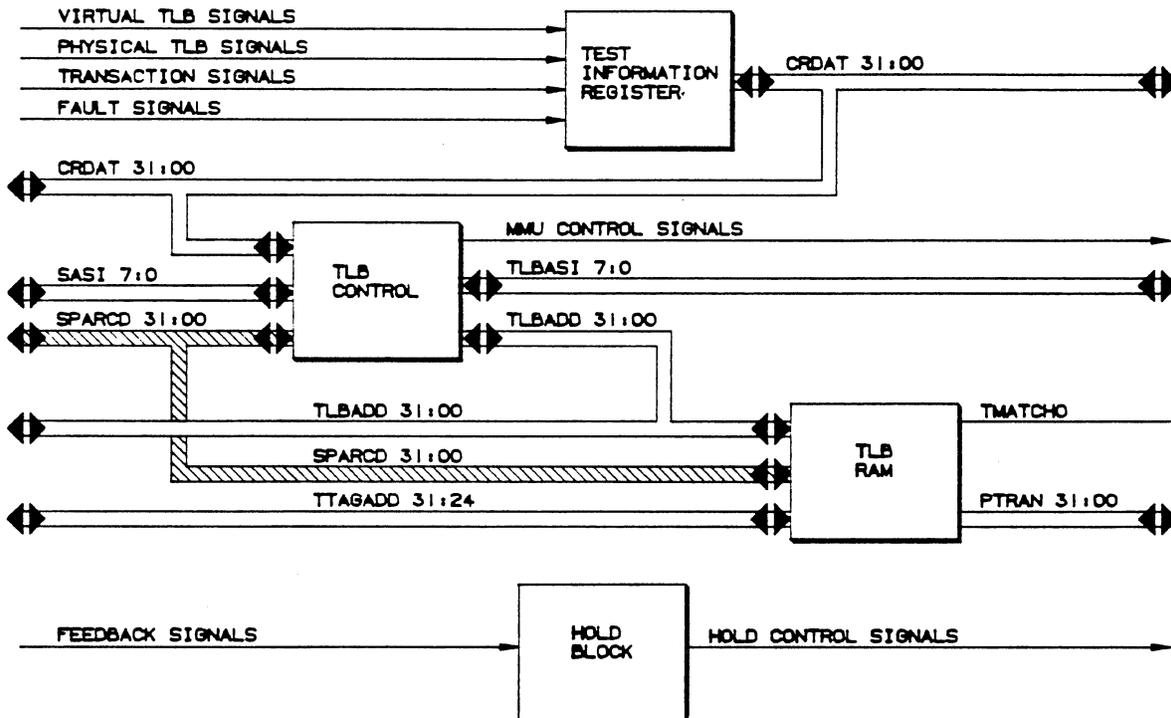


Figure 3-5. MMU Subsection Block Diagram

The hold block contains two PALs which coordinate the timing of signal transfers through the MMU.

The TIR holds the state of signals from many blocks on the CPU Board. On command from a diagnostics program, the information held in the TIR is loaded onto the CRDAT data bus for analysis.

The TLB block is made up of static 2 Kbyte x 8 RAM chips which hold the translated address tables used by the CPU for converting logical addresses to physical addresses. The TLB block receives the virtual address from the CPU block over the TLBADD address bus. It compares the address it receives with the address held in the translation table and when the two addresses match, the TLB block asserts the TMATCH line. Translated addresses are placed on the PTRAN bus for the Kbus subsection bus launcher section.

Control for the MMU process is achieved through several control PALs located in the TLB section of the block. The TLB block contains transceivers for the transfer of data between the

SPARCD data bus and the CRDAT data bus. This block also transfers the information of the SASI bus to the TLBASI bus which is issued to the Kbus subsection bus launcher section.

### 3.3.3 Cache Tag Subsection

The cache tag subsection of the CPU Board interfaces with the other sections of the CPU Board in its control of cache consistency. This subsection monitors the activities of the memory cache consistency processes and performs cache transactions based upon the status of these activities. The cache tag subsection is comprised of tag control and tag RAMs. The tag control section is further divided into the hit detection block and the tag control block. The tag RAMs section consists of virtual tag RAMs, physical tag RAMs, and valid tag RAMs. (See Figure 3-6.)

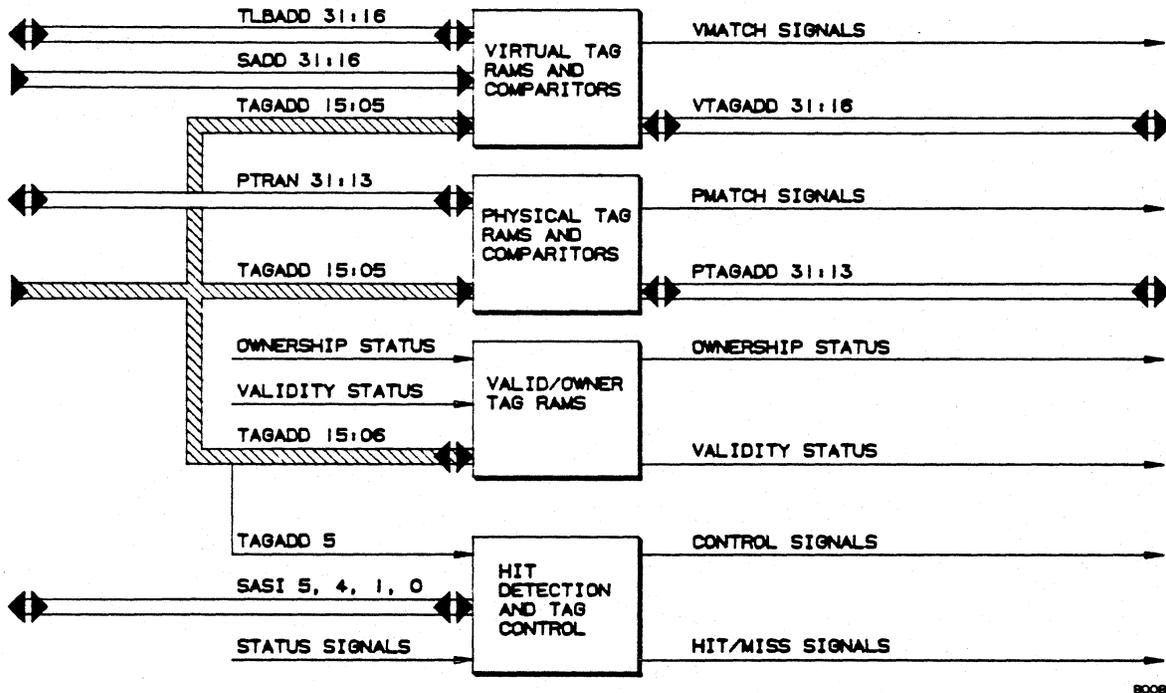


Figure 3-6. Cache Tag Subsection Block Diagram

The hit detection block receives cache consistency status signals from many sources on the CPU Board. It then determines issues virtual and physical hit and miss signals to the rest of the functional areas involved in the cache process. The hit detection block contains virtual hit and physical hit determination PALs which issue virtual and physical hit signals. The tag control block contains two control PALs, both of which control the storage of data to and release of data from the tag RAMs.

TAGADD address bus bits 15:5 can address RAM locations within the virtual tag RAMs, physical tag RAMs, and valid tag RAMs. The virtual tag RAMs store the upper two bytes of the TLBADD bus after they've passed through transceivers to the VTAGADD bus. The physical tag RAMs store the upper two bytes of the PTRAN address after they've passed through transceivers to the PTAGADD bus. The virtual tag RAMs store the physical and virtual valid and own status bits for transfer to the hit detection circuit.

The virtual and physical tag RAM blocks also issue virtual and physical match signals when the addresses held in the tag RAMs match the logical address. The physical tag RAM block asserts the PMATCH line when the address held in the physical tag RAMs matches the address on bits 31:13 of the PTRAN bus. The virtual tag RAM block asserts the VMATCH line when the address held in the virtual tag RAMs matches the address on bits 31:16 of the TLBADD bus.

### 3.3.4 Cache Subsection

The cache subsection is divided into a cache control block and a cache RAM block. The cache control block supplies all control lines for loading, storing, and releasing data held in the cache RAM block (see Figure 3-7).

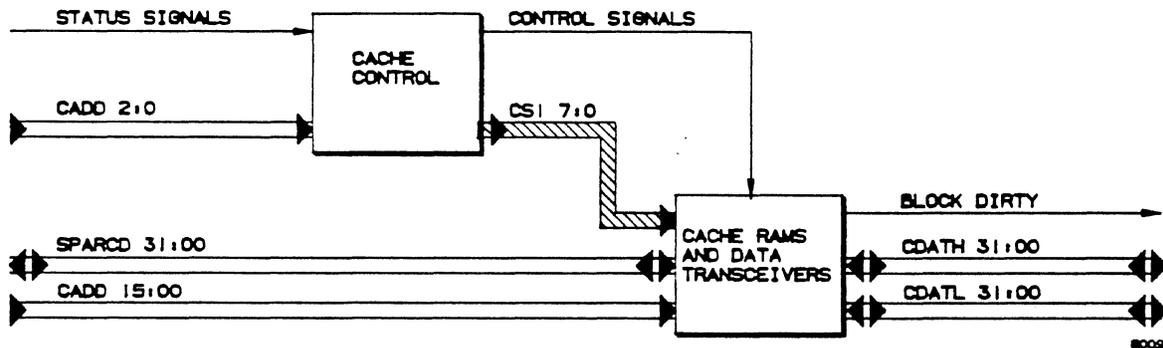


Figure 3-7. Cache Subsection Block Diagram

The cache control block contains two control PALs and other logic gates. One PAL is the chip select PAL which decodes the lower three bits of the CADD address bus. The chip select PAL sends the eight chip select signals, CS1 [7:0], to the cache RAM block. This PAL decodes the CADD bits and asserts the appropriate chip select signal to enable one of the cache RAM chips. The other control PAL along with other logic chips in the cache control block use memory management status signals to set other appropriate cache control signals.

The cache RAM block consists of a low bank and a high bank of 8 Kbyte by 8 cache RAM chips. These banks are supported by two sets of octal bus transceivers which transfer data between the SPARCD data bus and the high and low cache data buses, CDATH and CDATL. Two transfers of 32 bits on the SPARCD data bus make up the high and low 32-bit transfers of the CDAT buses. CDATL is loaded first, followed by CDATH. The third bit of the cache address, CADD 2, determines when the high and low buses receive data. When CADD 2 is low, the low address bus gets loaded. When CADD 2 is high, the high address bus gets loaded.

The cache RAM block also issues the BDIRTY signal which asserts during writes to cache. This signal is used by the Kbus interface circuit in its function of maintaining cache consistency on the Kbus.

## 3.4 Kbus Interface Section

The Kbus interface section of the CPU Board is involved in the coordination of data transfers between the CPU and the Kbus. This section of the board contains the following subsections:

- ROM

- Bus launcher
- Bus watcher
- ECC logic

Providing the 256 Kbyte Boot ROM on the CPU card enables greater flexibility for on board diagnostic testing. The goal for the diagnostics is to provide information which detects hardware failure to a specific block of circuitry.

The main function of the launcher section is to dispatch the necessary Kbus transactions or internal cycles to control registers or boot ROM to service a cache miss. If a Kbus transaction is initiated, it performs arbitration in accordance with the Kbus protocol and launches the transaction. The launcher determines if a cache block is to be flushed or if an aliased cache block is being referenced in the cache. If an aliased block exists in the cache it reruns the transaction to complete the transfer.

The bus watcher section works in coordination with the launcher section for implementing the cache protocol. It contains cache tags that allow the CPU's cache to be accessed as a single set direct mapped 64 Kbyte physical cache. As part of this, it contains a reverse translation map to reconstruct the virtual index into the cache so that data can be extracted by another CPU.

With this information, the watcher can extract data from the CPU cache at the request of another CPU, or it can supply data to the cache in response to a cache miss. The watcher is responsible for maintaining concurrency of its own and the CPU's physical address and status tags. The watcher is also responsible for initiating register-to-I/O transaction on Kbus.

The ECC logic section connects the 64-bit cache data bus to the 64-bit Kbus data bus. When data is moved from the cache to memory and another CPU, the data is extracted under control of the watcher section and passed to Kbus. As the data appears on the Kbus, the eight-bit ECC tag field, which has been generated by the ECC logic section, also appears on Kbus for each cache line. When data is being received from Kbus, the ECC logic section takes the eight-bit ECC and 64-bit data fields from the Kbus and performs on-the-fly correction of any single bit error before storing the data into the cache. An error signal is generated if an uncorrectable, multiple-bit error occurs.

### 3.4.1 ROM Subsection

The ROM subsection of the CPU Board contains four 512 Kbyte by 8 PROMs which are located on the CRADDR address bus and issue data on the CRDAT data bus (see Figure 3-8).



Figure 3-8. ROM Subsection Block Diagram

The boot ROMs contain all the software code needed to get the system up and running upon power up. The code executed first is diagnostic code. This code runs a series of tests on all the pertinent hardware functions of the system prior to loading the UNIX kernel from the hard disk. Each diagnostic test is represented by a two-digit display on the CPU LEDs. If one test fails to complete and the CPU stops executing code, the LEDs indicate the test number that failed and, if known, the condition under which it failed. See the *System Power-on Self Test Manual* (101486)

for additional information.

### 3.4.2 Bus Launcher Subsection

The bus launcher subsection is responsible for launching Kbus transactions on the Kbus. This subsection is comprised of several blocks as shown in Figure 3-9:

- Launcher control
- Kbus address generator
- Kbus address receiver
- Watcher tag RAMs

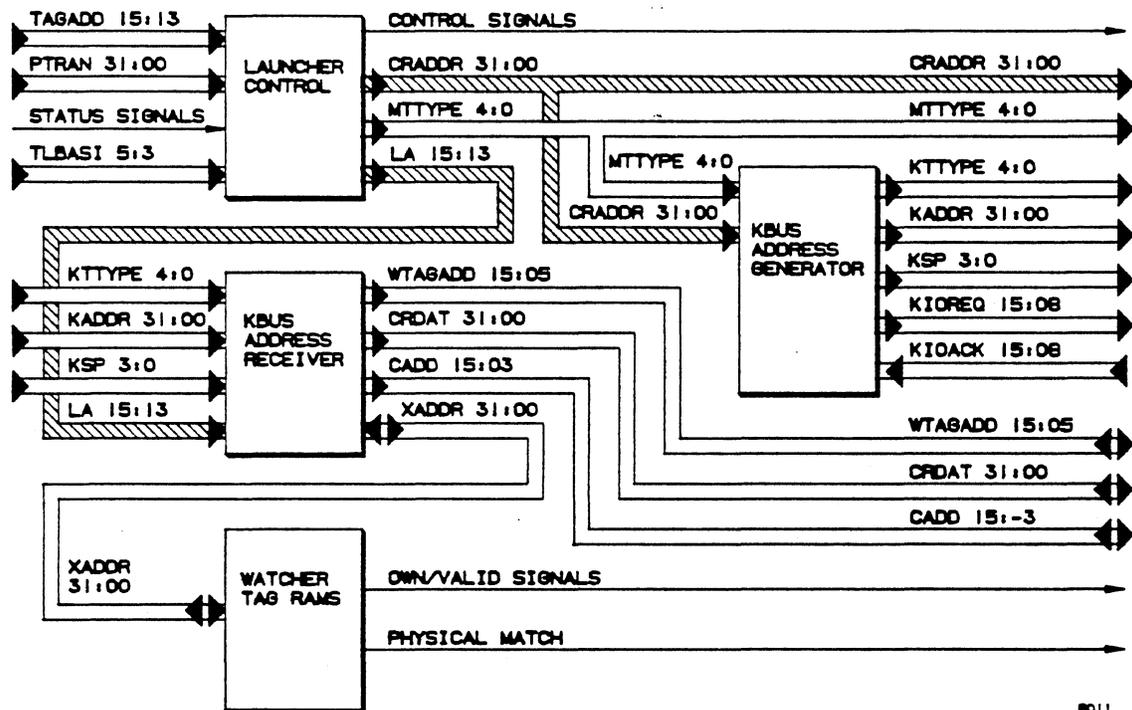


Figure 3-9. Bus Launcher Subsection Block Diagram

The launcher control block generates the control signals required for coordinating and synchronizing transactions between the Kbus and the CPU. The launcher control block contains six PALs which control usage of the CRADDR bus. The CRADDR bus is used throughout the Kbus interface subsection. Bits 5:3 of the TLBASI bus are used by the launcher block in its coordination of transfers. The launcher block transfers the translated physical address from the PTRAN bus onto the CRADDR bus. The translated address can then be latched by the Kbus address generator block. The launcher block also generates the signals used for identifying the type of transaction being sent to the Kbus. Transaction type information is sent out on the MTTYPE bus. Finally, the launcher block latches the three upper-most bits of the CPU blocks TAGADD and provides them to the Kbus address receiver block during address transfers from the Kbus to the CPU. These three bits are latched address bits LA [15:13]. In the Kbus address receiver block, these three bits are used for tag and cache address modification.

The Kbus address generator block issues to the Kbus all the necessary signals for successfully transferring data over the bus. Information provided includes: a Kbus address, transaction type, I/O space, and the Kbus request/acknowledge signals. The Kbus address generator block contains octal D-type flip-flops for loading the KADDR address bus with the address from the CRADDR bus. The same type of octal flip-flops are used for transferring MTYPE bus information to the KTTYPE bus and for loading the KSP I/O space bus. Before the Kbus address generation block can write to the Kbus, it must send a request to and receive an acknowledge from the Kbus. The address generator I/O request PAL encodes the upper four bits of the CRADDR bus to generate the appropriate KIOREQ bus signals. Once it has requested access to the Kbus, the Kbus address generator block monitors the KIOACK bus lines to know when it has control of the Kbus. When it receives the acknowledge, this block then makes the transfer.

The Kbus address receiver block performs the reciprocal role of the address generator block; it receives Kbus address, transaction type, and I/O space information from the Kbus. Several octal, edge-triggered flip-flops in the block latch the KADDR address and the KTTYPE bus lines. The KTTYPE bus bits become XTTYPE bus bits and are used by the watcher control circuit for controlling activity in the bus watcher block. The KADDR address bus feeds the XADDR bus which is used in the block for transfer to the CRDAT bus and part of the WTAGADD and CADD address buses. All bits of the XADDR bus enter the CRDAT bus directly except Bits 3 and 4. These two bits are controlled by a PAL which also controls the fault physical address register (FPAR). The 32 bits of the XADDR bus also address the watcher tag RAMs. In addition to receiving and distributing the Kbus address, the Kbus address receiver block contains an eight-bit comparator which compares the four bits of the KSP bus and KADDR [27:24] with a hardwired byte. If the two bytes match, the identification space select signal, IDSPSEL, asserts. If the two do not match, IDSPSEL does not assert. The Kbus address receiver block provides the CPU subsection TAGADD Bits 12:15 to the WTAGGADD and CADD address buses in place of XADDR bits. These bits are used in the cache tags when updating the address index in the cache tag RAMs.

The bus watcher tag RAM block holds the valid address index for the current valid address on the Kbus. This block is comprised of two identical circuits, each containing two sets of 4 Kbyte by 4 RAMs and an eight-bit comparator. One set is used for the third byte of the address, XADDR [23:16]. The other set is used for the fourth byte, XADDR [31:24]. The RAMs are addressed by XADDR bus bits 15:5. These RAMs hold the cache tag values for the watcher subsection. The values stored in these RAMs are compared with the tags of the current transaction to see if the tags need updating. If the tags do not match, the comparators assert physical modify signals PM0 and PM1. In addition, there is a 1 Kbyte by 4 RAM chip which is addressed by XADDR bits 15:6. This RAMs holds the status of physical ownership and valid bits for the cache tags.

### 3.4.3 Bus Watcher Subsection

The bus watcher section monitors activity on the Kbus. This subsection consists of the blocks listed below and shown in Figure 3-10:

- Watcher control block which includes Kbus execution circuitry, synchronization chips, and reconstructed bus clock gates
- Light emitting diode/ID block

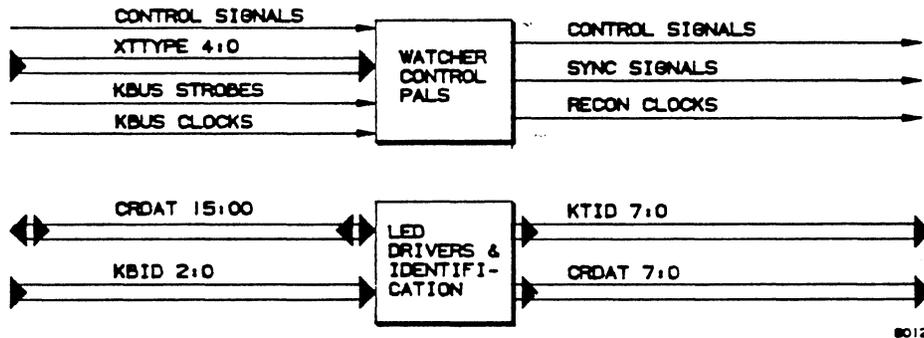


Figure 3-10. Bus Watcher Subsection Block Diagram

The watcher control block contains ten control PALs which monitor the Kbus and control when the CPU runs transactions on the Kbus. Seven of the PALs control the watcher subsection functions while the other three are Kbus execution PALs which deal mostly with Kbus transfer signal protocols. The seven watcher PALs are:

1. Transaction type encoder PAL
2. Ownership and valid selector PAL
3. Ownership and valid data generator PAL
4. Watcher generator PAL
5. Read counter and I/O write request PAL
6. Request generator PAL
7. Write generator PAL

In addition, the watcher control block performs the duty of generating six bus clocks for the CPU Board that are taken directly from the Kbus clock, BC. This block also synchronizes signals between the watcher subsection and other circuits of the CPU Board.

The light emitting diode block takes the lower two bytes of the CRDAT bus and places status codes on the two-digit CPU LEDs. This block also contains two eight-bit line drivers. One places the board identification bits BID 2:0 on the CRDAT bus. The other places the board identification bits BID 2:0 on the KTID bus.

### 3.4.4 Error Check Circuit Logic Subsection

The ECC logic subsection performs data transfers between the CPU Board and the Kbus (see Figure 3-11). It consists of

- ECC generator
- Diagnostic RAM

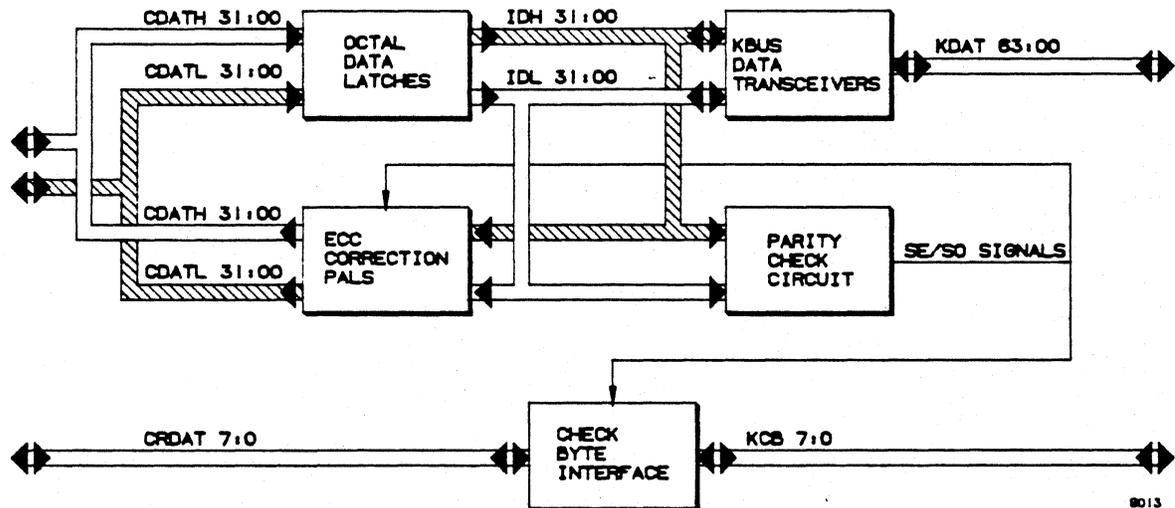


Figure 3-11. ECC Logic Block Diagram

The ECC generator block is involved in the transfer of data between the Kbus data bus, KDAT [63:00], and the high/low CPU data buses, CDATH [31:00] and CDATL [31:00]. The ECC generator block performs error checking on the data that passes through it and corrects simple single-bit errors as needed. When the CPU Board is putting data on the Kbus, the ECC generator block first uses a series of octal D-type flip-flops to latch the data from the CDATH and CDATL buses. This data passes via the internal data buses, IDL and IDH, to both the Kbus data transceivers and the parity checking circuit.

When the CPU is taking data from the Kbus, the Kbus data transceivers pass the data via the IDL and IDH buses to the parity checking circuit and to the ECC correction PALs. The parity check circuit issues even and odd status signals, SE and SO, to each ECC correction PAL. The state of these status signals is used in the PAL programs to determine the state of the data bit driven onto the CDATH and CDATL bus.

The diagnostic RAM block has a 2 Kbyte by 8 RAM chip which is addressed by CRADDR bus bits 13:3. This RAM chip reads and writes diagnostic data on the lowest byte of the CRDAT bus.

### 3.5 Interrupt Handler Section

The interrupt handler section contains the control and data registers to implement the Kbus serial interrupt link protocol. This protocol has a limited message passing capability. It features:

- Prioritized interrupt packets

- Interrupts that can be directed to a specific CPU
- Interrupts that can be directed to the lowest priority CPU

Processor interrupt level is adjusted by loading a register in the interrupt block with the appropriate value. Interrupts below this value do not pass to the CPU. The interrupt logic only signals the CPU if a interrupt of high enough priority has been received. The circuitry also conducts arbitration for receiving undirected interrupts without CPU intervention.

The CPU card contains an interrupt circuit that can receive and transmit interrupts. In addition to normal interrupt processing, the interrupt circuit can be used as a communication port to the CPU. This port is used to monitor the on-board diagnostics testing during manufacturing burn in.



## Section 4: Series5 CPU Board Theory of Operation

### 4.1 Introduction

This section gives the theory of operation for the Solbourne Series5 CPU Board, which is based on a Cypress CY7C601 SPARC integer unit and Weitek 3171 floating point unit. The Solbourne Series5 CPU Board operates at 33 MHz.

The Cypress CY7C601 and Weitek 3171 chip sets are used to create a single-CPU high performance RISC processor.

The Series5 CPU Board can be used to upgrade a current Series4/600 or Series4/500 workstation. The upgrade consists of removal and replacement of the Series4 CPU Boards with the Series5 CPU Boards and the installation of OS/MP 4.0C. No other hardware or software changes to the system are required.

Series5 and Series4 CPU Boards cannot be mixed in a single machine. The Series5/600 can contain up to five CPU Boards while the Series5/500 can contain up to three CPU Boards.

The integer performance of a single CPU is roughly 22 Sun MIPS and floating point performance is about 3.4 MFLOPS. In a multiprocessor configuration, four CPU Boards can provide about 65 Sun MIPS.

### 4.2 Series5 CPU Board Features

The main hardware features of the CPU Board are:

- Cypress CY7C601 SPARC CPU and Weitek 3171 floating point unit operating at 33 MHz
- Physical 128 Kbyte cache
- Hardware assisted memory management
- Improved I/O performance
- 4 Gbyte (32 bit) virtual address space
- 256 Mbyte (28 bit) physical address space
- 64-bit data bus with error check and correction (ECC)
- Kbus hardware cache consistency



**Launcher Section** The Launcher Section initiates transactions on Kbus in response to a cache miss by the CPU Section. The Kbus transaction results in the transfer of data on Kbus. The CPU indicates that it has taken a cache miss by asserting the CMISS signal to the Launcher Section. The CPU Section also provides information (MISS TYPE) about the type of cache miss; read or write, I/O or cacheable, or flush before fill. Based upon the information provided by the CPU Section, the Launcher arbitrates for mastership of Kbus and launches a transaction on Kbus with the appropriate TTYPE (transaction type). The FILLDONE signal is sent to the CPU Section after operations for the cache fill have been completed and allows the CPU to resume execution.

**Kdat Section** Transfer of data between the cache and Kbus is handled through the Kdat Section. Each 32 byte cache block is transferred as four sequential cache lines of 8 bytes each with a ninth Check Byte (CB) for Error Check and Correction. As data is received from Kbus, this block performs a single bit ECC function on each cache line and corrects single-bit errors, detects double-bit errors, and detects some multi-bit errors. Conversely the Kdat Section generates the 8 bit CB (Check Byte) field for each 8 byte cache line that is transferred from the cache to Kbus.

**Watcher Section** The transfer of data between Kbus and the cache is controlled by the Watcher Section. The Watcher Section observes transactions that have a cacheable TTYPE that have been initiated on Kbus and decides if any movement of data between the cache and Kbus is required in accordance with the cache consistency protocol. The Watcher Section maintains a list of which cache blocks are resident in the cache and whether they are writeable.

When the transfer of data is required between the cache and Kbus, the Watcher Section holds off the CPU Section and gains direct control of the cache to move data. The Watcher Address Bus indexes the selected cache block to be transferred.

#### 4.4 CPU Section

Figure 4-2 shows the exploded block diagram of the CPU Section, which contains the CPU/FPU block, the Cache, and the GTLB/FTLB.

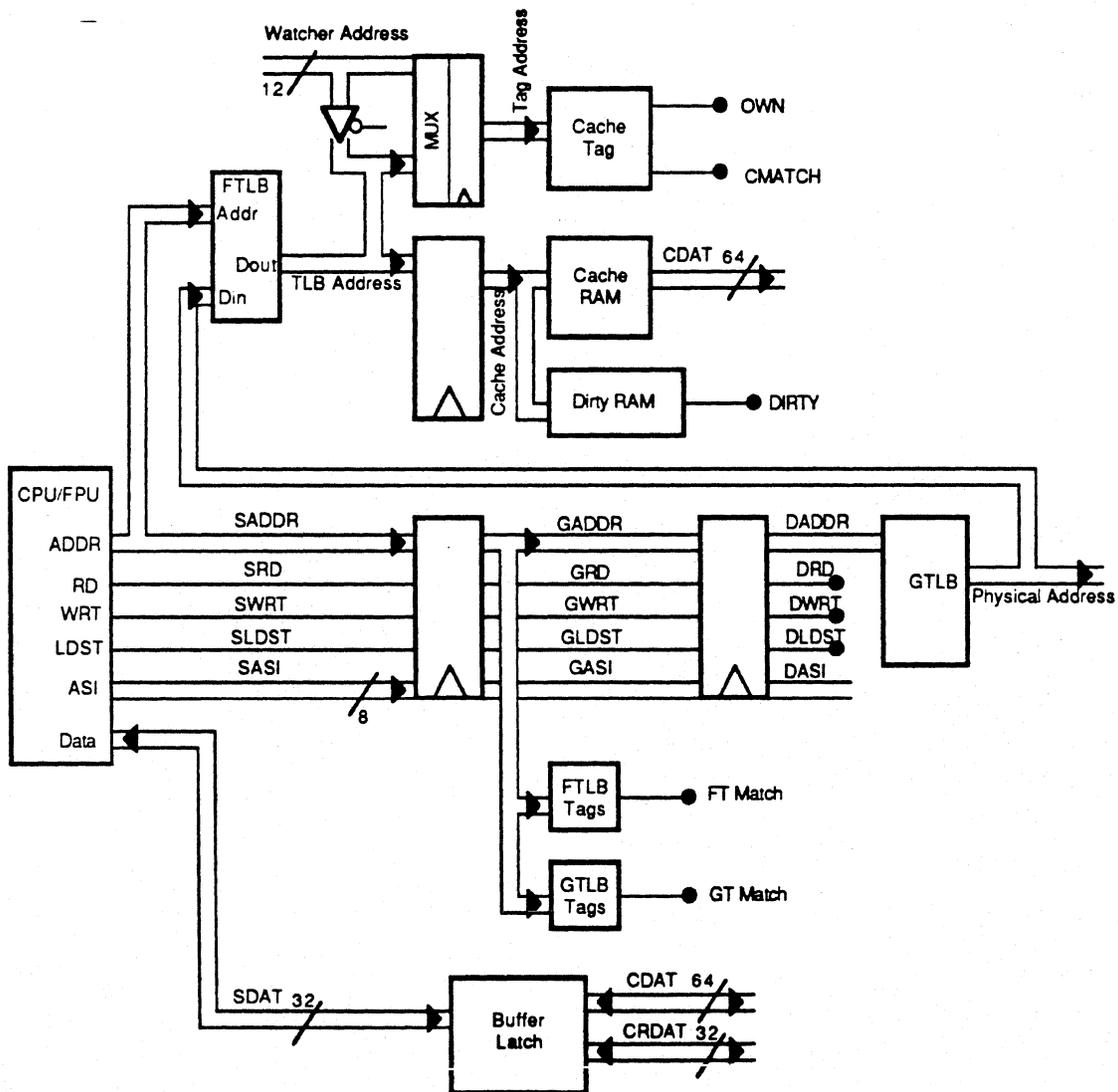


Figure 4-2. Block Diagram of the CPU Section

The CPU/FPU block contains the Cypress SPARC CY7C601 IU and the Weitek 3171 FPU. Program flow is directed by the IU which provides the address for all instruction and data fetches on SADDR. The IU generates addresses for both integer and floating point operations. Instructions and data flow over the SDAT data bus, which is shared by the IU and FPU. Floating point instruction opcodes are latched by the FPU as they appear on SDAT. The IU then instructs the FPU which FP opcodes should be executed.

The CY7C601 as a RISC processor, ideally executes one instruction every clock cycle and therefore must fetch a new instruction on every clock. As such, the IU generates a new address

and address space identifier (SASI) and expects to receive the corresponding new data every clock cycle.

#### 4.4.1 Address Pipeline

There are three sequential stages to the CPU Section's address pipeline. Addresses and address qualifiers sequentially move from the S stage, to the G stage, and then to the D stage: The S stage contains the most recent address (the address being currently issued by the IU). The G stage contains the address issued by the IU on the previous clock while the D stage contains the address issued by the IU two clocks previous.

In addition to addresses, each pipeline stage contains qualification information about each address. These qualifiers include ASI (address space identifier), RD (read), WRT (write) and LDST (atomic load store). As each signal flows down the pipe, a suffix is added to generate the name of the signal in that stage. For example, SASI is the ASI field currently being issued by the SPARC IU. SASI is latched in the G stage of the pipe to form GASI and DASI is similarly generated in the D stage of the pipe by latching GASI.

The S stage address pipe signals are issued directly by the IU and are valid only for a short time before and after the rising edge of clock. These signals are latched during the narrow time window when they are valid to form the G stage of the pipe. In contrast, G and D stages of the pipe remain valid over the entire clock period.

#### 4.4.2 Pipeline Operations

Various operations are performed at each stage of the address pipeline. The S stage of the address pipeline is for the most part dedicated to getting the information from the IU latched for the G stage and in performing the translation from virtual-to-physical addresses in the FTLB.

By using high speed GaAs (gallium arsenide) RAMs for the FTLB block, virtual addresses from the IU are translated through a table look up into physical addresses, which can reference system memory and the physical cache.

The 40 MHz IU, which is running at 33 MHz, provides a larger window before the rising edge of clock to permit sufficient time for the translation to occur and still latch the translated address for the G stage of the address pipe. The translated address is latched in the G level of the pipe to form the TAGADDR (tag address bus) and the CADDR (cache address bus). Thus, there is no performance penalty for implementing the physical cache since TAGADDR and CADDR have the same pipeline timing as they would in a virtual cache.

During the G stage of the address pipe, the physical address that is latched on CADDR, is used to reference the data cache RAMs and provide 64 bits of data on CDAT. The Buffer Latch Block selects the upper or lower 32 bits of CDAT and drives SDAT in time for the IU and FPU to sample data before the rising edge of clock. The G stage physical address latched on TAGADDR is simultaneously used to perform a cache tag lookup (check the list of valid blocks to see if the addressed cache block is in the cache). If the cache tags match, then CMATCH asserts before the next rising edge of clock (i.e., the block is already in the cache). If the CMATCH signal is not asserted, then control circuitry halts the IU (because the cache block referenced by the physical address on CADDR and TAGADD was not in the cache) and asserts CMISS to cause the cache block to be fetched in from Kbus.

At the same time that the cache tag check is occurring, the virtual address that is latched on GADDR is used during the G stage of the address pipeline to reference the FTLB and GTLB tags. These tags contain the list of virtual addresses whose translations are contained in the FTLB and GTLB. If the FTMATCH or GTMATCH signals are asserted, then the virtual-to-physical address

translation referenced in each of the FTLB or GTLB, respectively, are present and valid.

If the FTMATCH signal is not asserted, then the state of the CMATCH signal is invalid because the wrong or invalid translation entry is present in the FTLB and was placed on CADDR and TAGADDR. Both the FTLB and cache must contain the matching entries for the CPU to continue execution. If either FTMATCH or CMATCH is not asserted a “cache miss” occurs.

The CPU cannot continue execution of its current program until the correct translation entry is placed in the FTLB. While the GaAs FTLB translation RAM is extremely fast it contains only 256 entries. If the FTLB entry doesn't match, it is much more likely that the larger GTLB which contains 8 Kbyte entries will match. If the FTLB doesn't match but the GTLB matches, then the virtual-to-physical translation from the GTLB is quickly copied into the FTLB by control hardware and then the cache is checked for CMATCH with the new physical address translation.

Control circuitry causes an automatic FTLB miss when a GASI value greater than 63 is generated by the IU. Values from 0-to-63 are reserved for normal cacheable operations while values from 64-to-127 are used to address specific control registers and test functions. An automatic FTLB miss is generated when a GTLB translation entry is marked I/O since these translations are not dropped into the FTLB.

The IU and FPU read data on the rising edge of clock at the end of the G stage of the address pipe. The CMATCH and FTMATCH signals become valid just before the same rising edge so there is not enough time to prevent the IU and FPU from reading the wrong data if a cache miss occurs. Instead, the CY7C601 has a mechanism of halting and backing up after reading invalid data on a cache miss. When the IU is halted, the D stage of the address pipe contains the address which caused the cache miss.

Address and qualifier information for the cache miss is held in the D stage of the address pipe as long as the IU is halted. The virtual miss address DADDR is fed into the GTLB translation RAM to generate a physical address that can be used to refill the FTLB (if the FTLB missed and GTLB matched). The same physical address is also used by the Launcher Section to reference main memory when fetching a block into the cache.

If both the FTLB and GTLB do not contain the required virtual-to-physical translation, then a memory exception is taken for that access. The execution of the current program is suspended until the kernel exception handler can refill the GTLB with a valid translation (and re-executes the instruction that caused the memory exception).

#### 4.4.3 Cache Organization

The 128 Kbyte direct mapped cache in the CPU Section is organized into 4096 cache blocks with unique address tags. Each cache block, containing 32 bytes of data, is the smallest packet of information which is moved between the cache and system memory on Kbus. Byte, double byte, word and double word read and write operations are supported between the IU/FPU and the cache. The cache contains local current copies of main system memory (with 32 byte cache block granularity) which have been most recently accessed by the CPU.

Once a cache block is resident in the cache, IU/FPU reads and writes occur as required to the block, with no additional Kbus activity. If the block has been modified via a write, main system memory will be updated before the block is replaced with a block having a different address. While this copy back caching strategy is somewhat more complex, it provides much lower level of Kbus traffic than would be achieved in a write through cache where every CPU write simultaneously updates both the cache and main system memory.

Once a block is modified, it becomes the only valid copy of that cache block in the system. On occasion, the Watcher Section must halt the IU to read a cache block for another bus master requesting the current copy of a cache block.

A physical cache is accessed by addresses that have been translated from the virtual address generated by the IU. Since the same physical addresses are used to address the cache as are used to address main system memory, maintaining cache consistency between caches on multiple CPU Boards in the system is greatly simplified. In machines like the Series4, which have a virtual cache, an extra set of reverse translations must be maintained to translate the physical address generated by the Watcher Section back into a virtual index which can reference the cache. Circuitry on the Series4 also guarantees that virtual-to-physical and physical-to-virtual mapping is 1:1 and unique. The 1:1 mapping constraint on the Series4 makes the 64 Kbyte cache appear smaller than it actually is since extra data must be flushed to memory to prevent aliased mapping situations. This performance degradation is the price paid for multiprocessing in Series4. This situation is completely avoided with the physical cache on Series5.

#### 4.5 Launcher Section

The Launcher Section is responsible for determining what action is to be taken when a cache miss occurs and the CPU Section asserts CMISS. The Launcher Control block not only receives CMISS but also additional information about the type of miss that has occurred. After the Launcher Control Block has serviced the cache miss, FILLDONE is asserted to indicate that the cache fill operation is complete and the CPU Section can continue.

An exploded block diagram of the Launcher Section is shown in Figure 4-3.

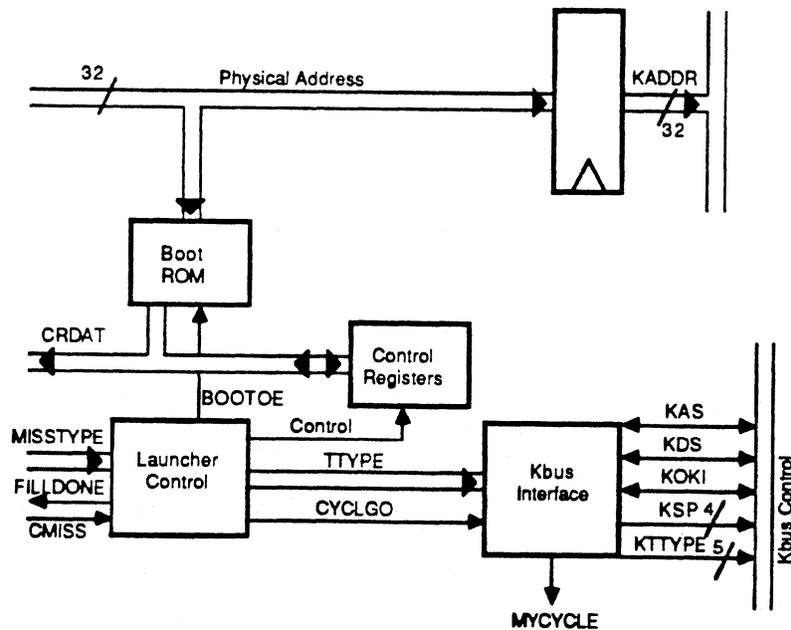


Figure 4-3. Block Diagram of the Launcher Section

The Launcher Control block has the choice of initiating a BootROM, control register or Kbus access to service the cache miss. A BootROM access is selected if the virtual address of the IU access is in the range 0xFEXXXXXX, or if the MMU is disabled and the IU access is in the range 0x0XXXXXXX. The BootROM data is returned to the CPU Section's Buffer Latch block on CRDAT<sub>1</sub>, the control register data bus and finally given to the IU/FPU.

The MISSTYPE information includes DASI which is the ASI of the cache access that caused the cache miss. If this value is between 96 and 127 then a Kbus control register is selected by the Launcher Control block.

If a BootROM access or Kbus control register access is not selected when CMISS is asserted, the Launcher control block asserts CYCLGO to initiate a Kbus transaction. A TTYPE is passed from the Launcher Control block to be placed on Kbus. If a cache miss was taken on an IU/FPU read, then a Cacheable Read TTYPE is generated. If the cache miss occurred on a write, then a Read and Invalidate TTYPE is used. If the current block in the cache is valid and dirty (been written to by the IU/FPU) then an IOB transaction is generated to first flush the cache block out to main system memory before the cache location is filled with new data.

The Kbus Interface block begins initiating a new Kbus transaction when its CYCLGO input is asserted. The CPU Section provides the Launcher Section with the physical cache miss address by translating DADDR through the GTLB which is then clocked out to Kbus. If it is not already the Kbus master, the Interface block first acquires mastership and then drives the bus with KAS, KSP, KTTYPE and KADDR in the next available time slot. These signals remain asserted as long as there is a pending KOKI or KDS handshake from the previous transaction.

Once the transaction has been launched onto Kbus, MYCYCLE signal is asserted and sent to the Watcher Section to indicate that the current transaction on Kbus has been initiated by this CPU Board.

## 4.6 Kdat Section

The Kdat Section contains a 64 bit data path for transferring data between Kbus and the cache. Each cache block is transferred as four sequential cache lines of 8 bytes each.

A block diagram of the Kdat Section is shown in Figure 4-4.

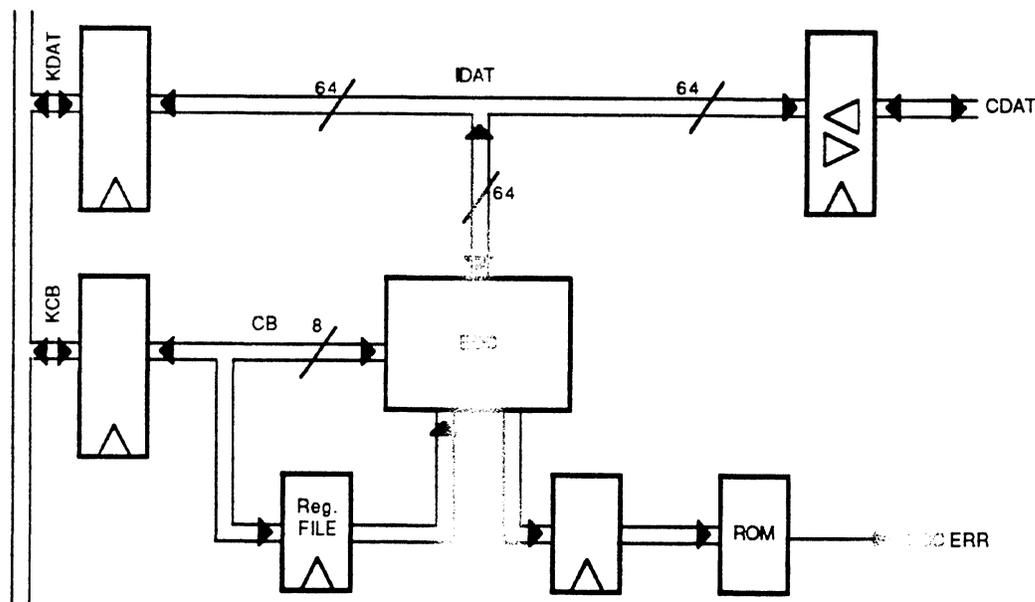


Figure 4-4. Block Diagram of the Kdat Section

The Kdat Section checks incoming data from Kbus for any ECC errors as the data is written into the cache. The EDC (Error Detect and Correct) block checks each of the four cache lines as they are clocked onto IDAT against the corresponding check byte on CB. The four check bytes are saved in a register file. Once all four cache lines are received and written into the cache, a control state machine checks if an error was detected on any of the four cache lines. If no error is detected, the CPU continues. However, if an error was detected, the CPU remains in a halted state until the EDC state machine corrects the error. If an error occurred, then the control state machine reconfigures the EDC for correction (which is much slower than detection) and sequentially reads each of the four cache lines from the cache, corrects each using the corresponding retained check byte value in the register file, and then rewrites the data into the cache. The control state machine then signals the CPU Section as to whether a single bit error (which is always correctable), or an uncorrectable multi-bit error occurred. A single bit error is signaled as an interrupt to the CPU Section, while a multi-bit error generates a memory exception which usually results in a kernel panic. As opposed to Series4 which had on the fly ECC detection and correction, the Series5 has optimized the speed of the detection function at the expense of the correction since correction is seldom invoked while detection occurs on every cache fill operation.

The Kdat section is also responsible for generating the CB field for each cache line as data is moved from the cache onto Kbus. In this mode, data is clocked from CDAT onto IDAT. During the time the data is on IDAT, the EDC generates the check byte for the cache line. On the next clock cycle, both the cache line on IDAT and the corresponding check byte on CB are clocked onto Kbus.

## 4.7 Watcher Section

The Watcher Section controls the movement of data between the cache and Kbus. It observes transactions which various masters including its own Launcher have initiated on Kbus and decides what actions are required.

A block diagram of the Watcher Section is shown in Figure 4-5.

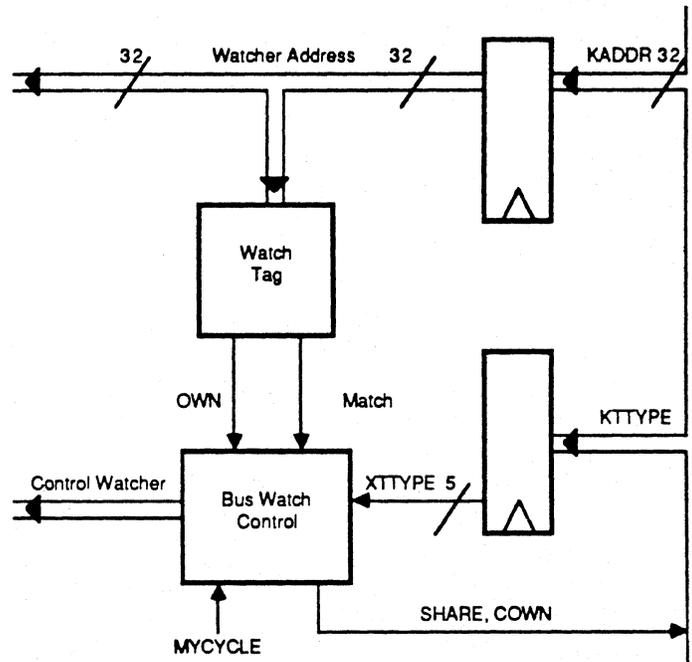


Figure 4-5. Block Diagram of the Watcher Section

As transactions appear on Kbus, the address of each cacheable transaction is latched on the Watcher Address bus. This address indexes into the Watcher Tag block, which contains a list of all cache blocks resident in the cache. If the block is present, a match signal is asserted to the Bus Watcher Control block which then causes the SHARE signal to be asserted on Kbus, indicating that a valid copy of the cache block is present outside of system memory. The OWN status from the Watcher Tag block is similarly echoed out to Kbus as COWN if a writeable copy of the block is in the cache.

If another bus master requests a copy of a cache block that is in the list of cache blocks contained in the cache and if that block is also owned, then the Watcher Section gains control of the cache, and provides a copy of the block to the requesting device on Kbus. The address that is driven onto the Watcher Address bus is used to reference the corresponding location in the cache.

The Bus Watcher Control block responds differently to Kbus transaction requests made by its Launcher as opposed to other Kbus masters. For example, on a Cacheable Read or Read and Invalidate transaction type, if the transaction was initiated by the Launcher block then the Bus Watcher Control block can expect to receive data from Kbus to be placed in the cache. If it was initiated by another master, it might need to supply data if it was the owner of the block. If the Read and Invalidate was initiated by the Launcher, then the Bus Watcher Control block will also

## Section 5: System Board Operation

### 5.1 Introduction

The System Board handles all I/O processes for the Series4 and Series5 maintenance functions for the system and the Kbus. The I/O processor Board include interface support for the VMEbus, Ethernet, SCSI, serial, and mouse. The System Board provides 16 levels of interrupts for the I/O processor. System maintenance functions provided by the System Board include RAM refresh, Kbus clocks, and bus timeout circuitry. The System Board can be a Kbus master or Kbus slave, depending on the type of transaction in which it is involved. The System Board is sectioned into five basic functional areas: the Kbus interface section, the VMEbus interface section, the SCSI/Ethernet interface section, the slow bus section, and the I/O ASIC section (see Figure 5-1).

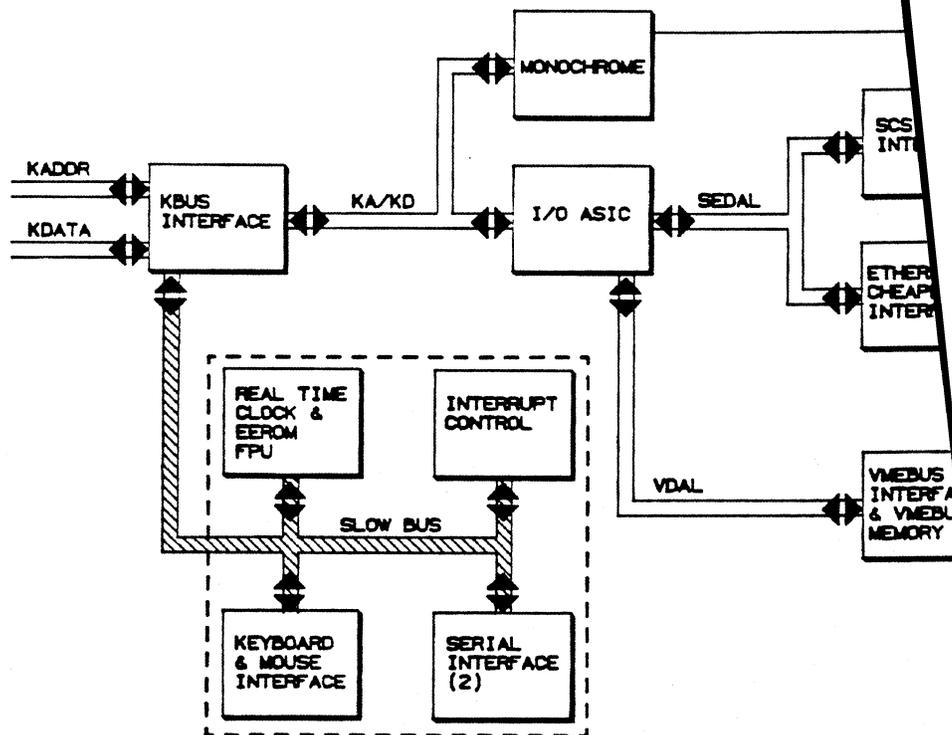


Figure 5-1. System Board Block Diagram

The Kbus interface section synchronizes all signals that pass between the other System Board and the Kbus. The SCSI/Ethernet section interfaces all internal SCSI devices and the Ethernet to the I/O ASIC. The VMEbus interface section of the board controls all data transfers between the VMEbus and the I/O ASIC or the Kbus interface.

The slow bus section of the system is dedicated to receiving and transmitting data between the Kbus interface section and all the I/O ports (keyboard, mouse, RS-423-A, etc.) This section is also devoted to handling all I/O interrupts which includes VMEbus interrupts. The I/O ASIC section of the board includes all support buffers and transceivers that aid this ASIC in its operation of the System Board functions.

## 5.2 Kbus Interface Subsection

The Kbus interface subsection of the System Board is the direct link between the Kbus and all other subfunctions of the System Board. This section is split into five subsections: System Board control, clock generation, Kbus address latches, Kbus data latches, and the latches for the System Board slow bus (see Figure 5-2).

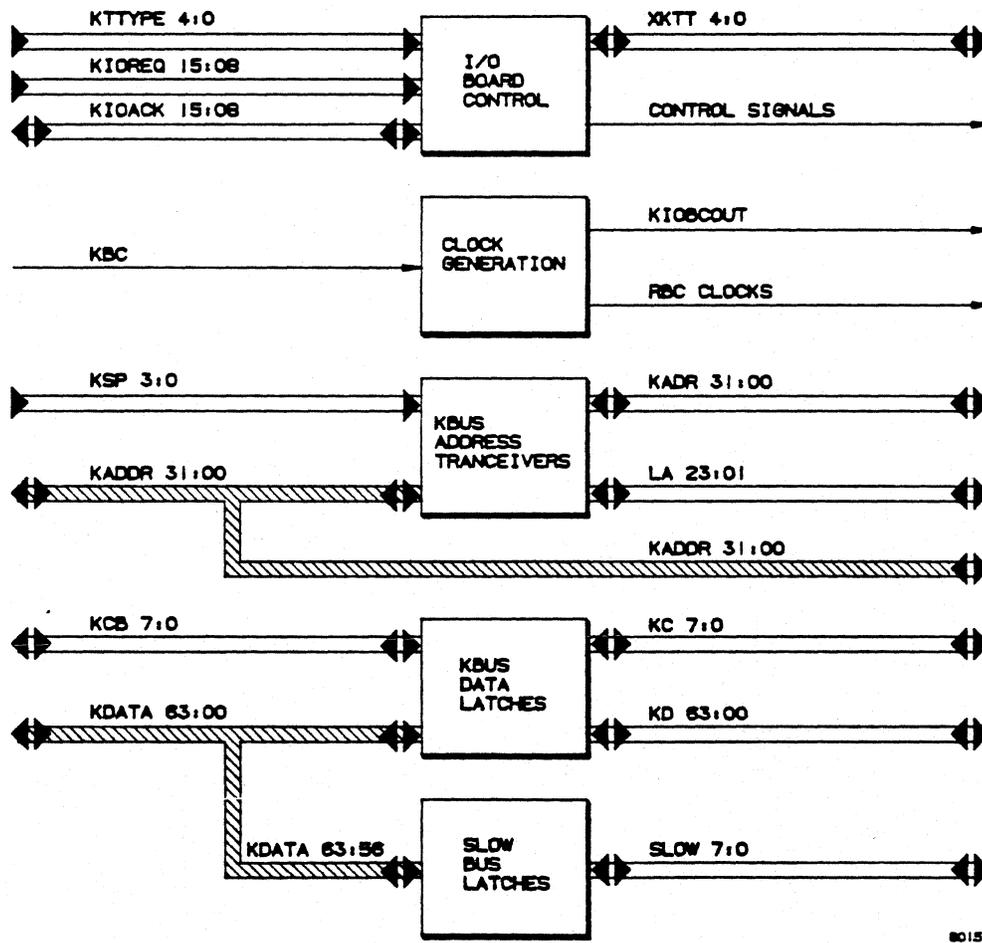


Figure 5-2. Kbus Interface Subsection Block Diagram

The System Board control block is comprised of many PALs which are each involved with the monitoring of and interaction between the Kbus and the various other subfunctions of the System Board. One PAL, the arbitor PAL, controls requests and acknowledge signals between the Kbus and the I/O devices. These devices can be located on the VMEbus, SCSI bus, or Ethernet. There are three PALs which are devoted to timing and interface with the Kbus as well

as decoding RIO space lines. The remaining PALs are devoted to issuing control signals for the other subsections, namely a slow bus control PAL and a VMEbus PAL. The rest of the System Board control block is comprised of latches and buffers for holding and driving the various Kbus and I/O control signals.

The clock generation block performs **two basic functions: it provides the 20 MHz clock (KIOBCOUT) from which the Kbus clocks are derived and it reconstructs the incoming Kbus bus clocks (KBC and KBCO) into RBC clock signals used on the System Board.** The remaining three blocks of the Kbus interface section include the various latches and transceivers for interfacing with the Kbus.

The Kbus address transceivers block has a set of four octal bus transceivers which transfer 32 address bits between the Kbus address bus and the KADR address bus. The KADR address bus sends addresses to the I/O ASIC block. A bank of octal D-type flip-flops latches the lower 28 bits of the KADR bus and the four KSP bits, resulting in the latched address bus. Three bytes of the latched address bus, bits LA [23:01], go to the VMEbus interface block for transferring addresses to the VMEbus.

The Kbus data latches block consists of two banks of eight octal latches which transfer the 64 bits of data between the Kbus KDATA data bus and the System Board KD data bus. The KD data bus enters the I/O ASIC block. The upper eight bits of the Kbus data bus, KDATA [63:56], are also transferred to and from the eight bits of the slow bus, SLOW [7:0], via two octal latches.

### 5.3 VMEbus Interface Subsection

The VMEbus interface supports communication between VMEbus boards and the Kbus interface section of the System Board. The VMEbus interface subsection consists of: VMEbus interface control block, two address buffer blocks, a set of static RAMs, and data transceivers.

The VMEbus interface control block issues all control signals used for interfacing with the VMEbus. The control signals come from the VMEbus interface PAL in the Kbus interface section of the System Board.

The two address buffer blocks in the VMEbus interface section are used for transferring address lines between the VMEbus and the System Board. Addresses generated for output to the VMEbus are issued on latched address bus LA bits 23:01. Addresses coming in from the VMEbus are driven onto the multiplexed VMEbus address/data bus, VDAL [31:00]. Of the addresses coming in from the VMEbus, only the lower twelve bits, A 12:01, are received and transferred directly to the VDAL bus as the lower bits of the translated address. The upper eleven bits, A 23:13, are used to address the 2 Kbyte locations of RAM which hold the page table entry for virtual memory mapping the 16 Mbyte VMEbus space into the 4 Gbyte system space.

All 32 bits of the VMEbus data bus are transferred to and from the multiplexed VMEbus address/data bus, VDAL [31:00]. Four octal bus transceivers transfer each of the four bytes upon commands from the VMEbus control PAL located in the Kbus interface section (see Figure 5-3).

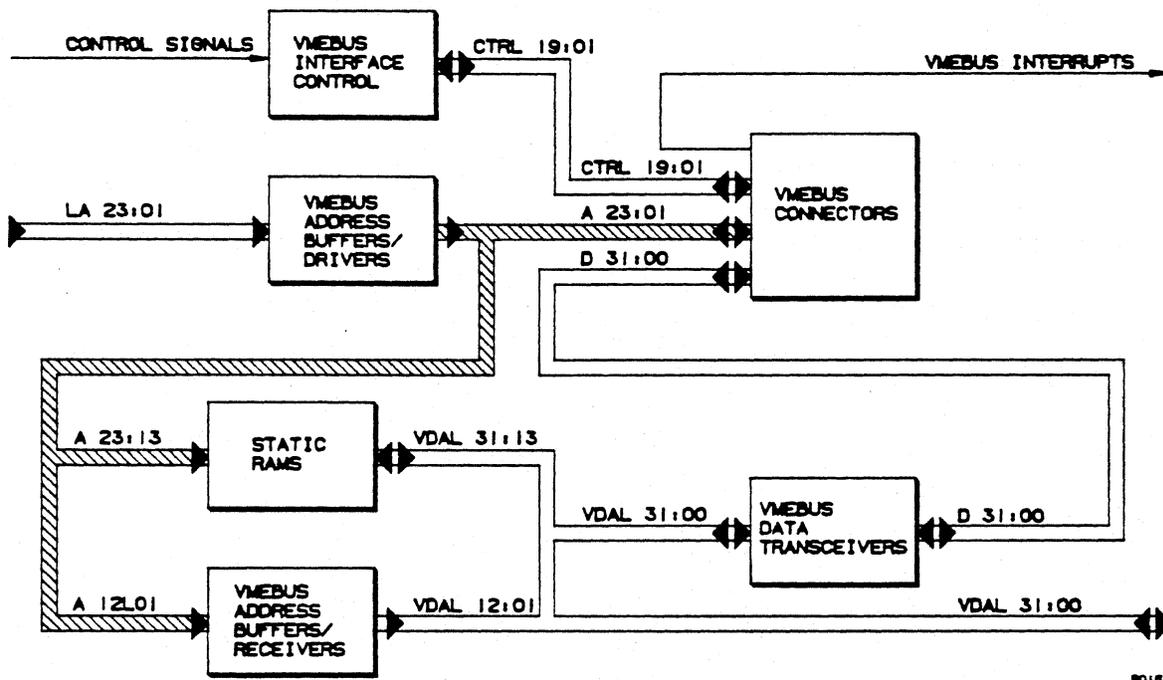


Figure 5-3. VMEbus Interface Subsection Block Diagram

#### 5.4 SCSI/Ethernet Interface Subsection

The SCSI/Ethernet interface subsection supports SCSI and Ethernet interface. Addresses and data are transferred to and from this block on the 16-bit SCSI/Ethernet multiplexed data and address bus, SEDAL. The SCSI controller uses only one byte of the SEDAL bus and the Ethernet controller uses all 16 bits of the SEDAL bus.

The SCSI interface circuit communicates with up to seven mass storage devices. These devices include tape drives, hard disk drives, and other SCSI-compatible devices. The SCSI interface block consists primarily of a Western Digital 33C93 SCSI controller running in synchronous mode with support for disconnect and reconnect. The DMA rate of the controller is in the 2.0 to 4.0 Mbyte transfer rate range. The disconnect/reconnect feature improves performance significantly by overlapping seeks from one drive to the next. (see Figure 5-4).

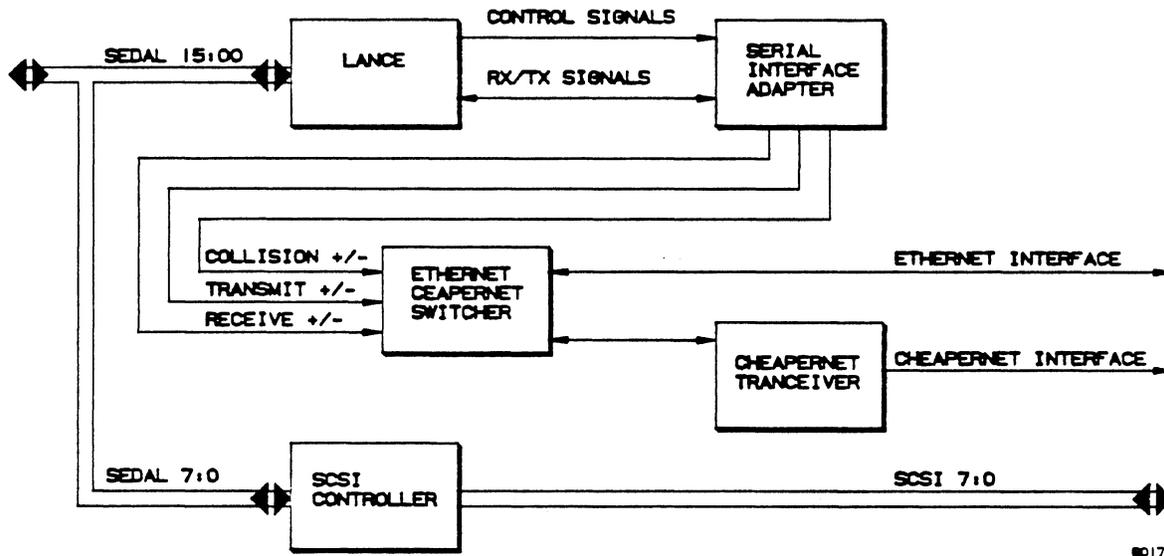


Figure 5-4. SCSI/Ethernet Interface Subsection Block Diagram

The SCSI chip has physical (single page) direct memory access (DMA) capability through the use of a DMA byte counter. It does not have an address register; thus the I/O ASIC contains a 32-bit address counter for the SCSI chip. This 32-bit counter can halt on any page boundary. If the SCSI chip requests that the counter cross a page boundary, an interrupt is generated and the SCSI chip stops transfers. The CPU can then change the address register while the SCSI chip is stopped. Once the address is updated to the next page, the SCSI chip is allowed to continue the DMA transfer.

The Ethernet circuit can handle connection to both Ethernet and Cheapernet networks. A Cheapernet transceiver package supplied by the user, is required to transition from Ethernet to Cheapernet. A trio of chips lie at the heart of the Ethernet circuit:

- Am7990 local area network controller for Ethernet (LANCE) chip
- Am7992 serial interface adapter (SIA) chip
- Am7996 transceiver

The Am7990 LANCE chip saves CPU time by performing all the functions of packaging the data to be transmitted over either network and unpackaging data coming in from either network. It accesses various types of information utilizing three base registers. The LANCE outputs data on the 16 bits of the SEDAL address bus. The lowest 13 bits select the byte within a page and the upper 3 bits select the type of transaction and the base register.

The LANCE chip interfaces directly with the Am7992 SIA which performs the task of decoding incoming data signals and encoding outgoing data signals into the Manchester format. The SIA

suppresses transient network line noise and ensures that the signals generated meet the requirements of the IEEE 802.3 standard.

The Am7996 transceiver performs the functions of detecting collisions, receiving data and transmitting data directly over the network. This chip contains noise filters which minimize the possibilities of data errors due to induced signals on the network.

The final remaining block on the Ethernet circuit is the switcher block that switches between the two types of networks. This switcher senses the type of cable that is connected to the System Board. If an Ethernet drop line is plugged into the Ethernet connector, the board switches to Ethernet. If a Cheapernet cable is plugged into the Cheapernet connector, the board uses Cheapernet. This switcher circuit does not support both networks simultaneously.

### 5.5 Slow Bus Subsection

The slow bus is an eight-bit data bus that runs at a slower speed than the other circuits on the System Board. The functional blocks that lie along the slow bus are relatively slow devices that deal primarily with the transfer of data between the various I/O devices namely the keyboard, the mouse, and two serial ports. In addition, a real time clock, an EEROM, and the System Board interrupt circuit sit on the slow bus (see Figure 5-5).

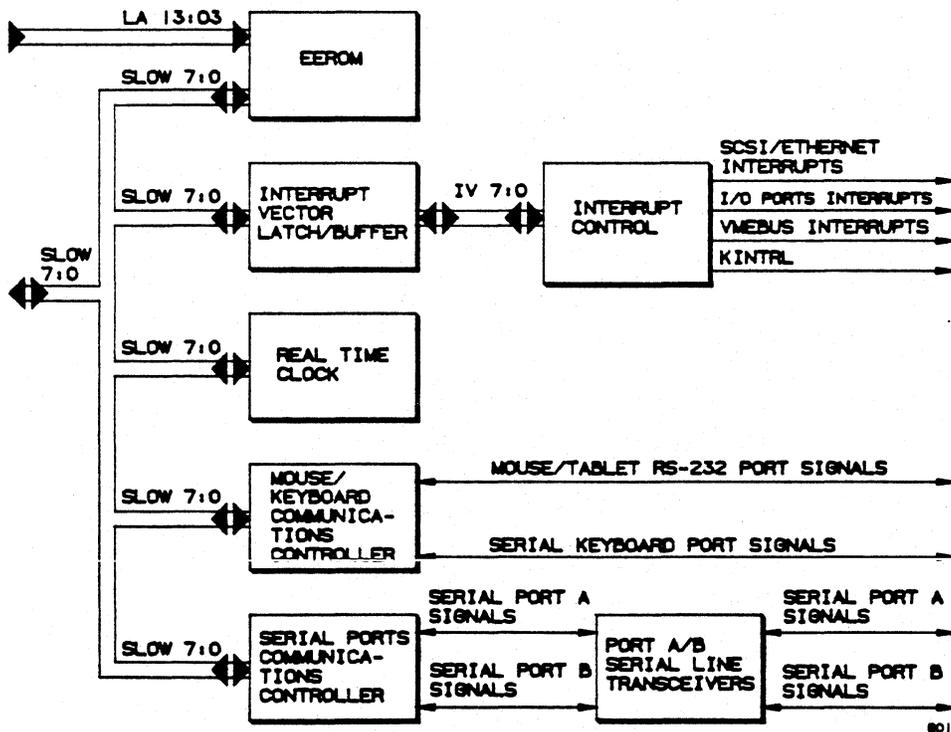


Figure 5-5. Slow Bus Subsection Block Diagram

The EEROM sitting on the slow bus is a 2 Kbyte by 8 erasable ROM that contains the machine serial number, Ethernet address, and other pertinent system information. The first half of the ROM (1Kbyte) is read-only and can only be programmed at the factory. The second half of the ROM is read/write and contains user configuration information. The ROM is addressed with

bits 13:03 of the LA address bus. Data transfers to and from the ROM over the slow bus.

Interrupts enter the System Board from all the I/O devices in the Series4 and Series5 family . The interrupt control block deals with the interrupts coming from the various sources. The interrupt control block can handle 16 levels of interrupts. It generates one level of interrupt for each I/O device. In case of more than interrupt being received at one time, the highest priority interrupt level is sent to the Kbus first.

There are even and odd interrupts. Each interrupt enters the interrupt control block by setting the positive output of a D-type flip-flop. The outputs of all flip-flops go into either an even or an odd eight-to-one multiplexer. These multiplexers then alert the interrupt PAL of the pending interrupt. The PAL sorts through the interrupts in a descending order until it finds the highest level interrupt asserted. It then accepts the interrupt and clears the interrupt's flip-flop.

Table 5-1. System Board Interrupts

Interrupt Level	Interrupting Device
Base+0	Unused
Base+1	SCSI
Base+2	VME-IRQ1
Base+3	SCSI Page Overflow - Illegal I/O Transfer
Base+4	VME-IRQ2
Base+5	Monochrome Graphics
Base+6	VME-IRQ3
Base+7	ETHERNET
Base+8	VME-IRQ4
Base+9	SERIAL A/ SERIAL B
Base+A	VME-IRQ5
Base+B	Keyboard / Mouse
Base+C	VME-IRQ6
Base+D	System Timer (no servicing required)
Base+E	VME-IRQ7
Base+F	Profile Timer (no servicing required)

When an interrupt is received, the interrupt control block generates an interrupt vector which is then transferred to the slow bus by the interrupt vector latch/buffer block. The interrupt vector latch/buffer block receives the interrupt vector via the IV bus. The interrupt vector holds more interrupt information which can be read by the servicing CPU.

The battery-backed, real time clock maintains the current time and date for the processes of the Series4 and Series5 family . This block is comprised of a Statek RTC-58321 real time clock chip. This chip is accurate to within  $\pm 4.5$  seconds per day.

The remaining two blocks in the slow bus subsection include Z8530 serial interface communications controller chips which each have two ports. One block is for the mouse and keyboard ports. The other block is for the two RS-423-A communications ports.

Port A of the mouse/keyboard communications controller is a port for a mouse RS-232-C input. This port operates at a transfer rate of 1200 baud. Port B is used for interface to the keyboard. Each key returns three bytes: the first byte when the key is pressed and the next two bytes when the key is released. Each byte returns an interrupt to the interrupt controller. Commands are sent to the keyboard through this communications controller.

The two serial communications ports are driven by the same Z8530 communications controller. These ports can run asynchronously at rates up to 57.6 Kbaud or synchronously at rates up to 921 Kbaud. The ports are EIA Standard RS-423-A, which is reverse compatible with RS-232-C.

## 5.6 I/O ASIC Subsection

The I/O ASIC is a 64-bit chip implemented in complementary metal-oxide semiconductor (CMOS) technology. It interfaces with most other sections of the System Board including: Kbus interface, VMEbus interface, SCSI, and Ethernet interface sections. The I/O ASIC chip synchronizes and coordinates the transfer of data between the extremely fast Kbus and the somewhat slower I/O areas.

## 5.7 Monochrome Video Subsection

The Series4 and Series5 Monochrome Graphics subsection provides RAM and video drivers for the Series4 and Series5 monochrome frame buffer. The control register and the frame buffer memory of the Monochrome Graphics subsection reside in I/O space and are addressed through RIO transactions on the Kbus. The control register sits in I/O space 1 and the frame buffer sits in any one of seven I/O spaces between 8 and 15. This board supplies video signals to the Series4 monochrome display monitor.

The frame buffer consists of 256 Kbytes of memory that can support two resolutions of display:

1. A low-resolution monitor having a 1152 x 900 pixel display refreshed at 69 Hz
2. A high-resolution monitor having a 1600 x 1280 pixel display refreshed at 66 Hz

The low-resolution mode uses 126 Kbytes of memory. The high-resolution mode uses 250 Kbytes of memory. The frame buffer is organized in a linear array with one screen pixel for each bit of video data.

The Monochrome Graphics subsection is sectioned into four basic subsections: Kbus interface (shared with other System Board devices), video control, video RAM, video drivers. See Figure 5-6.

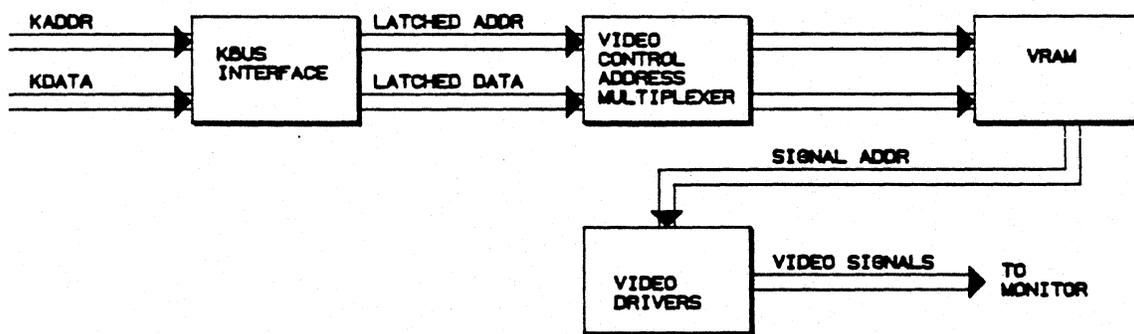


Figure 5-6. Monochrome Graphics Subsection Block Diagram

### 5.7.1 Kbus Interface for Monochrome Graphics

The Kbus interface watches for video addresses on the Kbus. The Kbus interface responds to the video addresses by latching both the Kbus address and the Kbus data. It then supplies the latched address and data to the video control subsection. The Kbus interface section is divided into: a Kbus interface control block, Kbus address latches, Kbus data latches, and a transaction type and identification block. See Figure 5-7.

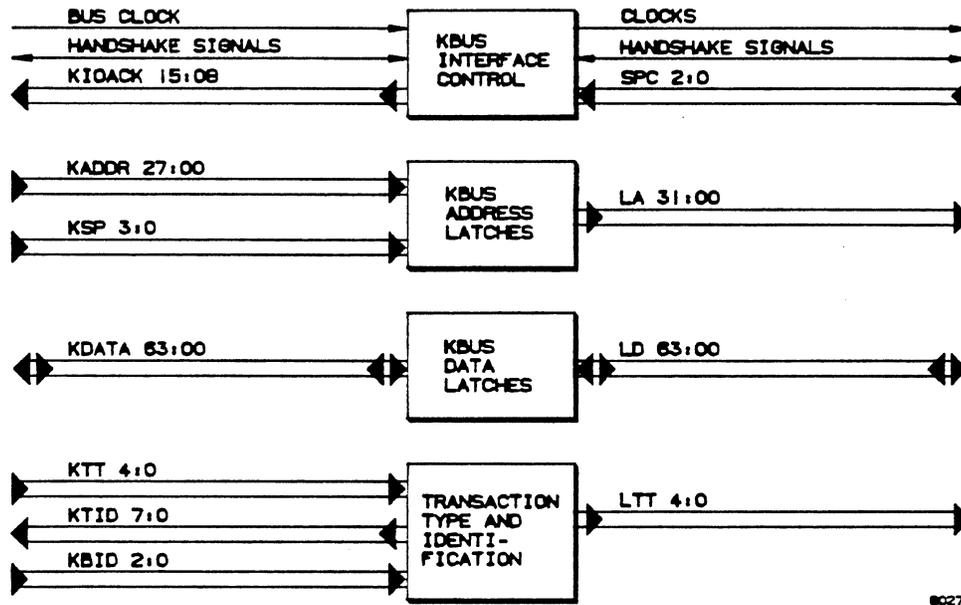


Figure 5-7. Kbus Interface Subsection Block Diagram

The Kbus interface control block makes sure that all the Kbus handshaking requirements are met by the Monochrome Graphics subsection. In the handshaking process, this block receives status signals from other subsections of the Monochrome Graphics subsection. The Kbus interface control block also receives the KBCO Kbus clock and generates internal reconstructed bus clocks for all other circuits on the Monochrome Graphics subsection. Since the Monochrome Graphics subsection resides in I/O space, it must respond to the I/O request/acknowledge protocol. The Monochrome Graphics subsection can only be a Kbus slave, so the interface control block issues the response on the IOACK [15:08] bus. A three-to-eight decoder issues the appropriate IOACK signals after decoding the three bits of the space bus, KSP [2:0], from the video control subsection's control status register.

The Kbus address latch block and the Kbus data latch block are similar in function. The Kbus address latch block consists of four octal D-type flip-flops which serve to latch the Kbus address and I/O space bits off the Kbus. The resulting 32-bit address is then transferred to the latched address bus LA [31:00]. The Kbus data latch block consists of 16 octal D-type flip-flops which serve to transfer data. Eight of the chips latch data flowing to the Graphics Board from the Kbus data bus. The other eight chips latch data from the LD [63:00] bus for transfer to the Kbus.

The transaction type and identification block performs the duty of latching the Kbus transaction type lines, KTTYPE [4:0], and identifying the board's location for the configuration program. One octal D-type flip-flop latches the KTTYPE lines. This latch then places the transaction type code on the latched transaction type bus, LTT [4:0]. The board's identity is placed on the KTID [7:0] bus by a dual-quad transceiver. This transceiver includes in the KTID value, the state of

the three board identity bits from the KBID lines.

## 5.7.2 Video Control Subsection

The video control subsection prepares the latched address and data for input to the video RAMs. It also contains the Monochrome Graphics subsection control/status register which controls all activity on the board including whether the board issues video signals to a high-resolution monitor or a low-resolution monitor.

In this block, the latched address is decoded to determine if the Kbus data is destined for the video board. If it is, the latched addresses gets multiplexed and sent to the RAMs. Since the video RAM array is only 32-bits wide, the incoming 64-bit latched data is split into high and low 32-bit data words for transfer to the video RAMs. The video control subsection is comprised of three functional blocks: video control, address multiplexers, and data divider latches. See Figure 5-8.

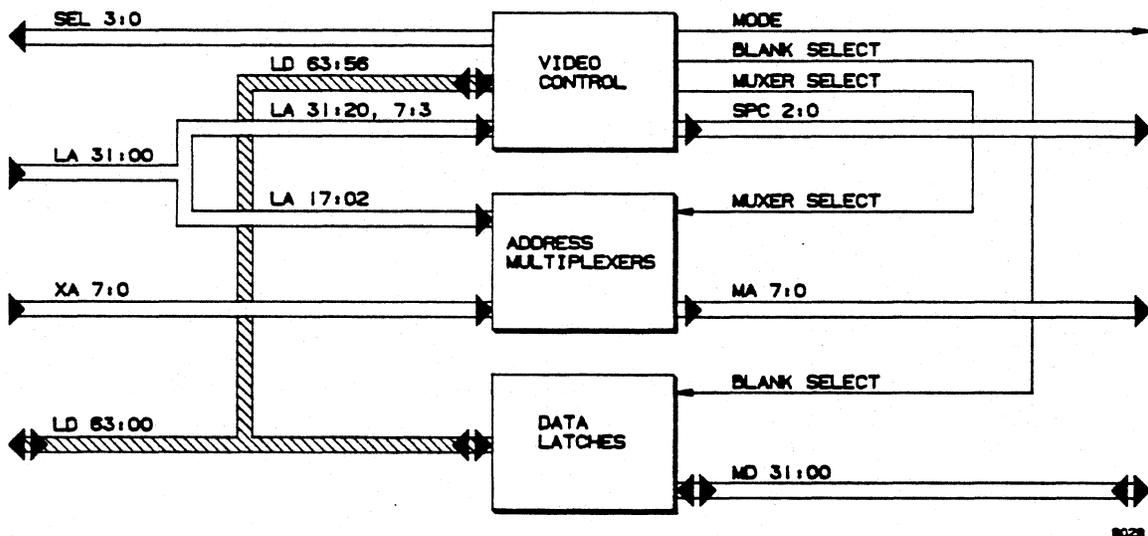


Figure 5-8. Video Control Subsection Block Diagram

The video control block receives the latched addresses and data from the Kbus interface subsection. Two PALs connect to the LA address bus: an address decoder PAL and a register/RAM selector PAL. Identification information is stored in the EAROM on the System Board.

The address decoder PAL also receives the latched transaction type lines, LTT [4:0], and three space bits identifying the location of the frame buffer. The three space bits are set by software in the control/status register and allow for the 8-15 I/O space locations to which the frame buffer responds. The address decoder PAL decodes all the bits of data from its inputs and determines if the transaction on the Kbus is intended for the control registers of the Monochrome Graphics subsection or the frame buffer.

The register/RAM selector PAL further decodes bits of the latched address and latched transaction type busses. If the transaction on the Kbus is intended for control registers, this PAL selects the exact register. If the transaction on the Kbus is intended for the board's frame buffer, the register/RAM selector PAL issues the appropriate code on the four-bit SEL [3:0] select bus to select up to the four banks of video RAM.

The video control block contains the control/status register which is a octal D-type flip-flop which latches data bits off the latched data bus, LA [63:57]. This register gets set by software during initial system configuration. It issues board control signals which enable the frame buffer, enable the video signal to the display monitor, and enable interrupts. The control/status register also issues the MODE signal which selects between the low and high resolution monitors. Finally, the control/status register holds in the three space bits, KSP [2:0], the value of the I/O space to which the frame buffer responds.

The address multiplexer block provides the row and column addresses for the four banks of the video RAMs. This block consists of four dual four-to-one multiplexers which combine 16 latched address bits LA [17:02] with eight transfer address bits XA [7:0]. This block outputs the eight-bit multiplexed address bus MA [7:0]. Coordination of the multiplexing process is done through multiplexer select signals that are fed back to the multiplexer block from the video RAM control block in the video RAM subsection.

The data latches block transfers data between the 64-bit data bus of the Kbus interface subsection and the 32-bit data bus of the video RAM subsection. This block takes the data coming in from the Kbus and splits it into a high and low half. Each half is then transferred to the video RAMs. Conversely, data being sent to the Kbus is collected from the RAMs in two 32-bit data transfers before being sent on. The data latches block consists of 16 octal D-type flip-flops used for latching the data. Eight of the chips latch data off the LA latched address bus and drive the MA multiplexed address bus. The other eight chips transfer data from the MA multiplexed address bus to LA latched the address bus.

### 5.7.3 Video RAM Subsection

The video RAM subsection of the Monochrome Graphics subsection contains eight 64 Kbytes x 4 video RAM chips. Each pair of chips makes up a 64 Kbyte bank. There are four banks. So, there is a total of 256 Kbytes of memory for the frame buffer. The video RAM chips store the data supplied by the data latches of the video control subsection. This data then is transferred to the video driver subsection for preparation for the monitor. The video RAM subsection is divided into two functional areas: the four video RAM banks and the video RAM control circuit. See Figure 5-9.

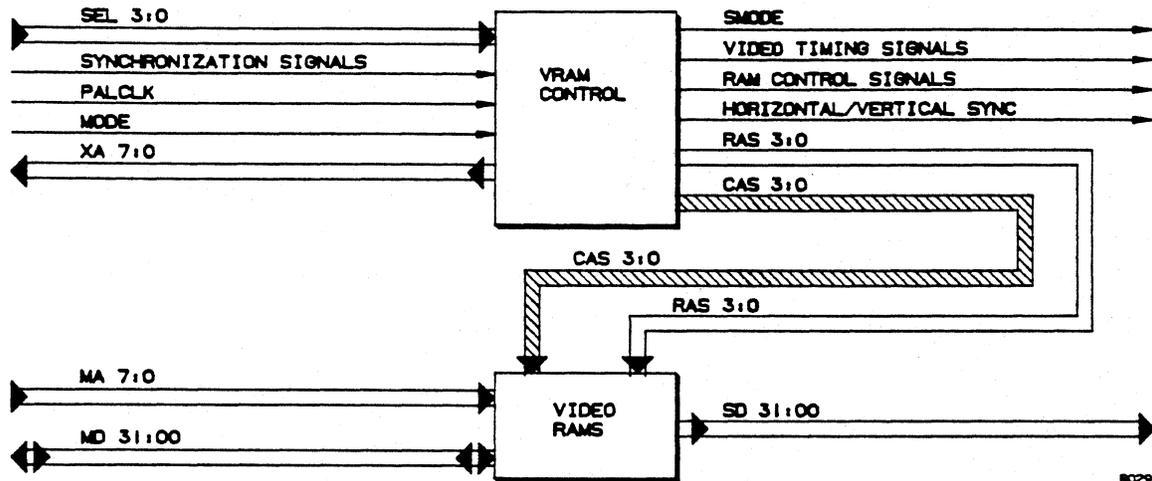


Figure 5-9. Video RAM Subsection Block Diagram

The video RAM banks store all frame buffer data. The banks are addressed through the eight bits of the MA multiplexed address bus. Each RAM chip is addressed by the MA bus. The selection of the appropriate bank is done by the video RAM control circuit with instructions from the video control subsection. Frame buffer data is loaded into the RAMs over the MD multiplexed data bus and is transferred to the video driver area shift registers via the 32-bit SD serial data bus. Each of the eight RAM chips holds four bits of data. The frame buffer responds to three sizes of Kbus RIO transactions:

1. Byte reads and writes which deal only with one of the four banks
2. Doublebyte reads and writes which deal with two of the four banks
3. Quadbyte reads and writes which deal with all four of the banks

The video RAM chips have two ports through which data can transfer: the four-bit parallel I/O port which can be accessed by the system CPU and the four-bit serial output port which drives the video circuitry. The use of two ports on each RAM chip permits the video driver half of the Monochrome Graphics subsection to operate independently without interfering with the CPU's need to access the video frame buffer. On command from the video RAM control block, a 1024-bit register in each RAM chip sends the data held in the chip's RAM array out the serial port. This can occur even though the system CPU may be preparing to write new data into the frame buffer. Since the serial ports of the RAM arrays are running much faster than the parallel ports, any update to the frame buffer from the CPU is reflected in the serial output almost immediately.

Control signals are issued to each RAM chip from the video RAM control block. These control signals include:

- Row address strobe (RAS) which indicates that the address on the MA address bus is valid for locating the row of the array in which data is stored
- Column address strobe (CAS) which indicates that the address on the MA address bus is valid for locating the column of the array in which data is stored
- Write enable (WE) signal which permits the RAM chip arrays to be loaded with data from the MD data bus
- Output enable (OE) signal which permits the RAM chip arrays to output data either to the MD data bus or to the SD serial data bus, depending on the state of RAS when OE asserts
- Serial enable (SE) signal which lets the RAM chip arrays output data to the serial ports
- Serial control (SC) input clock which, on the Monochrome Graphics subsection, ensures that the serial port doesn't output data until after a CPU write cycle is complete

The video RAM control block manages the process of moving data in and out of the video RAM chips. It generates the strobes for the row and column addresses, issues transfer addresses to the address multiplexers in the video control subsection, generates vertical and horizontal synchronization signals, and coordinates the timing of the transfer of data to the video drivers subsection. This block contains seven PALs which help coordinate the entire process. Three of the PALs are involved with the coordination of data coming into the block from the Kbus side of the circuit. The other four PALs are involved in the coordination of data being transferred out the monitor side of the circuit.

The three video RAM control block PALs that deal with the incoming data are the:

1. RAM cycle generator PAL
2. Transfer address counter PAL
3. Shift counter PAL

The RAM cycle generator PAL issues the WE and OE control signals to the RAMs. It also issues the RAS and CAS signals which are combined with the four SEL bank select bits to generate RAS and CAS signals for each chip in the RAM array. This PAL runs on an RBC clock to keep it in sync with the activities on the Kbus-side of the board.

The transfer address counter PAL and the shift counter PAL work together to issue the correct transfer address to the address multiplexers of the video control subsection. The counter PAL keeps track of the shift count. The transfer address PAL issues the addresses to the XA transfer address bus and has the capability of recalculating the timing depending on whether the Monochrome Graphics subsection is driving a low resolution monitor or a high resolution monitor. These two PALs must keep in sync with the video output circuits in order to ensure that the bits in RAM truly represent the bits displayed on the monitor screen. So, these two PALs are clocked at the same rate as the PALs involved with the video output signal with the SLOCLK clock.

One of the four PALs that coordinate the transfer of data out to the monitor is the shift/slow clock generator PAL. It performs the task of coordinating the timing of data transfers to the video shift registers. This PAL runs on the PALCLK timing signal from the video drivers subsection. It provides the SLOCLK clock for synchronizing the transfer of addresses to the video output circuits. It also signals the video driver shift registers when to start shifting the bits out.

The other three PALs involved in the video data transfer process generate horizontal and vertical synchronization signals for the video drivers. These PALs are the:

1. Horizontal sync and blank generator PAL
2. Vertical sync and blank generator PAL
3. Vertical line counter PAL

The horizontal sync and blank generator PAL issues the horizontal synchronization signal to the monitor. This PAL runs on the SLOCLK clock for timing purposes. When the cathode ray tube (CRT) raster generator beam reaches the end of a vertical line, it must retrace across the screen and begin another line. When it retraces, this PAL asserts a horizontal blank signal to the rest of the control circuit. This signal combines with the vertical blank signal to keep data from being sent out to the screen while the beam retraces.

The vertical sync and blank generator PAL issues the vertical synchronization and blanking signals to the monitor. It too runs on the SLOCLK clock for timing purposes. Along with the horizontal blank signal, when the CRT raster generator beam retraces the screen, this PAL asserts the vertical blanking signal. This PAL is assisted by the vertical line counter PAL.

The vertical line counter PAL keeps a count on the number of lines that the beam has passed through. With the count kept by this PAL, along with some vertical and horizontal incrementing flags, the video RAM control circuit can keep track of the location of the CRT electron beam in the display monitor.

### 5.7.4 Video Drivers Subsection

The video drivers subsection takes the 32-bit serial-input data from the video RAM subsection and develops a single-bit serial-output data signal through internal shift registers. The data that passes into this stage gets converted from transistor-to-transistor logic (TTL) to emitter coupled logic (ECL). The output then becomes a differential-ECL signal which drives the video input of the display monitor. Horizontal and vertical synchronization pulses are supplied to the monitor as separate signals.

The video drivers subsection is split into five blocks: a shift register block, a TTL-to-ECL converter/shift register block, a differential signal/clock generator block, a clock divider circuit, and a circuit for driving the horizontal and vertical synchronization signals. See Figure 5-10.

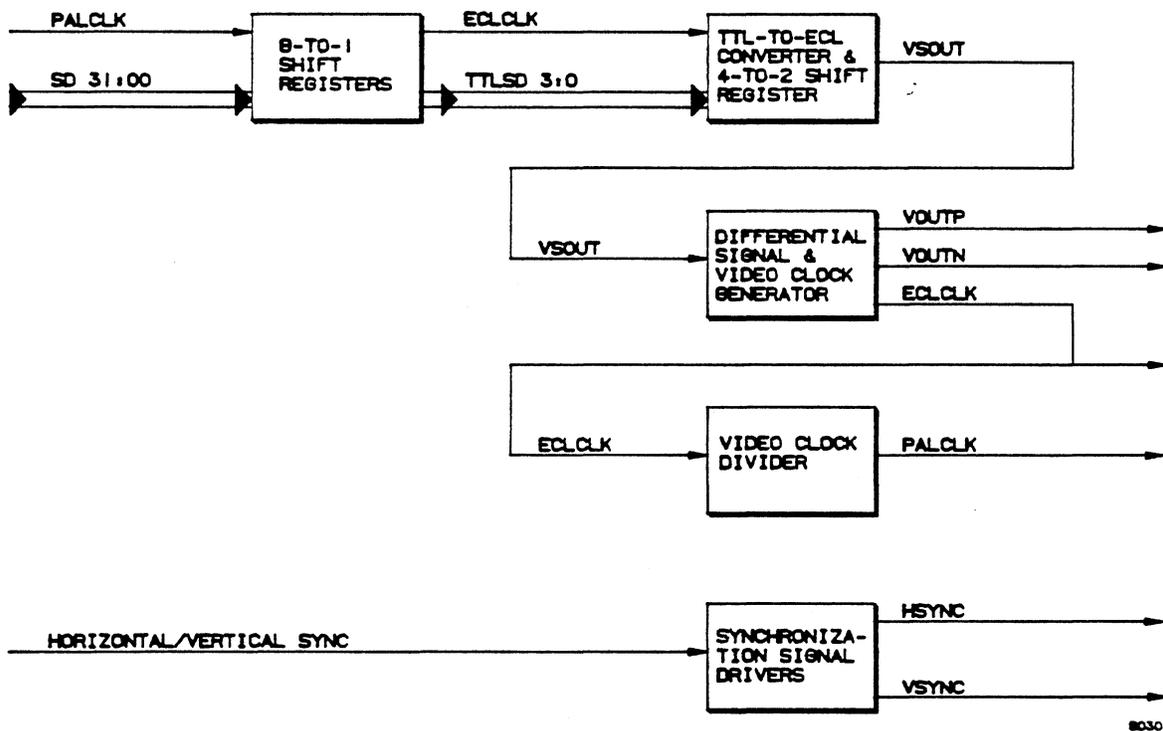


Figure 5-10. Video Drivers Subsection Block Diagram

The first set of shift registers in the video signal flow take the 32 serial input lines, SD [31:00], from the video RAMs and shifts them down to four serial output lines, TTLSD [3:0]. There are four eight-to-one shift registers involved in the process. Each shift register receives serial data from every fourth line of the SD [31:00] bus. The clock used for the shift rate is PALCLK which is running at  $\frac{1}{4}$ th the speed of the clock driving the differential ECL video signal out to the monitor. So, the serial signals issued from this block are running at  $\frac{1}{4}$ th the speed of the final video output.

The next stage of the video signal is the a TTL-to-ECL converter/shift register block. This block changes the voltage levels of the four shift register serial lines from TTL levels to positive-biased ECL levels. These four ECL signals then enter a four-to-one shift register to further concentrate the serial video signal onto one line. Since the four serial lines that enter this block are transferring serial data at  $\frac{1}{4}$ th the speed of the final output, and the output of the four-to-one shift register must run at the final output speed, the shift register runs on ECLCLK, the final

video transfer rate clock. The video signal is now ready for its final conversion.

In the final stage of the video signal preparation process, the video serial output signal, VSOUT, passes into the differential signal/clock generator block. This block serves two functions:

1. It generates the video clock upon which all chips involved in the serial video signal transfer process are clocked
2. It takes the positive-based video serial ECL output signal from the four-to-one shift register circuit and converts that serial signal to true negative-based differential ECL output signals, VOUTN (-) and VOUTP (+).

The final output video clock can run at 100 MHz or 200 MHz depending on whether the monitor being used has a low-resolution or high-resolution screen. A 200 MHz crystal creates the original clock signal. The serial mode signal, SMODE, from the video RAM control block, controls the video clock rate. SMODE determines whether the 200 MHz clock passes on to the other serial data transfer circuits or gets divided in half before it passes on to the other serial data transfer circuits.

The video clock that is generated in the clock generator block then enters the clock divider circuit. This block takes the video clock and divides it into the  $\frac{1}{4}$  clock and  $\frac{1}{8}$  clock needed for timing the serial video lines through the shift registers. The final output video clock can run at 100 MHz or 200 MHz depending on whether the monitor being used has a low-resolution or high-resolution screen. With a low resolution screen the 100 MHz clock is used, resulting in a 25 MHz ECLCLK and a 3.125 MHz PALCLK. With a high resolution screen the 200 MHz clock is used, resulting in a 50 MHz ECLCLK and a 6.25 MHz PALCLK.

The last block in this subsection is the circuit for driving the horizontal and vertical synchronization signals. This small circuit receives the two synchronization signals from the video RAM control block and drives them out to the monitor.

The output of the Monochrome Graphics subsection's video drivers block is the differential ECL video signal and horizontal and vertical synchronization signals.

## Section 6: 16 Mbyte Memory Board Operation

### 6.1 Introduction

This Memory Board provides 16 Mbytes of memory on a single Kbus board. The 16 Mbyte Memory Board interfaces directly to the Kbus backplane and occupies one slot. Each Memory Board occupies 16 Mbytes of Kbus address space. Up to five Memory Boards may be configured into a system. This configuration provides up to 80 Mbytes of memory on the Kbus.

The Memory Board is four-way interleaved and has four memory banks. During a Kbus transaction, each of the cache lines in a cache block accesses a separate memory bank. The Memory Board uses one 256 Kbyte by 4 CMOS dynamic RAMs. General features of the Kbus Memory Board include:

- 16 Mbytes of dynamic RAM
- 72 bit (64 data and eight check bits) Kbus data interface
- Software settable base address
- Software settable enables for memory reads and writes

The Memory Board is split into two basic sections: the Kbus interface/memory control section and the RAM arrays section (see Figure 6-1).

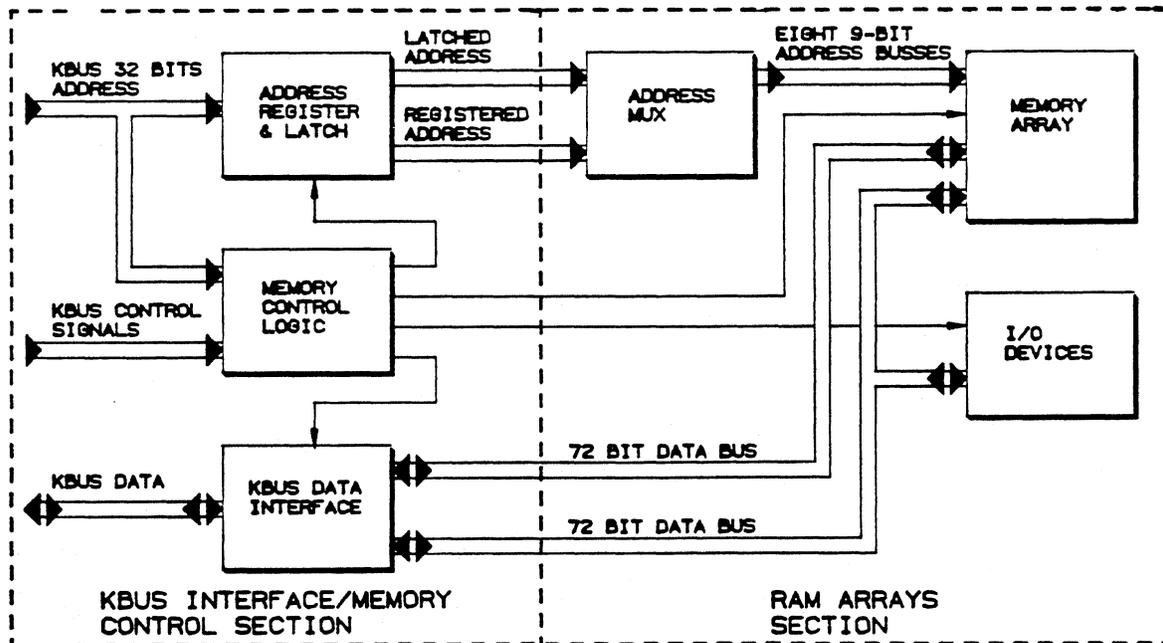


Figure 6-1. Memory Board Block Diagram

The Kbus interface/memory control section is further divided into three subsections: the address register and latch, the memory control, and the Kbus data interface subsection. The RAM arrays section, too, is divided into three subsections: an address multiplexer, an I/O device subsection, and the RAM memory arrays.

The memory is located on even 16 megabyte address boundaries on the Kbus. Kbus address bits 31:24 determine whether an address is located on a particular Memory Board. Kbus address bits 23:05 determine the address of a cache block within a Memory Board. The Memory Board only supports 32 byte cache block memory transactions. The memory is four way interleaved. The first cache line within a cache block accesses bank 0, the second cache line accesses bank 1, the third cache line accesses bank 2, and the fourth cache line accesses bank 3. In addition the memory and each bank is divided into a high and low halves. Kbus address bit 23 determines whether an address is in the high half (KA 23 = 1) or the low half (KA 23 = 0) of a bank. The Memory Board is 72 bits wide. Sixty-four are the normal Kbus data bits and eight are ECC check bits. The base address of a Memory Board is settable by writing to the board's memory location register.

## 6.2 Address Register and Latch Subsection

During any memory transfer, the Kbus address doesn't remain valid during the entire transaction. It is important, though, for the Memory Board to store the lower 24 bits of the Kbus address, KADDR 23:00, in order to complete the transaction. The Kbus address register and latch section of the Kbus Memory Board performs the function of latching and holding these bits during memory transactions. The address register and address latch each consists of three octal D-type flip-flop latches (see Figure 6-2).

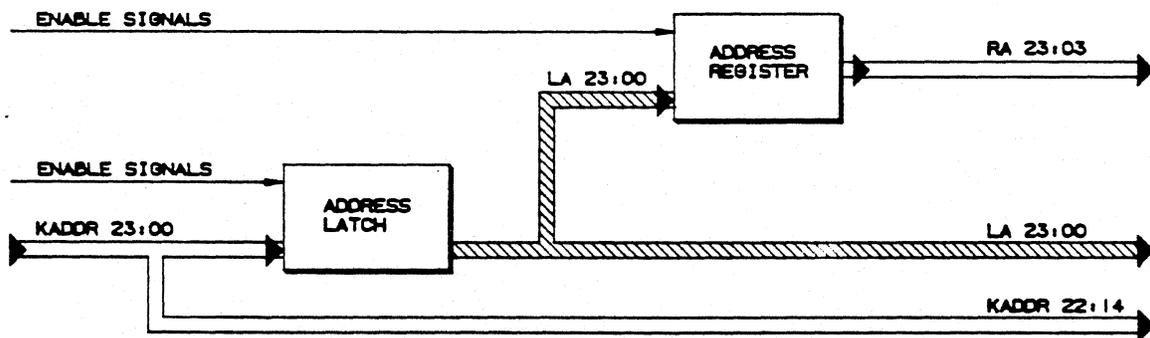


Figure 6-2. Address Register and Latch Subsection Block Diagram

The three octal D-type flip-flops of the address latch load the 24 Kbus address bits when the enable signal (EN) asserts. EN is a buffered form of the Kbus address strobe, KASL. The address latch outputs latched address bus bits LA [23:00]. These bits then are placed at the inputs of the three octal D-type flip-flops of the address register. On the rising edge of the first clock after KASL deasserts, the LA [23:00] bits are clocked into the three octal D-type flip-flops that form the address register. ARENL enables the address register. The address register then outputs register address bus bits RA [23:00]. The latched address bus bits LA 23:05 provide the address for Banks 0 and 1 of the RAM arrays. The registered address bits RA 23:05 provide the address for RAM array Banks 2 and 3. The RA bits are also used during byte I/O transactions.

### 6.3 Memory Control Subsection

The memory control subsection determines when a Kbus transaction belongs to the Memory Board, decodes the transaction type, provides the necessary control signals to the memory banks sections of the Memory Board, controls the flow of data through the data buses within the Memory Board, and issues the handshake signals to the Kbus to complete the transaction (see Figure 6-3).

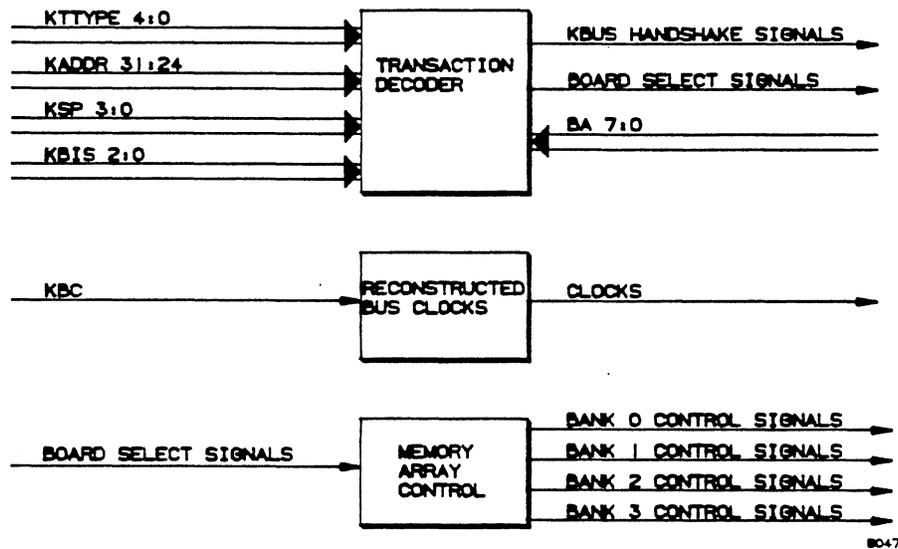


Figure 6-3. Memory Control Subsection Block Diagram

The memory control subsection consists of: a block that reconstructs bus clock, a transaction decoder, and a memory array control block that issues row and address control signals to the memory arrays. The reconstructed bus clock (RBC) block performs a function similar to the RBC circuits of the other Kbus boards. It derives board-level clocks from the Kbus clock, KBC.

The transaction decoder block determines whether the current Kbus transaction is one that deals with the Memory Board. It decodes the Kbus transaction type to see if it's the type of transaction that the board needs to be concerned about. If it is, then an eight-bit comparator in the transaction decoder block compares the address on the Kbus with the base address held in the memory location register. If the address falls within the 16 Mbyte boundary represented by the Memory Board, the transaction decoder block issues signals that reflect that the board has been selected. If the transaction is not a regular cache memory transaction but is a transaction in I/O space, another eight-bit comparator in the transaction decoder block checks to see if the transaction is for one of the on-board I/O devices. This comparator looks at the signal on the KBID bus, the KSP bus, and bits KADDR 27:24 of the Kbus address bus. If one of the on-board I/O devices is being addressed, the transaction decoder circuit issues the appropriate select signals.

The memory array control block is made up of control PALs and signal drivers. The control PALs coordinate the issuance of high and low row address strobes (RAS) and high and low column address strobes (CAS) for each of the four memory banks. The PALs of the memory array control block also issue high and low output enable (OE) signals for each of the four memory banks. Further, these PALs direct and coordinate the flow of data between the memory banks and the Kbus through control of high and low write (WRT) signals for each bank.

of memory. If the WRT signal is asserted, data is transferred into memory. If the WRT signal is not asserted, the Kbus transaction is a read transaction and data is from memory to the Kbus.

Each of the signals generated by the memory array control PALs must be further divided into an A, B, and C component. The signal drivers in this block serve to divide the PAL signals. For each bank of RAM, then, the signal drivers issue an A, B, and C components of the high and low RAS, CAS OE, and WRT signals.

#### **6.4 Kbus Data Interface Subsection**

The Kbus data interface subsection performs the task of transferring data and check bits between the internal data buses of the Memory Board and the Kbus. There are two internal 64-bit data buses (MDA 63:00 and MDB 63:00) and two internal eight-bit check bit buses (MCA 7:0 and MCB 7:0) on the Memory Board. These two sets of buses each interface with the single Kbus data and check bit buses. The need for two internal sets of buses arises from fact that the Kbus can move data faster than the memory chips can respond.

The "A" set of buses, MDA and MCA, transfer data to and from the Bank 0 and Bank 2. The 'B' set of buses, MDB and MCB, transfer data to and from the Bank 1, Bank 2, and the I/O devices (see Figure 6-4).

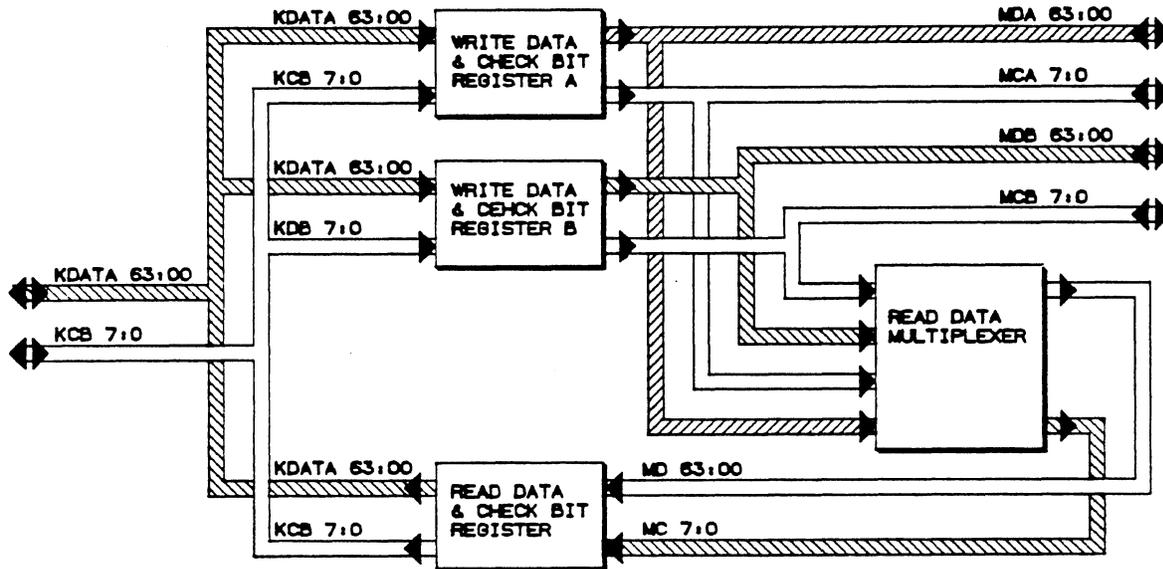


Figure 6-4. Kbus Data Interface Subsection Block Diagram

There are two write data and check bit registers, each which is comprised of nine octal D-type flip-flops. The first one feeds Kbus data to the MDA and MCA data and check bit buses. The second one feeds Kbus data to the MDB and MCB data and check bit buses. During a Kbus data write transaction, the two write data and check bit registers alternate between cache lines when moving the memory cache block to the RAM. These two data and check bit registers output data the internal data buses when the output enable signal MDAOEL and MDBOEL assert. MDAOEL and MDBOEL come from the memory control subsection.

A third data and check bit register, which is also comprised of nine octal D-type flip-flops, is used during data read transactions to drive the 64-bit Kbus data bus and the eight-bit Kbus check bit bus. This register ensures that data transferring to the Kbus is properly synchronized during memory and I/O read transactions. The OBOEL signal from the memory control subsection coordinates the outputs of the read data and check bit register. This register interconnects the Kbus KDATA and KCB buses with the 64-bit MD bus and the eight-bit MC bus. The MD and MC buses are the inputs to read data and check bit register. These two buses originate in the read data multiplexer.

The read data multiplexer block consists of 18 quadruple two-to-one multiplexers. These read data multiplexers combine the MDA/MCA and MDB/MCB sets of buses into a single 64-bit data and eight-bit check bit bus, MD [63:00] and MC [7:0]. The MDA and MCA buses are driven by Banks 0 and 2. The MDB and MCB buses are driven by Banks 1 and 3. Also, read data from the I/O devices appears on MDB bits 63:56. The selection between the MDA/MCA and MDB/MCB buses is controlled by the SELA0L and SELA1L signals. These signals are generated at the proper time, during a read by the memory control logic.

## 6.5 Address Multiplexers Subsection

The address multiplexer subsection provides row and column addresses required by the dynamic RAM arrays in the four memory banks at the proper time in the memory cycle. This subsection receives its inputs from the address register and latch and provides eight (two for each bank) nine-bit address buses to the memory array (see Figure 6-5).

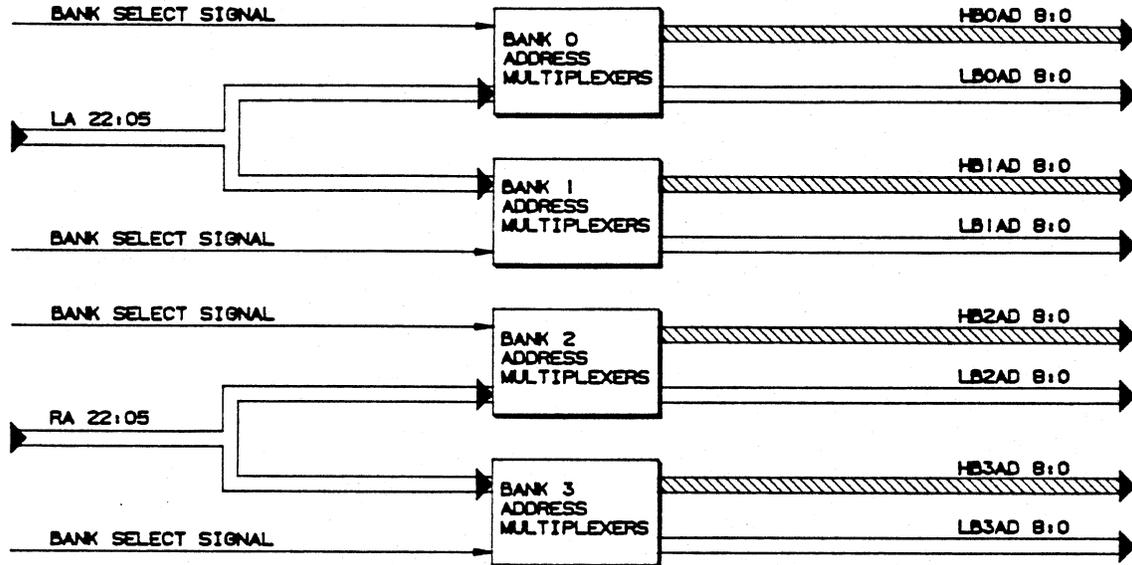


Figure 6-5. Address Multiplexers Subsection Block Diagram

The address multiplexer is comprised of 20, two-to-one multiplexers arranged in four groups. Each group provides two row and column addresses for one of the banks. There are row and column addresses for both the high and the low halves of each bank.

The multiplexers for Banks 0 and 1 generate addresses from the 18-bit latch address bus, LA 22:05. The multiplexers for Banks 2 and 3 generate addresses from the 18-bit register address bus, RA 22:05. Each multiplexer block multiplexes one of the 18 bit address buses down to a high and a low 9-bit address bus for issuance to each bank of the array. LA 22:05 is multiplexed down to four buses for Banks 0 and 1: LB0AD 8:0, HB0AD 8:0, LB1AD 8:0, and HB1AD 8:0. RA 22:05 is multiplexed down to four buses for Banks 2 and 3: LB2AD 8:0, HB2AD 8:0, LB3AD 8:0, and HB3AD 8:0.

The output of each multiplexer block is controlled by one of four address select signals: ASEL0L, ASEL1L, ASEL2L, and ASEL3L. When the select signals are high, bits 22:14 of either the LA or RA bus are output, sending the row address to the banks. When the select signals are low, bits 13:05 of either the LA or RA bus are output, sending the column address to the banks. Address select signals ASEL0L - ASEL3L are generated by the memory control subsection as a result of the high or low state of Kbus address bit 23. If Kbus address bit 23 is a zero, the address is located in the low section of the banks. If address bit 23 is a one, the address is located in the high section of the banks.

## 6.6 Memory Array Subsection

The memory array provides the actual 16 Mbytes of RAM storage. It is four way interleaved and has four memory banks. Each bank is accessed in sequence with each cache line in the cache block during a Kbus memory read or memory write transaction. Each bank is comprised of two sets of 36 256 Kbyte by 4 dynamic RAM chips. The data flows to and from the RAM arrays over the two 72-bit (64 data bits and eight check bits) buses. These buses are driven by the Kbus data interface subsection during write transactions and by the RAM arrays during read transactions.

Each of the four banks is designed the same and is divided into identical low and high halves. Each half contains 18 256 Kbyte by 4 bit dynamic RAM chips. Each chip receives a nine-bit address, four bits of the data bus, and control signals including: row address strobe, column address strobe, output enable, and write/read directional signal. Of the 18 bits in each half of a bank, 16 hold four bits each of the 64 bits in the data bus and two hold four bits each of the eight bits in the check bit bus.

Four pairs of nine-bit address buses issue row and column addresses to each RAM bank. There is a low bank address and a high bank address issued for each row strobe and for each column strobe. Bank 0 is addressed by LB0AD and LH0AD. Bank 1 is addressed by LB1AD and LH1AD. Bank 2 is addressed by LB2AD and LH2AD. Bank 3 is addressed by LB3AD and LH3AD.

Two pairs of internal data buses transfer data to and from the banks. Each pair of data buses consists of a 64-bit data bus and an eight-bit check bit bus. The data bus pairs are labeled 'A' and 'B'. Bank 0 and Bank 2 are served by the 'A' pair, MDA 63:00 and MCA 8:0. Bank 1 and Bank 3 are served by the 'B' pair, MDB 63:00 and MCB 8:0. These buses reside on separate data buses because the RAM's have a slow output disable time relative to the speed of the data transfers over the Kbus.

While one pair of banks is writing or reading data on the internal bus, the other bus can be transferring data to or from the Kbus (see Figure 6-6).

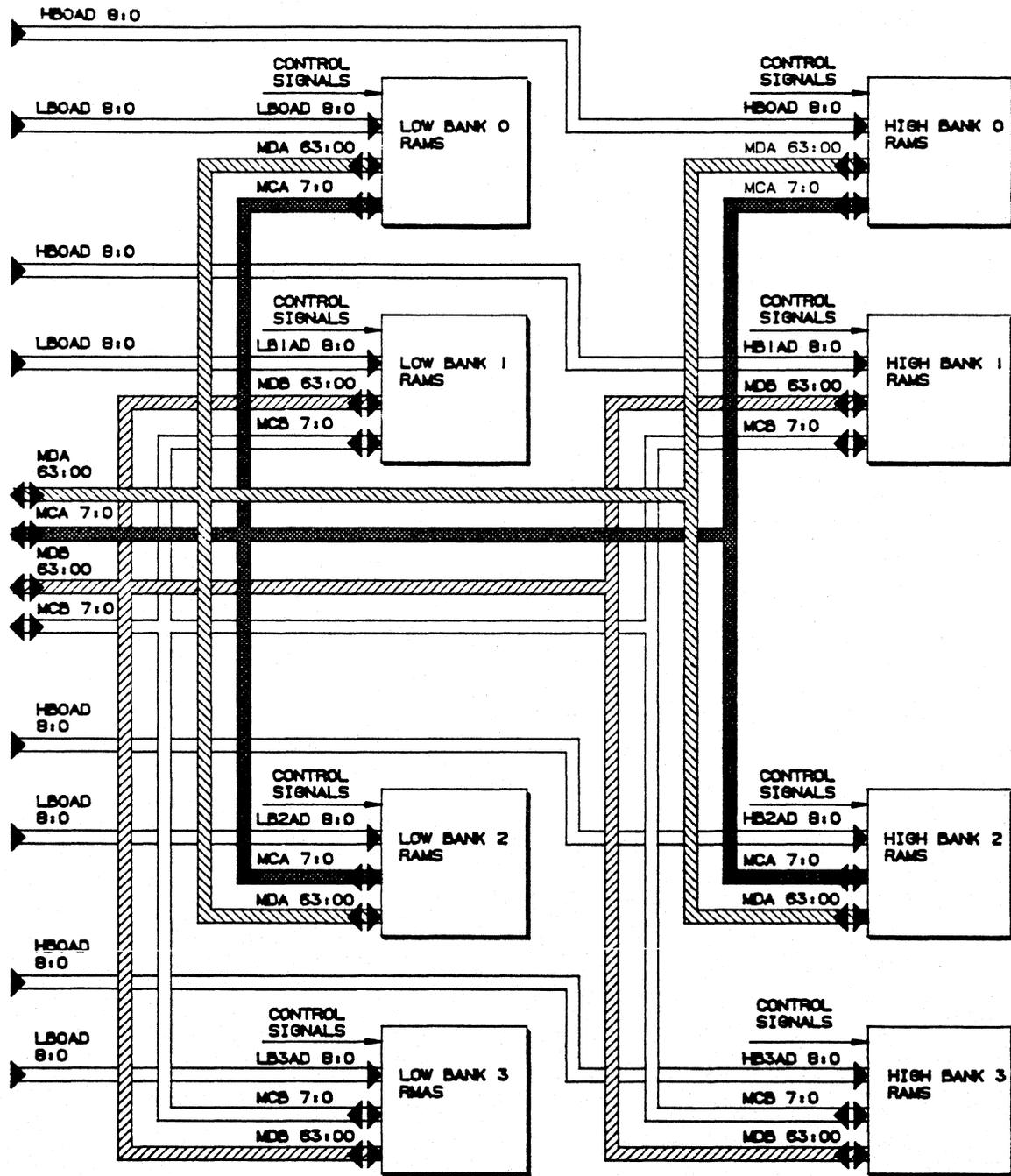


Figure 6-6. Memory Array Subsection Block Diagram

During any memory transaction, the banks are accessed sequentially from Bank 0 to Bank 3.

Accessing is achieved through separate sets of control signals that originate in the memory control subsection. Each high and low half of every RAM bank receives RAS, CAS, OE, and WRT signals. Due to the potential current drain, each signal is triplicated resulting in identical A, B, and C components. These three components control the activities of the RAM chips. The dynamics of the signals activities vary depending on the type of transaction being run.

### 6.7 I/O Devices Subsection

There are three devices that are used during the system configuration process to set up the Memory Boards within a system. These devices are accessed through RIO transactions on the Memory Board and are, therefore, I/O devices. These devices include the identification (ID) PROM, the memory location register, and the enable register.

These I/O devices can be read or written using byte RIO read or write transactions on the Kbus. The ID PROM holds information on the type and configuration of the Memory Board. The operating system software reads the information held in the ID PROM during system configuration. The memory location register establishes the base address in Kbus address space in which the board's 16 Mbytes of memory sits. The enable register permits the Memory Board to execute read and write transactions (see Figure 6-7).

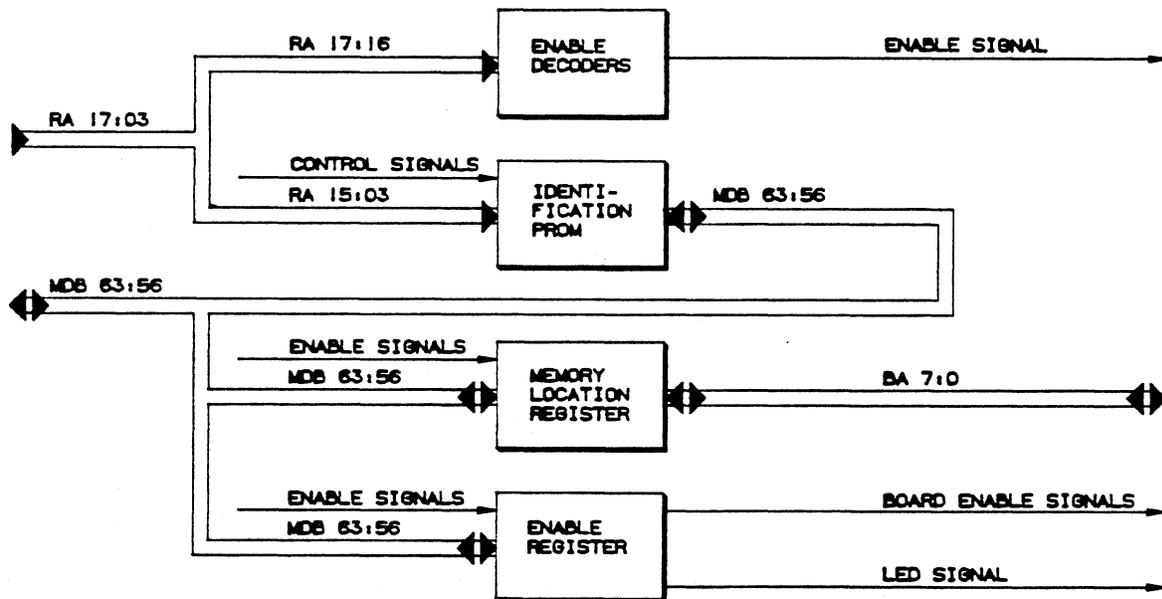


Figure 6-7. I/O Devices Subsection Block Diagram

The Memory Board responds to I/O transactions in I/O space 0001, when Kbus address bits, KADDR 27:24, match the backplane slot ID of the Memory Board. All I/O devices on the Memory Board are byte-wide and respond only to byte-read or byte-write transactions. During I/O read or write operations to any Memory Board only Kbus data bits 63:56 contain valid data. The I/O devices are addressed by register address bus bits RA 17:03 and are controlled by the enable decoders. Two two-to-four decoders decode the state of two RA bits (17:16) into read and write enables for the I/O devices. Once enabled, the I/O devices transfer data over the MDB bus.

The ID PROM block consists of an 8 Kbyte by 8 PROM and an octal D-type flip-flop. It is addressed through register address bus bits RA 15:03. The ID PROM occupies I/O addresses YX00000 through YX0FFFF (where Y = the board's slot ID and X = don't care). Addresses XY10000 through XY1FFFF are reserved for expansion of the ID PROM. At the present time only the first two locations in the ID PROM are defined.

The memory location register is comprised of an octal D-type flip-flop and an octal buffer/line driver. Each chip sits between the MDB data bus and the base address bus, BA 7:0, which provides the base address to the memory control subsection for memory transaction determination. The octal flip-flop latches the base address setting it for use by the memory control circuit. The base address latch can be read by the operating system software through the octal buffer. During a Kbus memory transaction the contents of the memory location register are compared to Kbus address bits 31:24 to determine if the pending Kbus transaction belongs to the Memory Board. The base address register occupies I/O address YX20000 through YX2FFFF (where Y = the board's slot ID and X = don't care).

The enable register is comprised of an octal D-type flip-flop and a quadruple buffer/line driver. This register allows read and write operations to each Memory Board to be independently enabled or disabled. In addition, a third bit in the register controls whether the board's error LED is on or off. Although the enable register is a byte-wide register, bits 0-2 are defined:

- Bit 0 corresponds to Kbus data bit 56 and controls when the Memory Board is enabled for read transactions. Performing an I/O byte write transaction to the enable register with Kbus data Bit 56 equaling 1 enables the Memory Board for memory read transactions.
- Bit 1 corresponds to Kbus data bit 57. Performing an I/O byte write transaction to the enable register of a Memory Board with Kbus data Bit 57 equaling 1 enables the Memory Board for memory write transactions.
- Bit 2 corresponds to Kbus data Bit 58. Bit 2 turns the Memory Board's error LED on and off. Performing an I/O byte write operation to the enable register with Kbus data bit 58 equaling 1 turns the error LED on. It remains on until the enable register is written with Kbus data Bit 58 equaling 0.

The enable register occupies I/O addresses YX3000 through YX3FFFF and responds to every eighth address within that region. The MBREN and MBWEN outputs of the enable register control when the memory control subsection is enabled to respond to Kbus memory read or write transactions.

## 6.8 Kbus Transactions

The Memory Board responds to three basic types of Kbus memory transactions: memory read, memory write, and refresh transactions. The Kbus transaction type bits, KTT 4:0, define which transaction is taking place on the Kbus. These bits are valid at the same time as the Kbus address.

If the upper byte of the Kbus address, KADDR 31:24, matches the contents of the Memory Board's memory location register and a memory read or memory write transaction type is detected, the Memory Board performs the operation indicated by the transaction type. The Memory Board disregards the Kbus address during refresh transactions. If the KCOWNL signal is asserted at the proper time following KASL the Memory Board converts the read transaction into a write transaction. This feature is called memory update on transfer (MUT).

## Section 7: 32 Mbyte Memory Board Operation

### 7.1 Introduction

The Memory Board provides 32 Mbytes of memory on a single Kbus board. The Memory Board interfaces directly to the Kbus backplane and occupies one slot. Each Memory Board occupies 32 Mbytes of Kbus address space. Up to five Memory Boards may be configured into a system. This configuration provides up to 160 Mbytes of memory.

The 32 Mbyte Memory Board is four-way interleaved and has four memory banks. During a Kbus transaction, each of the cache lines in a cache block accesses a separate memory bank. The 32 Mbyte Memory Board uses 1 Mbyte by 1 CMOS dynamic RAMs. General features of the Kbus Memory Board include:

- 32 Mbytes of dynamic RAM
- 72 bit (64 data and eight check bits) Kbus data interface
- Software settable base address
- Software settable enables for memory reads and writes

The Memory Board is split into two basic sections: the Kbus interface/memory control section and the RAM arrays section (see Figure 7-1).

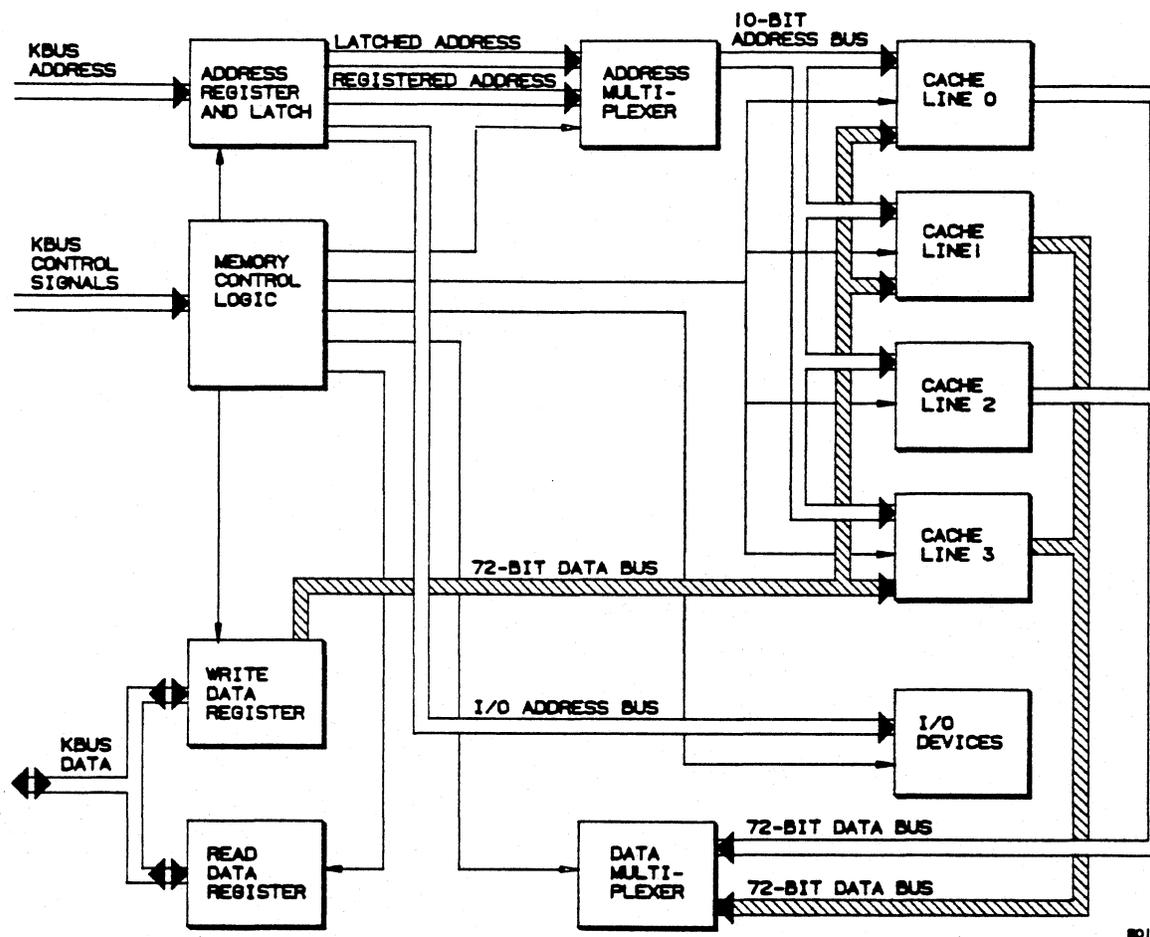


Figure 7-1. 32 Mbyte Memory Board Block Diagram

The Kbus interface/memory control section is further divided into three subsections: the address register and latch, the memory control, and the Kbus data interface subsection. The RAM arrays section, too, is divided into three subsections: an address multiplexer, an I/O device subsection, and the RAM memory arrays.

The memory is located on even 32 Mbyte address boundaries on the Kbus. Kbus address bits 31:25 determine whether an address is located on a particular Memory Board. Kbus address bits 24:05 Determine the address of a cache block within a Memory Board. The Memory Board only supports 32 byte cache block memory transactions. The memory is four way interleaved. The first cache line within a cache block accesses bank 0, the second cache line accesses bank 1, the third cache line accesses bank 2, and the fourth cache line accesses bank 3. The Memory Board is 72 bits wide. Sixty-four are the normal Kbus data bits and eight are ECC check bits. The base address of a Memory Board is settable by writing to the board's memory location register.

## 7.2 Address Register and Latch Subsection

During any memory transfer, the Kbus address doesn't remain valid during the entire transaction. It is important, though, for the Memory Board to store the lower 25 bits of the Kbus address, KADDR 24:00, in order to complete the transaction. The Kbus address register and latch section of the Kbus Memory Board performs the function of latching and holding these bits during memory transactions. The address register and address latch each consists of three octal D-type flip-flop latches (see figure 7-2).

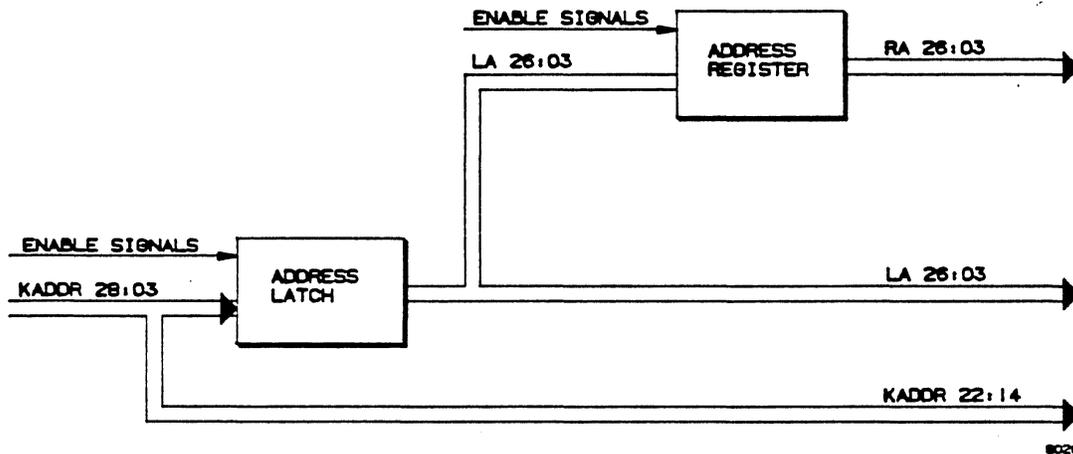


Figure 7-2. Address Register and Latch Subsection Block Diagram

The three octal D-type flip-flops of the address latch load the 25 Kbus address bits when the enable signal (EN) asserts. EN is a buffered form of the Kbus address strobe, KASL. The address latch outputs latched address bus bits LA 24:00. These bits then are placed at the inputs of the three octal D-type flip-flops of the address register. On the rising edge of the first clock after KASL deasserts, the LA 24:00 bits are clocked into the three octal D-type flip-flops that form the address register. ARENL enables the address register. The address register then outputs register address bus bits RA 24:00. The latched address bus bits LA 24:05 provide the row address and the column address for Bank 0. The latched address bus bits LA 24:15 provide the row address for Bank 1. The registered address bits RA 14:05 provide the column address for Bank 1 of the RAM arrays. The registered address bits RA 23:05 provide the address for RAM array Banks 2 and 3. The RA bits are also used during byte I/O transactions.

### 7.3 Memory Control Subsection

The memory control subsection determines when a Kbus transaction belongs to the Memory Board, decodes the transaction type, provides the necessary control signals to the memory bank sections of the Memory Board, controls the flow of data through the data busses within the Memory Board, and issues the handshake signals to the Kbus to complete the transaction (see Figure 7-3).

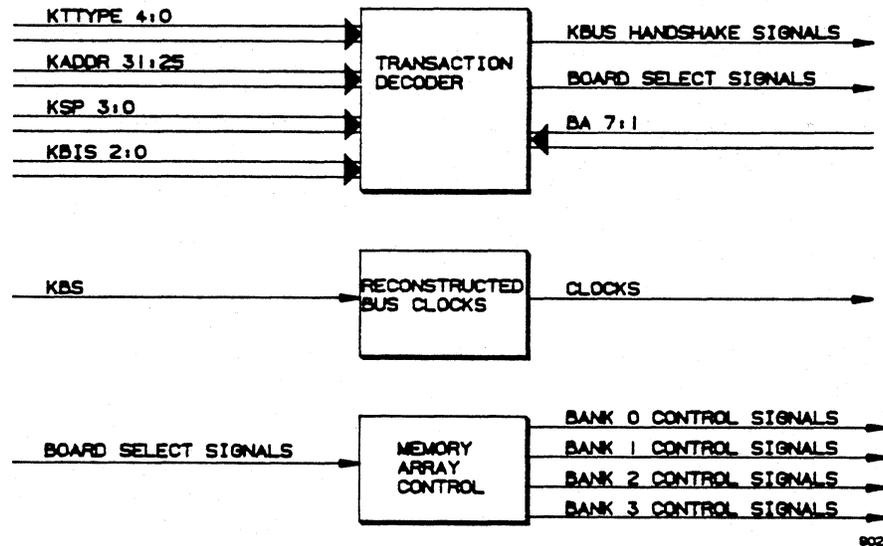


Figure 7-3. Memory Control Subsection Block Diagram

The memory control subsection consists of: a block that reconstructs bus clock, a transaction decoder, and a memory array control block that issues row and address control signals to the memory arrays. The reconstructed bus clock (RBC) block performs a functions similar to the RBC circuits of the other Kbus boards. It derives board-level clocks from the Kbus clock, KBC.

The transaction decoder block determines whether the current Kbus transaction is one that deals with the Memory Board. It decodes the Kbus transaction type to see if it's the type of transaction that the board need be concerned about. If it is, then an eight bit comparator in the transaction decoder block compares the address on the Kbus with the base address held in the memory location register. If the address falls within the 32 Mbyte boundary represented by the Memory Board, the transaction decoder block issues signals that reflect that the board has been selected. If the transaction is not a regular cache memory transaction but is a transaction in I/O space, another eight bit comparator in the transaction decoder block checks to see if the transaction is for one of the on-board I/O devices. This comparator looks at the signal on the KBID bus, the KSP bus, and bits KADDR 31:25 of the Kbus address bus. If one of the on-board I/O devices is being addressed, the transaction decoder circuit issues the appropriate select signals.

The memory array control block is made up of control PALs and signal drivers. The control PALs coordinate the issuance of high and low row address strobes (RAS) and high and low column address strobes (CAS) for each of the four memory banks. Further, these PALS direct and coordinate the flow of data between the memory banks and the Kbus through control of high and low write (WRT) signals for each bank of memory. If the WRT signal is asserted, data is transferred into memory. If the WRT signal is not asserted, the Kbus transaction is a read

transaction and data is from memory to the Kbus.

Each of the signals generated by the memory array control PALs must be further divided into an A0:2, B0:2, C0:2, and D0:2 components. The signal drivers in this block serve to divide the PAL signals. For each bank of RAM, then, the signal drivers issue an A0:2, B0:2, C0:2, and D0:2 components of the high and low RAS, CAS, and WRT signals.

#### 7.4 Kbus Data Interface Subsection

The Kbus data interface subsection performs the task of transferring data and check bits between the internal data busses of the Memory Board and the Kbus. There are two internal 64-bit data buses (DIN 63:00, DOUTA 63:00, and DOUTB 63:00) and three internal eight-bit check bit busses (DIN 71:64, DOUTA 71:64, and DOUTB 71:64) on the Memory Board. These three sets of busses each interface with the single Kbus data and check bit busses. The need for three internal sets of busses arises from fact that the Kbus can move data faster than the memory chips can respond. The DOUTA set of busses, MDA and MCA, transfer data to and from the Bank 0 and Bank 2. The DOUTB set of busses, MDB and MCB, transfer data to and from the Bank 1, Bank 2, and the I/O devices (see Figure 7-4).

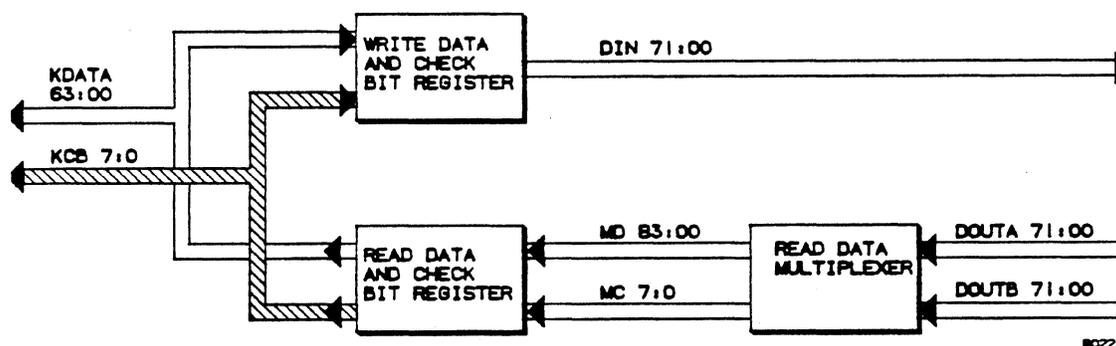


Figure 7-4. Kbus Data Interface Subsection Block Diagram

There is one write data and check bit register, which is comprised of nine octal D-type flip-flops. The write data and check bit register feeds Kbus data to DIN data and check bit bus. This register outputs data to the internal data buses when the output enable signal MDAOEL and MDBOEL assert. MDAOEL and MDBOEL come from the memory control subsection.

A second data and check bit register, which is also comprised of nine octal D-type flip-flops, is used during data read transactions to drive the 64-bit Kbus data bus and the eight-bit Kbus check bit bus. This register ensures that data transferring to the Kbus is properly synchronized during memory and I/O read transactions. The OBOEL signal from the memory control subsection coordinates the outputs of the read data and check bit register. This register interconnects the Kbus KDATA and KCB busses with the 64-bit MD bus and the eight-bit MC bus. The MD and MC busses are the inputs to read data and check bit register. These two busses originate in the read data multiplexer.

The read data multiplexer block consists of 18 quadruple two-to-one multiplexers. These read data multiplexers combine the DOUTA and DOUTB sets of busses into a single 64-bit data and eight-bit check bit bus, MD 63:00 and MC 7:0. The DOUTA bus is driven by Banks 0 and 2. The DOUTB bus is driven by Banks 1 and 3. Also, read data from the I/O devices appears on DOUTB bits 63:56. The selection between the DOUTA and DOUTB buses is controlled by the

SELA0L and SELA1L signals. These signals are generated at the proper time, during a read by the memory control logic.

## 7.5 Address Multiplexers Subsection

The address multiplexer subsection provides row and column addresses required by the dynamic RAM arrays in the four memory banks at the proper time in the memory cycle. This subsection receives its inputs from the address register and latch and provides eight (two for each bank) eight-bit address busses to the memory array (see Figure 7-5).

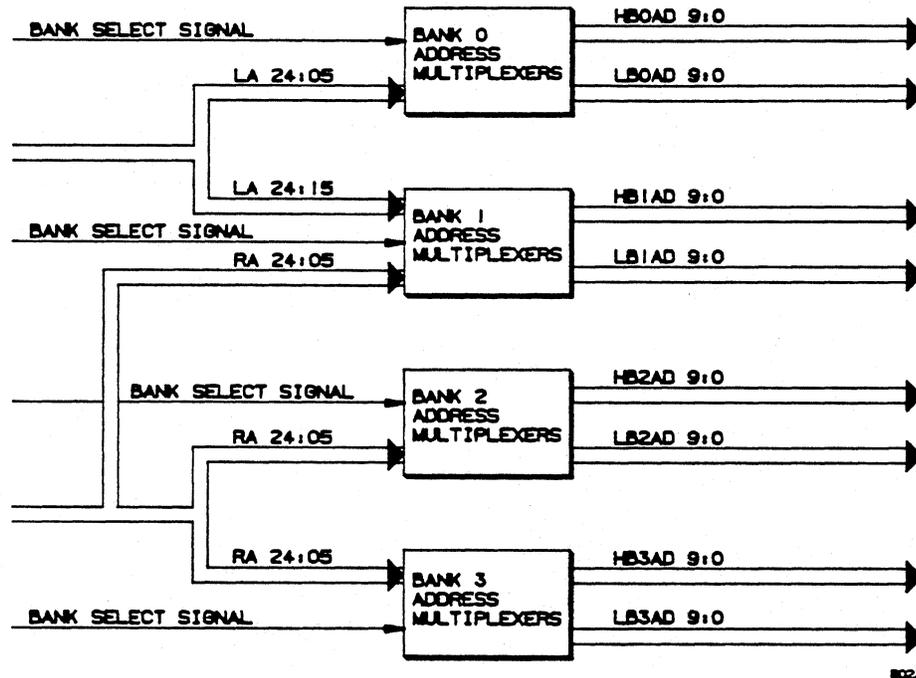


Figure 7-5. Address Multiplexers Subsection Block Diagram

The address multiplexer is comprised of 20, two-to-one multiplexers arranged in four groups. Each group provides two row and column addresses for one of the banks.

The multiplexers for Bank 0 generate addresses from the 20-bit latch address bus, LA 22:05. The multiplexers for Bank 1 generate addresses from the 1-bit latch address bus, LA24:15 and the 10-bit register address bus RA 14:05. The multiplexers for Banks 2 and 3 generate addresses from the 18-bit register address bus, RA 22:05. Each multiplexer block multiplexes one of the 20-bit address busses down to a high and a low 9-bit address bus for issuance to each bank of the array. LA24:15 is multiplexed down to two buses for Bank 0, LB0AD9:0 and HB0AD9:0. LA24:15 and RA14:05 are multiplexed down to two buses for Bank 1, LB1AD9:0 and HB1AD9:0. RA 22:05 is multiplexed down to four busses for Banks 2 and 3: LB2AD 8:0, HB2AD 8:0, LB3AD 8:0, and HB3AD 8:0.

The output of each multiplexer block is controlled by one of four address select signals: ASEL0L, ASEL1L, ASEL2L, and ASEL3L. When the select signals are high, bits 22:14 of either the LA or RA bus are output, sending the row address to the banks. When the select signals are low, bits 13:05 of either the LA or RA bus are output, sending the column address to the banks. Address select signals ASEL0L - ASEL3L are generated by the memory control subsection as a

result of the high or low state of Kbus address bit 23. If Kbus address bit 23 is a zero, the address is located in the low section of the banks. If address bit 23 is a one, the address is located in the high section of the banks.

## 7.6 Memory Array Subsection

The memory array provides the actual 32 Mbytes of RAM storage. It is four way interleaved and has four memory banks. Each bank is accessed in sequence with each cache line in the cache block during a Kbus memory read Or memory write transaction. Each bank is comprised of two 72 1 Mbyte by 1 dynamic RAM chips. The data flows to the RAM arrays over one 72-bit (64 data bits and eight check bits) bus, DIN 71:00. The data flows from the RAM arrays over two 72-bit (64 data, 8 checkbit) buses, DOUTA 71:00 and DoutB 71:00. These busses are driven by the Kbus data interface subsection during write transactions and by the RAM arrays during read transactions.

Each of the four banks is designed the same. Each bank contains 72 one Mbyte by one dynamic RAM chips. Each chip receives a 10-bit address, a bit data input, a bit data output, and control signals including: row address strobe, column address strobe, and write/read directional signal.

Four pairs of 10-bit address buses issue row and column addresses to each RAM bank. There is a bank address issued for each row strobe and for each column strobe.

Internal data busses transfer data to and from the banks. Each data bus consists of a 64-bit data bus and an eight-bit check bit bus. The data buses are labeled 'A' and 'B'. Bank 0 and Bank 2 are served by the 'A' bus, DOUTA 71:00. Bank 1 and Bank 3 are served by the 'B' bus, DOUTB 71:00. These buses reside on separate data busses because the RAMs have a slow output disable time relative to the speed of the data transfers over the Kbus. While one pair of banks is writing or reading data on the internal bus, the other bus can be transferring data to or from the Kbus (see Figure 7-6).

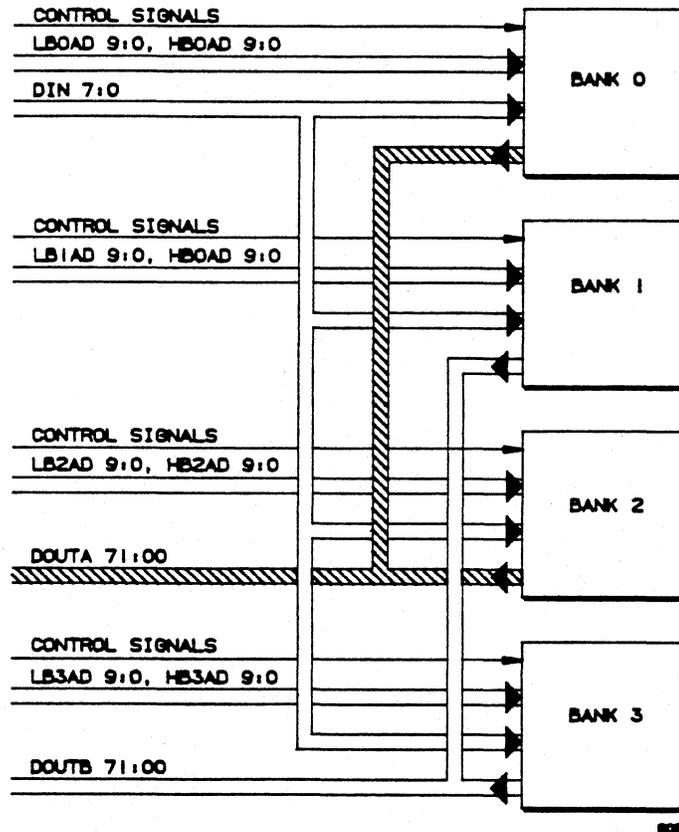


Figure 7-6. Memory Array Subsection Block Diagram

During any memory transaction, the banks are accessed sequentially from Bank 0 to Bank 3. Accessing is achieved through separate sets of control signals that originate in the memory control subsection. Each high and low half of every RAM bank receives RAS, CAS, and WRT signals. Due to the potential current drain, each signal is divided resulting in identical A2:0, B2:0, C2:0, and D2:0 components. These 12 components control the activities of the RAM chips. The dynamics of the signals activities vary depending on the type of transaction being run. Section 6.8 provides more details on the signal dynamics.

## 7.7 I/O Devices Subsection

There are three devices that are used during the system configuration process to set up the Memory Boards within a system. These devices are accessed through RIO transactions on the Memory Board and are, therefore, I/O devices. These devices include the identification (ID) PROM, the memory location register, and the enable register.

These I/O devices can be read or written using byte RIO read or write transactions on the Kbus. The ID PROM holds information on the type and configuration of the Memory Board. The operating system software reads the information held in the ID PROM during system configuration. The memory location register establishes the base address in Kbus address space in which the board's 32 Mbytes of memory sits. The enable register permits the Memory Board to execute read and write transactions (see Figure 7-7).

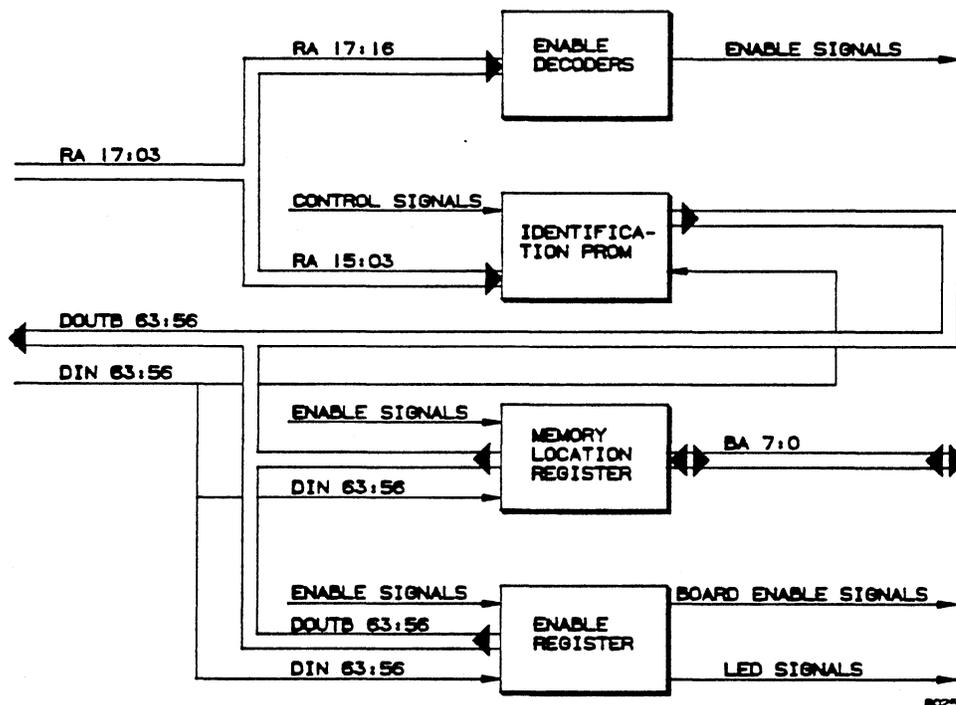


Figure 7-7. I/O Devices Subsection Block Diagram

The Memory Board responds to I/O transactions in I/O space 0001, when Kbus address bits, KADDR 27:24, match the backplane slot ID of the Memory Board. All I/O devices on the Memory Board are byte-wide and respond only to byte-read or byte-write transactions. During I/O read or write operations to any Memory Board only Kbus data bits 63:56 contain valid data. The I/O devices are addressed by register address bus bits RA 17:03 and are controlled by the enable decoders. Two two-to-four decoders decode the state of two RA bits (17:16) into read and write enables for the I/O devices. Once enabled, the I/O devices transfer data over the DOUTB bus.

The ID PROM block consists of an 16 Kbyte by 8 PROM and an octal D-type flip-flop. It is addressed through register address bus bits RA 15:03. The ID PROM occupies I/O addresses YX00000 through YX0FFFF (where Y = the board's slot ID and X = don't care). Addresses

XY10000 through XY1FFFF are reserved for expansion of the ID PROM. At the present time only the first two locations in the ID PROM are defined.

The memory location register is comprised of an octal D-type flip-flop and an octal buffer/line driver. Each chip sits between the data buses (DIN and DOUTB) and the base address bus, BA 7:1, which provides the base address to the memory control subsection for memory transaction determination. The octal flip-flop latches the base address setting it for use by the memory control circuit. The base address latch can be read by the operating system software through the octal buffer. During a Kbus memory transaction the contents of the memory location register are compared to Kbus address bits 31:25 to determine if the pending Kbus transaction belongs to the Memory Board. The base address register occupies I/O address YX20000 through YX2FFFF (where Y = the board's slot ID and X = don't care). BAO is not used for comparing. This bit exists only for reasons of software compatibility.

The enable register is comprised of an octal D-type flip-flop and a quadruple buffer/line driver. This register allows read and write operations to each Memory Board to be independently enabled or disabled. In addition, a third bit in the register controls whether the board's error LED is on or off. Although the enable register is a byte-wide register, bits 0-2 are defined:

- Bit 0 corresponds to Kbus data bit 56 and controls when the Memory Board is enabled for read transactions. Performing an I/O byte write transaction to the enable register with Kbus data Bit 56 equaling 1 enables the Memory Board for memory read transactions.
- Bit 1 corresponds to Kbus data bit 57. Performing an I/O byte write transaction to the enable register of a Memory Board with Kbus data Bit 57 equaling 1 enables the Memory Board for memory write transactions.
- Bit 2 corresponds to Kbus data Bit 58. Bit 2 turns the Memory Board's error LED on and off. Performing an I/O byte write operation to the enable register with Kbus data bit 58 equaling 1 turns the error LED on. It remains on until the enable register is written with Kbus data Bit 58 equaling 0.

The enable register occupies I/O addresses YX3000 through YX3FFFF and responds to every eighth address within that region. The MBREN and MBWEN outputs of the enable register control when the memory control subsection is enabled to respond to Kbus memory read or write transactions.

## Section 8: CG 40 Color Graphics Board

### 8.1 Introduction

The Solbourne CG 40 Color Graphics Board is a simple color frame buffer with an eight-bit image plane and two overlay planes. The Color Graphics Board resides on the Kbus. These bit planes reside in the I/O space assigned to the CG 40. The CG 40 is controlled by a set of Control Registers which reside in its ID space.

General features of the CG 40 include:

- 1152 by 900 resolution
- 66 Hz frame rate
- One 8-bit image plane
- Two 8-bit overlay planes
- 24 bit color lookup table providing 16,777,216 different colors

A block diagram of the CG 40 is shown in Figure 8-1.

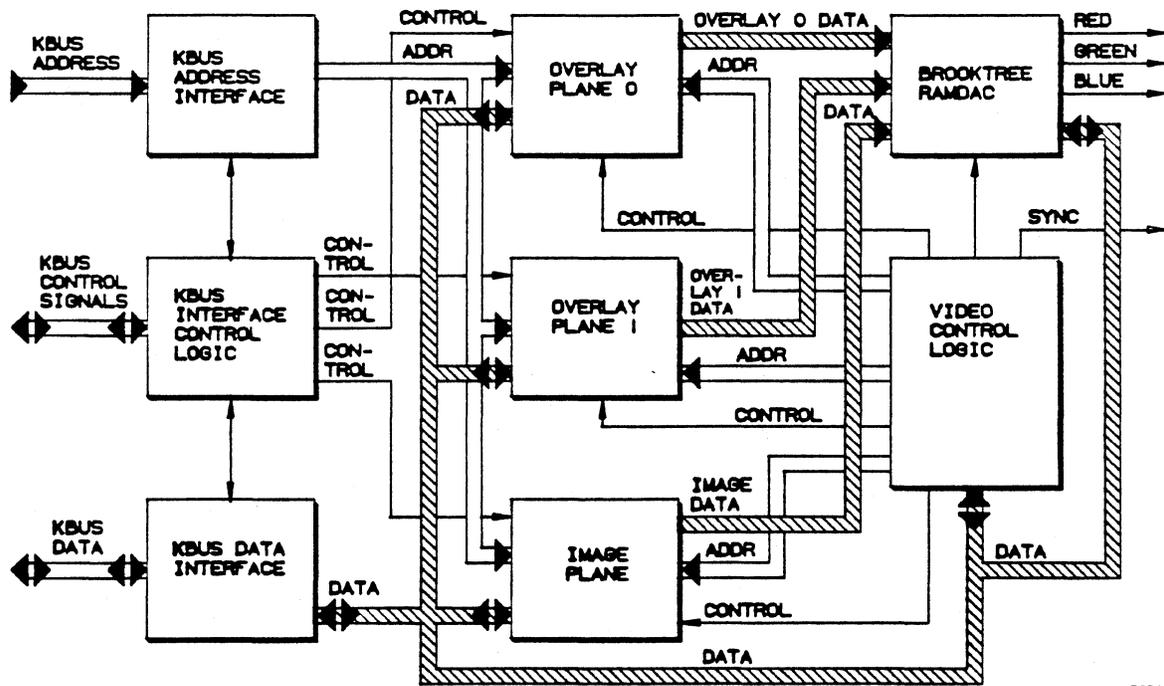


Figure 8-1. CG 40 Block Diagram

This following sections describe the specific features of the CG 40 from the system performance point-of-view.

## **8.2 Kbus Address Interface**

The Kbus address interface converts Kbus addresses to local addresses used within the CG 40. This allows any Kbus master to access the various devices on the CG 40.

## **8.3 Kbus Interface Control Logic**

The Kbus interface control logic provides the handshake signals required for Kbus masters to complete transactions with the CG 40. It also allows the CG 40 to generate Kbus interrupts on vertical sync.

## **8.4 Kbus Data Interface**

The Kbus data interface provides the interface between the Kbus and various devices on the CG 40.

## **8.5 Overlay Planes**

A Control Register, accessible to the software, is used to set the Kbus I/O space where the CG 40 resides. Within that space, the first overlay plane occupies addresses from 0-1FFFF. The second overlay plane occupies addresses from 20000-3FFFF. The function of these overlay planes is determined by software accessible Control Registers within the Brooktree Bt458 random access memory digital-to-analog converter (RAMDAC). The two overlay planes may be read or written using byte, double byte, or quad byte I/O read or I/O write transactions on Kbus.

## **8.6 Image Plane**

The eight bit image plane occupies addresses from 100000-1FFFFFF within the Kbus I/O space assigned to the CG 40. The image plane may be accessed using byte, double byte and quad byte I/O read and write transactions on Kbus. Pixels are output left to right, top to bottom, starting in the top left hand corner of the screen. Address 100000 is the address of the first pixel. ID PROM are allowed, but the write transaction has no effect.

## **8.7 Brooktree RAMDAC**

The Brooktree RAMDAC (Bt458) is designed specifically for high performance, high resolution color graphics.

The architecture of the Bt458 enables the display of up to 1280 by 1024 bitmapped color graphics (eight bits per pixels plus up to two bits of overlay information), minimizing the use of ECL interfacing, as most of the high speed (pixel clock) logic is contained on the chip. The multiple pixel ports and internal multiplexing enables TTL compatible interfacing (up to 32 MHz) to the frame buffer, while maintaining the 125 MHz video data rates required for sophisticated color graphics.

The Bt458 contains a 256 by 24 color lookup table with triple eight-bit video D/A converters. It generates RS-343-A compatible red, green, and blue video signals.

### 8.7.1 Bt458 Registers

The Brooktree Bt458 RAMDAC has an eight-bit wide MPU interface. It allows read/write access to the internal control registers and color/overlay palates. The color palette RAM and overlay registers are dual ported and may be updated without contention while the display is being refreshed. The MPU port of the Bt458 is accessed by performing byte I/O reads and writes to YX4 0000-YX4 0018<sup>†</sup>. For a detailed discussion of registers, refer to the Brooktree Product Databook.

### 8.7.2 Kbus Reads and Writes to the Bt458

The lists in this subsection discuss how the Kbus bits control the operation of the Brooktree Bt458.

- Writing to the color palette RAM:

1. Perform a byte write to Address YX4 0000 with KDATA[63:56] set to the Address within the Bt458 to be written. (This sets up the Address Register within the Bt458).
2. Perform a byte write to Address YX4 0008 with KDATA[63:56] set to the RED value to be written to the Address specified in step 1.
3. Perform a byte write to Address YX4 0008 with KDATA[63:56] set to the Green value to be written to the address specified in step 1.
4. Perform a byte write to Address YX4 0008 with KDATA[63:56] set to the Blue value to be written to the address specified in step 1.

On receiving the write of the blue value, the Bt458 updates the addressed location with the new data and increment the address register. If the next sequential location within the Bt458 is to be written, the address register need not be written and the three color values may be written by performing three more sequential write to Address YX4 0010.

- Reading the color palette RAM:

1. Perform a byte write to Address YX4 0000 with KDATA[63:56] set to the Address within the Bt458 to be read. (This sets up the Address Register within the Bt458).
2. Perform a byte read to address YX4 0008. The RED value at the location specified in step 1 is output on KDATA[63:56].
3. Perform a byte read to address YX4 0008. The Green value at the location specified in step 1 is output on KDATA[63:56].
4. Perform a byte read to address YX4 0008. The Blue value at the location specified in step 1 is output on KDATA[63:56].

After the Blue value is read, the Bt458 increments its address register. If the next location in the Color Palette RAM is to be read, the three color value reads may be performed without reloading the address register.

---

<sup>†</sup> In the CG 40 section, all address are given with a YX prefix. The Y is the slot ID bit of the Color Graphics Board and the X is a don't care bit.

- Writing the overlay registers:

1. Perform a byte write to Kbus Address YX4 0000 with KDATA[63:56] set to the Address of the Overlay Register to be written. The following table shows the byte and register information.

**Table 8-1. Writing to the Overlay Register**

Byte	Register
00	Overlay Register 0
01	Overlay Register 1
02	Overlay Register 2
03	Overlay Register 3

2. Perform a byte write to Address YX4 0018 with KDATA[63:56] set to the RED value to be written to the Overlay Register specified in step 1.
3. Perform a byte write to Address YX4 0018 with KDATA[63:56] set to the Green value to be written to the Overlay Register specified in step 1.
4. Perform a byte write to Address YX4 0018 with KDATA[63:56] set to the Blue value to be written to the Overlay Register specified in step 1.

On receiving the write of the blue value, the Bt458 updates the Overlay Register with the new data and increment the address register. If the next sequential Overlay Register within the Bt458 is to be written, the address register need not be written and the three color values may be written by performing three more sequential write to Address YX4 0018.

- Reading the overlay registers:

1. Perform a byte write to Kbus Address YX4 0000 with KDATA[63:56] set to the Address of the Overlay Register to be written. The following table shows the byte and register information.

**Table 8-2. Reading the Overlay Register**

Byte	Register
00	Overlay Register 0
01	Overlay Register 1
02	Overlay Register 2
03	Overlay Register 3

2. Perform a byte read to address YX4 0018. The RED value at the location specified in step 1 is output on KDATA[63:56].
3. Perform a byte read to address YX4 0018. The Green value at the location specified in step 1 is output on KDATA[63:56].
4. Perform a byte read to address YX4 0018. The Blue value at the location specified in step 1 is output on KDATA[63:56].

After the Blue value is read, the Bt458 increments its address register. If the next Overlay Register is to be read, the three color value reads may be performed without reloading the

address register.

- Writing the other registers:

1. Perform a byte write to Kbus Address YX4 0000 with KDATA[63:56] set to the Address of the Register to be written. The following table shows the byte and register information.

**Table 8-3. Writing to Other Registers**

Byte	Register
04	Read Mask Register
05	Blink Mask Register
06	Command Register
07	Test Register

2. Perform a byte write to Kbus Address YX4 0010 with KDATA[63:56] set to the value to be written to the selected register.

- Reading the other registers:

1. Perform a byte write to Kbus Address YX4 0000 with KDATA[63:56] set to the Address of the Register to be read. The following table shows the byte and register information.

**Table 8-4. REading Other Registers**

Byte	Register
04	Read Mask Register
05	Blink Mask Register
06	Command Register
07	Test Register

2. Perform a byte read to address YX4 0010. The contents of the register specified in step 1 is output on KDATA[63:56].

## 8.8 Video Control Logic

The video control logic provides the video timing. It provides the signals to the Brooktree RAMDAC that are required to generate the video. It also provides addresses to the video RAMs so the video information is output to the Brooktree RAMDAC in proper sequence to generate the image on the monitor.

## 8.9 Control Registers

The CG 40 contains a variety of Control Registers. These registers are read/write accessible to the software by means of byte I/O read and write transactions to the CG 40 ID space (space 0001). During these transactions only Kbus Data bits 63:56 contain valid data. All Control Registers are byte wide and reside on even eight byte boundaries (only every eighth address is defined). Not every bit within all of the Control Register is defined. Some Registers (ID PROM) are read only. Writing to a read only register causes no action to take place.

### 8.9.1 CG 40 Board Control Register

The Board Control Register controls the operation of the Color Graphics Board. It resides at Kbus address YX1 0000. The Board Control Register is a byte wide register that may be read or written by the software. Currently four of the eight bits in the register are defined.

- **BIT0** - This is Kbus data bit 56; the Video Enable (VIDEN). When this bit is a 1 the video timing chain on the CG 40 is enabled to operate and horizontal and vertical timing signals are generated for the monitor.
- **BIT1** - This is Kbus data bit 57; the Video Blank (VBLK). When this bit is set to a 0 the video display is blanked. This bit is to be used for blanking the monitor under software control rather than VIDEN, since the refresh of the VIDEO RAMS depends on the operation of the video timing chain.
- **BIT2** - This is Kbus data bit 58; the Interrupt Enable (INTEN). The CG 40 has the capability on generating a Kbus interrupt on each vertical sync. When set to a 1 this bit enables the interrupt to occur.
- **BIT7** - This is Kbus data bit 63; the light emitting diode (LED). This bit, when set to a 1, turns on the LED on the Color Graphics Board.

The Remainder of the bits in the Board Control Register are undefined. Setting or clearing them has no effect. The CG 40 de-asserts all of the bits in the Board Control Register when Kbus Reset is asserted.

### 8.9.2 Space Register

The Space Register which Kbus I/O space the CG 40 occupies. It resides at address YX2 0000 in its ID Space. Only bits 0-3 (Kbus data bits 59-56) are defined. Bit 0 is the LSB and bit 3 is the MSB of the I/O space number.

### 8.9.3 Video Control Registers

The Video Control Registers control the horizontal and vertical timing chains within the CG 40. They allow most video timing parameters to be set by the software. Each of the Video Control Registers is 8 bits wide and byte I/O read/write accessible by the software.

- **Horizontal Pixel Count Register** - This register is loaded, during software configuration, with the number of pixels in a horizontal line divided by eight. In the case of an 1152 pixel width display the value is 144 decimal (90 hex). The Horizontal Pixel Count Register resides at address YX3 0000.
- **Horizontal Front Porch Count Register** - This register controls the delay between the assertion of horizontal blanking and the assertion of horizontal sync. This time is called the horizontal front porch and varies with the type of monitor being used. An initial value based on our chosen monitor is five. This value may change slightly based on experimentation. The Horizontal Front Porch Count Register is at address YX3 0010.
- **Horizontal Sync Width Count Register** - This register controls the width of the horizontal sync pulse. Again, this parameter depends on the monitor type and may change slightly during bring up. A good initial value is 17 decimal (11 hex). This register resides at YX3 0018.
- **Horizontal Back Porch Width Count Register** - This register controls the delay between the de-assertion of horizontal sync and the de-assertion of horizontal blanking. This delay time

is called the Horizontal Back Porch. This is also a parameter which depends on the monitor type and could change during bring up. A good initial value is 25 decimal (19 hex). The Horizontal Back Porch Count Register is located at address YX3 0020.

- **Line Count Registers** - These two registers make up a 16 bit counter which is loaded with the number of horizontal lines to be displayed +256, minus two. In the case of a 900 line monitor the counter is loaded with a value of 1054. The 16 bit counter is divided up into two eight bit counters. The low byte of the counter is located at address YX3 0040. The high byte of the counter is at YX3 0048. The low byte counter should be loaded with 82 hex and the high byte counter should be loaded with four.
- **Vertical Front Porch Count Register** - This register controls the delay between the assertion of vertical blanking and the assertion of vertical sync. This delay time is called the vertical front porch. This parameter varies with the monitor type and may change slightly during bring up. The initial value should be three. The address of this register is YX3 0050.
- **Vertical Sync Width Count Register** - This register controls the width of the vertical sync pulse. This parameter also varies with monitor type and may change during bring up. A good initial value is two. This register is located at YX3 0058.
- **Vertical Back Porch Count Register** - This register controls the delay from the de-assertion of vertical sync to the de-assertion of vertical blanking. This delay is called the vertical back porch. This parameter varies with monitor type and may change during initial bring up. A good initial value is 31 decimal (1F hex). This register is located at address YX3 0060.

## 8.10 Interrupt Information and Interrupt Vector Registers

The IIR resides at Kbus Address YX5 0000 and the IVR resides at Kbus address YX6 0000. Together, these two registers comprise the Interrupt Transmit Register as described in the Kbus Specification Manual. Each register is read/write and, together, they provide the capability for directed or non directed interrupts. These registers are both byte wide and are accessed via byte I/O read and write transactions.

The CG 40 has the capability of generating a Kbus interrupt message on each vertical sync. Bits 0-7 of the Interrupt Vector Register correspond to KD56-KD63 and provide the interrupt vector portion of the interrupt message. Bits 0-7 of the Interrupt Information Register correspond to KD56-KD63. IIR Bit 7 is always 0, IIR Bit 6 is Directed (IIR6 = 1) or Undirected (IIR6 = 0). The remainder of the bits comprise the Destination ID (for Directed Interrupts) or Information Field (for Undirected Interrupts).

## 8.11 ID PROM

The ID PROM occupies addresses from YX0 0000 through YX0 FFFF. The ID PROM is an 8 Kbyte by 8 Kbit device. Only every eighth Kbus address is defined. Both byte I/O read and write transactions to the



## Section 9: CG 30 Color Graphics Board

### 9.1 Introduction

This section contains a description of the function and design of the Solbourne Computer CG 30 Color Graphics Board. The CG 30 Color Board replaces the earlier color frame buffer, CG 40, in the Solbourne Series4 and Series5 product lines.

### 9.2 Major Design Goals

The CG 30 Color Graphics Board consists of a 1 Mbyte RAM frame buffer plus number of hardware features that support increased performance when operating software graphics packages. The board provides approximately a 2-5X performance advantage over the Solbourne CG 40 Color Graphics Board.

The main functional portion of the board is Sun CG3-compatible, meaning that the CG 30 provides compatibility with Sun-4 applications that address the frame buffer directly, as well as to the Sun-4 version of Sunview, Pixrect, and NeWS drivers and applications. Any software application that operates under Sun-4 architecture and uses Sun Microsystem's CG 3 Board runs on the CG 30.

### 9.3 CG 30 Color Board System Connection

The board is physically connected to the Kbus backplane. Figure 9-1 shows the relationship of the CG 30 board to the Kbus and a CPU Board.

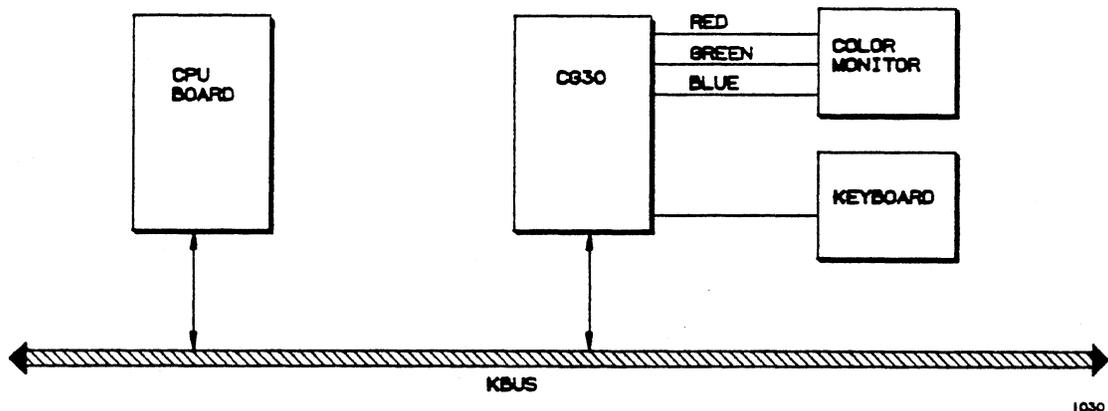


Figure 9-1. CG 30 System Connections

Features of the CG 30 Board include:

- 1152 by 900 resolution at 66 Hz refresh (1024 X 1024 resolution implemented but not currently supported)
- Eight raster operation (ROP) chips each capable of launching logical display cycles into the frame buffer for high-speed graphics effects.
- Alternate planar and pixel organization of the frame buffer for more graphic flexibility
- 24-bit color lookup table providing 256 colors at any one time out of a total of over 16,000,000 colors through remapping the table
- Hardware cursor for flicker-free scrolling
- Keyboard connection
- Multiple terminals (multiple boards) supported; not multiple keyboards

## 9.4 Functional Block Diagram

Figure 9-2 shows the primary functional blocks on the board and their internal and external interfaces.

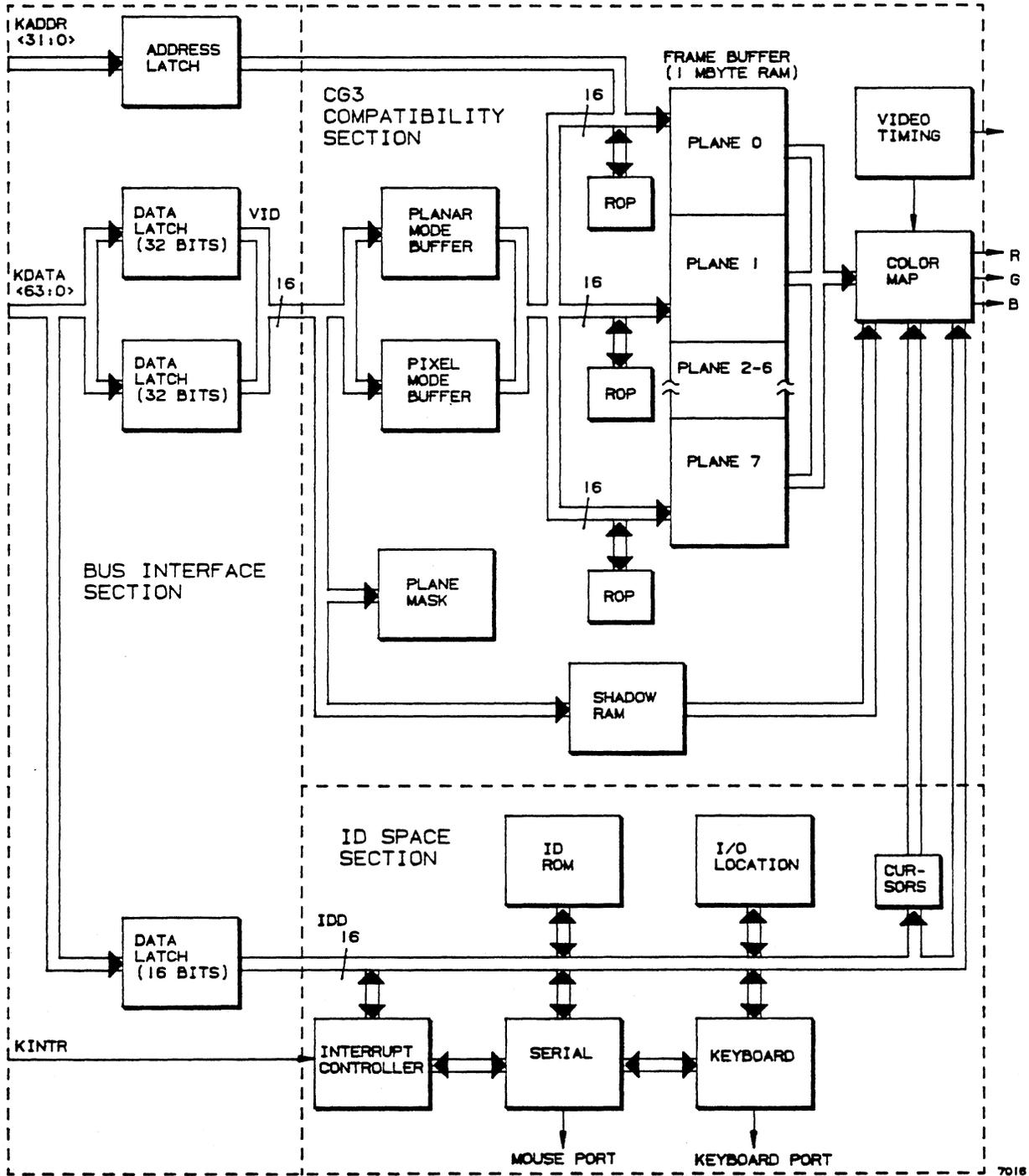


Figure 9-2. CG 30 Functional Block Diagram

## 9.5 Kbus Interface

The Kbus interface latches data and address lines from Kbus.

The latches in the interface section provide data to two main buses on the CG 30 board, a VID (Video Data) bus and an IDD (ID data) bus. Devices that respond to cycles in ID space are connected to the IDD bus. These include the ID ROM, the I/O Location register, the cursors, and the color map. The VID bus connects the CG3 compatibility section to the Kbus and the CPU. Control information, such as the movement of the cursor from one spot to another is provided through cycles in ID space and onto the IDD bus. Pixel data, the content of the screen image is run through cycles to I/O space and ends up on the VID bus on the color board.

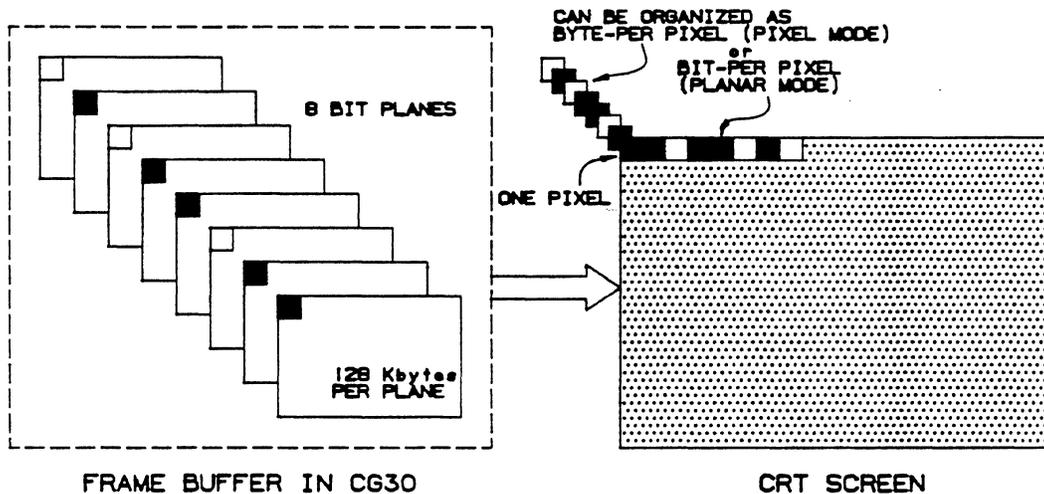
In addition to the KADDR and KDATA lines, the KINTR line is connected to the CG 30 board. Interrupt vectors are assigned to the serial (keyboard and mouse) ports so interrupt requests can be communicated to the external world from the keyboard and mouse. Also interrupt requests can come from vertical blanking.

## 9.6 CG 3 Compatibility Section

The CG3 compatibility block includes the frame buffer, the raster operation (ROP) chips, and the color map. The CG3 compatibility area is the primary functional area on the board, and is organized to be completely compatible with applications designed for the Sun Microsystems CG3 board. All RAM, buffer, and register locations are identical to the addresses on the CG 3 board.

## 9.6.1 The Frame Buffer

The frame buffer is the primary functional element of the CG 30. It is a RAM array consisting of a gross total of 1 Mbyte. Figure 9-3 shows conceptually two ways that a video image can be organized with the CG 30.



7014

Figure 9-3. Relation Between Video Image and Frame Buffer RAM

The video RAM in the CG 30 is organized in eight 128-Kbyte bit planes. The 1152 x 900 pixel video screen contains approximately 1 million pixels. Thus, there are approximately eight times as many bits in the frame buffer as there are pixels in the video image, as  $128K \times 8 \times 8 = 8 \text{ Mbit}$ .

The frame buffer is accessible with 8-, 16-, and 32-bit operations. In addition, three modes of access to the frame buffer and to the video image are possible:

1. Planar mode
2. Pixel mode
3. ROP Mode

In planar mode, the first eight bits of the frame buffer can map eight pixels on the video screen. In pixel mode, the first eight bits in the frame buffer map to the first pixel in the video image, eight bits deep. These two different organizational modes are for different purposes. An eight-bit-deep (or byte-per-pixel) access allows the pixel information to contain color codes the color map can translate to colors on the video screen. Planar (bit-per-pixel) mode allows eight times more pixels to be accessed per cycle than pixel mode.

The ROP mode is more complex and involves a CPU-like cycle to be run from hardware on the CG 30 Board. Data remains on the CG 30 Board, processed by the ROP chips, rather than being transferred by the bus. This allows much higher processing rates (i.e., 16 x eight bits rather than only eight bits)

ROP operations, also commonly called BITBLT (bit block translation), allow bit-mapped images to be combined and manipulated by logical operators. VTI rasterop chips are used for this application. Types of operations that can be performed locally on the board include movements

of graphics elements from one point on the screen to another and various ways of merging and combining of graphic elements

### 9.6.2 Color Map

The color map is connected to the 1 Mbyte frame buffer. The color map provides of 256 combinations of red, green, and blue color at any one time. As (in pixel mode) each pixel is represented by one byte, 256 ( $2^8$ ) different values could be associated with each pixel. The color map interprets each pixel as one of the available 256 colors, some combination of red, green, and blue. In this way the pixel data forms patterns of light, dark, and various colors, which make up the text, windows, and background you see on the screen.

The color map has two main ports: the one from the frame buffer and another that comes through ID space and which is used to change the color mappings. By writing to the color map address through ID space, the CPU can change the mappings of color combinations to numbers. For example, if (to simplify the formula somewhat) the number 119 represents 10 percent blue, 70 percent red, and 20 percent green in one mapping, it might represent 100 percent blue, 0 percent red, and 0 percent green in another mapping. In total, over 16,777,216 color combinations can be created through remapping the color map.

### 9.6.3 Shadow RAM

The shadow RAM exists in I/O address space and contains a copy of the color mappings in the color map. The shadow RAM may be copied to the hardware color map every vertical retrace. This allows smooth color changes as the colors change only during vertical blanking period. As stated above, the CPU can write directly to the color map to change mappings through ID space.

### 9.6.4 Video Output Signals

Output of the board consists of four elements: video syncing, red signals, green signals, and blue signals. The red, green, and blue control the pixel data that appears on the screen. Sync controls the timing of the monitor. Video timing consists of horizontal and vertical timing. The electron beam in the monitor requires just under 1000 horizontal sweeps to make one vertical sweep; 66 vertical sweeps occur each second. Table 9-1 shows the details of vertical and horizontal timing.

Video timing is the output of the video sync circuit, which is tied to the green output. The sync port extending through the cover plate of the board is not used in the current implementation. The following additional video timing specifications characterize the CG 30.

**Table 9-1. Video Timing**

Type	Parameter	1152 x 900	1024 x 1024 <sup>1</sup>
Horizontal	Period	16.16 $\mu$ s	N/A
	Blank	4.04 $\mu$ s	N/A
	Front Porch	337 ns	N/A
	Sync	1.43 $\mu$ s	N/A
	Frequency	61.88 KHz	N/A
Vertical	Period	15.141 ms	N/A
	Blank	598 $\mu$ s	N/A
	Front Porch	48.5 $\mu$ s	N/A
	Sync	48.5 $\mu$ s	N/A
	Frequency	66 Hz	N/A

## 9.7 The ID Space Block

The ID Space block responds to addresses in Kbus ID space, 0x1Y000000 and up. The 1 is the I/O space allocated to ID in the Series4 architecture; Y is the board slot location, corresponding to an ID Space. ID space is used for non-compatible (with Sun's CG3) components. These include the I/O location register, the ID ROM, the cursor chips and the color map.

### 9.7.1 Cursors

A 64 X 64-bit user definable cursor is implemented in hardware on the CG 30 Board. The hardware cursor provides a mask that prevents the cursor from being obscured when over a background of the same color.

The cursor chips communicate with the color map directly instead of with the frame buffer. This way, they act as an overlay to the pixel data, and provide a steadier, flicker-free cursor.

The cursor may be moved off the top, bottom, left, or right of the screen without wraparound.

### 9.7.2 CG 30 I/O Ports

The following I/O ports are implemented on the CG 30 board.

- Red video signal
- Blue video signal
- Green video signal (includes video sync signal output)
- Keyboard

<sup>1</sup> Not supported.

The color signal ports connect to the red, green, and blue input connectors of the color monitor. There are sync connectors on both the monitor and on the color board; neither sync connector is currently used as the sync output is combined with the green output.

# Index

- 1**
- 16 Mbyte Memory Board, 6-1
    - Address multiplexers subsection, 6-6
    - Address Register and Latch subsection, 6-2
    - Features, 6-1
    - I/O devices subsection, 6-9
    - I/O read transaction, 6-12
    - I/O write transaction, 6-11
    - Kbus interface subsection, 6-4
    - Kbus transactions, 6-10
    - Memory array subsection, 6-7
    - Memory control subsection, 6-3
    - Memory read transaction, 6-13
    - Refresh transaction, 6-12
- 3**
- 32 Mbyte Memory Board, 7-1
    - Address multiplexers subsection, 7-6
    - Address register and latch subsection, 7-3, 7-4
    - Features, 7-1
    - I/O devices subsection, 7-9
    - Kbus data interface subsection, 7-5
    - Memory array subsection, 7-7
- C**
- CG 40, 8-1
    - Control registers, 8-5
    - Features, 8-1
    - ID PROM, 8-7
    - Image planes, 8-2
    - Interrupts, 8-7
    - Kbus address interface, 8-2
    - Kbus control logic interface, 8-2
    - Kbus data interface, 8-2
    - Kbus reads and writes to the Bt458, 8-3
    - Overlay planes, 8-2
    - Space register, 8-6
    - Use of Brooktree RAMDAC, 8-2
    - Use of Bt458, 8-3
  - Video control logic, 8-5
  - Video control registers, 8-6
- D**
- DMA:
    - SCSI host adapter, 5-5
- I**
- I/O ASIC, 5-8
  - I/O interrupts:
    - on System Board, 5-7
  - Illustration:
    - Bus launcher, 3-16
  - Illustrations:
    - 16 Mbyte Memory Board, 6-1
    - 32 Mbyte Memory Board Block diagram, 7-2
    - Address multiplexers subsection, 7-6
    - Address multiplexers subsection block diagram, 6-6
    - Address register and latch block diagram, 6-2
    - Address Register and Latch subsection, 6-2
    - Address register and latch subsection, 7-3, 7-4
    - Bus Watcher Subsection, 3-18
    - Cache Subsection, 3-13
    - Cache Tag Subsection, 3-12
    - CG 40 block diagram, 8-1
    - CPU Board, 3-2
    - CPU Section Buses, 3-3
    - CPU Subsection, 3-9
    - ECC Logic, 3-19
    - I/O devices subsection, 7-9
    - I/O Devices subsection block diagram, 6-9
    - Kbus data interface subsection, 7-5
    - Kbus Interface Section Buses, 3-7
    - Kbus interface subsection block diagram, 6-5
    - Memory array subsection, 7-8
    - Memory array subsection block diagram,

## Solbourne Computer, Inc.

- 6-8
  - Memory control subsection block diagram, 6-3
  - MMU Subsection, 3-11
  - ROM Subsection, 3-14
  - SCSI/Ethernet interface, 5-5
  - Series4 and Series5 Components and Connections, 1-3
  - Slow bus subsection, 5-6
  - System Board block diagram, 5-1
  - System Board Kbus interface, 5-2
  - System Board VMEbus interface, 5-4
- K**
  - Kbus:
    - Backplane, 2-2
    - Masters and slaves, 2-1
    - Signals, 2-3
  - Keyboard port:
    - on System Board, 5-7
- L**
  - LANCE:
    - Ethernet chip, 5-5
- R**
  - RS-232-C:
    - on System Board, 5-8
  - RS-423-A:
    - on System Board, 5-8
- S**
  - Series4 and Series4 family:
    - memory boards, 1-4
  - Series4 and Series5 family:
    - Color Graphics Board, 1-4
    - CPU Boards, 1-4, 1-6
    - expansion bus, 1-5
    - introduction, 1-1
    - major functions, 1-1
    - System Board, 1-4
    - system bus, 1-5
  - Series4 CPU Board:
    - Address generator, 3-16
    - Boot ROMs, 3-14
    - Bus launcher, 3-16
    - Bus watcher, 3-18
    - Cache, 3-13
    - Cache Tag Block, 3-8
    - Cache Tag Section, 3-12
    - CDAT bus, 3-7
    - Clock Generation, 3-9
    - CRDAT bus, 3-7
    - ECC, 3-19
    - Floating point controller, 3-10
    - Interrupt handler, 3-19
    - Kbus Interface, 3-13
    - Major Buses, 3-2
    - MMU Section, 3-11
    - PTRAN bus, 3-7
    - SASI bus, 3-4
    - SDATA, Series4, 3-4
    - TAGADD bus, 3-4
    - TLB Block, 3-11
    - TLBASI bus, 3-7
    - Virtual Cache, 3-8
    - WTAGADD bus, 3-4
  - Series4 CPU:
    - Features, 3-1
    - Support for multiprocessing, 3-1
  - Series5 CPU Section, 4-2, 4-4
    - Address pipeline, 4-5
    - Cache organization, 4-6
    - Pipeline operations, 4-5
  - Series5 CPU Theory, 4-1
    - Features, 4-1
    - Overview, 4-2
  - Series5 Kdat Section, 4-3, 4-9
  - Series5 Launcher Section, 4-3, 4-7
  - Series5 Watcher Section, 4-10, 4-3
  - Signals:
    - KADDR, 2-4
    - KAHALTL, 2-3
    - KASL, 2-4
    - KBC and KBC0, 2-3
    - KBDRL, 2-3
    - KBID, 2-5
    - KBSLOC, 2-4
    - KCB, 2-5
    - KCOWNL, 2-4
    - KDATA, 2-5
    - KDIAGL, 2-3
    - KDSL, 2-5
    - KECCONL, 2-5

KHOWNL, 2-3  
KINTRL, 2-3  
KIOACK, 2-4  
KIOBCOUT, 2-3  
KIOREQ, 2-4  
KNMIL, 2-3  
KOKIL, 2-5  
KOWNL, 2-3  
KRESETL, 2-3  
KSHAREL, 2-4  
KSP, 2-4  
KTERRL, 2-5  
KTID, 2-5  
KTTYPE, 2-4  
PALCLK, 2-3  
System Board:  
Clock, 5-2  
Functions, 5-1  
Interrupts from I/O devices, 5-7  
LANCE, 5-5  
SCSI/Ethernet interface, 5-4  
Slow bus subsection, 5-6  
VMEbus interface, 5-3

