HP 3000 Computer Systems

SERIES 64
REFERENCE/TRAINING MANUAL

HEWLETT PACKARD
19447 PRUNERIDGE AVENUE, CUPERTINO, CA 95014

Part No. 30140-90005
E0483

Printed in U.S.A. 04/83

FEDERAL COMMUNICATION COMMISSION RADIO
FREQUENCY INTERFERENCE STATEMENT

The United States Federal Communications Commission (in Subpart J, of Part 15, Docket 20780) has specified that the following notice be brought to the attention of the users of this product:

> ***"Warning: This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual may cause interference to radio communications. As temporarily permitted by regulation it has not been tested for compliance with the limits for Class A computing devices pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference."***

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition, and lists the dates of all changed pages. Unchanged pages are listed as "ORIGINAL". Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars and dates remain. No information is incorporated into a reprinting unless it appears as a prior update.

Second Edition.....April 1983

Effective Pages	Date
ALL	APR 1983

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date on the title page and back cover of the manual changes only when a new edition is published. When an edition is reprinted, all the prior updates to the edition are incorporated. No information is incorporated into a reprinting unless it appears as a prior update. The edition does not change.

First Edition . . . . . APR 1982
Second Edition . . . . . APR 1983

CONTENTS

CONTENTS (Continued)

CONTENTS (Continued)

CONTENTS (Continued)

CONTENTS (Continued)

CONTENTS (Continued)

PREFACE

This manual contains hardware reference information for the HP 3000 Series 64 Computer (32460A and 32460B). Information for the new Power System in the 32460B systems has been included in Section VIII. All references to the Series 64 (32460A) also include the Series 64 (32460B) unless specifically mentioned otherwise. The System Status and Display Panel (SSDP) for the 32460B has been designated as SSDP-B for ease of reference.

This manual contains material for persons attending Hewlett-Packard's 3000 Series 64 Computer hardware training courses. Since the information in this manual is approximately the same as that presented in classroom lectures, the manual should be used for classroom reference, note-taking, and post-class reference.

SYSTEM HARDWARE OVERVIEW

SECTION I

The HP 3000 Series 64 Computer is a general purpose system that includes a dual-ALU CPU and up to 8 Mbytes of Main Memory. It executes under the control of Hewlett-Packard's Multiprogramming Executive (MPE) operating system.

In this manual, the computer's hardware is described in three major modules: CPU and Cache Memory, Main Memory, and the I/O System. These modules are organized around three major buses: the Central System Bus (CSB), the Intermodule Bus (IMB), and the Hewlett-Packard Interface Bus (HP-IB).

This section provides an overview of the computer's major hardware modules and buses.

1-1. THE MAJOR BUSES

The CSB is the computer's main inter-module line of communication. See Figure 1-1. It is a synchronous bus, carrying 66 signal lines of information, control, and system status.

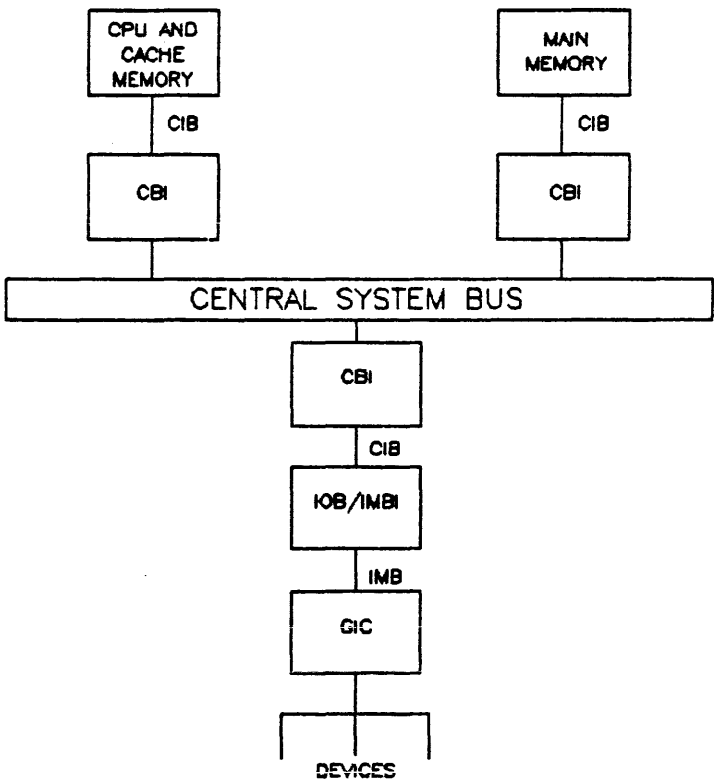


Figure 1-1. Major Bus Structure, HP 3000 Series 64 Computer

When one of the three major modules (CPU and Cache Memory, Main Memory, or I/O System) wants to communicate with another one, it first requests the CSB. Before the request reaches the CSB, it propagates through a Common Bus Interface (CBI) PCA. See Figure 1-1; there is one CBI PCA between each major module and the CSB. The CBIs are identical and interchangeable. Refer to Section II for a detailed description of the CSB and the CBI PCAs.

The IMB and HP-IB are part of the computer's I/O System. The IMB carries signals between the I/O Channels [such as the General I/O Channel (GIC)] and the Intermodule Bus I/O Adapter (IMB IOA). The HP-IB provides the signal interface for peripheral devices. Refer to Section V for a list of signals carried on the IMB and HP-IB.

Figure 1-2 shows the computer configured with two IMBs.

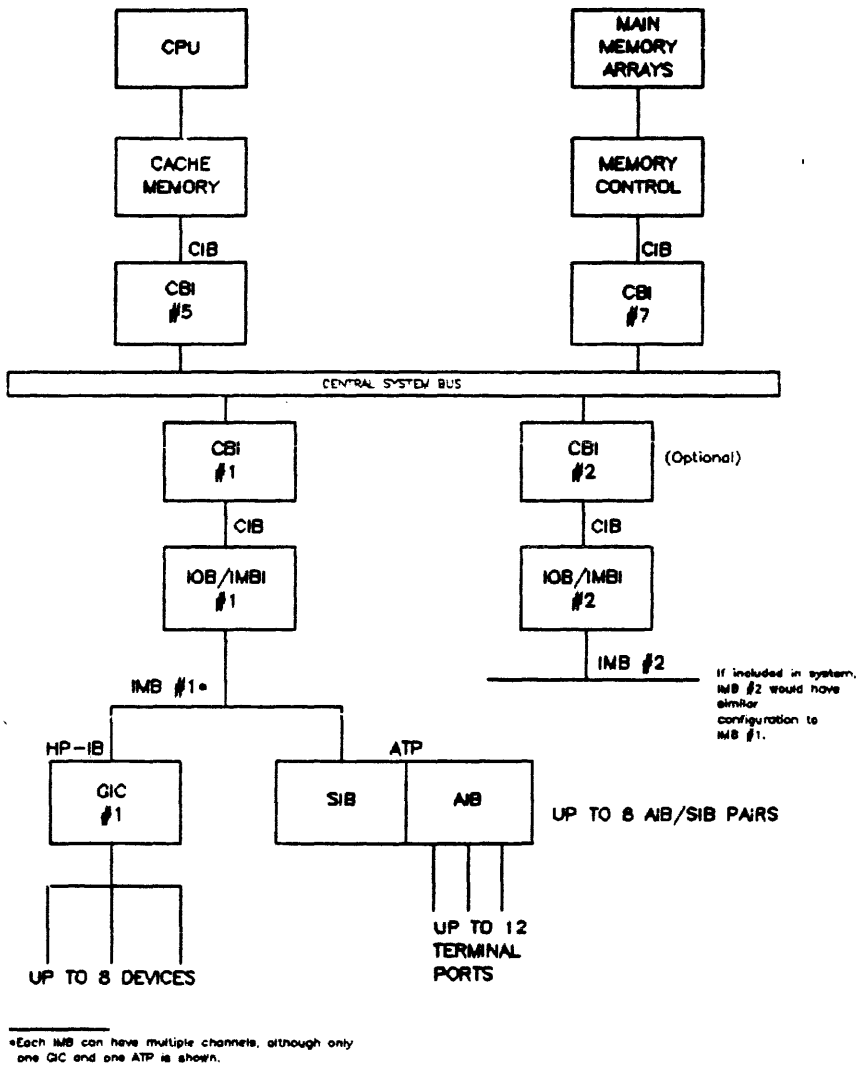


Figure 1-2. Simplified Computer Block Diagram

1-2. CPU AND CACHE MEMORY MODULE

The CPU does the data processing and computing. It includes a Writable Control Store and a dual ALU.

1-3. Writable Control Store (WCS)

The two WCS PCAs store the HP 3000 Machine Instruction Set and channel program microcode, or any special diagnostic microcode entered from the system console or from an I/O device. At power-on, bootstrapping code is loaded into the WCS from Read Only Memory (ROM) on the Diagnostic Control Unit (DCU) PCA. The bootstrap code then loads the system microcode into the WCS from the system disc. Thus, as the machine instruction set or the channel program microcode changes during the life of the computer, the changes can be made, via Installation Tapes, without changing the hardware.

1-4. Dual ALU

The CPU has two 16-bit ALUs. Thus it is compatible with the 16-bit machine instructions, but throughput is increased by either processing two operations simultaneously or by linking the ALUs together to do 32-bit calculations.

When the CPU needs information from another major module, it sends a request to its Cache Memory.

1-5. Cache Memory

Cache Memory is an 8-kbyte memory dedicated to the CPU. Its job is to reduce wait-time when the CPU wants data from Main Memory. To reduce the wait-time, Cache Memory fetches an entire data block (eight 16-bit words) when the CPU requests one word, on the strong probability the CPU will subsequently request the following word in that same block. Thus, when the CPU requests the next word, Cache has it immediately available, and no time is lost requesting and waiting for the word to come from Main Memory. The Cache is designed for a 95 percent "hit ratio," meaning it is expected to anticipate the CPU's needs and have the information immediately available 95 percent of the time. As far as the CPU is concerned then, the Cache Memory "IS" Main Memory (i.e., the CPU is unaware of what transpires beyond the Cache Memory).

The Cache Memory is also responsible for its contents with respect to the rest of the computer. For example, if the I/O System asks Main Memory for a block of data, and the Cache Memory holds the most current copy of that data, Cache Memory aborts Main Memory's response to the request and supplies the data to the I/O System.

1-6. CPU PCA Descriptions

Eleven PCAs are used in the CPU. They are briefly described as follows (unless otherwise noted, the CPU contains one of each PCA):

CONTROL A

The Control A (CTLA) PCA's chief job is to perform control functions for the ALUs, registers, shifters, and look-ahead logic.

CONTROL B

The Control B (CTLB) PCA performs control functions for ALUs, interrupts, and the CPU timer and clock distribution networks.

REGISTER/ALU

The four Register/ALU (RALU) PCAs contain ALUs, input multiplexers for the ALUs, and various registers.

SKIP/SPECIAL

The Skip/Special (SKSP) PCA contains the logic for the Skip and Special microcode fields.

CURRENT INSTRUCTION REGISTER

The Current Instruction Register (CIR) PCA contains the register of the same name, the Next Instruction Register, the Look Up Table, and VBUS breakpoint logic.

V BUS

The V Bus (VBUS) PCA contains microcode ranking logic, logic to control data intended for the WCS, and the Return Address Register (RAR).

WRITABLE CONTROL STORE

The two Writable Control Store (WCS) PCAs contain high-speed memory to store system and diagnostic microcode, as previously described.

1-7. Cache Memory PCAs

The Cache Memory includes two PCAs: a Cache Array Controller (CAC) and a Cache Memory Array (CMA). The CAC generates the Cache Memory control signals and directs interfacing with the rest of the computer. The CMA is the Cache Memory data store. It also controls incoming messages.

Refer to Section III for a detailed description of the CPU and Cache Memory Module.

1-8. MAIN MEMORY MODULE

The Main Memory Module is assigned Module No. 7, depicted by its association with CBI #7, as shown in Figure 1-2. This gives it a higher priority than either the CPU and Cache Memory Module or the I/O System Module when requesting the use of the CSB. The Main Memory Module contains one Memory Correction and Storage (MCS) PCA, one Main Memory Control (MMC) PCA, and up to eight Main Memory Array (MMA) PCAs.

1-9. Memory Size

The Main Memory Module provides from 1 to 8 Mbytes of storage. Each Main Memory Array (MMA) PCA provides 1 Mbyte of storage, and 1 to 8 MMA PCAs can be installed in the computer. (The minimum recommended configuration is two MMAs.)

1-10. Memory PCA Descriptions

The MMC PCA contains the control logic for memory accesses, messages, refresh and power down sequences.

The MCS PCA contains error detection, correction, and logging logic; address and data registers for memory accesses; memory status registers; and a message register. A register file is used to store messages, addresses and data.

The MMA PCAs contain (in addition to the RAMs) logic for decoding addresses, determining memory size and controlling the RAMs.

Refer to Section IV for a detailed description of the Main Memory Module.

1-11. I/O SYSTEM MODULE

The I/O System Module includes two major hardware divisions: an Intermodule Bus I/O Adapter (IMB IOA), and one or more Intermodule Bus I/O channels. See Figure 1-2. The I/O System, like the other major modules, has a CBI PCA to interface with the CSB.

1-12. IMB IOA Functions

The IMB IOA includes an I/O Buffer (IOB) PCA, an Intermodule Bus Interface (IMBI) PCA, and a CBI PCA. The IMB IOA performs the following functions:

a. Accepts messages from the CPU and either acts on them or passes them on as IMB commands. In either case it returns a response to the CPU.

b. Initiates messages to the CPU.

c. Transfers data and channel programs between the Main Memory Module and the IMB.

d. Enforces CSB conventions regarding Main Memory operations and messages.

e. Responds to system signals such as Power On (PON) and Power Fail Warning (PFW).

f. Supplies diagnostic information to the CPU or the Diagnostic Control Unit PCA, as requested.

1-13. Supported Peripherals

Refer to the HP 3000 Computer Systems Configuration Guide (P/N 5953-7438) for a list of supported peripherals.

1-14. DIAGNOSTIC CONTROL UNIT (DCU)

The DCU is a microprocessor-based PCA that provides access to all PCAs connected to the CPU Bay card cage backplane. It also controls the computer's clocks.

The DCU has two diagnostic roles. First, it tests a specific kernel of CPU hardware. Second, it enables the group of diagnostics called Fault Locating Diagnostics (FLDs) to be used to isolate faults to the board level.

1-15. Diagnostic Functions

The main diagnostic function of the DCU PCA is to interrogate the CPU registers. This is done via serial shift strings. Each PCA connected to the CPU backplane of the CPU Bay card cage has a shift string which the DCU can access by connecting all the strings and shifting them circularly. The DCU controls shifting by using the SYNC lines which control the computer's clocks. By using the the computer's clocks, shift strings can be controlled in any combination.

1-16. DCU Operating Modes

The DCU has four modes of operation: Initialization, Control, Maintenance, and Idle.

Initialization occurs at power-on or at auto-restart following a power failure. The DCU is initialized and a self-test is run. IF the self-test is successful, the DCU either reloads the WCS and Look Up Table (LUT) from the disc and restarts MPE (in the auto-restart case), or (in the power-on case) it enters the Control or Maintenance Mode.

The Control Mode is entered after the Initialization Mode self-test, or when the operator enters CNTL-B at the system console [the key switch near the Power Control Module (PCM) must be in the correct postion]. In the Control Mode, the operator can command functions such as Run, Halt, Load, Start, and Dump.

Maintenance Mode is entered when the operator enters CNTL-B if it is enabled by the Remote/Local Diagnostic Control Switch. In Maintenance Mode, the operator can command hardware functions, such as micro-halting the computer and listing PCA strings. Maintenance Mode is also entered when the computer detects an error and sets the SYSSTOP line (the key switch near the PCM must be in the correct position).

When the DCU is not in any of the active modes, it is in Idle Mode. In Idle, the DCU looks for interrupts that call one of the active modes, performs self-tests, and monitors the power system via the Power System Controller (PSC) PCA in the Series 64 (32460A) and via the Power Distribution Monitor (PDM) in the Series 64 (32460B).

Refer to Section VI for a detailed description of the DCU.

1-17. POWER DISTRIBUTION AND POWER SYSTEM CONTROLLER (PSC)/ POWER DISTRIBUTION MONITOR (PDM)

1-18. AC Power

In the Series 64 (32460A), the AC Power System consists of a Power System Controller (PSC) and a three-phase isolation transformer, both mounted in the bottom of the I/O Bay. Cable harnesses provide distribution.

In the Series 64 (32460B), the AC Power System is consists of 3 ferro-resonant transformers mounted in the bottom of the I/O Bay. Cable harnesses provide distribution.

The computer must be connected to 208 VAC, 60-Hz, three-phase power in the United States. In other countries, it can be connected to 415 VAC, 50-Hz, or 380 VAC, 50-Hz three-phase power.

1-19. DC Power

DC power in the Series 64 (32460A) is provided by seven power supplies and a battery pack. A bus bar and cable harnesses provide distribution.

DC power in the Series 64 (32460B) is provided by six power modules arranged in four module sets, including a battery pack. Bus bars and cable harnesses provide distribution.

Battery backup in the Series 64 (32460A and 32460B) supplies 5 volts to Main Memory when normal AC power is supplied to the computer. When AC power is interrupted, the backup supplies 5 volts for at least 15 minutes, depending on the total computer load.

Backup in the Series 64 (32460A) consists of a dual DC-to-DC converter, a Battery Control Module (BCM) PCA, and a battery assembly containing 12 five-ampere-hour cells.

Backup in the Series 64 (32460B) consists of a Battery Charger, an Off Battery Converter, and a battery assembly containing 12 five-ampere-hour cells.

1-20. Power System Controller (PSC)/ Power Distribution Monitor (PDM) PCA Functions

The PSC/PDM PCA is the interface between the power supplies and the rest of the computer. It has two major functions: it monitors and controls the power supplies to ensure they provide valid outputs; and it aids the DCU is diagnosing and troubleshooting power problems. The PDM also distributes +5VB and +/-12V.

Information gathered from the power supplies is sent to the DCU PCA for processing. The DCU uses the information to control the level of computer activity.

Refer to Section VII for a detailed description of the Series 64 (32460A) power supplies and the PSC.

Refer to Section VIII for a detailed description of the Series 64 (32460B) power supplies and the PDM.

CENTRAL SYSTEM BUS AND COMMON BUS INTERFACES

SECTION II

The HP 3000 Series 64 Computer's three major modules (CPU and Cache Memory, Main Memory, and the I/O System), are organized around a Central System Bus (CSB). See Figure 2-1. Each module uses a Common Bus Interface (CBI) PCA to interface with the CSB. (Communication between each module and its CBI PCA takes place over a Common Interface Bus, CIB).

This section describes the CSB and the CBI PCAs. Refer to Section III for a description of the CPU and Cache Memory Module, Section IV for the Main Memory Module, and Section V for the I/O System.

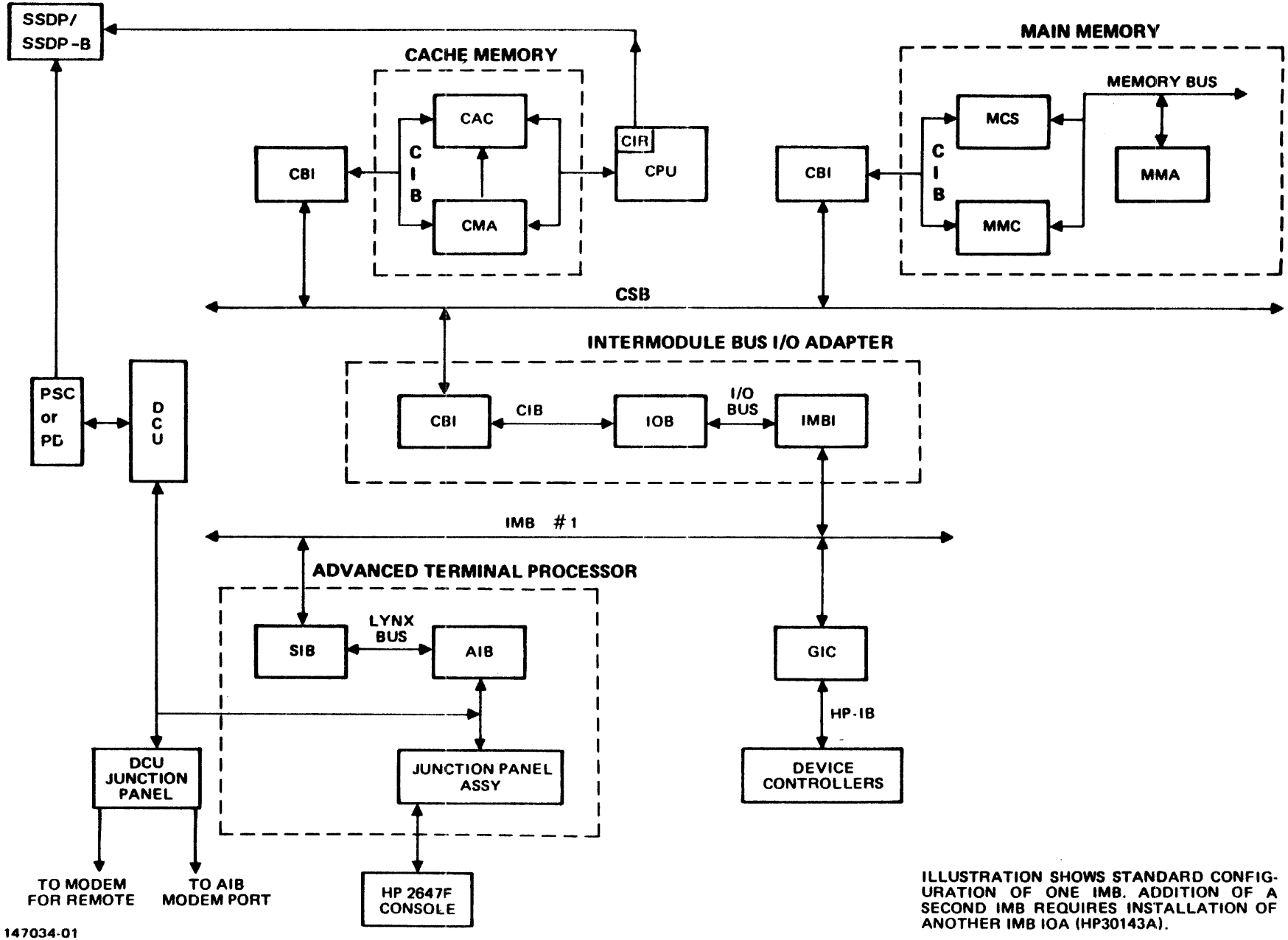


ILLUSTRATION SHOWS STANDARD CONFIGURATION OF ONE IMB. ADDITION OF A SECOND IMB REQUIRES INSTALLATION OF ANOTHER IMB IOA (HP30143A).

Figure 2-1. Major Modules of the Computer

2-1. CENTRAL SYSTEM BUS

The CSB and the CBI PCAs form the communications link between the major modules of the computer. This organization permits the connection of up to four major modules, as follows:

Module	Module No.
Main Memory	7
CPU and Cache Memory	5
Intermodule Bus I/O Adapter (IMB IOA) No. 1	1
Intermodule Bus I/O Adapter (optional) No. 2	2

Data-sharing among the modules is allowed, and mutual exclusion for reads/writes and message transfers is guaranteed.

The CSB is a synchronous, general-purpose, high-speed bus. To obtain the CSB, any module must go through a two-clock, request/selection cycle; no module has implied control of the bus. Data is transferred over the bus on a block basis. A block consists of eight 16-bit words (it is sometimes referred to as four 32-bit double-words).

2-2. CSB SIGNALS

CSB signals can be grouped into three categories. They are defined in the following paragraphs.

a. Information:

AD00-31
ADP0-1
BUSOP0-3
TO0-2
FROM0-2
CPARITY

b. Bus control lines

Busy lines:

BUSY1-7

Request lines:

CONTXFR
REQ7-1

Other bus control lines:

HOLD
ABORT

c. System Status:

ACK
NACK

2-3. Information Signals

The information signals are defined as follows:

AD00-31 and ADP0-1

The 32 AD lines are used for both Addresses and Data (a BUSOP code indicates which interpretation is valid). ADP0 is the parity bit for AD00-15; ADP1 is the parity bit for AD16-31.

BUSOP0-3

The BUSOPs (Bus Operation codes) are:

			TO:	
CODE	INTERPRETATION	CPU	IOA	MEM
0000	NOP			
0001	(Illegal Opcode)			
0010	(Illegal Opcode)			
0011	Read Block Private			yes
0100	Write Old Block			yes
0101	(Illegal Opcode)			
0110	(Illegal Opcode)			
0111	(Illegal Opcode)			
1000	Data	yes	yes	yes
1001	Error	yes	yes	
1010	Send Word	yes	yes	yes
1011	Send Address	yes	yes	
1100	(Illegal Opcode)			
1101	(Illegal Opcode)			
1110	(Illegal Opcode)			
1111	Check Block	yes		

Read Block Private

This signal indicates that the 32 bits on the AD lines are interpreted as Address. If the block is to modified, it will be the only valid copy available when the write is complete. Reading it with a Read Block Private signal will result in all other copies of the block being marked invalid.

Write Old Block

This signal also indicates that the 32 bits on the AD lines are interpreted as Address. A module that has a modified "dirty" copy of a data block that is to be overwritten must first write that block back into Main Memory, as it is the only valid copy. All other copies of the block are marked invalid.

Data

This opcode indicates that the 32 bits on the AD lines are interpreted as Data. The Data Opcode is used by Main Memory to return data in response to a read request. Other modules use the Data Opcode for write operations.

Error

The Error Opcode replaces the Data Opcode during a read when Main Memory detects a multiple bit error, or attempts a read from an address for which no Main Memory Array PCA exists in the computer. The 32 bits of AD are not valid in this case.

Send Word

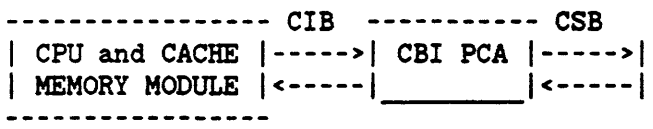
This opcode also means the 32 bits on the AD lines are to be interpreted as a Message.

Send Address

The 32 bits of AD are interpreted by the receiving module as a pointer into Main Memory. The first word describes the structure of the rest of the message.

Check Block

Check Block is used for diagnostics by the CPU and Cache Memory Module. It is a dummy BUSOP that allows the module to send and receive information through the CBI PCA data path, as shown:



During a Check Block, Cache Memory sends a message to itself to test the TO code and FROM code. These codes specify the TO and FROM module numbers assigned to the seven request lines. They are assigned as follows:

To/From Code	Request Line	Module
001	1	IMB IOA #1
010	2	IMB IOA #2 (optional)
011	3	(Not Used)
100	4	(Not Used)
101	5	CPU and Cache Memory
110	6	(Not Used)
111	7	Main Memory

CPARITY

A parity bit used on the FROM, TO, and BUSOP opcodes.

2-4. Bus Control Signals

The bus control signals are defined as follows:

BUSY7

This is the Main Memory Module BUSY signal. It is driven high when memory is busy or when a transfer to memory must be prevented. The latter case occurs while a module is transmitting to memory, or when a module other than memory finds it must supply a data block. For example, if the IMB IOA requested a block from Main Memory and the CPU's Cache Memory had a modified copy of the block, the Cache would abort Main Memory and supply the block. The intercepting module holds Main Memory BUSY after aborting the read until it can supply the data block.

BUSY6-1

For the CPU and Cache Memory and the I/O System modules, BUSY means the module's message buffer is full. The BUSY line is not checked for data returns because a module that requested a read must be ready to accept that data. Similarly, messages that are responses to a Send Word command also ignore the BUSY lines. When selected, the sending module drives the receiving module's BUSY line high. On succeeding clocks, the receiving module holds its own BUSY line high.

CONTXFR

The Continue Transfer line is used to obtain the bus for multiple clock transfers. For example, writes to memory require five consecutive bus transfers (address followed by four data words.) To do a write to memory, a module requests the bus and after selection, CONTXFR is used to request the bus for the following transfer. CONTXFR is only valid for the last module that used the bus. All other modules must use normal request/selection cycles.

REQ7

Request 7 is the request line for the Main Memory Module. When this line goes active, REQ7 is sent to Modules 6 through 1.

NOTE

Main Memory, which has the highest priority number, has the highest request priority on the CSB.

REQ6-1

These are the CSB Request lines for the modules other than Main Memory. The request lines determine which module gains access to the CSB. The lines are structured such that REQ6 is sent to Modules 5 through 1; REQ5 is sent to Modules 4 through 1; etc.

HOLD

HOLD is asserted when a module requests the CSB for a transfer to memory, but finds the bus already busy. This causes subsequent requests for the CSB to queue. Thus queues get formed for those modules that are asserting the HOLD line, and for those modules that are being held off by the HOLD line.

ABORT

ABORT is asserted no later than three clock periods after a Read Block Private appears on the CSB, by any module that finds it has a modified copy of the requested block. The read to memory is aborted and the aborting module then supplies the block to the requesting module. For a Read Block Private, the aborting module marks its block Invalid and the receiving module's "Dirty." (Memory would not have been aborted if the copy had not been modified, i.e., was "clean.")

2-5. System Status Signals

System Status signals ACK and NACK are asserted two clock periods after the opcode to which they refer. The module to which the opcode was directed responds with an ACK if the transmission was correct. ACK is asserted by the module to which it is directed for every transfer (i.e opcode, data). Certain opcodes must also be received by modules other than the one to which the command was sent. These modules acknowledge correct reception of these opcodes by asserting NACK.

The following lists the ACK/NACK expectations of the sender for each opcode:

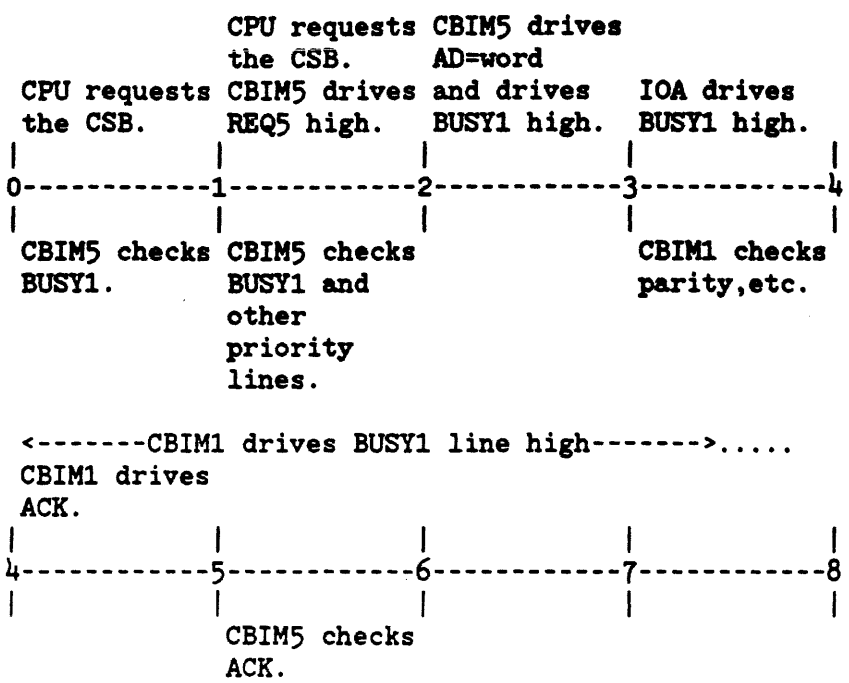
Opcode	ACK	NACK	Module(s) Directed To
Read Block Private	X	X	Memory
Write Old Block	X		Memory
Data	X		CPU, I/O, Memory
Error	X		CPU, I/O
Send Word	X		CPU, I/O, Memory
Send Address	X		CPU, I/O
Check Block	X		CPU

2-6. CSB OPERATION EXAMPLES

In the following examples of CSB operations, Main Memory has BUSY7 and REQ7 as its busy and request lines, as it is connected to CBI Module 7. Similarly, the CPU and Cache Memory Module is attached to CBIM5 and has BUSY5 and REQ5; the IMB IOA is attached to CBIM1 and has BUSY1 and REQ1. The description above the timing diagram is the initial action; the description below the diagram is the response.

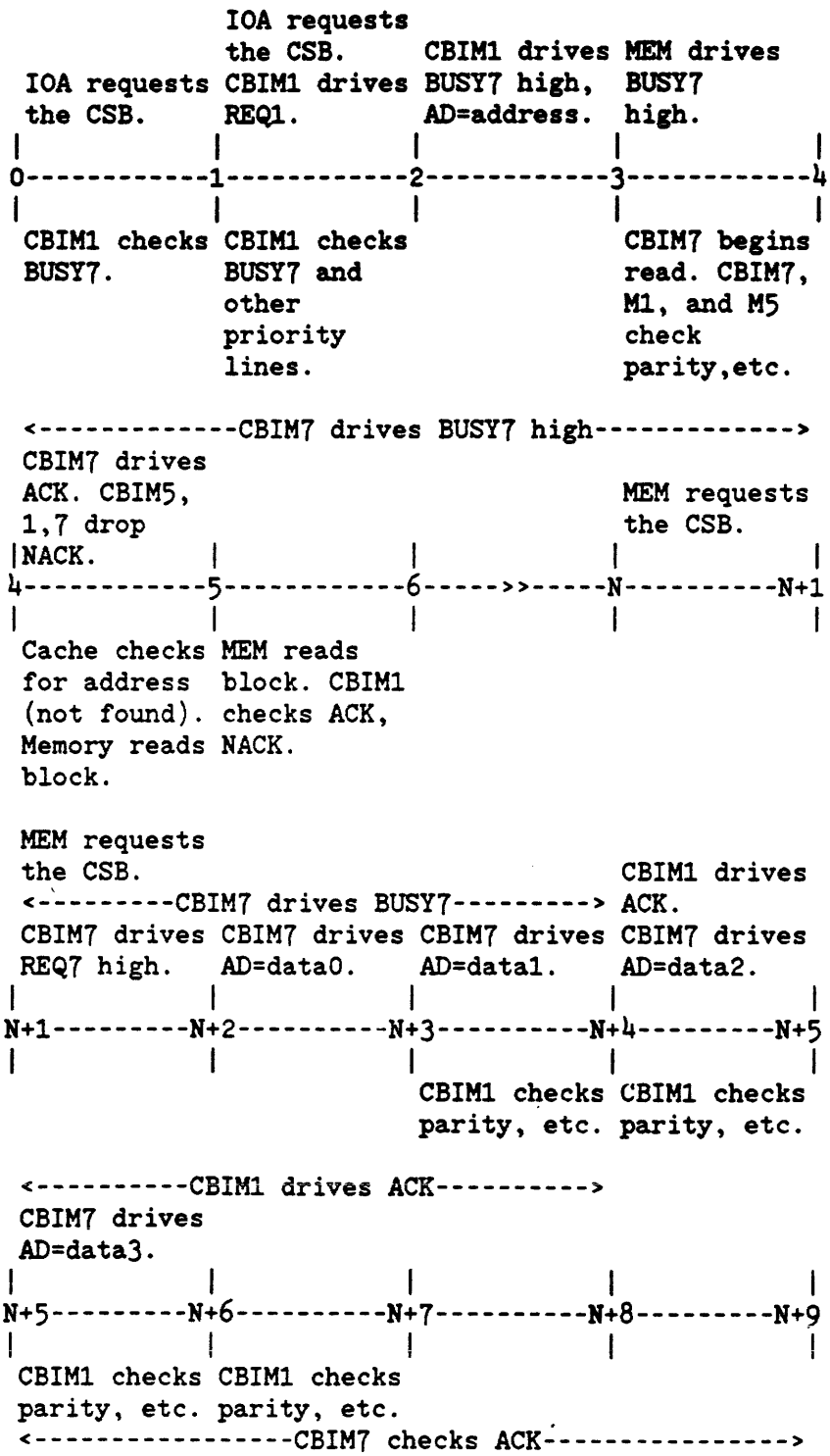
2-7. Send Word

In this example, the CPU (CBIM5) issues a Send Word message to the IMB IOA (CBIM1).



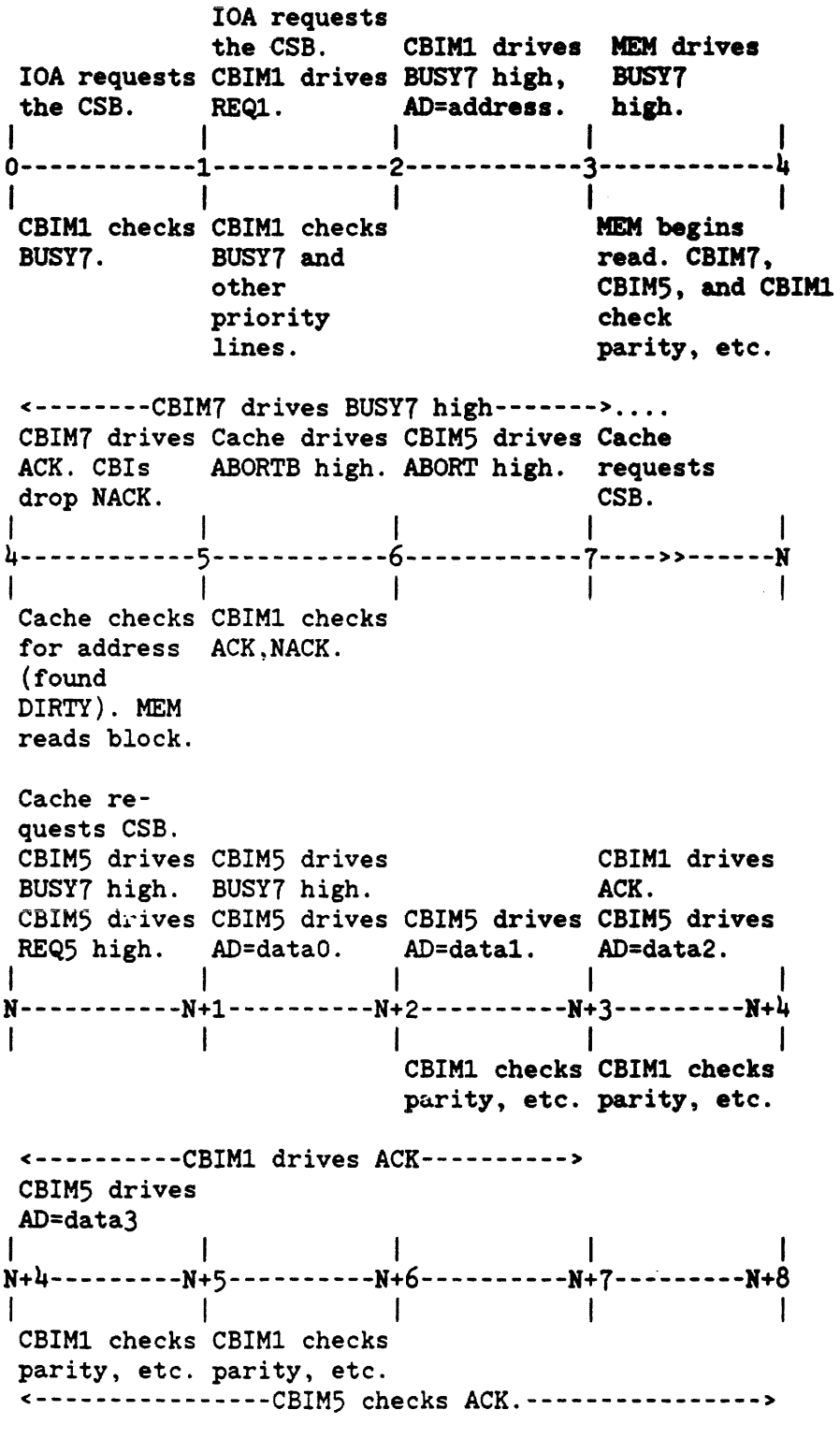
2-8. Read Block Private (No Modified Copy)

In this example, the IMB IOA (CBIM1) issues a READ BLOCK PRIVATE to Main Memory (CBIM7) for a block it needs to modify. The CPU and Cache Memory Module checks for that block and finds it does not have a copy.



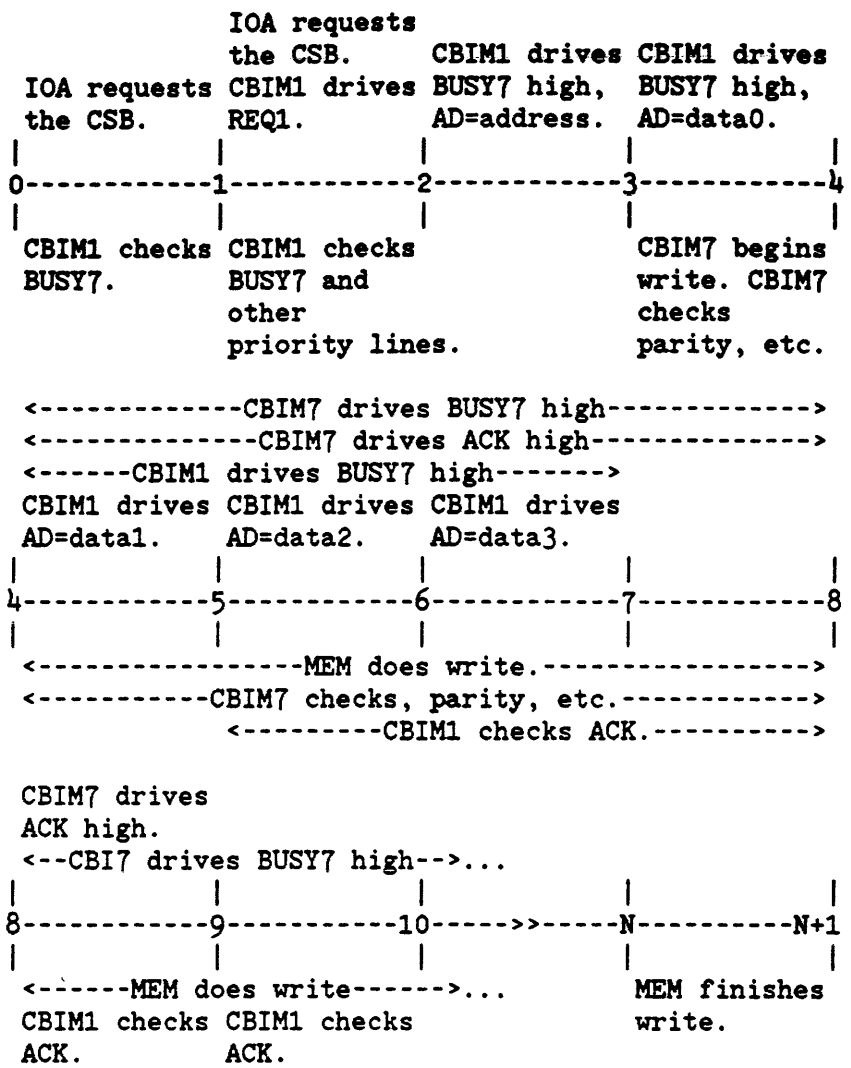
2-9. Read Block Private (Modified Copy)

This example is the same as the preceding one, except that the CPU and Cache Memory Module has a modified ("dirty") copy of the block.



2-10. Write Old Block

In this example, the IMB IOA (CBIM1) issues a write to Main Memory.



2-11. COMMON BUS INTERFACE PCA

The Common Bus Interface (CBI) PCA is a major module/CSB interface. Each of the three major modules (CPU and Cache Memory, Main Memory, and I/O System) requires a CBI PCA to communicate with any other module on the CSB. The CBI has two ports: one to the CSB and another to the Common Interface Bus, CIB. See Figure 2-2.

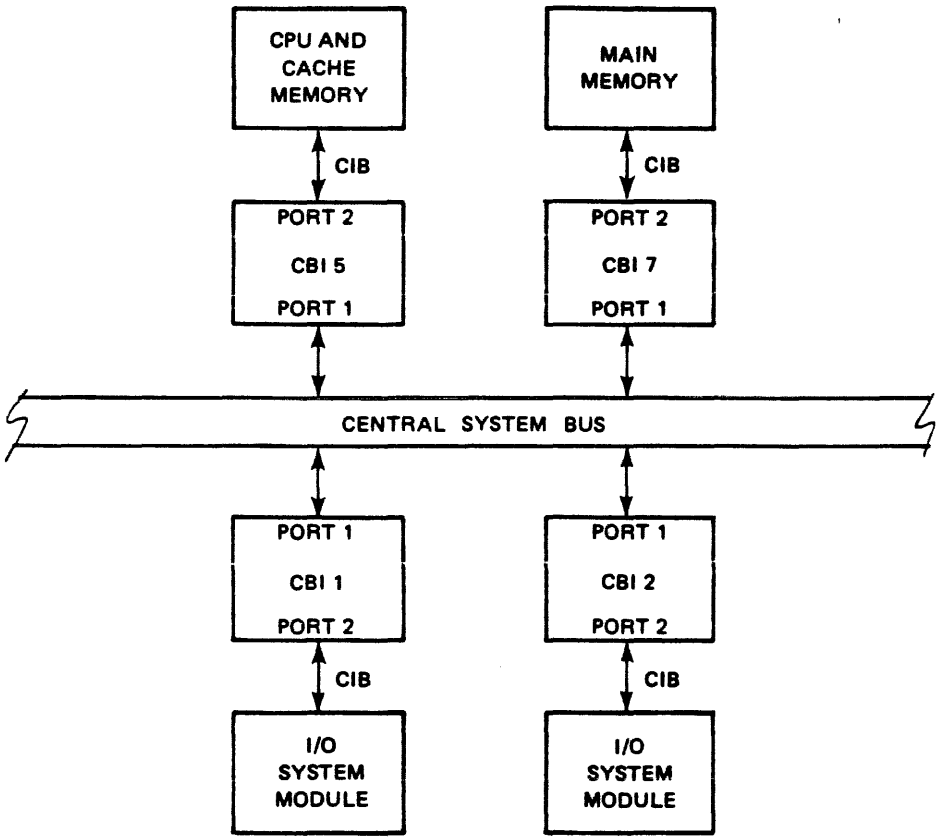


Figure 2-2. CBI PCAs and CIBs

2-12. CBI Logic

There are two main logic sections on the CBI PCA. The receiving (listening) logic does Bus Operation (BUSOP) handling. The sending (talking) logic requests the use of the CSB. All discussion of receiving/sending is between the CBI and the CSB.

The listening logic contains two independent 34-bit registers to store addresses, data, or messages from the CSB. Two three-bit registers store an associated FROM code. Error checking is done each time a module receives an opcode. To receive an opcode implies that one of the BUSOPs (Data, Error, Send Word, or Send Address) appeared on the CSB with that module's TO code, or that a read or write was sent to Main Memory.

The module numbers used by the CBI PCA to identify the TO and FROM codes are wired on the backplane. They are assigned as follows:

TO/FROM Code	Module
1	IMB IOA1
2	IMB IOA2 (optional)
3	(Not Used)
4	(Not Used)
5	CPU and Cache Memory
6	(Not Used)
7	Main Memory

The talking logic does several jobs. It stores address/data and control information from the module to be transmitted on the CSB; it requests the CSB, and, after selection, drives the CSB with this information; and it checks for ACK and NACK.

2-13. System Error Handling By The CBI

The CBI interrupts the I/O System as well as the CPU and the Diagnostic Control Unit (DCU) when it detects any of the following conditions:

CSB Address/Data parity errors
Control parity errors
Illegal BUSOP codes
Error BUSOP codes (caused by a memory double-bit parity error or "No MMA" error)
Module data parity error

The CBI interrupts the CPU with the signal CBI INT. This notifies the CPU of a CBI error condition; it is then up to the CPU to initiate any necessary action.

The CBI interrupts the DCU with a System Stop (CBI SYSSTOP) signal. This halts the entire computer. At that point the DCU investigates the CBI SYSSTOP by shifting each CBI PCA looking for the active SYSSTOP bit.

2-14. CBI-to-Module Signals

The signals that travel between each CBI PCA and its module are carried on a cable called a Common Interface Bus (CIB). All CIB's are identical, but the useage of the CIB signals are controlled by the module to which the CIB is attached. These signals may be grouped in five categories: Information, Control, System Status, Clocks, and CSB Drive. The signals are listed next.

The Information signals are:

ADATA00-31 (Address or Data)
ADATAP0-1 (Parity on Address or Data)
SWORD(L) (Send Word)
FROMIN0-2 (FROM from CSB)
RBP (Read Block Private)
WOB (Write Old Block)
BCOP (Broadcast op)
DATOP (Data op)
SWOP (Send Word op)
SAOP (Sent Address op)
DCB (Diagnostic Check Block)
ABORTIN
DATA00-31
DATAP0-P1
OP0-3 (BUSOP to CSB)
EXTTO0-2 (TO to CSB)
BUSY
ME0-2

The Control signals are:

Register Control:

TOENB0(L) (TO Enable 0)
TOENB1(L) (TO Enable 1)
ENBREGA(L) (Enable Register A)
ENBREGB(L) (Enable Register B)
CLKFRMA(L) (clock FROMA)
CLKREGA(L) (clock registerA)
CLKREGB(L) (clock registerB)

Data Control:

GENPAR (generate parity)
CHKPAR(L) (check parity)
MUXSTAT (mux status)
CLRSTAT (clear status)

Request Control:

AWAKE(L)
IGNBUSY(L) (ignore busy)
CONT (continue)
DSELCT
SELECT

Diagnostic Control:

DCUSHFT (DCU shift)
DCUSI (DCU shift in)
CBISO (CBI shift out)
CIBSHFT (CIB shift)
DISSTOP (Disable CBISYSTOP)
RESET

Other Control: ONE (logical one)

The System Status signals are:

CBIINT (CBI Interrupt)
CBISYSTOP
ERROR

The Clock signals are:

CBICLK
CBICLK(L)
CBISYNC

The CSB Drive signals are:

REQME (Request Me)
BUSYME (Busy Me)

2-15. CBI Physical Description

The CBI PCAs connect to the CPU Bay card cage backplane in the slots shown below (numbers 3, 4, and 6 are not used):

CBI No.	Slot	Module
1	20	IMB IOA No. 1
2	22	IMB IOA No. 2 (optional)
5	19	CPU and Cache Memory
7	26	Main Memory

The CBI PCAs are identical and interchangeable. Their numbers, as listed above, are determined by the slot each is plugged into. In other words, if you plug a CBI into Slot 20, it functions as CBI #1. But if you plug that same CBI into Slot 19, it functions as CBI #5. This enables the use of multiple CBIs, without identification switches. Figure 2-3 shows the physical outline of the PCA.

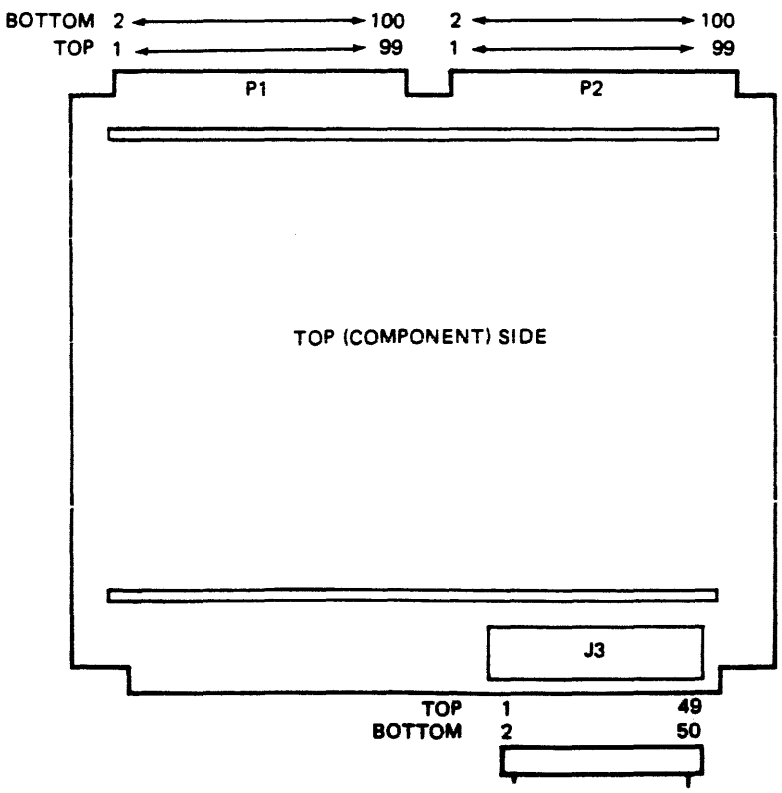


Figure 2-3. CBI PCA Physical Outline

2-16. CBI THEORY OF OPERATIONS

The CBI operations are described in the following paragraphs. The CBI is considered either a "talker" (sender) or a "listener" (receiver) on the CSB, and performs only one of these functions at a time.

2-17. Access To The CSB

Access to the CSB is implemented by a "round-robin" scheme which prevents a high priority module from having continuous exclusive access of the bus. This is done by prioritizing CBI numbers and using a CBI State Machine. The priorities are as follows:

CBI	Module	Priority
7	Main Memory	Highest
5	CPU and Cache Memory	|
2	IMB IOA 2 (optional)	|
1	IMB IOA 1	Lowest

As shown in Figures 2-4 and 2-5, the sequence followed for CBI selection onto the CSB is controlled by the CBI State Machine. The State Machine has six states: (1) Idle, (2) Wait, (3) Request, (4) Select, (5) Inside, and (6) Outside.

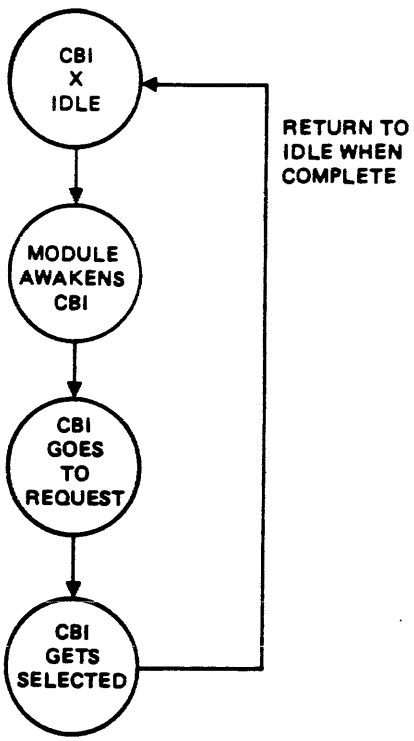


Figure 2-4. CBI State Machine, One Module Needing Service

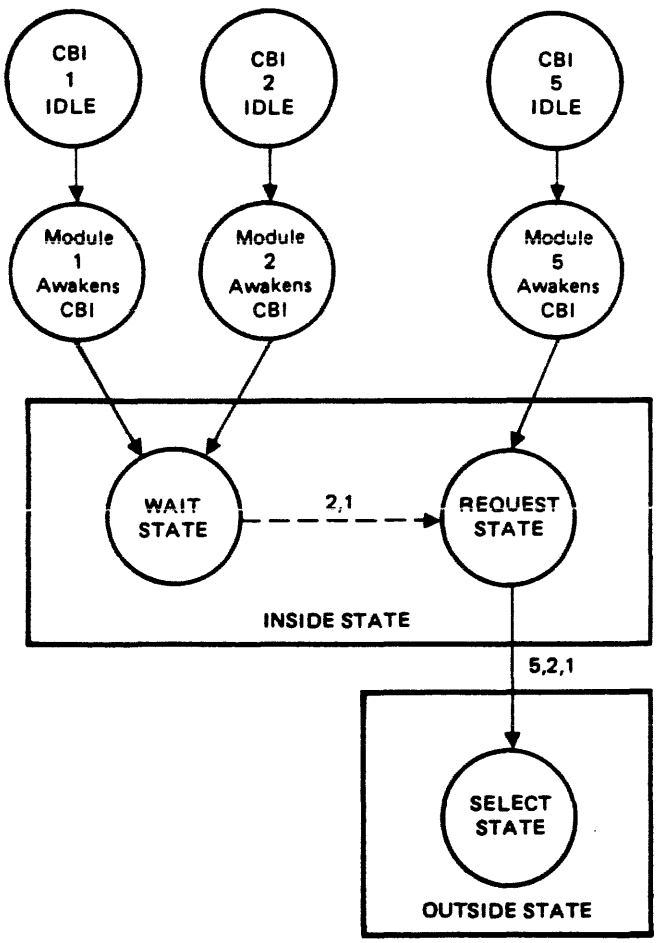


Figure 2-5. CBI State Machine, Multiple Modules Needing Service

In the Idle State, there is no activity. Request means the CBI is requesting the CSB. Select means the CBI is being selected. The Wait or Request State always follow the Idle State; likewise, Select always follows Request.

In Figure 2-4, one module needs CSB access, so it awakens its CBI. Since CBI "X" is the only active CBI, it goes through three states, then back to Idle.

In Figure 2-5, multiple modules want the CSB at the same time. Each awakens its CBI. CBI 5 enters the Request State first, as it has the highest priority number (i.e., 5 is higher than 1 or 2). CBIs 1 and 2 thus enter the Wait State. Once CBI 5 is selected (goes to the Outside State), CBIs 2 and 1 take their turns.

The CBIs do not go to the Inside State for message transfers. Those are handled on a first-come, first-served basis. Modules requiring message transmissions go to the Wait State if an Inside State exists until all CBIs Inside are serviced.

2-18. CBI Talking Logic

All information coming from a module is latched into the Talking Logic Output Registers on the CBI PCA, then sent to the CSB. See Figure 2-6. Those registers are gated to the CSB via the Select line. Parity is generated for Control information (the BUSOP, TO code or FROM code).

The TOENB (TO Enable) lines select either the INTTO (Internal TO), ONE, or EXTTO (External TO) signals to multiplex through to the Output Registers. INTTO is latched on the CBI and is accessible to the CIB. ONE is used to designate the module the information is intended for. EXTTO is an external TO code coming from the module.

The following paragraphs provide further descriptions of the Talking Logic circuitry. See Figure 2-6.

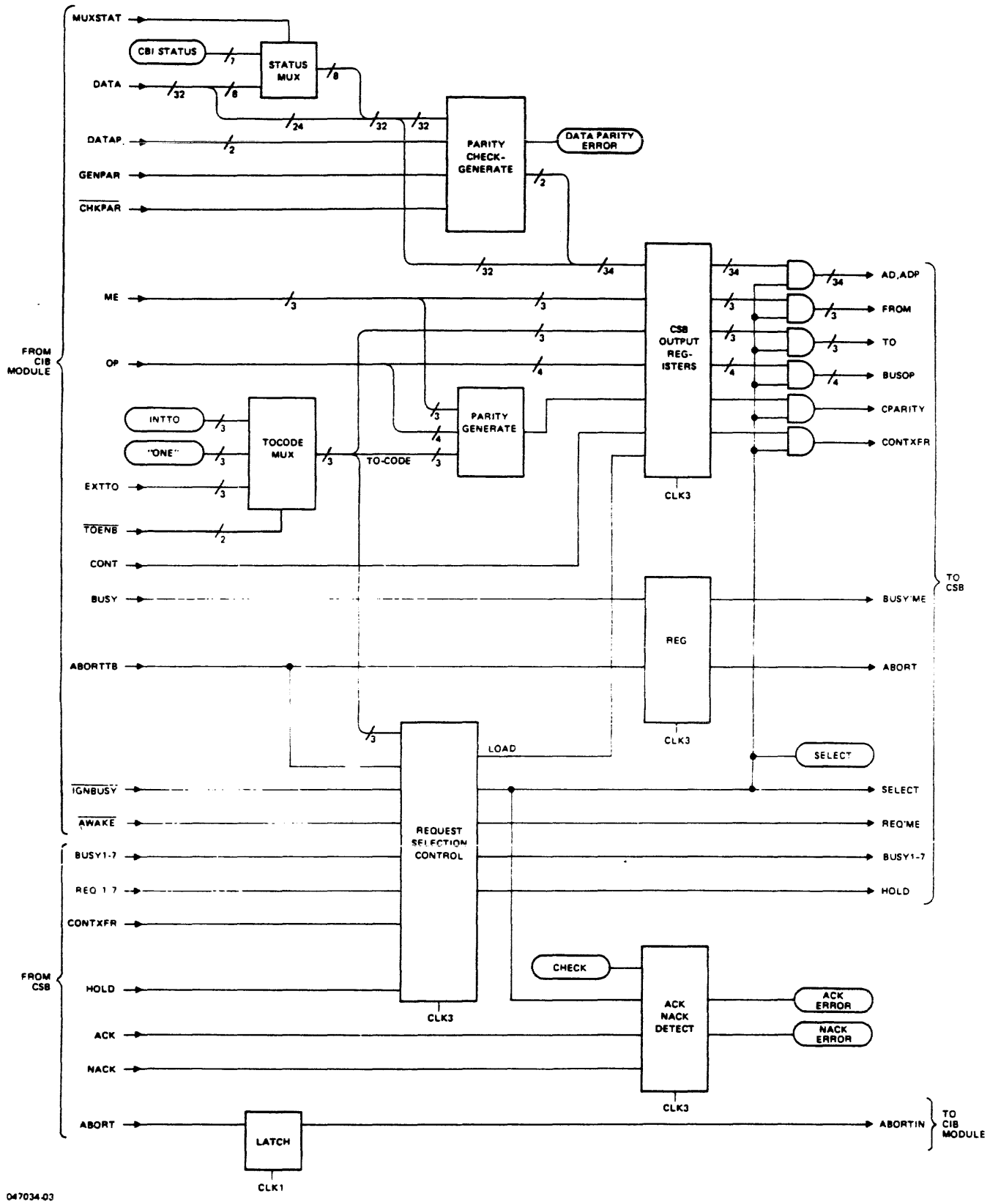


Figure 2-6. CSB Talking Logic on CBI PCA

2-19. Output Registers

Information to the CBI PCA (DATA00-31 and DATAP0-1) from the module is always being loaded into the CBI output registers. If the CBI does not enter the Select state one cycle following the Wait or Request state, the CBI will stop loading these output registers. This means that for multi-cycle transfers, such as writes, the module can send the address to the CBI for two cycles and then start sending the first data word. If the CBI goes into the Select state, the module will continue to send the rest of the data words; if not, the address will be held in the CBI output registers with the first data word as the input to these registers.

2-20. Generating Parity

The CBI can generate parity (GENPAR=1) on the 32 bits of information (DATA00-31) being sent from the module to the CBI PCA. This signal must be valid at the CBI the cycle after the request was made.

2-21. Multiplexing Status

The module can multiplex (MUXSTAT=1) the CBI status bits into the first 7 bits of the information (DATA00-06) being sent to the CBI. When these bits are to be sent as part of a message, parity must be generated on the CBI PCA (GENPAR=1).

2-22. Checking Parity

The module can have the CBI PCA check parity (CHKPAR(L)=0) on the 32 bits (DATA00-31) and the 2 parity bits (DATAP0-1). The data and parity bits must be valid whenever the CBI is checking parity, otherwise the CBI will generate a Data Parity Error (DPE=1). Parity on the CSB is even.

2-23. TO Code Selection

The module can select one of four sets of TO codes for transmission on the CSB:

TOENB0(L)	TOENB1(L)	Description
0	0	To memory.
0	1	To the FROM code stored in association with Register B.
1	0	To the external TO code (EXTTO0-2) specified by the module.
1	1	To the FROM code stored in association with Register A (FROMIN0-2 must be connected to EXTTO0-2).

For example, the CPU and Cache Memory Module sends reads and writes to Main Memory, supplies data for an aborted read to the module specified by the FROM code stored in association with Register B, and supplies a TO code via EXTTO for messages. Main Memory, by tying FROMIN to EXTTO on the backplane, returns data and messages to the module specified by the FROM code associated with Register A.

2-24. CBI Listening Logic

All information coming from the CSB to a module is clocked into the Listening Logic Input Register on the module's CBI PCA. See Figure 2-7. Parity (even parity) is checked on Address/Data information as well as Control signals. Any BUSOP signals are decoded and sent to the module. The module decides what it is being asked to do, and clocks the corresponding register (CLKREGA or CLKREGB). When the module is ready to use the information, it enables it to itself.

If an error was detected in the information being transmitted, a bit in the CBI Status Control logic is set, and a CBI Interrupt or a SYSSTOP is issued. In that case, the module does not acknowledge receipt of valid information by sending ACK or NACK to the originating module.

The following paragraphs provide further descriptions of components of the Listening Logic. See Figure 2-7.

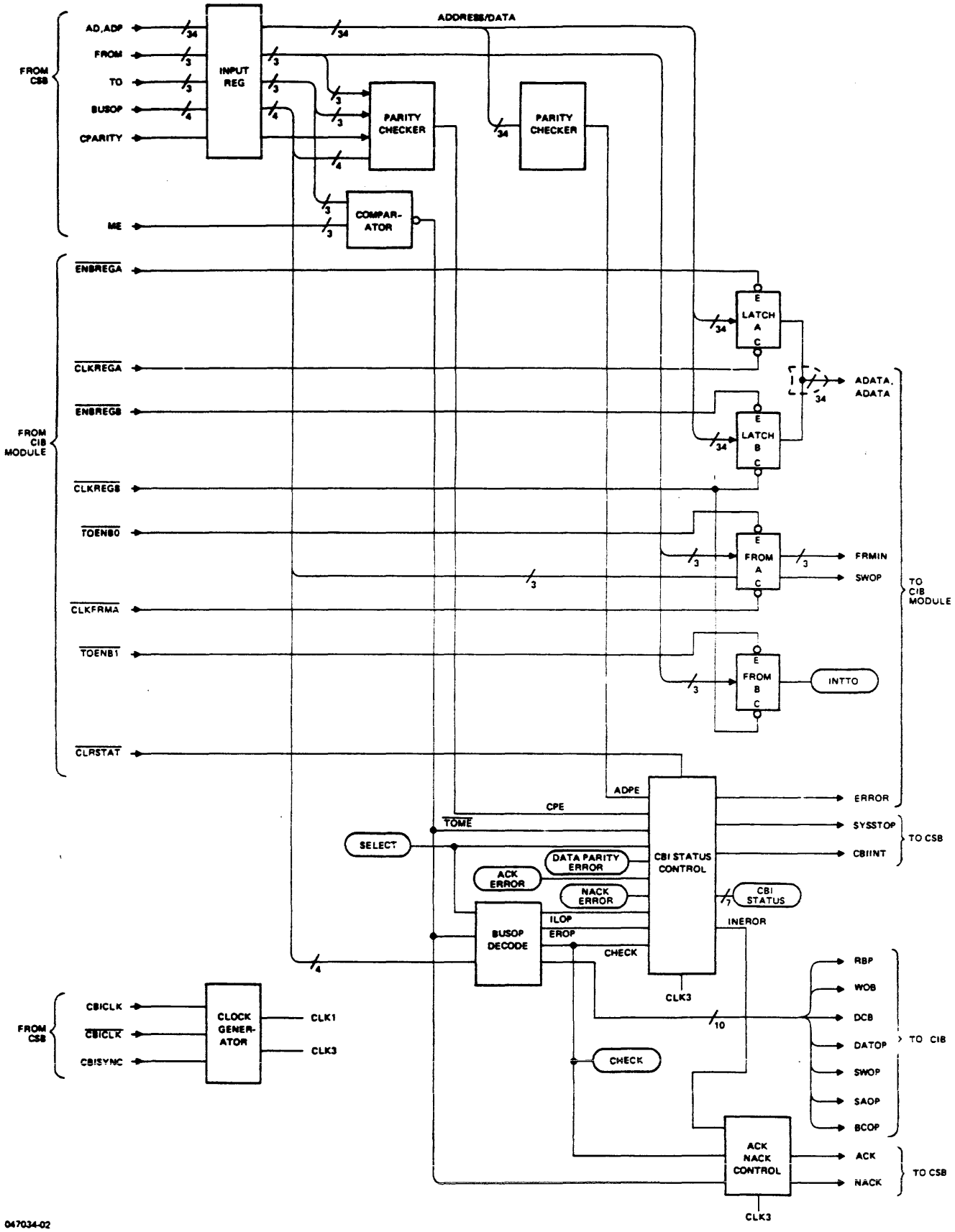


Figure 2-7. CSB Listening Logic on CBI PCA

CBI INPUT REGISTERS AND LATCHES

The information on the CSB is loaded into each CBI PCA every clock cycle. AD00-31, ADP0-1, FROM0-2, and CPARITY are loaded into registers; BUSOP0-3 and TO0-2 are loaded into latches. Parity is checked on the outputs of the registers storing the data (AD00-31 and ADP0-1) and on the outputs of the registers and latches storing the control information (BUSOP, TO, FROM, CPARITY); and if there is an address/data parity error (ADPE=1) or a control parity error (CPE=1), address/data parity error delayed (ADPED=1) or control parity error delayed (CPED =1) are stored at the end of that same cycle on CLOCK3.

The CBI responds to two different sets of BUSOPs; one set requires that the TO code be the same as the CBI's ME code; the other set always has Main Memory's TO code.

BUSOP DECODING

If the TO code and the ME code are equal, the BUSOPs must be one of the following: Data or Error (DATOP = 1), Send Word (SWOP = 1), Send Address (SAOP = 1), and Diagnostic Check Block (DCB = 1). Any other BUSOP will be decoded as an illegal opcode (ILOP = 1).

If the TO code is intended for memory (TO0-2 = 7) and the CBI is not in the Select state, the BUSOPs must be either Read Block Private (RBP) or Write Old Block (WOB). Any other BUSOP will be decoded as illegal opcode (ILOP).

DATA PATHS

There are two sets of latches on the CBI PCA to store information from the CSB. Register A is 34 bits wide to store AD00-31 and ADP0-1; FRMA is a four-bit latch used to store the FROM code and least significant bit of the BUSOP associated with the data in Register A. The least significant bit of the BUSOP is used to distinguish between Send Word and Send Address. Register B is also used to store AD00-31 and ADP0-1, and the FROM code associated with the data.

The outputs of the data and parity portions of Registers A and B are OR-tied and enabled by the module (ENBREGA(L) = 0 or ENBREGB(L) = 0). The module also controls the loading of these latches (CLKREGA(L) = 0, CLKREGB(L) = 0). The FRMA latch is loaded by the module (CLKFRM(L) = 0) and the outputs enabled (TOENB0(L) and TOENB1(L) = 1) to the module (FROMIN0-2, SWORD(L)).

2-25. CBI Error Conditions Status Bits

As previously described, there are seven error condition status bits on the CBI PCA. They can be read during a message by the module, using MUXSTAT; they can also be cleared by the module, using CLRSTAT.

SBACKE

This is the ACK error status bit. If a talking CBI did not receive an ACK (acknowledgement) from the intended destination module, the talking CBI will will set SBACKE and CBIINT. SYSSTOP is set to halt the machine.

SBNCKE

This is the NACK error status bit. If a talking CBI did not receive a NACK from all other CBIs on the CSB the talking CBI will set SBNCKE and CBIINT. SYSSTOP is set to halt the machine.

SBEROP

This is the Error busop status bit. The bit is set when a CBI receives the Error busop. This busop is only generated by memory as a result of an uncorrectable error condition in the data sent to a requesting module. The module that originally requested the data will have SBEROP set. CBIINT and SYSSTOP is also set.

SBILOP

This is the illegal opcode status bit. If some CBI sends a busp that is not defined the result will be an illegal busop. This illegal busop is detected and SBILOP, CBIINT, and SYSSTOP are set.

SBCPE

This is the control parity error status bit. This bit is set if event parity is not detected on CPARIN, FROMIN, BUSOP (incoming busop not in the shift string), and TOCODE (incoming tocode not in the shift string).

SBADPE

This is the address/data parity error status bit. This bit will be set if even parity is not detected on ADIN and PARIN01 on the CBIs. This parity error is detected on data coming in from the CSB.

SBDPE

This is the data parity error status bit. This error is caused by not detecting even parity on the data path from the module to the CBI. The module has the opportunity to enable or disable the test for parity on this data path. Very little time is allowed for checking the parity on this data path so most CBI modules are not able to use this feature.

The CBI can also check parity on data being transmitted from the module to the CBI. If a parity error occurs and the CBI is supposed to check parity, the CBI will store a status bit for the data parity error.

CPU AND CACHE MEMORY

SECTION III

This section describes the CPU and Cache Memory Module of the HP 3000 Series 64 Computer. The CPU and Cache Memory Module accesses the rest of the computer via the Central System Bus (CSB), and has a Common Bus Interface (CBI) PCA to communicate with the CSB. See Figure 3-1.

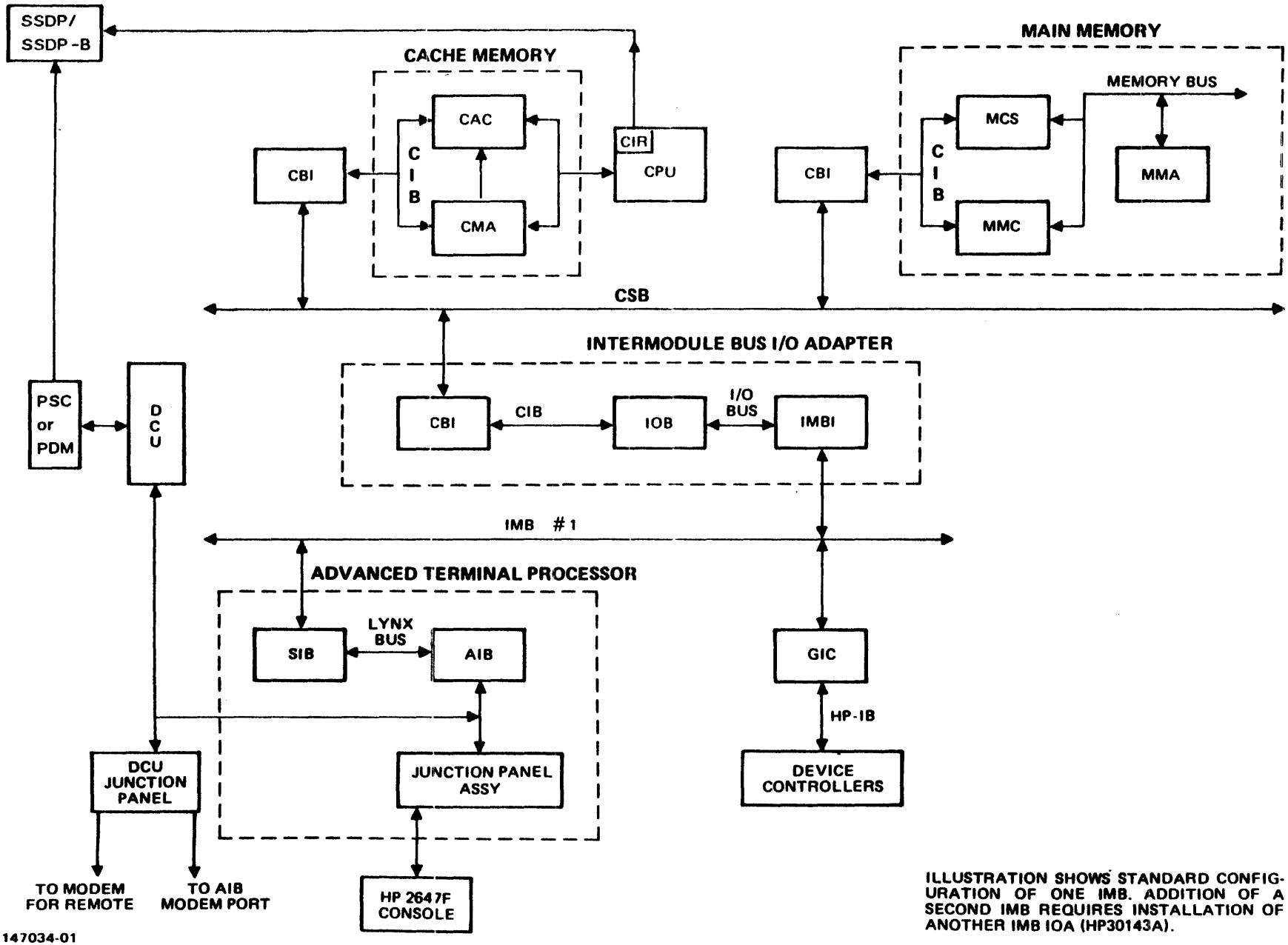


ILLUSTRATION SHOWS STANDARD CONFIGURATION OF ONE IMB. ADDITION OF A SECOND IMB REQUIRES INSTALLATION OF ANOTHER IMB IOA (HP30143A).

Figure 3-1. Major Modules of the Computer

3-1. CPU FUNCTIONAL OVERVIEW

The CPU does the computer's data processing and computing. They take place on the Register and Arithmetic Logic Unit (RALU) PCAs. Each RALU has two 16-bit Arithmetic Logic Units (ALUs) which operate in parallel. See Figure 3-2. The outputs UBA and UBB are routed to the Cache Memory and back to the CPU registers. After processing by the registers, the data can once again be made available to the ALUs.

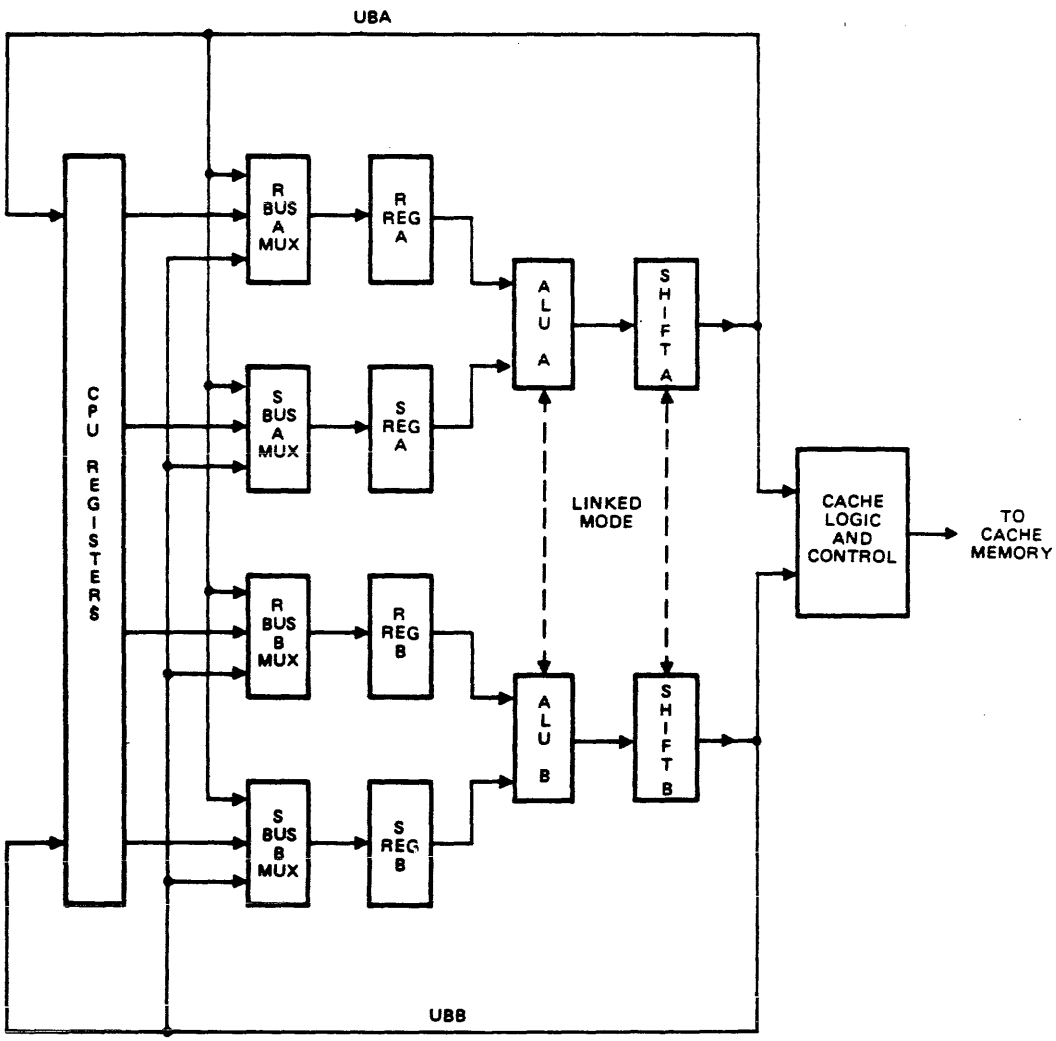


Figure 3-2. ALUs and Inputs

Control signals are generated on the Control A (CTLA) and Control B (CTLB) PCAs. The CTLA PCA has the control logic for the CPU Registers, the operand multiplexers, the RBUS and SBUS, the operand registers RREG and SREG, and the ALUs and shifters. CTLB controls the HDATA, MDATA, Bank, and Y Registers, and multiplexers that interface the CPU to the Cache Memory. For the location of these and other logical circuit blocks, see the HP 3000 Series 64 Computer Block Diagram, part no. 30140-90004.

The CTLB PCA has the System Stop logic, SYSSTOP, which halts the system in the event of an operating error. Control signals for the counters and interrupts are resident on CTLB. The CTLB has system clocks for all PCAs in the computer.

The CPU is driven by machine code and microcode supplied by the computer's Multi-Programming Executive (MPE) operating system. The machine code is loaded into the Next Instruction Register (NIR) from a system input device, via the Cache Memory. See Figures 3-3 and 3-4. The NIR supplies data to the Current Instruction Register (CIR). The CIR supplies the address to the Look Up Table, which generates the next microcode address.

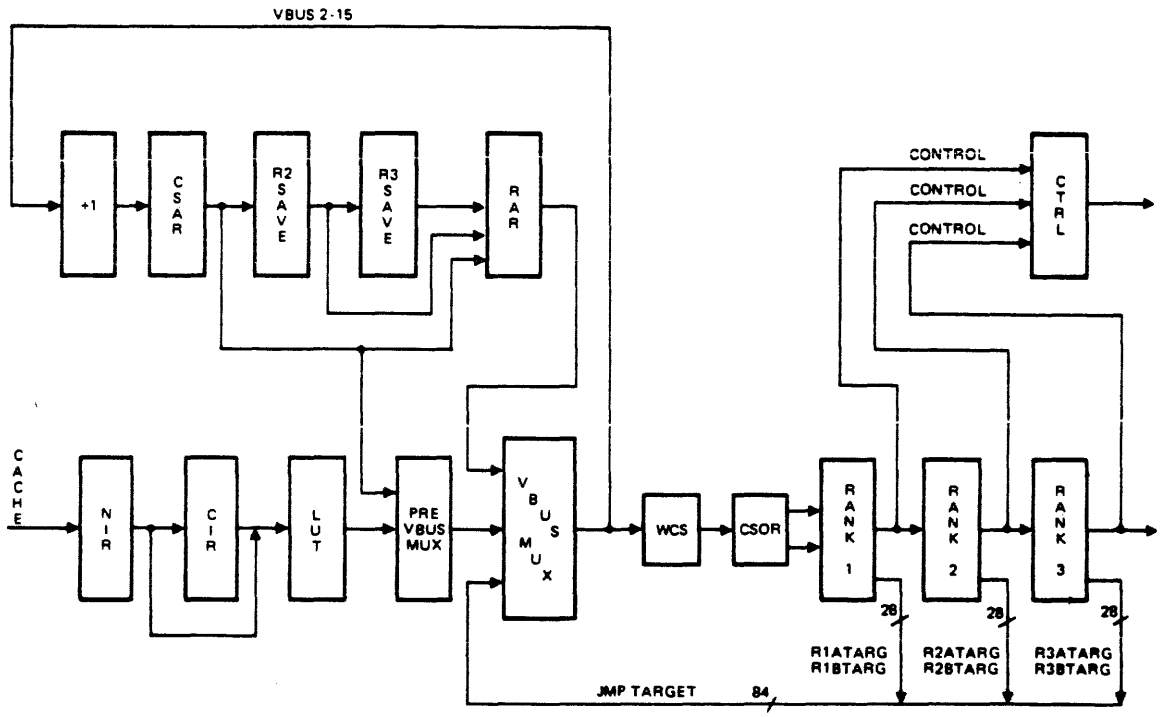


Figure 3-3. Machine and Microcode Pipelines

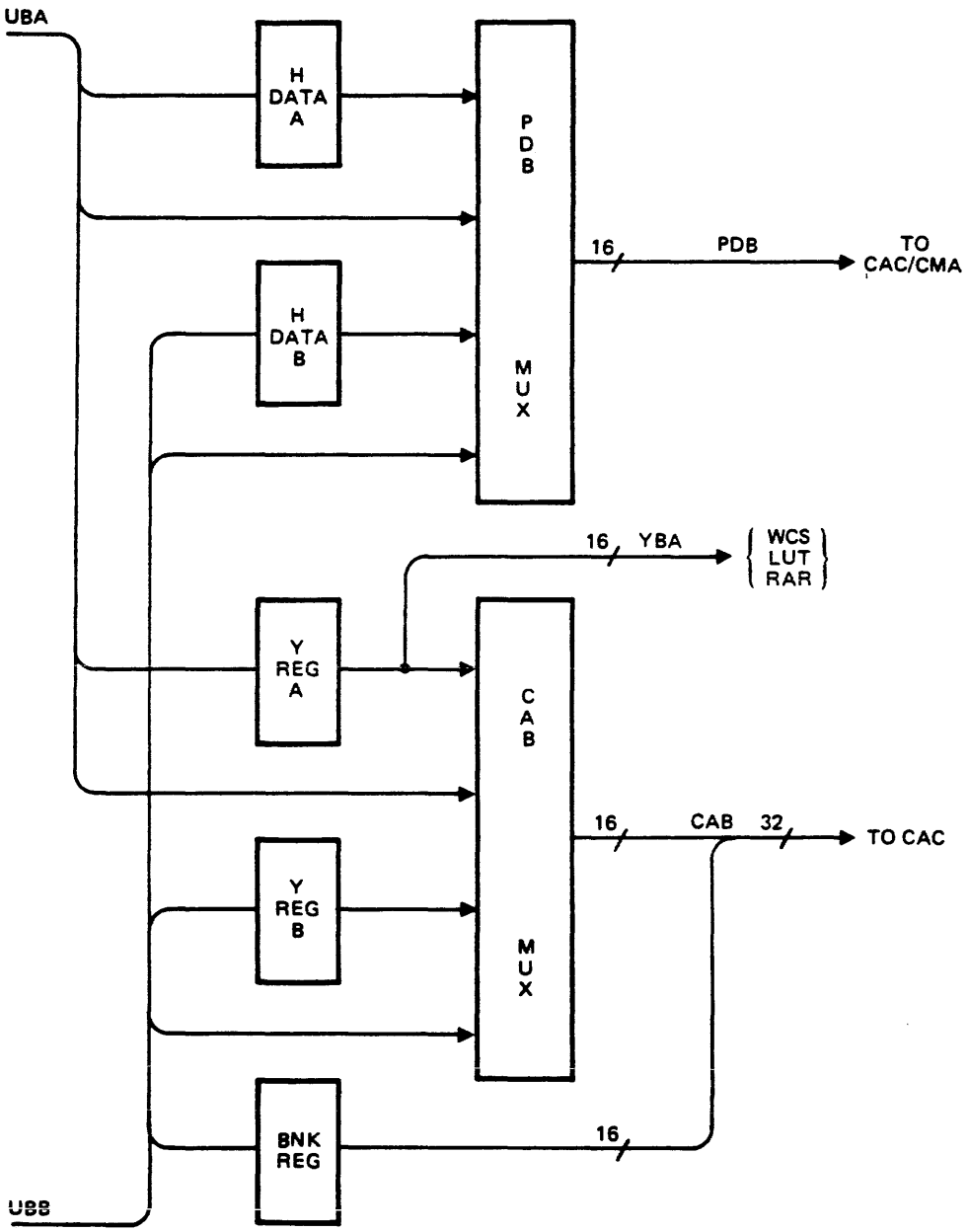


Figure 3-4. Cache Logic and Control

The microcode is stored in the Writable Control Store (WCS). It is loaded from an external device at system initialization and remains unchanged until the system is initialized again. A microinstruction is supplied on each clock cycle in response to an address from the VBUS Mux. The address from the VBUS can be derived from any of nine sources: Ranks 1A, 1B, 2A, 2B, 3A, 3B, the LUT, the Control Store Address Register (CSAR), and the Return Address Register (RAR).

The microcode instruction is processed on the Skip Special (SKSP) and VBUS PCAs. The instruction is first stored in the Control Store Output Register (CSOR), then decoded in the pipeline. The SKSP PCA supplies the control signals for the VBUS Mux. The controls and fields of data decoded are used primarily to determine the control signals generated on the CTLA and CTLB PCAs.

3-2. PROCESSING THE DATA

Data processing takes place in the ALUs of the RALU PCAs. Data to be processed is selected from the CPU Registers by the RBUS and SBUS Multiplexers. The operands are stored in registers RREG and SREG for one clock cycle and then sent to the ALU. The operands are then processed according to the function selected by the CTLA PCA.

The CPU has four RALU PCAs. Each contains two ALUs, ALUA and ALUB, that operate in parallel on the same microcode instruction. Each has two operands: RREGA and SREGA, or RREGB and SREGB. The operands are selected by multiplexers with the signals SBUSA, SBUSB, RBUSA, or RBUSB. The ALUs are similar but differ in operation as each has access to a different set of registers. ALUB has priority over ALUA for operations of the same rank. Both ALUs can access the Top-of-Stack Registers, the SR Register, SP4A, the Environmental Registers, The Cache Data Bus (CDB), and the output of either ALU. Other registers are only available to one ALU or the other.

Each ALU is a 16-bit device and normally operates independently. Two ALUs, however, allow the CPU to perform two operations with each microcode instruction. Alternatively, the operation of the two ALUs can be linked for 32-bit arithmetic.

Control signals for the ALUs come from the CTLA PCA. The control signals are derived from microcode decoded on the VBUS and SKSP PCAs. They select the arithmetic or other function that the ALU will perform on the operands. The 16-bit output from each ALU is available to the A and B Shifters. The shifters can shift the data left or right, swap its bytes, or pass it through unaltered.

All 16-bit data buses and registers are spread across the four RALU PCAs, so that each PCA gets four bits of ALUA and four bits of ALUB. Differences in bit assignments among the RALU PCAs are achieved by backplane and frontplane wiring.

Microcode is stored on the Writable Control Store (WCS) PCAs. There are two identical WCS PCAs, each containing 32 bits of each 64-bit microcode word. The same data, address and control inputs go to each WCS. The PCA in slot 0 of the CPU Bay card cage supplies Bits 0-31 of the microcode; the PCA in slot 1 supplies bits 32-63.

Most controls for the RALU PCAs come from the CTLA and CTLB PCAs. Instructions for the control operations are decoded from the microcode by the Skip Special PCA (bits 0-31) and the VBUS PCA (bits 32-63). The CIR PCA contains the Current and Next Instruction Registers, and the LUT. These supply the next microcode address in the event of a jump to a new macro-instruction routine.

The ALU outputs can be tested for zero, carry, or arithmetic overflow. The outputs of the shift registers can be tested for a positive, negative, odd or even number.

3-3. Data Inputs to the Registers

The outputs of the A and B Shifters are available on UBA and UBB. This data is available to the Processor Data Bus (PDB) from the MDATA Register, which is obtained either from the HDATA Registers (A or B) or directly from UBA or UBB.

The 32-bit Cache Memory address is sent on the Cache Address Bus (CAB). The upper 16-bit word is obtained from the Bank Register. The lower word is selected by the Cache Data Bus Mux from Y Register A or B (YREGA, YREGB), or from UBUS A or B. YREGA is also used to load the WCS and LUT during initialization. The Return Address Register (RAR) is also loaded with the return address from YREGA.

UBUSA and UBUSB transmit data from the shifters to the CPU Registers. This means that data from an ALU can be loaded into another register so that it can be used as an operand in a subsequent operation.

3-4. Other Inputs to the Registers

The Cache Data Bus (CDB) receives data from Cache Memory when a successful read of Cache Memory is made into the NIR. If an error condition occurs, a CSHERR signal to the system stop logic halts the CPU. The type of error can also be determined by examining the CAC shift strings.

The CPU can access the contents of the LUT and CIR via the 16-bit LUTCIR bus from a multiplexer on the CIR PCA. The current microcode instruction can be accessed over the 16-bit WCSSBB bus from the Control Store Output Register (CSOR) on either the VBUS or SKSP PCA.

The system interrupts are stored in the CPX Registers. The function of each signal differs on each RALU PCA.

The Stack Register (SR) tells the number of valid registers in the TOS.

The following CPU registers supply outputs to the operands, and also send their contents to other PCAs in the CPU:

a. CPX 1 and CPX 2. System interrupt data which is transmitted to the Control B PCA for decoding.

b. Counters. The contents of the CTR and CTX registers are sent to the Overhead Multiplexer on the VBUS and SKSP PCAs as an address input to Rank 1 or 2 operations, and can address registers or stores programmatically.

3-5. The Machine Code

The machine code of HP's Multiprogramming Executive (MPE) operating system comes from the Cache Memory through the Paired Stackop Multiplexer (Mux) to the Next Instruction Register (NIR) of the CIR PCA. The Paired Stackop Mux handles situations where two stack operations are contained in one microcode instruction.

A portion of the output of the NIR is the address for the Look Up Table (LUT). The LUT output is used for the beginning address of a microcode instruction sequence. This occurs at the end of a current instruction sequence, signalled by a NEXT operation. NEXT allows the next machine instruction to be clocked into the CIR for the next sequence. Other bits of the LUT output are used for SR Preadjust to determine the number of valid TOS Registers in an operation.

During initialization, the LUT is loaded from the RALU via the YBUSA under the control of a decoder using Store Field A. The LUT is 32 bits wide. YBUSA is only 16 bits wide, so each 32-bit LUT "double word" is loaded in two operations.

3-6. The Microcode Store and Pipeline

The microcode store is loaded when the CPU is initialized. Data is input via the RALU YREGA and the YBUSA lines. Once loaded, the microcode is not changed during normal operation.

The microcode is stored on two Writable Control Store (WCS) PCAs, each having 32 bits. The microcode address is output by the VBUS Mux and transmitted to the WCS PCAs. Each half of the microcode word has the same address on each PCA, so the 13-bit VBUS bus addresses both PCAs simultaneously. The 64-bit instruction resident at that address is output to the SKSP PCA (first 32 bits) and the VBUS PCA (second 32 bits). This 64-bit instruction is called Rank 0.

The microcode is loaded into the Control Store Output Registers (CSORs) on the VBUS and SKSP PCAs. The output of the CSOR is the instruction for Rank 1. Register fetches and unconditional jumps are determined by the microcode instruction. Anything not completed by the end of Rank 1 is stored in registers to be executed in Rank 2.

During Rank 2, the ALU operation is performed and the contents of all stores are transmitted to the registers, non-data dependent jumps are performed and data dependent conditions are tested.

Most operations are completed in Rank 1 and 2. Thus, the only ones performed in Rank 3 are data-dependent jumps. Microcode and the LUT are also loaded during Rank 3, while initializing the system.

The microcode address is either the next store field address to continue the routine, or a jump to another routine. In the first case, the current microcode address is incremented by one to fetch the next microcode instruction. This address is then stored in the CSAR register. Previous microcode addresses are saved for two ranks in the R2SAVE and R3SAVE Registers on the VBUS PCA, so the CPU can load the Return Address Register (RAR) with these addresses. The Return Address Counter (RAC) can be incremented or decremented under the control of the SKSP PCA to manipulate the microcode return address stack in the RAR.

Microcode skips and jumps are decoded by the jump logic on the SKSP PCA. If a skip is indicated, the Skip Logic will cause a "No Operation" (NOP) of Rank 2 execution of the next microcode line in the same ALU.

The SKSP PCA jump logic controls the next address as selected by the VBUS Mux. Generally, Rank 3 takes precedence over Rank 2, and Rank 2 takes precedence over Rank 1, as Rank 3 is the oldest instruction. ALUB normally has priority over ALUA. During memory references however, ALUA has priority over ALUB.

3-7. The Control PCAs

There are two control PCAs that produce control signals for the RALUs. These control signals are based on the data decoded from the VBUS and SKSP PCAs. The CTLA PCA contains ALU control; shifter control; RBUSA, RBUSB, SBUSA and SBUSB Mux controls; register controls; carry lookahead; zero detection; bounds violations; and store decode signals. The CTLB PCA holds Cache Access logic; SR and Namer logic; CPX Interrupt logic; and Freeze and SYSSTOP logic. The CTLB also has clock signals for all PCAs in the computer.

3-8. System Clocks

The system clock on the CTLB PCA is derived from an oscillator. The oscillator output is divided by two, then transmitted to every PCA in the computer. Each PCA receives this Two-Times-Clock (2XCLK) signal and its complement from the CTLB, and a synchronizing signal (SYNC) from the Diagnostic Control Unit (DCU) PCA. Timing circuits on each PCA divide the 2XCLK by two, using SYNC to determine the phase. There is a separate SYNC signal for each PCA in the computer; therefore, the DCU can allow clocks to occur on any PCA or combination of PCAs.

3-9. CPU PCA-LEVEL FUNCTIONAL DESCRIPTIONS

The following paragraphs provide functional descriptions of the CPU logic circuitry, at the PCA-level.

3-10. Current Instruction Register (CIR) PCA

The CIR PCA takes the machine code from the Cache Data Bus (CDB) and stores it in the NIR. The contents of the NIR are passed to the CIR, and a portion of this code is used as the address for the Look Up Table. There are two forms of input to the NIR: a 16-bit input and a Paired Stackop in which two separate macro-instructions are put into one 16-bit macro-instruction. This speeds up the operation of the computer, and saves memory space.

The LUT is loaded from YREGA during initialization, and stores the VBUS address for a jump to a new instruction routine. It also has the SR Preadjust, Displacement Field, and decoders for the X Register and base. The LUTCIR Bus allows the RALU to look at the contents of these stores.

3-11. PAIRED STACKOP MUX. The Paired Stackop Mux is used to select certain bits of the Cache Data Bus for transfer to the NIR. First, the microcode option NIRS is low and the Mux selects CDB04-15 for transfer to the 04 thru 15 input lines of the NIR. Then, if there is a right STACKOP, the Overhead line issues a "Read to NIR Shifted using Blank P" (RNSP) so the same instruction is read again with NIRS set, and the CDB is shifted. NIRS is set high and CDB10-15 are shifted through the Mux to the 04-09 input lines to the NIR. Bits 10-15 to the register are held low. CDB00-03 are fed directly into the NIR.

When the Paired Stackop function is not required, CDB00-15 are passed directly to the input of the NIR.

3-12. NIR AND CIR. The NIR and CIR hold the next and current microcode instructions. Each register is 16 bits wide. The NIR is loaded from the CDB and Paired Stackop Mux after an RONP or RNSP Special microcode option, when TONIR from the CTLB PCA is high. NIR is transferred to CIR when NEXT is in Rank 2, if no interrupts are pending. Bits 00-03 and 10-15 from the NIR go to the Stackop Pending Decoder, and Bits 00-11 from the NIR are used as the address for the Look Up Table. These two functions occur simultaneously.

The CIR is loaded when NIRTOCIR(L) (from CTLB) is active, and decoded into CIRS1 and CIRS2 by the CIR selector. When FRZ is active, the CIR is placed in the hold mode to prevent it from changing until the freeze is lifted.

3-13. LUT STORE AND PARITY CHECKER. The LUT is a set of thirty-two 4K by 1k RAMs. The format of its 32-bit, double-word output is:

bits 0-2	Not used.
bits 3-15	CIRVBUS. This points to the starting microcode instruction of the next software instruction.
bits 16-18	SR pre-adjust code. This specifies the minimum number of the Top-of-Stack Register that must be valid to start the next instruction.
bit 19	NIRSUB. Used to set or clear F2 (flag 2) on the Overhead line.
bits 20-22	Displacement Field Code. These determine which CIR bits are used for code displacement.
bits 23-24	Base Register Code. These specify the base register related to the addressing. This will be P, Q, DB, or SM.
bits 25-26	Index Conditional Code. The Index Conditional Code (XC) specifies the type of indexing for the software instruction: normal, double word, by bytes, or none.
bit 27	LUT parity bit to parity checker.
bit 28	Indicates whether the software instruction is direct or indirect.
bits 29-31	Not used.

The 32-bit double words are written into the LUT 16 bits at a time, from the YREGA on the RALUs. The STF3A5-7 signals from the CTLA are decoded into LUT0 and LUT1 commands to load the first 16 bits into positions 00-15, and the second 16 bits into positions 16-31. The load commands are timed by the system SYNC signal.

The parity checker looks at all 32 bits from the LUT store and generates LUT parity error (LUTPE) to the CTLB PCA if the LUT word parity is even. In that case, bit 27 is written into the LUT to make the parity odd. The checker is enabled by the signal LPEN from the LUT control decoder. The checker is disabled when TONIR from the CTLB PCA is active,as this changes the LUT store address and its output will be invalid until the address has settled in the NIR. LUTPE is held in a register for one clock cycle so that only valid parity errors go to the CTLB to cause a SYSSTOP. The same register also has signals OHD(L) and SHOPRF. OHD(L) selects the proper inputs for the Index Registers XC0 and XC1.

3-14. SR PRE-ADJUST ENCODER. This encoder compares DESR0-2 (the next value of ESR) from the CTLB with bits 16-18 of the LUT store (SRP0-2) to generate SRPREQ for the CTLB. If there are not enough TOS Registers valid to start the next instruction (SRP0-2>DESR0-2), an SR pre-adjust is required. In that case, the Overhead line is modified to jump to a microcode routine to bring another word from memory into the TOS Register. The Overhead line is then repeated, and tested again for SR Preadjust. If there is a Bounds Violation (BNDV), SR Preadjust is prevented, so the Overhead line will branch to the interrupt service routine.

3-15. DISPLACEMENT FIELD DECODER. This decoder uses bits 20-22 from the LUT store and CIR8-11 to generate DISPL8(L)-11(L)to the LUT/CIR Mux. If the microcode asks for the displacement (DSPL) in SREGA, it will get the four to eight least significant bits from the CIR. These normally are used to give the addressing displacement from a base register within the instruction. They can also be used for an immediate operand or to get the last four bits of the instruction to finish decoding.

When selected, the four bits are loaded into position 8-11 of the LUT/CIR Mux. The Displacement Bits are updated at the beginning of each software instruction.

3-16. XC/BASE DECODER AND MULTIPLEXER. This circuitry controls the X (Index) Register and X Mux. The circuitry uses OHD(L) to select either the stored bits for the instruction or the combined bits from the LUT.

3-17. LUT/CIR MULTIPLEXER. This multiplexer (mux) uses SFA6 and 7 from the SKSP PCA to transfer CIR00-15, LUT00-15, or the displacement field to the RALU, via LUTCIR00-15.

When the displacement field is selected, DISPL8-11 is combined with CIR12-15 to produce LUTCIR8-15. LUTCIR00-07 are held low in this case.

3-18. SSDP MUX. The SSDP Mux normally selects CIR00-15 for display. When SHOPRF from the LUT control decoder is active, the contents of the Performance Register (PERF) are displayed. The output from the performance register remains selected until SHOPRF is reset by STF3A5-7 in the LUT control decoder.

3-19. STACKOP PENDING DECODER. This circuitry decodes NIR00-03 and NIR10-15 into Paired and Paired(L) signals to the RALU and VBUS PCAs for paired and stack operations. When the Shift Register Preadjust Request (SRPREQD1) signal is true, OHDINC1-2 increments the jump targets on the Overhead line.

3-20. CLOCK CIRCUITS. The CIR PCA clock circuits receive CIRCLK and CIRCLK(L) from the CTLB PCA and CIRSYNC from the DCU. CIRCLK is a 2X clock signal. CIRSYNC is phased to gate every other positive transition of CIRCLK. The result is 70-nanosecond clock signal, used to drive the logic on the PCA.

3-21. The V Bus PCA

The VBUS PCA supplies the microcode address of the instruction to be used in the next CPU operation to the WCS. This address is obtained from one of nine inputs to the VBUS Mux.

The VBUS PCA also decodes function fields, special fields in ALUB, store fields, and R Fields in ALUA and ALUB from the the least significant half of the microcode word.

3-22. VBUS MUX OUTPUT. The Next microcode address output from the VBUS Mux to the WCS PCAs via VBUS02-15. The VBUS Mux output is selected by the VSEL and VSEL(L) signals from the SKSP PCA. This output is also sent to incrementing logic on the VBUS PCA. There the microcode address is incremented by one and clocked into the Control Store Address Register (CSAR).

3-23. VBUS MUX INPUTS. There are nine inputs to the VBUS Mux. The first two are preselected by a pre-VBUS multiplexer:

a. The CSAR output, which is the previous address from the VBUS Mux incremented by one.

b. A new address derived from the LUT on the CIR PCA in response to a NEXT.

The other seven VBUS Mux inputs are:

a. The Return Address Register (RAR). This is a 16-bit register file organized as a stack that can store up to 16 levels of microcode subroutine. The RAR is addressed by the Return Address Counter (RAC).

b. Jump Targets. The targets for microcode jumps are decoded from the Control Store Output Register (CSOR) for ALUB, or from from the OHD Mux for ALUA. These are held in the registers for two more cycles for medium speed and slow speed jumps. Thus R1ATARG and R1BTARG come from the current microcode instruction; R2ATARG and R2BTARG come from the previous instruction; and R3ATARG and R3BTARG come from the instruction before that. All of these addresses are possible inputs to the VBUS Mux.

3-24. THE OVERHEAD MUX. If the next instruction follows the last (i.e., if there are no Jump to Subroutine commands before the next instruction), the microcode address sent to the WCS via the VBUS Mux is the address from the CSAR incremented by one. Whether a jump or skip is required is determined by the Overhead Mux.

3-25. STORE FIELDS. During normal operations, WCS39-43 bits represent Store Fields A and B. They are used to determine which registers store the data from ALUA and ALUB. They are extracted from the microcode in Rank 2.

3-26. The WCS PCAs

The WCS holds the CPU's microcode. The microcode is loaded when the system is initialized and remains in WCS until the next initialization.

The CPU has two WCS PCAs, each storing 32 bits of microcode. The PCAs are addressed in parallel and they read out the microcode in parallel. A distinction is made between the PCAs when new data is being written in, as data is input 16 bits at a time.

3-27. THE WCS MEMORY ARRAY. The WCS memory array consists of static RAMs that do not need to be refreshed. The first two 16-bit words to enter WCS are loaded on WCS 0; the third and fourth words are loaded on WCS 1. The PCA address is determined by the CPU backplane.

3-28. LOADING THE WCS. Data intended for the WCS is received from Y Register A. Four load sequences are required to load each 64-bit microcode instruction. Data location is determined by the Write Enable and WCSONE signals. WCSONE is a high output which is fed back to the WCSHO input on PCA 0 but not on PCA 1. The Write Enable determines where the YBUSA data will be written. WCS0-15, 16-31, 32-47, and 48-63, STFA5-7, and WCSH0 must be interrogated by each PCA. STFA5 must be low for a write to occur. STFA6 is compared to WCSH0 and, if they are opposite,

then write will occur on the PCA. If the signals are the same, write is disabled. If these conditions are met, STFA7 is used to determine whether word 0;2 (STFA7 low) or word 1;3 (STFA7 high) is written.

3-29. READING A MICROCODE INSTRUCTION. When a microcode instruction is to be read, its address is received from the VBUS PCA. The bits addressed are read out of the RAMs over the WCS Bus, the most significant bits (00-31) from PCA 0 and the least significant bits (32-63) from PCA 1. All bits are output simultaneously. The most significant bits are transmitted to the SKSP PCA and the least significant bits to the VBUS PCA.

3-30. Skip Special (SKSP) PCA

The SKSP PCA contains most of the CPU microsequencer control logic. It processes conditional jumps and skips, handles the microprogrammable hardware flags, and selects jumps and Returns from Subroutine (RSBs) for the next microinstruction. In addition, it receives the 32 most significant WCS bits and processes them as required. It also does most of the decoding of the Special and Skip microcode fields.

3-31. MICROSEQUENCER CONTROL. Usually, each microinstruction is followed by the one with the next higher address. The three exceptions are JSBs, RSBs, and the start of new software instructions.

JSBs specify the address of the next microinstruction to be executed, which is in the part of the WCS word on VBUS. RSBs call for the address of the microinstruction following the one that last executed a JSB (exception; see PSHR/POPR). A new software instruction can be started from a microcode NEXT option, or as a side effect of a bounds violation. In that case, the LUT (Look Up Table) specifies the address to be executed after execution of the hardware Overhead line.

SKSP determines which parts of the microcode line should not be executed, either because of a jump or skip, or because a microcode field is part of a literal or jump target and cannot be used. It suppresses execution with various NOP signals; NOP2A, NOP2B, SKNOP2A, and SKNOP2B.

3-32. JUMP AND SKIP CONDITIONS. For every JSB there is a conditional test and the JSB is only taken if the condition is true. If the condition is true, stores, special options, and skip conditions are disabled in the next microcode line (IN THAT ALU ONLY).

There are three types of JSBs: unconditional ("fast"), non- data-dependent ("medium"), and data-dependent ("slow").

Unconditional JSBs are resolved in Rank 1. CSOR is checked for these "fast" JSBs, because there are none in the Overhead line. Thus, Rank 1 JSBs and RSBs must be inhibited if: a. the Overhead line is being executed; b. there is a non-data dependent skip in that ALU on the previous line; c. there is a data-dependent skip in that ALU on the previous line.

"Medium" JSBs are executed in Rank 2. Non-data-dependent tests are either tests of the value of SR, or of whether the microcode flags are true or false.

Data-dependent ("slow") JSBs are executed in Rank 3. Most data-dependent skips depend on an ALU output, so the information is not available until the end of Rank 2. A true data-dependent skip will directly set NOP2 and SKNOP2 in the appropriate ALU to inhibit Rank 2 execution of the following microcode line in that ALU.

3-33. JUMPS TO SUBROUTINES. JSBs are specified in the Function Field. If the skip condition is true, the next line of microcode to be executed will be at the target address. In ALUA, the target address is encoded in the last three bits of FCNFA and in STFA, so STFA is always NOP'ed on any microcode line with a JSB in ALUA. In ALUB, the target address is encoded in the last three bits of FCNFB and in SPFB, so SPFB is always NOP'ed on any microcode line with a JSB in ALUB. If FCNFB(2,3) = 00, there is a long target address for the JSB. In that case, RFB is also used for the address, and RREGB is loaded with a 0.

3-34. RETURNS FROM SUBROUTINES. RSBs are specified in SPSKFA or SKFB, and are checked at the output of CSOR, since there are no RSBs on the Overhead line. RSBs cause the microcode to continue execution at the line after the one that last had a JSB, unless the RAR was manipulated using PSHR, POPR, or LDRAR. RSBs are inhibited if:

a. the Overhead line is being executed;
b. there is a true non-data-dependent skip on the previous line;
c. there is a data-dependent skip on the previous line.

3-35. JUMP PRIORITY/ VBUS SELECTION. When more than one JSB or RSB is valid at one time, the microsequencer must decide which to take. Highest priority goes to the one that originated from the earliest microcode line. This is the one from Rank 3 (the "slow" jump). Next is the medium speed Rank 2 JSB or RSB. Lowest priority goes to the unconditional JSB and RSB. When in the same line, JSBs in ALUB have priority over those in ALUA. JSBs in either ALU have priority over RSBs.

Once the source of the next microcode address has been chosen, the choice is sent to the VBUS Mux via the VSEL lines.

3-36. NEXT OPTIONS. NEXT is a Skip Option which is available in both ALUs. NEXT forces the appropriate action to start the next software instruction. It causes the transfer of ESR to SR, ENAMER to NAMER, and, if no interrupt or SR preadjust is pending, NIR is transferred to CIR. On the following clock cycle, the Overhead line is executed. During Rank 2 execution of the Overhead line, CTX, CTR, F1, F3A, F4B, TFF, and RFLAG get reset to zero. Any JSB, RSB, or true non-data-dependent skip in the other ALU will override the Next, but true medium speed and slow JSBs will not prevent the transfer of SR, NAMER, or CIR.

3-37. FORCENEXT (BNDV). If a Bounds Violation (BNDV) is detected, the current instruction is aborted, and execution of the Overhead line is forced to give control to the Interrupt Handler. A TEST is then executed on the Overhead line. In this case, TEST is true, and transfers control to the Interrupt Handler microcode. To prevent damage from the violating instruction, all NOPs are invoked for Rank 2, while the Overhead line is in Rank 1, and the transfer to SR, NAMER, and CIR that usually accompanies a NEXT is prevented.

3-38. REPEAT LOOPS. By using the Repeat on Condition (REPC) or Repeat N Times (REPN) function in FCNFB, a microcode line can be repeated. These options freeze the line in Rank 1 until the specified condition is met, or for the specified number of times, as explained in the following two paragraphs.

3-39. REPEAT ON CONDITION. If there is no true skip condition on the line with a REPC, the following line will be repeated until a true skip condition is evaluated in ALUA or ALUB. At that time, one extra copy of the repeated line is about to be executed in Rank 2, so the line is NOP'ed. This is accomplished by freezing the repeated line in CSOR and the address of the next line to be fetched in CSAR.

3-40. REPEAT N TIMES. REPN works like REPC with the following exceptions: The microcoder specifies a number, N, for the number of times the line is to repeat. The microassembler decrements this by two and loads it into SPFB on the line with the REPN. When the REPN line is executed, SPFB is loaded into CTR. When the repeated line has a true skip condition, the repeated line is executed one final time (instead of being NOP'ed). Thus, if the repeated line has a special option of DCTR (Decrement CTR) and a skip condition of CTR0 (true if CTR=0), the repeated line will occur N times.

3-41. RANK 2 NOPS. The microsequencer sometimes must prevent sections of microcode from executing, either because a skip was invoked, a medium or slow JSB brought invalid microcode into the pipeline, or part of the microcode line was used for a JSB target or literal. This is typically done in Rank 2 when stores, JSBs, skips, and most specials are evaluated or executed.

3-42. Control A (CTLA) PCA

CTLA PCA logic circuitry includes Function Field decoders, ALU Shift Control decoders, ALU function control decoders, and ALU register control decoders.

3-43. FUNCTION FIELD DECODERS. Function Fields in ALUA and B from the SKSP and VBUS PCAs enter the CTLA PCA. The decoders act as a first-stage decode for FCNFA0-4 and FCNFB0-5. These signals generate enables (EA and EB) with SHFTA and FNXA for the ALU Shift Register, the multiply and divide (MPAD and DVSB) signals, and the linking and shift signals.

3-44. ALU SHIFT CONTROL DECODERS. The ALUA and ALUB Shift Control Decoders generate the control signals for the ALUA and ALUB Shifters. The decoders use SHFTA, FNXA, FCNFA5-7, FCNFB5-6, and LNKALL from the FCNFB pre-decoder as their 8-bit input address. They output 12-bit words. Ten of the 12 bits are used for the ALUA Shift Control commands to the ALUA shift registers. These tell the ALU to do a shift, to swap bytes, or to pass the words straight through.

The ALUBSC commands are formed in a similar manner. The decoders use LNKALL, SHFTB, FCNFB4-6 and FCNFA5-7 as their input addresses. The 12-bit word is held in a register for one clock cycle and 10 of the bits are used for the ALUBSC commands. The remaining two outputs of the registers are used in the DCU shift string.

3-45. ALU FUNCTION CONTROL DECODERS. The function control decoding for the ALUs is performed by two ALUAFC decoders. The first decoder is always enabled. It uses FCNFA0-6 as its input address and delivers 4-bit function commands ALUAFC0-3 to ALUA. The second ALUAFC decoder is enabled when LNKALL is decoded in the shift control logic. This links ALUA and ALUB and sends them both function commands decoded from FCNFA. When the ALUs are not linked, FCNFB0-6 are decoded by the ALUBFC.

3-46. ALU REGISTER CONTROL DECODERS. These decoders receive NAMER0-2 (the three-bit logical address of the top-of-stack) from the CTLB PCA and subtract it from SFA0-2, SFB0-2, RFA0-2 and RFB0-2 from the SKSP PCA, to generate the physical addresses of the TOS A and B Registers.

The decoders also decode the four-bit select commands for the RBUSA, RBUSB, SBUSA, and SBUSB multiplexers on the RALU PCAs, from the SFA and B and RFA and B fields. These are fed from the CTLA PCA to the RALUs as SFAS0-3, SFBS0-3, RFAS0-3 and RFBS0-3, to select the register contents for the ALU operand. The SFA and SFB fields are also decoded into HSREG and OP1 signals. The Hold S Register signals are used elsewhere on the CTLA. The OPA1 and OPB1 signals modify the select commands and also go to the CTLB PCA. If either OPA1 or OOPSA is high and TOOPA(L) is low, SRAS0-3 will be high. OPB1, OOPSB and TOOPB(L) perform a similar function on SFBS0-3.

Bits 4 and 5 of SFB also go to a mux which places a high level on one of its four output lines, BITM0-3. BITM0-3 go to the RALUs where SFB6 and 7 are combined with them to mask any of the 16 bits of the SBUSB.

The R fields are also decoded into hold R reg commands and RALU R bus mux select lines. The RFA decoder PROM uses RFA0-3, XC0, XC1 and FCNFA0 as its input address. XC1 is buffered out as XCS. RFA0 is buffered out as RRAENA to enable the R bus A mux on the RALUs. Similarly, HRREGB and RFBS0-3 are decoded from RFB0-3 and FCNFB0 and 1. And, RFB0 is buffered to the RALUs as RRBENA.

3-47. Control B (CTLB) PCA

The CTLB PCA, like the CTLA, contains CPU control logic circuitry. In addition, the CTLB contains the computer's system clock, which drives the clock circuits on each individual PCA. The CTLB's control logic circuitry includes system interrupt and SYSSTOP logic, SR and NAMER control logic, RALU output multiplexer select logic, Y Register control decode logic, CPX interrupt logic, and NIR control logic.

3-48. SYSTEM CLOCK GENERATION AND DISTRIBUTION. The CTLB generates the computer's system clock and distributes this signal to all PCAs in the computer. An oscillator generates a frequency of 57.14 Mhz. This frequency is divided by two and is outputted to each PCA in the computer. Each PCA then has its own divide-by-two circuitry. The result is a clock signal with a 70-nsec period on each PCA.

3-49. INTERRUPTS AND SYSSTOP LOGIC. These circuits halt the computer by issuing a SYSSTOP to the DCU PCA. Several conditions cause the computer to halt. The first of these are the Run Mode Interrupts INTA and INTBD1-3, which come from the Interrupt Registers on the RALU PCAs. Other interrupts result from parity errors in the LUT or WCS, an error from the Cache Memory, or an ALU overflow bit from the Status Register on the RALU PCA. Other interrupts include DCU Interrupt, Power Fail Warning Interrupt, and CPU Reset Interrupt.

3-50. SR AND NAMER LOGIC. These circuits control the Top-of-Stack (TOS) Registers on the RALU PCA. The SR0-2 signals indicate how many words in the stack have valid data. The Namer signals NAMER0-2 convert logical addresses to physical addresses.

Two copies of SR and NAMER are stored in the logic. The extra copies, called ESR and ENAMER, are temporary. They permit changes to be done to the temporary copies during an instruction cycle without changing the permanent copies.

3-51. RALU OUTPUT MUX SELECT LOGIC. This circuitry decodes SPSKFA0-5 and SPFB0-5 into a variety of requests that are stored, queued if necessary, and then prioritized into commands to the RALU and Cache Memory PCAs.

3-52. Y REGISTER CONTROL DECODE LOGIC. This is the control logic for YREGA and YREGB. YRPA1-3 and YRPB1-3 are the Y Register propogate lines, and indicate which RALU PCA the signals come from. These combine with the decoded signals from READ, FLAG, and BSEL which encode the carry logic. The outputs DENA and CBNA are active when an increment or decrement of the register is required.

3-53. CPX INTERRUPT AND NIR LOGIC. Interrupts to the CPX Interrupt Logic include INTBD1-3 in Run Mode, interrupts from the DCU, RUNHLTINT (Run Halt Interrupt), PFWINT (Power Fail Warning Interrupt), CPURSTINT (CPU Reset Interrupt) and Overflow from the CPU Timer. The Run Mode Interrupts are gated by RUNFF, PWRFINFF and SYSHLTFF inputs. The outputs are STORSTAM which enable a Status Register load, and NIRTOCIR which transfers a machine instruction from the NIR to the CIR.

3-54. RALU PCAs

The Register and Arithmetic Logic Unit (RALU) PCAs contain registers and Arithmetic Logic Units (ALUs) to process the data in the CPU. After processing, the data can be routed to the Cache Memory, to other CPU PCAs, or directed back to the registers on the RALU PCA for storage and further processing. Control for RALU functions comes from the CTLA and CTLB PCAs.

There are four RALU PCAs; they are identical and interchangeable. All 16-bit data buses are bit-sliced across the four PCAs, with bits 0-3 on RALU 0; 4-7 on RALU 1; 8-11 on RALU 2; and 12-15 on RALU 3. Some inputs have different functions on the different RALUs; which function is performed is determined by the slot position of each RALU PCA in the CPU Bay card cage.

3-55. RALU DATA PROCESSING. Data to be processed is selected by the RBUSA, SBUSA, RBUSB, and SBUSB multiplexers, then stored in the RREGA, SREGA, RREGB, and SREGB registers. Each contains a 16-bit operand for an ALU. Data is multiplexed and loaded into the RREG and SREG registers between Rank 0 and Rank 1. At the same time the control inputs to the ALUs, ALUAFC0-3 and ALUBFC0-3, are stored in the ALU Control Register for one clock. Then the ALUs are set up to process the data.

During Rank 1, the operands from RREG and SREG are processed by the ALUs and appear on the outputs. The data is then applied to Shifters A and B.

The shifters, which are under the control of ALUASC0-9 and ALUBSC0-9 from the CTLA, shift right, left, and perform byte-swapping. The outputs of the shifters are referred to as UBA, from ALUA, and UBB, from ALUB.

The UBUS information is available to the Cache Memory, via the Cache Processor Data Mux and the Processor Data Bus (PDB). The Hold Data A (HDATAA) Register holds data from UBA, and the HDATA B Register holds data from UBB for the Cache Processor Data Bus (PDB). This data is held for one clock in the MDATA Register. The HDATA Registers hold queued requests; Rank 2 requests are obtained directly from the UBUS. The Cache Processor Data Mux selects data from HDATAA or HDATAB Registers, or data directly from UBA or UBB, as determined from the CTLB PCA.

The Bank Register is loaded with the upper 16 CAB address bits from UBB, CAB00-15. This is a register file which has one input and two output ports. The write address is STFB5-7. The read addresses are MBANK0-2 which reads the output to the CAB, and SFB5-7 which reads out the data for the SB operand. The lower 16 bits of the Cache Address Bus, CAB16-31, are selected by the CAB Mux.

When addressing the Cache Memory, the YREGA and B registers are counters which count up for the next word, or count down for the previous word. This output is available to the CTLB PCA which returns the YRCA and YRCB0-3 signals as a lookahead when a decrement or increment is expected.

The registers P,D,S and X3-7 specify the sixteen most significant bits of a memory address. The register addressed depends on the microcode. The outputs are selected by either the MBANK for the Cache output, or the SFB for the Bank output into the SBUSB Mux.

Data on the UBA and UBB are available to the following registers: UBA supplies data to SP4A, STA, X, SPOA, XTRA Registers, and to both R and S MUXes.

UBB supplies data to TCLK, SP3B, Z PL, CTRS/CTX, RBUS, SBUS, TOS, and to the Environmental and Bank Registers.

3-56. ALU INPUTS. The following are the major inputs available to the ALUs:

a. LITA00-15. Literal A. This is comprised of Special Skip Field A, RBUSA, and Function Field A. It goes to the RREGA Mux.

b. CDB0-15. Cache Data Bus. This bus receives data from the Cache Memory. Data can be sent directly to the SA or SB Mux as an operand. Alternatively, the data can be stored in the OPA or OPB Register under the control of the TOOPA(L) and TOOPB(L) signals, and then sent to the SREGA or SREGB MUX'es.

c. CAC0-15. Cache Status Bus. This bus receives the status from the Cache Array Controller PCA. This data goes to SREGA.

d. LUTCIR0-15. Look Up Table. These signals access the microcode address currently being read from the LUT. They go to SREGA.

e. DCU0-15. This is a data bus from the DCU to SREGA.

f. SR0-2. Shift Register. This carries the address of the valid registers in the Stack Registers RA-RH to RREGA and RREGB.

g. WCS0-15. Writable Control Store. This 16-bit bus allows the RALU to see the Microcode, 16 bits at a time, as stored in the CSOR, via SREGB.

3-57. THE CPU REGISTERS. Obviously, there are many registers in the CPU. However, the following are known collectively as The CPU Registers: the Top-of-Stack Registers, Bank Registers, Index Register, Environmental Registers, Status Register, Counter Register, Counter X Register, Timer Clock Register, and the Scratchpad Registers. These registers are distributed across the four RALU PCAs, one nibble (4 bits) for ALUA and one nibble for B per PCA. They are discussed in the following paragraphs.

3-58. TOP-OF-STACK REGISTERS. Within the CPU, up to seven words from the top-of-stack are stored in registers for quick access. The same data is also stored in the Main Memory.

The Top-Of-Stack (TOS) is located in an eight-word register file used circularly. A three-bit register NAMER is used for the address translation from the logical address to the physical address in the file. Another three-bit file, Stack in Register, is used to keep track of how many words of the stack are valid in the registers at any time. Second copies of both of these registers exist for the software instruction, called ENAMER and ESR. This allows SR and NAMER to remain unchanged until the successful completion of the instruction. Updating occurs with the execution of a NEXT microinstruction.

In any one cycle, up to four TOS registers can be read into SREGA, RREGA, SREGB, RREGB; and any one TOS register can be written from UBUSA or UBUSB. If both are trying to write to a TOS register, UBUSB has priority.

SR ranges from zero to seven, so at most seven words in the stack can be stored in the registers. But all eight are addressable in firmware. This leaves one that can be used as scratchpad. Also, since the file is addressed circularly, items are pushed onto the stack by writing to the eighth word in the file, then incrementing SR and NAMER so it becomes the first word in the TOS.

Each ALU has one SREG and one RREG to hold operands for the ALU for Rank 2 operation. Each one (SREGA, RREGA, SREGB, RREGB) has a unique set of sources. In addition, RREG and SREG are used to hold the operands and results of multiplication and division in the MPAD and DVSB micro-instructions.

3-59. BANK REGISTERS. There are eight bank registers used to specify the most significant 16 bits of a memory address: BNKD, BNKP, BNKS, and BNKX3-7. Which is used depends on the microinstruction used for the memory reference. In hardware, these are stored in a register file. Since BKX5 and BKX7 cannot be used for memory references, they are used as scratchpads, because, in the allocation of codes for the system modules, BKX5 is used for the Cache and BKX7 is used for the Main Memory.

3-60. INDEX REGISTER (X). The Index Register is used for address computation for addressing. It can be accessed either normally or multiplied or divided by two, for byte or double word instructions, using the XC (Index Conditional) option in RREGA.

3-61. ENVIRONMENTAL REGISTERS. The Environmental Registers include:

a. Stack Limit (Z) - contains the highest allowable address for the stack for the executing user's segment.

b. Program Limit (PL) - contains the highest executable address in the user's segment.

c. Data Limit (DL) - physical address of the first data word in the user's stack.

d. Data Base (DB) - physical address of the base of the user's stack.

e. Stack Marker (Q) - address of the stack marker of the currently executing routine.

f. Stack in Memory (SM) - physical address of the last valid word of the stack in memory. SM + SR = S, where S is the logical Top-Of-Stack.

g. Program Base (PB) - lowest physical address in user's program segment. PB is used for bounds checking.

h. Program Counter (P) - address of the next machine instruction to be fetched from memory. Normally this points to the NEXT instruction to be executed except during the first line of the instruction or when halted.

3-62. STATUS (STA) REGISTER. This register contains information about execution status:
bit 0 - Privileged mode;
bit 1 - Interrupt Enable;
bit 2 - User Trap Enable;
bit 3 - Stack Op B Pending;
bit 4 - Overflow;
bit 5 - Carry;
bits 6 and 7 - Condition Code;
bits 8 to 15 - number of the currently operating segment.

3-63. COUNTER (CTR) REGISTER. This is a special purpose 8-bit register counter. It can be incremented, decremented, or loaded by microcode; tested if equal to zero for jumps and skips; used with REGN to access any CPU register, based on the value of CTR; used with REGN for single line microcode loops; or used with T1CB to automatically access the TOS.

3-64. COUNTER X (CTX) REGISTER. This is an eight-bit register used for looping. It can be loaded from UBUSB, decremented or cleared in Special Field B, and tested for jumps if equal to zero.

3-65. TIMER CLOCK (TCLK) REGISTER. This register decrements every cycle, and is used as a timer. It has a range of 2**16-1 cycles or about 4 milliseconds. TCLK can be set to interrupt at underflow.

3-66. SCRATCHPAD REGISTERS. These registers (SP0A, SP1B, SP2B, SP3B, SP4A) are used to hold intermediate values in computation. In addition, SP0A, SP3B, and SP4A are linked to shift with SREG and RREG to hold the multiplicand/product or dividend/quotient in multiplication or division operations.

3-67. CPX1 AND CPX2 REGISTERS. The CPX Registers are used to field interrupts and store system flags. CPX1 stores all signals that can cause an interrupt, dependent on various flags in CPX2.

There are two types of interrupts:

a. Those that occur only during the run mode;

b. Those that interrupt whether the system is running or halted. All interrupts are disabled if the Power Fail Inhibit Flip-Flop (PWRFINHFF) is active. Run mode interrupts also require that the Run Flip-Flop (RUNFF) be active and that the System Halt Flip-Flop (SYSHLTFF) be inactive. Except for OVFLO, all CPX bits are set when the described conditions are valid. They remain set until cleared by microprogram control. CPX2 stores various signals from throughout the computer, and holds flags set by the CPU to communicate with the rest of the computer.

o THE CPX1 BITS

BIT00 CPURST	This bit can only be set using the DCU shift string. Its use is currently undefined.
BIT01 OVFLO	This is the only bit that is not a flip-flop. OVFLO is true if the user trap bit and the overflow bit in the status register are both set. This is a Run mode interrupt.
BIT02 BNDV	This bit is set if there is no carry out of an ALU when there is a bounds test microinstruction, and either it is a universal bounds test (UBNE, UBNG), or the machine is in an unpriviliged mode. This is a Run mode interrupt.
BIT03 WCSPE	WCS parity is computed on VBUS and SKSP. A WCS parity error is defined by even parity across CSOR during normal microinstruction sequencing. Error detection is inhibited while reading or writing WCS, during execution of the overhead line, and after a bounds violation. After a parity error, the incorrect line will be in Rank 1. Bit 03 will be set, and a WCS parity error will cause a SYSSTOP.
BIT04 R/HSW	Run/Halt Switch is set by the Z80 in the DCU to tell the microcode to toggle RUNFF.
BIT05 LUTPE	LUT parity is computed in the CIR. A LUT parity error is defined as even parity across the LUT outputs with LUT parity checking enabled (by means of the LPON store field A option). Bit 05 will be set, and an LUT parity error will cause a SYSSTOP.
BIT06 TCLKINT	This bit is set when there is a carry out of TCLK. TCLK decrements every cycle. This is a Run mode interrupt.
BIT07 CPUTIMER	This bit is set during the cycle after the CPU timer on CTLB overflows. The CPU timer is reset every time a TEST is performed for interrupts, so the timer overflow in 2**16 cycles. In addition to setting BIT07, CPU timer overflow causes a SYSSTOP.
BIT08 DCUINT	This bit is reserved for a future communication interface from the DCU to the firmware.
BIT09 MSGINT	This bit is set by (SAOPM5 OR SWOPM5) which is (Send Address or Send Word). It is set whenever the CBI has information to give the Cache. This is a Run mode interrupt.
BIT10 CBIINT	Common Bus Interface Interrupt is set whenever any CBI recognizes a recoverable error, instead of causing a SYSSTOP. This is a RUN mode interrupt.
BIT11 BKPT	This bit is set by the BPVA to allow interrupts based on accessing a certain memory address. This is a RUN mode interrupt.
BIT12 PFWINT	Power Fail Warning is set by the Z80 in the DCU to signal that power is failing and there is only about five milliseconds before dc power goes away. Used for going down softly.

BIT13 XINT1	Not currently used.
BIT14 XINT2	Not currently used.
BIT15 XINT3	Not currently used.

o THE CPX2 BITS

BIT00 PONINT(L)	Power ON Interrupt is set by the DCU after DC power is valid, and before transferring control to the CPU.
BIT01 XFLAG0	Not currently used.
BIT02 PAUSEFF	This bit is set by the CPU to tell the DCU that the CPU is currently in the pause state.
BIT03 ICSFLAG	This bit is set by firmware to indicate that it is executing from the Interrupt Control Stack. It is set when the CPU is servicing an interrupt.
BIT04 DIAGM	This bit is set when a DIAG microinstruction is performed. It also does a SYSSTOP.
BIT05 XFLAG1	Not currently used.
BIT06 RUNFF	This bit is set by the CPU to enable Run mode interrupts.
BIT07 DSPFLAG	Dispatcher flag is under the control of the microcode. It is normally used to indicate that the operating system is in the dispatcher.
BIT08 VPFF	Virtual Page Fault Flag is set from the BPVA. It is intended to be used with virtual addressing to indicate a cache miss. It can also be tested for skips and jumps in ALUA.
BIT09 CSHERR	Cache error originates on the CAC and indicates a hardware error in the cache. This generates a SYSSTOP.
BIT10 SYSHLTFF	System Halt FF is set by the CPU to disable Run Mode interrupts.
Bit11 FLUSH	FLUSH is set by firmware to tell the CAC to flush (transfer) the contents of Cache Memory to Main Memory.
BIT12 TVE	Tag Verify Error is only used in Cache Memory diagnostics. The diagnostic will expect a certain value for the tag, and transmit that value to Cache Memory. If the tag value is not the same, TVE is set.
BIT13 XFLAG2	Not currently used.
BIT14 PWRFINHFF	Power Fail Inhibit Flip-Flop is set by the CPU and inhibits all interrupts. It is used to power-down the computer.
BIT15 DINTFF	Deferred Interrupt Flip-Flop is set by the firmware to record that it has an interrupt pending that it is not ready to handle.

3-68. MACHINE INSTRUCTION SEQUENCING

3-69. The Machine Instruction Pipeline

Each machine instruction to be executed is pre-fetched into the NIR. The upper 12 bits of NIR address the LUT, which outputs information pertaining to the instruction in NIR. When execution is complete for the instruction in the CIR, a NEXT option in Rank 2 causes a transfer of NIR to CIR, and forces the NEXT-Overhead line onto Rank 1. This initiates a pre-fetch for the next machine ("macro") instruction. If the next instruction has not yet been pre-fetched into NIR when a NEXT option is encountered in Rank 1, the CPU is frozen until the instruction fetch is complete (i.e. NIR is loaded). This ensures that NIR is valid before it is transferred to CIR, and that the LUT data is valid before execution of the Overhead line.

If the instruction in NIR is a paired-stack operation, the Overhead line specifies a pre-fetch of the same instruction, except that the right stackop portion of the opcode is mapped into the left stackop portion.

If either an interrupt is pending or an SR pre-adjust is required, the transfer of NIR to CIR during a NEXT is inhibited. (The transfer is delayed until the corresponding microcode routine is exited with a NEXT, providing no interrupts are pending and no further SR pre-adjust is required).

Due to the nature of the interrupt checking mechanism during the NEXT sequence, a store to the Status Register (STA) is not allowed in the same microinstruction as a NEXT if it might result in clearing or masking off a pending interrupt condition.

3-70. The Look Up Table

When the next instruction is loaded into NIR, it addresses the LUT for information about that instruction. The LUT specifies the following information about the NEXT instruction:

a. The WCS entry point target address;

b. The number of top-of-stack registers which must be valid;

c. Flag 2 initialization;

d. The width of the opcode's displacement field;

e. The base register and the type of indexing to be used for memory addressing;

f. Whether indirect addressing is to be employed.

In addition, each LUT word contains a parity bit which is used to create odd parity across all of the data in the LUT.

3-71. Flag Register Initialization

The following flags are cleared on the clock signal which clocks the NEXT-Overhead line out of Rank 2: F1, F3A, F4B, TFF (Target Fetch Flag), and RFLAG (Register FLAG). Also, CTX and CTR are loaded with zero, ESR is transferred to SR, and ENAMER is transferred to NAMER.

On the same clock, F2 is copied from the NIRSUB bit in the LUT. This can be used in conjunction with the ALU Add/Subtract function, which specifies an add or subtract based on F2, to implement two similar software instructions with the same microcode routine.

F5B is also set on this clock for a Branch on Condition Code instruction in CIR which will not be taken, or cleared otherwise.

3-72. MICRO-INSTRUCTION SEQUENCING

3-73. Control Store Loading and Accessing

Microcode is stored on the two WCS PCAs. Each contains 32 bits of the 64-bit microcode word. At LOAD/START, bootstrap code is loaded into the WCS from the DCU. When executed, this code loads the system microcode into the WCS from either disc or tape. With the system microcode loaded, the WCS contains all of the microcode necessary to implement the HP 3000 Machine Instruction Set.

Several system clock periods are required to fetch and execute the microcode. These clock periods are referred to as Ranks 0-3. Each rank is a time period during which certain functions occur, depending on the instruction. The ranks are overlapped, so while one instruction is being executed in Rank 2 the next consecutive instruction is being executed in Rank 1, etc. Thus each Rank contains different instructions.

3-74. Rank 0 Operations

A microcode instruction is fetched during Rank 0. The WCS is addressed by the Control Store Address Register (CSAR) on the VBUS PCA. Unless an instruction such as JUMP or RETURN is decoded, the CSAR is loaded with the next sequential WCS address (the current WCS address plus one). The microcode is fetched and executed from sequential WCS locations until a microcode option forces an out-of-sequence WCS address.

3-75. Rank 1 Operations

In Rank 1, unconditional Jump to Subroutines (JSBs) and Return from Subroutines (RSBs) are executed. Input registers are read to the R and S Registers (RREGA/B and SREGA/B), where they become operands for the ALUs. The output of the Control Store Output Register (CSOR) is multiplexed with a hardwired Overhead line. The multiplexer's output is decoded and executed. Any unconditional JSB in Rank 1 multiplexes the jump target onto the VBUS. The VBUS carries the WCS address of the current instruction. A check is made for an RSB and if a return address is found it is placed on the VBUS.

With the microinstruction in Rank 1, the registers needed for ALU operations are read and the data placed on the appropriate buses to be loaded into RREGA, SREGA, RREGB, and SREGB. Literals are decoded and placed on the appropriate buses; the ALU function codes are pre-decoded and the ALU and shifter controls are determined; many of the special operations are pre-decoded; and non-data dependent skip conditions are checked.

3-76. Rank 2 Operations

In Rank 2 the ALU functions are performed (i.e., the actual "computing" is done), using the RREG and SREG contents. The results are placed on TBUSA and TBUSB. The data on the TBUSes is transmitted to the shift logic to produce UBA and UBB. The results are stored according to the instructions in the store fields. In addition, "medium" speed jumps are executed from Rank 2, the special field options are executed, and data-dependent skip conditions are checked.

3-77. Rank 3 Operations

Data-dependent jumps are operated on in Rank 3, and written to the LUT and WCS.

3-78. Micro-Sequencing Jumps

The most common option to cause an alternate address to be placed on the VBUS is the JSB. JSB results in the transfer of microcode execution to the subroutine target address. When a JSB is executed, the target address from either Rank 1, 2, or 3 is placed on the VBUS, depending on the speed of the JSB (fast, medium, or slow, respectively), and the address of the following line of microcode is placed at the top of the Return Address Register (RAR).

Data-dependent jumps (such as those based on the results of the ALU operations) are executed from Rank 3. Non-data-dependent jumps (such as those based on flags), and unconditional jumps preceded by data-dependent skips, are executed from Rank 2. Unconditional jumps are executed from Rank 1. Jumps executing from Rank 3 override those from Rank 2 or 1. Jumps from Rank 2 override those from Rank 1. Jumps from any Rank in ALUB override those from the same Rank in ALUA. A Rank 1 jump is executed with no lost microinstruction cycles; a Rank 2 jump causes one clock cycle of No Operation (NOP); and a Rank 3 jump causes two cycles of NOP. NOP inhibits the execution of a store function when the data from that operation is not required. It occurs during the execution of microinstructions entering the pipeline after the jump, but prior to the execution of the jump.

3-79. Skips

In the event the skip condition indicated in the skip field for an ALU is true, NOP2 is set for that ALU when the following microinstruction enters Rank 2, and the store, special, jump, and skip operations specified in that microinstruction for that ALU are inhibited (i.e., skipped). All the skip conditions except those involving the SR Register are tested in Rank 2. Those involving the SR Register are tested in Rank 1, using the current value of the SR.

3-80. Returns

When an RSB skip field option is executed, the VBUS is driven by the RAR. This, like an unconditional JSB, can occur when the RSB is in either Rank 1 or 2. Jumps from all ranks override a RSB from Rank 1, while only jumps from Ranks 2 and 3 override a RSB from Rank 2.

The RAR is not automatically incremented or decremented for JSBs and RSBs. Incrementing is done in microcode, using Special Field options Push Register (PSHR) and Pop Register (POPR).

3-81. Repeats

There are two ALU Function Field options that cause the next sequential microinstruction to repeat:

a. REPC, in which the instruction repeats until a condition is met,

b. REPN, used to repeat an instruction N times.

REPC causes ALUB to perform an add, and if skip conditions specified in ALUA and ALUB are both false, the following instruction is repeated until the condition in ALUA or ALUB is true. The repetition of the instruction is accomplished by holding it in the CSOR until one of the skip conditions becomes true.

REPN is identical to REPC except that:

a. A counter is loaded with a constant (N) specified in the skip field of the instruction containing the REPN.

b. Rank 2 operations are NOT inhibited at the end of the repeat sequence.

3-82. NEXTs and Bounds Violations

A NEXT Skip Field Option in Rank 2 forces a NEXT-Overhead line onto Rank 1. The detection of a bounds violation causes a pseudo-NEXT, forcing the Overhead line onto Rank 1. While the Overhead line is being forced onto Rank 1, the next machine instruction entry point target (from the LUT) is forced onto the VBUS, thus transferring microinstruction execution to the next WCS entry point. However, this "pseudo-jump" can be overridden by either a fast jump to the Stack Register (SR) pre-adjust routine, or a slow jump to the interrupt handler from the Overhead line.

Three NEXT-Overhead lines are built into the hardware. One of them is mapped into Rank 1, depending on whether an SR preadjust is required before the execution of the next macro-instruction, and whether NIR contains a paired stack operation.

Usually, when an SR pre-adjust is not required and NIR does not contain a paired stackop, the "normal Overhead line" is executed. ALUA adds the Index Conditional register and the base register specified by the LUT. It also executes a jump to the interrupt handler if any interrupts are pending. ALUB increments The Program Counter (P) Register and executes a RONP special option, which presents the resultant address to the Cache Memory and pre-fetches the next macro-instruction into NIR.

If NIR contains a paired stackop when NEXT is executed, the Overhead line is altered slightly. P is not incremented, and an RNSP special option is executed. This causes the macro-instruction pre-fetch to be repeated, except that the right stackop portion of the opcode is mapped into the left stackop portion before the instruction is loaded into NIR. The right stackop is then executed as if it was a single stackop.

The Overhead line is also altered if an SR pre-adjust is required. This occurs when the number of valid top-of-stack registers in the CPU is less than the number specified for the next instruction in the SRP field of the LUT. In that case, ALUA reads the top-of-stack and executes a JSB to the SR pre-adjust routine. If an SR pre-adjust is required when NIR contains a paired stackop, the ALUA SR preadjust target address is decremented by two. Otherwise, the SR Overhead line is unaltered.

3-83. CPU INTERFACE WITH CACHE MEMORY

Cache Memory is an 8-kbyte, high-speed, buffer memory. Its purpose is to anticipate what data the CPU will request next, get that data from Main Memory, and have it ready when the CPU requests it. When the CPU requests its first data word, Cache Memory pulls the entire block (eight 16-bit words) containing that word from Main Memory. This is done on the strong probability that the CPU's next request will be for the next word in that same block.) Cache Memory is successful in having the word the CPU wants available when requested approximately 95 percent of the time. This is referred to as a "95 percent hit ratio."

3-84. Memory Reads

A read is initiated when a Read Request is output from the CPU. The data returned is received in the OPA or OPB Registers in the RALU. A Read Request will specify to the Cache Memory which Bank Register to use for the upper 16-bit address, and the lower 16-bit address from the UBUS, and the operand registers to receive the data. The Read Request goes to the Cache Memory during a clock cycle when the Read operation is in Rank 3. The data is stored in the Cache Memory two clock cycles later, assuming a "hit". If the data is not in the Cache at the time of the request, the CPU will freeze when the instruction calling for the operand register is in Rank 2. The freeze will hold until the Cache can transmit the data to the Rank, which will put it directly into the SREG and the operand register.

3-85. Memory Writes

A memory write starts with loading the 32-bit address into a Bank Register and a Y Register, either from a previous memory access or by using a Write Register option. The CPU then puts the data to be written onto the UBUS and specifies the data option. The address in the Y Register is incremented to point at the next address.

3-86. CACHE MEMORY OVERVIEW

To accomplish its job as described in Paragraph 3-83, the Cache Memory is located between the CBI PCA and the CPU (see Figure 3-1). CPU requests are sent to Cache Memory in the form of addresses across the Cache Address Bus (CAB). For CPU read operations, the address is compared to other addresses stored in Cache Memory, and depending on the success or failure of the comparison, data is supplied to the CPU immediately or after a memory cycle. For CPU write operations, data and address are sent to the Cache and the data is stored by address.

Cache Memory is also responsible for its contents with regard to the rest of the computer. For example, if the Cache Memory contains an updated copy of a block of data that the I/O System Module requests from Main Memory, the Cache Memory aborts Main Memory's response to the I/O System, and supplies its updated data to the I/O System.

3-87. Cache Buses

Cache Memory interfaces to the CPU via three buses and six control signals. The buses are: the Cache Address Bus (CAB - 32 bits) and the Processor Data Bus (PDB - 16 bits) for CPU writes, and the Cache Data Bus (CDB - 16 bits) for CPU reads. The control signals are: START, WRITE, BUSC(L), LOCK, FAIL, and FLUSH.

The CAB supplies addresses to Cache Memory from the CPU; it also carries CPU messages to the CSB. The CAB originates on the CPU RALU PCAs. CAB bit assignment is as follows:

BITS	FUNCTION
00-04	(Not Used)
05-20	16-bit data block tag; for a given data block, these bits are stored in the Cache Tag RAMs and are compared with their corresponding bits in a CPU- or CSB-provided address.
21-28	8-bit data block address used to address the Tag RAMs for block status and tag storage. These bits are also the high order address bits of the data RAMs in the Cache Memory Array.
29-30	Point to the specific double-word (32 bits) addressed in a data block.
31	LSB that points to the specific word (16 bits) in the currently addressed data block. This bit is used to multiplex the word out of Cache Memory Data Sets (the CDB is 16-bits wide). 0 = bits 0-15; 1 = bits 16-31.

The 16-bit CDB originates on the Cache Memory Array (CMA) PCA and is routed the RALU and Current Instruction Register (CIR) PCAs.

The 16-bit PDB carries data to the CMA PCA and bus commands to the Cache Array Controller (CAC) PCA.

3-88. CPU Control Signals

The CPU asserts six control signals to Cache Memory: START, WRITE, BUSC(L), LOCK, FAIL, and FLUSH. START indicates the beginning of a CPU data read or write operation. WRITE indicates a data write (WRITE high) or read (WRITE low) operation and occurs at the same time as START. LOCK indicates the CPU is going to modify data in Cache Memory and is used to deny Cache access to any other CSB module. The CPU asserts BUSC(L) during messages and diagnostics; it indicates that the information on the PDB is a command rather than data. The CPU asserts FLUSH during a power failure operation; FLUSH causes FAIL to occur only on "dirty" faults, causing the Cache to write only the "dirty" (modified) blocks to Main Memory.

If Cache Memory does not contain the data requested by the CPU, it sends FAIL to the CPU. FAIL will occur if:

a. the CPU asserts START and Cache Memory does not have a valid copy of a block containing the requested data;

b. the CPU asserts BUSC(L) to transmit a message to the CSB;

c. the CPU asserts START or BUSC(L) when the Cache is handling a CSB request.

3-89. Cache Interface with CBI PCA

Cache Memory interfaces with its CBI PCA via two buses and several control and status signals. The buses are: ADATA, which carries CSB addresses, data, and messages to the Cache Memory; and DATA, via which the Cache Memory sends addresses, data, and messages to the CSB. Each bus is 32 bits wide.

The control and status signals include:

ABORTIN	Indicates to Cache Memory that a CSB module has a valid copy of a data block and is supplying it to another module.
DATAOP	Indicates to Cache Memory that a CSB data transaction is in progress.
DSLECT	Indicates to Cache Memory that its request for the CSB is acknowledged and that it will be provided the bus on the next clock.
DCB	Diagnostic Check Block indicates to Cache Memory that the current CSB transaction involves a diagnostic check block.
RBP	Read Block Private indicates to Cache Memory that a CSB module is requesting to read a private block in memory.
SAOP	Send Address Operand indicates to Cache Memory that the CBI has received the address of an incoming message to the CPU.
SWOP	Send Word Operand indicates to Cache Memory that the CBI has received the information of an incoming message to the CPU.
FRMIN	From In is a three-bit code to the CPU indicating the source of a message.
WNB	Write New Block indicates to Cache Memory that a CSB module is writing a data block to the Main Memory. Cache invalidates a matching block if contained in the Data Set RAMs.
ADATAP	ADATA Parity (2-bits), part of the ADATA bus, to the Cache Memory for each double-word sent to Cache Memory.
ABORTTB	Indicates to the CSB that Cache Memory is aborting the Main Memory read to supply the valid data block from its own data sets.
AWAKE(L)	Tells the CBI that a Cache Memory request exists.
BUSY	Indicates to the CBI that Cache Memory is currently involved in a message transaction.

CLKFRMA(L) CLKREGA(L) CLKREGB(L)	Control signals from Cache that latch incoming CSB information into CBI registers.
ENBREGA(L)	Enables Register A on the CBI to send message information to the CMA.
ENBREGB(L)	Enables register B on the CBI to send data information to the CMA.
IGNBUSY(L)	Forces an outgoing message onto the CSB.
TOENB(L)	To Enable 3-bit code from Cache Memory to the CSB that enables the target module for a CPU message or data.
GENPAR	Generate Parity control signal from Cache Memory that causes the CBI to generate parity on an outgoing message or address.
CHKPAR(L)	Check Parity control signal from Cache Memory that causes the CBI to check the parity bits on data from Cache Memory.
MUXSTAT	Multiplex Status control signal from Cache Memory that causes the CBI to gate its status onto the CSB with Cache/CPU TO and FRM codes (5). During the next transaction, Cache reads the status like a normal message.
CLRSTAT	Control signal from Cache Memory that clears CBI status.
CONT	Continue Transfer status signal from Cache Memory that indicates Cache Memory has a data block to transfer.
EXTO (0:2)	A module number sent to the CSB that is simply buffered by the Cache (for messages only).
OP (0:4)	The bus operand codes for an outgoing message to the CBI.

3-90. Cache State Machine

All "on-board" Cache Memory operations are controlled by hardware. The hardware circuit responsible for Cache control is called the Cache State Machine. Cache Memory functions (reads, writes, and commands) are controlled during six states:

Cache Free (CSHFRE).

The default and idle condition of the Cache Memory. Cache Memory executes all CPU read hit, write hit, and bus command transactions (except Send Message) while in the CSHFRE state.

Wait For Bus (WFB).

A wait state while Cache Memory has CSB request (AWAKE(L)) asserted to the CBI. Cache Memory enters the WFB State for clean and dirty faults and CPU send message operations.

Send Data (SD).

The data write-to-CSB (Main Memory) state; the Cache enters the SD State for dirty faults only.

Receive Data (RCVD).

The read-from-CSB (usually Main Memory) state; Cache Memory enters the RCVD State for clean faults only.

Check (CHCK).

The CSB compare state. Cache Memory compares a current address on the CSB with a tag in the Cache Tag RAMs.

Be Memory (BEMEM).

The condition in which Cache Memory aborts Main Memory to supply valid data to another CSB module.

The five active states (WFB, RCVD, SD, CHCK, and BEMEM) vary in the time they are active. WFB is valid for a minimum of two clocks before Cache Memory moves to another state (depending on the current status of the CSB). SD is valid for four successive clocks. RCVD is valid for a minimum of 13 clocks - the time required for a memory read; RCVD may take less time in the event of an I/O abort. CHCK is valid for one clock - the time required to compare a CSB address with a Cache Memory Tag. BEMEM is valid for six clocks minimum.

3-91. Handling CPU Starts

START and BUSC(L) are mutually-exclusive CPU command signals. STARTs (CPU read or write transactions) may start sequentially, but will not overlap. The CPU can also assert START while Cache Memory is in the CHCK or BEMEM states, in which case Cache will assert Fail to the CPU until it has completed its check operation or BEMEM transaction and returned to the CSHFRE state.

Every clock cycle (except while Fail is asserted) Cache Memory will clock addresses on the CAB into the Cache Address Register (CAR). Cache Memory does not latch START, WRITE, LOCK, or FLUSH from the CPU, therefore, the CPU must hold these signals asserted for the entire time Cache Memory has Fail asserted.

When Cache Memory is in the CSHFRE state, it multiplexes the CAR Tag bits into the Compare gates. It also sends the block bits (multiplexed to the RAM address inputs with the word bits) to the CMA where they address the Data Set RAMs. THe low order bit selects the upper or lower word at the output of the CMA multiplexers. By this process, read and write hits will be fully executed without Cache Memory asserting Fail in the CSHFRE state. Only a Miss will cause Cache Memory to assert Fail and enter through the WFB state to the RCVD or SD states.

For a clean fault, Cache Memory reads the data block from Main Memory while in the RCVD state. Cache Memory then goes back to the CSHFRE state to complete the CPU transaction as a clean hit. For a dirty fault, the Cache Memory first writes the dirty block to the Main Memory, then proceeds with a clean fault transaction, as above.

3-92. Handling CSB Checks

Cache Memory responds to five opcodes from the CBI. The CBI generates these opcodes by decoding CSB busops. The opcodes are:

a. Diagnostic Check Block (DCB)
b. Write New Block (WNB)
c. Read Block Private (RPB)
d. Send Word Operand (SWOP)
e. Send Address Operand (SAOP) (used only for diagnostics)

The CBI contains two sets of latches dedicated to Cache Memory for messages and data operations. Latch Set A holds incoming messages; Latch Set B holds incoming data and addresses. The CBI also contains FROM latches which are used similarly.

3-93. Message Handling in the Cache

Message communication between the CPU and other major modules takes place over the Central System Bus (CSB). Cache Memory merely passes messages to and from the CPU without modifying or decoding them.

The CPU outputs messages on the CAB to the Cache Address Register on the CAC PCA. From there, the Cache switches them onto the DATA Bus to the CBI. At the same time the CPU puts the message on the CAB, it puts a TO Code on the PDB, and asserts BUSC to Cache Memory.

During a message transaction, Cache Memory asserts MESS which indicates the send status of the message to the CPU. Cache sets MESS one clock after it asserts Fail, and resets it one clock after it has been granted the CSB.

The CBI detects incoming messages to the CPU three quarters of the way through the transaction cycle. After detecting the message, the CBI decodes the busop (SAOP or SWOP). The CBI sends the decoded busop and asserts SAOP or SWOP to the Cache, causing the CAC to generate CLKREGA(L) and CLKFRMA(L). CLKFRMA(L) causes the CBI to latch the message and the From code. Also, the CAC asserts MSGINT at phase 0 of the cycle and holds it one clock period. The CAC sends MSGINT to the CPU to alert it to the incoming message. The CPU then asserts BUSC and puts the required busop code on the PDB to read the message.

A message includes three parts: a FROM code, a Send Word/Send Address bit, and the message itself. The upper and lower message words must be read separately by the CPU since it is a 16-bit machine. When MSGINT is asserted, Cache Memory latches the message into the CBI registers where it is held until another message comes in. Cache Memory decodes the PDB bus command for the part of the message requested by the CPU. It gates the requested part from the ADATA Bus through the CMA multiplexers to the CDB.

3-94. Handling CPU Bus Commands

In addition to opcodes for handling messages, bus commands from the CPU to cache include opcodes to read CBI status, Tag RAMs, and Cache Memory status. These opcodes are sent to the cache over the PDB. The busop codes and their bit assignments are:

PDB BITS 3	4	5	13	14	BUSOP
0	0	0	W	Y	Send message word (SNDWRD)
0	0	1	0	Y	Read upper 16 bits of incoming message (MSGUP)
0	0	1	1	Y	No operation
0	1	0	0	Y	Read lower 16 bits of incoming message (MSGLO)
0	1	0	1	Y	Clear CBI status (CLRSTAT).
0	1	1	0	Y	Read FROM Code of incoming message (ENFRM)
1	0	0	V	Z	(Not Used)
1	0	1	V	Z	(Not Used)
1	1	0	X	X	Write the Status RAMs; Keep old parity (RSTATW)
1	1	1	X	X	(Not Used)

Key:
V = 0 (real), 1 (virtual)
W = 1 (ignore busy)
X = don't care
Y = 0 (reset Busy F/F), 1 (set Busy F/F)
Z = 0 (A Tag Set), 1 (B Tag Set)

3-95. CACHE MEMORY OPERATION

The following paragraphs describe the functions of the logic on the two Cache PCAs: the Cache Array Controller (CAC) and the Cache Memory Array (CMA). See the system block diagram for the circuitry.

3-96. Interaction with the CPU

As previously described, Cache Memory is a data buffer between the CPU and Main Memory. It provides temporary storage for data being accessed by the CPU, thus reducing CPU use of the Central System Bus (CSB). When the CPU requests its first data word, Cache Memory reads the entire block that includes that word from Main Memory. (A block contains eight 16-bit words.) If the next word requested by the CPU is the next one in sequential order, the Cache Memory then has that word immediately ready. When the CPU requests a word the Cache Memory doesn't have, and the block that is in Cache Memory has been modified (is "dirty") Cache Memory writes that dirty block to Main Memory, then reads the block containing the requested word. Cache Memory has the word the CPU wants available when requested approximately 95 percent of the time (i.e., it has a "hit ratio" of approximately 95 percent).

3-97. Cache States for CPU Transactions

As previously described, the CPU interacts through Cache Memory to read or write data to Main Memory. Data read and write functions, however, depend upon whether the required data is currently in the Cache Memory. If the data is not available in Cache Memory, it must read the block in Main Memory containing the requested data. This data transfer becomes complex if the data currently in Cache Memory was modified in previous CPU transactions. In this case, the resident data block must first be written to Main Memory before the replacement block containing the requested data can be read.

To handle the conditions affecting CPU read/write transactions, the Cache Memory contains state control logic. This logic sets the Cache to one of four states that involve the CPU:

a. Cache Free (CSHFRE)
b. Wait For Bus (WFB)
c. Send Data (SD) to Main Memory
e. Receive Data (RCVD) from Main Memory

The CSHFRE state is the default state for all Cache Memory operations. When the CPU reads or writes data currently in Cache Memory, the Cache operates in this state. It also operates in the CSHFRE state for orders or commands associated with Bus Command (BUSC(L)). (BUSC(L), a low level signal, indicates to Cache Memory that the information on the processor data bus [PDB] is a diagnostic instruction to Cache Memory.

The WFB state sets Cache Memory into a standby mode while it is queued for CSB access for a Main Memory read or write operation. It enters the WFB state whenever data needed by the CPU for a read or write is not contained in the Cache.

When the Cache acquires the CSB to write data to Main Memory, it enters the SD state. At the conclusion of the write, the Cache returns to CSHFRE.

When the Cache acquires the CSB to read data from Main Memory, it enters the RCVD state. At the conclusion of the read, it returns to CSHFRE.

3-98. Cache Transaction Conditions

The condition of the data block currently in Cache Memory depends on prior and pending CPU transactions. The data block can be in one of four conditions:

a. Data requested by the CPU is in Cache Memory. This is called a Hit.

b. Data requested by the CPU is in Cache but is Invalid because an updated version of the same data exists in Main Memory.

c. Data requested by the CPU is not in Cache Memory and the data that is in Cache has not been modified. This is called a Clean Miss (or a Clean Fault).

d. Data requested by the CPU is not in Cache Memory and the data that is in Cache has been modified. This is called a Dirty Miss (or a Dirty Fault).

If the data addressed by the CPU is a Hit in Cache Memory, it simply reads it from or writes it to the Cache.

If the Cache Memory data is Invalid, Cache generates Miss and Fail signals. Cache Memory then goes to the RCVD state to read the data from Main Memory, and overlays the current contents of Cache Memory. Cache Memory then returns to the CSHFRE state where it removes the Fail signal and proceeds with the CPU transaction.

If the data required by the CPU results in a Clean Miss, Cache Memory goes through the same process as for an Invalid data condition.

In the case of a Dirty Miss, Cache Memory goes through the WFB state to write its current contents to Main Memory. When Cache Memory returns to the CSHFRE state, it executes the same functions as a clean miss.

3-99. CPU/Cache Communication

The CPU sends the addresses of needed data to Cache Memory over the CAB. The CAB also carries the message portion of Bus commands from the CPU.

The CPU sends write data to Cache Memory over the PDB. The PDB goes to the CMA and CAC PCAs.

Cache Memory sends CPU read data to the CPU over the Cache Data Bus (CDB). The CDB also carries messages from other CSB modules to the CPU.

Five signals from the CPU control Cache Memory operations:

a. START
b. WRITE
c. BUSC(L)
d. LOCK
e. FLUSH

The CPU sends START when is is addressing needed data. WRITE indicates the CPU will write data to Cache Memory. WRITE(L) indicates the CPU will read data from Cache Memory. BUSC(L) indicates the CPU either is sending a message to another CSB module, or sending diagnostic orders/instructions. The CPU issues a LOCK command to Cache Memory during a read operation to prevent external access to Cache during the read. The CPU issues FLUSH during a loss of primary power to prevent the loss of Cache data. During a FLUSH, the CPU forces the Cache Memory to write its dirty blocks to Main Memory. (Main memory is protected by battery backup; cache memory is not.)

3-100. Cache Interaction with the CBI and CSB

Cache Memory communicates with other CSB modules via its CBI PCA.
Such communications include:

a. Data block reads to access data required by the CPU.
b. Data block writes of dirty data to Main Memory to release Cache Memory space for data reads.
c. Data block writes to another module to supply data more current than that in Main Memory.

3-101. Cache States for CSB Transactions

Cache Memory's state machine steps the Cache through the WFB, SD, and RCVD states to handle data block reads and writes that are concerned with the CPU.

When Cache Memory is in the CSHFRE or WFB state, it constantly monitors CSB activity. When a transaction occurs (DCB or RPB), a WASANY status signal is set. If Cache Memory is in either the CSHFRE or WFB state, WASANY causes it to go to the CHCK state to compare the address on the CSB with the addresses on the data in Cache Memory.

For example, if an I/O module is reading data from Main Memory that also exists dirty in Cache Memory, Cache sends ABORT to Main Memory. It then goes into the Be Memory (BEMEM) state and seizes the bus. Cache Memory marks its block Invalid, then sends the data requested by the CSB module to that module. When the Cache Memory concludes the data transfer, it returns to the state it was in before the CSB transaction.

3-102. CBI/Cache Communication

Cache Memory communicates with its CBI PCA via two buses:

a. The ADATA bus, which carries addresses, data, and parity from the CBI to the Cache;
b. The DATA bus, which carries addresses, data, and parity from Cache to the CBI.

Cache Memory and the CBI also exchange control and status signals to cause data and message exchanges to take place. Included are five CSB operation codes. These codes, which cause Cache Memory either to enter the CHCK state or send an interrupt to the CPU, are:

a. Diagnostic Check Block (DCB) which indicates that the incoming block of data is part of a diagnostic test initiated by the CPU.

b. Write New Block (WNB) which indicates an external module is writing new data to Main Memory. This code generates a WASANY signal, causing Cache Memory to go to the CHCK state. If the address of the data being written matches that in Cache Memory, the Cache block becomes invalid. This opcode is not implemented on the Series 64 (32460B).

c. Read Block Private (RBP) which indicates another CSB module is attempting to read data from Main Memory. This code generates a WASANY signal, causing Cache Memory to go to a check state. If the address of the data being read matches that in Cache Memory, and the block is dirty, then ABORTTB is sent to the Main Memory and Cache Memory shifts to the BEMEM state.

d. Send Word Operand (SWOP) which indicates a message is coming from another CSB module to the CPU. This code causes Cache Memory to generate a Message Interrupt (MSGINT) signal to the CPU. It is decoded on the CBI from the Send Word opcode.

e. Send Address Operand (SAOP) which is the same as SWOP except that it is decoded on the CBI from the Send Address opcode.

3-103. DIVISION OF FUNCTIONS BETWEEN CACHE PCAS

As previously described, Cache Memory consists of two PCAs: the CMA and the CAC. The CMA contains memory, data bus multiplexers, parity generators, and some control logic. The CAC contains most of the Cache control logic, including the Cache state generator.

3-104. CMA Functions

The CMA primarily contains memory. The memory is divided into two data sets (A and B), each containing 256 blocks. A block consists of eight 16-bit words, giving Cache a capacity of 2K words per set, for a total of 8 kbytes (a byte is half a word).

3-105. CMA Data Organization

Data within Main Memory is divided into blocks. If the CPU requests the data word at address n and the Cache Memory does not contain it, the Cache Memory reads the entire Main Memory data block to which the requested word belongs. Main Memory data is transferred in double words (two 16-bit words) over the 32-bit CSB.

3-106. CMA Data and Control Paths

CMA PCA inputs and outputs consist of four data buses, control signals from the CPU and the CAC, status signals from the CAC, and signal lines from the CBI. These buses and signals include:

a. Processor Data Bus (PDB)
b. Cache Data Bus (CDB)
c. Cache output data bus (DATA)
d. CBI input data bus (ADATA)
e. Intra-Cache Bus (ICB)
f. WRITE signal line (from the CPU)
g. CSHFREE state status line
h. Control signals AUPWE(L), BUPWE(L), ALOWE(L), and BLOWE(L) from the CAC for selecting Set A or Set B data
i. CDB output selection control
j. CBI message FROM code

Data on the PDB from the CPU and the ADATA bus from the CBI are input through a two-to-one mux to the Set A and B RAMs, then to another two-to-one mux. WRITE from the CPU and CSHFREE from the CAC are ANDed to select the data output from the CPU. The Intra-Cache Bus (ICB) from the CAC PCA forwards the address from either the ADATA bus or the CAR to the address inputs of the Data Set RAMs. If the CPU is writing to Cache Memory, CMA Write Enable logic on the CAC determines whether the CPU is sending the upper or the lower part of a double-word. This address decoding logic applies the required enable signal to the data set RAMs for storing the CPU words. These enable signals are:

a. A data set upper word enable (AUPWE(L))
b. A data set lower word enable (ALOWE(L))
c. B data set upper word enable (BUPWE(L))
d. B data set lower word enable (BLOWE(L)).

The outputs of the data sets are applied to the seven-to-one Data Set Mux where, for a CPU read, the CDB output selection control signals select Data Set A upper or lower words, Data Set B upper or lower words, messages, or CBI message FROM code. The selection controls pick the output of the multiplexer as shown in Table 3-1.

Table 3-1. Data Set Mux Output Selection

OUTPUT TO CDB	A	B	C
A Upper Word	low	low	low
A Lower Word	ICB10	low	low
B Upper Word	low	CMAMISSB(L) high	low
B Lower Word	ICB10	CMAMISSB(L) high	low
MSG Upper	low	low	MSGUP
MSG Lower	MSGLO	low	MSGLO
Message FROM Code	ENBFRM	ENBFRM	ENBFRM

The Mux control truth table is as follows:

C	B	A	Mux output
0	0	0	A Bank Upper
0	0	1	A Bank Lower
0	1	0	B Bank Lower
0	1	1	B Bank Upper
1	0	0	Message Upper
1	0	1	Message Lower
1	1	0	Don't Care
1	1	1	From Code

The outputs of the Data Set Mux also are applied to the CMA parity generators. These generators output parity bits (high) on even parity. These parity bits are input to Cache Memory error logic on the CAC.

The outputs of the data sets are also applied to the DATA multiplexer which selects the output of data set A or data set B to send to the CBI on the DATA bus. The CMA least recently used (CMALRB) status signal from the CAC selects the output of the DATA multiplexer: low = Set B, high = Set A.

The CBI inputs the FROM code to the data set mux where the ENBFRM code from the CAC selects it for output to the CPU. This code is read by the CPU during read message transactions.

3-107. CAC Functions

As Cache Memory is a memory buffer for the CPU, its primary activities are CPU read and write operations. Cache Memory operates in the CSHFREE state up to 95 percent of the time. It is in the CSHFREE state that the CPU reads data from and writes data to Cache Memory. All other Cache activities in the CSHFREE state, except message transactions, are directed toward supplying the CPU with the requested data. Cache Memory operations thus can be described as follows:

a. Read Hit, where data for a CPU read transaction is currently in Cache Memory.

b. Write Hit, where data for a CPU Write transaction is currently in Cache Memory.

c. Clean Fault, where data needed by the CPU for a Read or a Write transaction is not in Cache Memory and the data currently in the Cache Memory has not been modified (is "clean"). In this

case, Cache Memory goes through the RCVD state to read the data block in Main Memory containing the data.

d. Dirty Fault, where data needed by the CPU for a Read or a Write transaction is not in Cache Memory, and the data that is in Cache has been modified (is "dirty"). In this case, Cache Memory goes through the SD state to write the dirty block to Main Memory and the RCVD state to read in the data block containing the required data.

e. Send Message, where the CPU is sending a message instruction to a module on the CSB.

f. Receive Message, where a module on the CSB is sending a message to the CPU.

g. CSB Check, where Cache Memory monitors CSB activity for Main Memory transactions involving data blocks currently in Cache.

h. Cache Write to the CSB, where Cache Memory detects a CSB module attempting to read a data block from Main Memory that is currently in Cache, and is valid and dirty.

The CAC performs virtually all Cache Memory control functions for these operations, i.e., data and address steering, CPU activity and data status, error sensing, parity, and state control. These functions are all under hardware control and consist of the following circuits:

a. State logic
b. Tag compare
c. Error
d. Fail
e. BUSC(L) decoding
f. BUSOP code
g. CPU interrupt
h. Status

The CAC's Tag compare logic compares the Main Memory address of the data requested by the CPU with the addresses of the data blocks contained in Cache Memory.

When a data block is first written into Cache Memory, its 16-bit tag (bits 5-20) is stored in a location addressed by bits 21-28. Each of these locations corresponds to one block stored in a data set. Since a particular tag corresponds to 256 blocks, the Tag RAMs may contain the same tag in several locations.

When a data block is addressed by the CPU or an I/O module, the CAC applies address bits 21-28 to the Tag RAMs and bits 5-20 to Address Comparators. The Tag RAMs each output a Tag (bits 5-20) that corresponds to the block addressed (bits 21-28). These Tag outputs are applied to the Address Comparators. If a compare results from one of the tags matching the tag block address, the CMA Write Enable logic enables the Data Set containing the addressed block for a Read or a Write operation. Address bit 31, input to the CMA Write Enable logic, selects the upper or lower word in the data set. This Tag comparison process occurs for each Cache Memory data operation.

3-108. Read Hit Operation

As previously described, a Read Hit condition exists when data addressed by the CPU exists in Cache Memory. For a Read Hit, assume the following conditions in the Cache Memory:

a. CSHFREE output from the State logic;
b. Fail signal out of the Fail Logic is low;
c. WRITE low and START high from the CPU are low;
d. BUSC(L)(L) from the CPU is high;
e. No current CSB activity resulting in BUSOP or other codes being input from the CBI.

When the CPU addresses data for a Read, it simultaneously sets START high and Write low. The data address from the CAB is input through the CAR to the CAR/ADATA Mux. Since CHCK and BEMEM are low at this time (Cache Memory is in the CSHFREE state), the CAR is output on the Address (ADR) bus to the A and B Tag RAMs address inputs, and on the Intra Cache Bus (ICB) to the A and B Data Set RAMs on the CMA PCA.

The Tag bits in the address are compared against tags stored in the Tag RAMs. Assume a match (Hit) occurs in the A Tag RAM which results in Miss(L) and Miss A(L) signals going high (indicating a Hit) output from the A Tag Address Comparator. Miss A is applied to the Cache Error Logic and, since it is low, CACERR out is low. If Miss A and Miss B were low at the same time (indicating a Hit in both Tag RAMs), a Cache Error status signal would be sent to the CPU. Miss(L) is applied to the Dirty Hit logic (part of the input logic to the State Machine), to the CSB (Bus Operation (Busop) logic, and to the Fail logic.

Since MISSA is high to the Dirty Hit logic, HITDTY(L) is high, and ABORT(L) (input to the State Machine) is high. Miss(L) is also high to the CSB Busop logic. Because no busop is in process, the logic outputs no Opcode. Miss(L) is input as a high signal to the Fail logic resulting in a low Fail signal out, i.e., no Fail condition.

MISSA is input low to the CMA Write Enable logic. This logic also has Address Bit 31 (ADR31 - assume it to be high) MISSB - low, Least Recently Used (LRU - don't care), Receive Data (RCVD - low; also WRITE is low) input to it. Because WRITE is high, the A Upper and Lower Word Enables (AUPWE and ALOWE) are high. When high, these signals disable a Write to the Data Set RAMs (setting the Data Sets to the Read default condition).

The WRITE signal from the CPU (low because the CPU is reading data) and the CSHFREE signal (high) from the State logic are applied to the ADATA/PDB Mux on the CMA. Given the condition of these signals in this discussion, the ADATA bus is selected as the mux output. However, this mux selection is incidental, since Cache Memory is in a Read Hit operation.

Data Sets A and B are addressed at the same time by the CPU, via the Intra-Cache Bus (ICB). The Data Sets are addressed with Address bits 21 through 30. These bits address double-words in the RAMs. Since a Read Hit on Data Set A exists in this discussion, both the upper and lower data words are available at the outputs of Data Set A. By virtue of ICB10 high input to the CDB Select logic, the Data Set Mux selects the lower word of Data Set A. The lower word as originally addressed is made available to the CPU on the CDB.

The output of the data Set Mux is also applied to the CMA Parity Checker. The parity bits are output back to the CAC Error Logic. If a parity error exists, the Error Logic outputs an Error status signal back to the CPU.

The Data Set outputs are available at the Data Mux for selection and output on the DATA bus to the CBI. However, since the CBI is not prepared to read data from Cache Memory at this time, this is a don't care condition.

Since Fail was not high at the rising edge of CLK3, the CPU receives the addressed data word, and it withdraws the Start signal, completing the Read operation.

3-109. Write Hit Operation

A Write Hit condition exists when data addressed by the CPU currently exists in Cache Memory. When the CPU addresses a data location for a Write, the CAC processes the operation similarly to a Read except for the differences described in the following paragraphs. The inputs to the CMA Write Logic are the same as for a Read operation except for the WRITE signal which is high. This combination of signals causes the ALOWE(L) to go low, allowing the lower word of the A Data Set RAM to be written to.

The WRITE signal from the CPU and the CSFREE signal from the State Machine are applied to the ADATA/CDB Mux. As a result, the Mux selects the PDB for output to the data inputs of the Data Sets.

The Data Set Mux selects the A Data Set lower word for output. This is a don't care condition, as the CPU is writing to Cache Memory. The Data Set outputs are applied to the Data Mux for selection and output to the DATA bus. This is also a don't care condition, as the CBI is not prepared to accept data from Cache Memory for this operation.

Since Fail is low, at the CLK3 edge the CPU finishes writing the data word into Cache Memory, and the START signal goes low, thus completing the Read Operation.

3-110. Clean Fault Operation

When the CPU addresses a word, Cache Memory uses bits 21-28 to address the block containing that word. If the block is not in either data set, and the Least Recently Used (LRU) block is "clean" (unmodified), Cache brings the required block from Main Memory and overwrites the LRU block.

When a Write or Read Clean Miss occurs, the Miss(L) signal goes active at the input to the Fail logic, causing a Fail signal to be sent to the CPU. When the CPU receives the Fail signal, it goes into a Freeze state and holds the START signal, and the Write signal at the desired level until the completion of the CPU operation. Recall that for a Clean Miss operation the Cache Memory sequences through WFB, RCVD, and back to CSHFREE, where it completes the CPU Write or Read operation. The Fail signal is also sent to the State Machine and to the Awake logic.

At the input to the State Machine, the Fail signal combines with CSHFREE to cause a WFB state. WFB is sent to the CSB Busop logic and causes a Read Block Private (RBP) Busop code to be sent to the CBI. WFB is also sent to the Awake Logic and to the Tag/Adr Mux. In the Awake logic, WFB causes the AWAKE(L) signal to be sent to the CBI. RBP indicates to the CBI that Cache Memory needs to read a data block from Main Memory. AWAKE(L) requests CSB access by Cache Memory and causes the CBI to initiate a CSB request. At the Tag/Adr Mux, WFB(L) selects CPU Address bits 5-20 out of the ADATA/CAR Mux and enables Address bits 0-4 and 21-31 from the CPU out of the ADR Block/Word Buffer for output to the CBI on the Data bus.

As soon as the CBI obtains the bus, it sends DSELCT to the CAC to acknowledge Cache Memory bus request. The CAC inputs DSELCT to the State Machine, the Word Count Select logic, and the Word Counter. DSELCT combines with WFB to cause the State Machine to shift Cache Memory state to RCVD and to select the Word Count 0 and 1 bits input to the ADATA/CAR Mux. The RCVD state signal, from the State Machine, is input to the CMA Write Enable logic where it combines with MISSA to produce AUPWE(L) and ALOWE(L). These signals are input to the Data Set RAMs to enable them to be written to.

The CSB and Main Memory require a minimum of nine system clock cycles to set up before data can be transferred to the CBI registers. As soon as Main Memory provides the first double-word (32 bits)

to the CBI, the CBI asserts a DATAOP signal to the CAC to indicate that the first word is ready for transfer. DATAOP generates a DATAIN signal which is applied to the Word Counter and, with RCVD, enables the Word Counter to count. DATAIN is also sent back out through the Enable CBI Registers logic as ENREGA(L) to the CBI where it enables the ADATA register onto the ADATA Bus. The CBI holds DATAOP (and thus DATAIN and ENBREG) asserted until the completion of the block read.

As soon as DATAIN is applied to the Word Counter, Word Count 0 is sent to the ADATA/CAR Mux, where it addresses word 0 of the block. This address is input to the Data Set RAMs in the CMA via the Intra-Cache Bus (ICB). CLKREGB(L) from the CAC then latches the addressed data block from Main Memory into the ADATA Registers in the CBI. Because Word Count Select (WCSEL) previously selected the WORD0 and WORD1 bits input to the ADATA/CAR Mux and the incoming double-word address was pre-loaded into this mux, the double-word appearing at the inputs to the Data Set RAMs is written into the location specified by the Word Counter.

The clock generation logic continues to clock data into the CBI data registers. When data becomes available at the inputs to the CBI data registers, the CBI asserts DATAOP. The resulting DATAIN toggles the Word Count from 0 through 3. This increments the double-word address to the Data Set RAMs. When the last word is written into the Data Set RAMs, CTR3 out of the Word Counter is asserted at the input to the State Machine, causing Cache to return to the CSHFREE state. As Cache reads the words of the data block from Main Memory, the Status logic changes the Valid bit associated with the block to indicate that the block is, in fact, in Cache Memory.

As START was asserted by the CPU at the beginning of the Clean Read Fault operation, Cache Memory proceeds with the CPU Read operation when it returns to the CSHFREE state.

3-111. Dirty Fault Operation

A dirty fault is a case where the data the CPU requests is not in Cache Memory, and the data that Cache Memory has in the required cache location, has been modified since it was copied from main memory. In this case, Cache Memory first writes its data to Main Memory, then reads the requested data from Main Memory. Cache Memory sequences from CSHFRE through WFB and SD, then back to CSHFRE to write the dirty block to Main Memory. It then sequences from CSHFRE through WFB and RCVD and back to CSHFRE to read the data block containing the requested data from Main Memory.

When a dirty Miss occurs, Cache Memory proceeds in the same manner as for a clean Miss: it sends Fail to the CPU which, in turn, goes into a Freeze state. Cache Memory then enters the WFB state. The WFB status signal combines with Miss and LRDTY in the CSB Busop logic to generate a Write Old Block Busop code. Cache Memory sends this code to the CBI where it indicates to the CBI that Cache Memory is going to Write an Old Block (WOB) to Main Memory.

When the CBI acknowledges a Cache Memory request with DSELCT, DSELCT combines with WFB and LRDTY to change the Cache Memory state to Send Data (SD). The SD status signal is then applied to the Status Update logic where it sets the DA(L) status bit associated with the dirty block high. This causes CMALRB to select the A upper and lower word inputs to the Data Mux on the CMA.

DSELCT and SD also go to the Word Counter to enable the word count, and to the WCSEL logic to enable WCSEL. Just as for a Read operation, WCSEL selects the word count bits (WORD0 and WORD1) out of the ADATA/CAR Mux. The ICB output of this mux addresses the word to be read out of the A Data Set.

With the word count started, each double word is read out of the A Data Set onto the DATA Bus. When the word count reaches 3, CTR3 is input to the State Machine and causes Cache Memory to change to the CSHFRE state. Since the CPU still has START asserted for its attempted read or write operation, Cache Memory immediately goes back to the WFB state to process a normal Clean Fault operation.

3-112. Send Message to CSB Operation

Messages from the CPU to the CSB may include instructions or requests for status addressed to modules on the CSB. The CPU transmits messages over the CAB and the Bus Command over the PDB. Cache Memory places them directly on the DATA Bus. The CPU signals the transmission of a message with a BUSC(L) command to the Cache Memory.

When the CPU sends BUSC(L) to Cache Memory and the PDB is coded for a Send Word command, Cache Memory asserts Fail and goes to the WFB state. The protocol for WFB during a BUSC operation is the same as for a FAULT. At the time CPU asserts BUSC(L), it also sends the Send Message opcode on the PDB to Cache Memory. PDB bits 3-15 are input to the BUSC (Send Word) Decoder which decodes MSGOP from bits 6-9 and applies it to the CSB Busop logic. The PDB bits 10-12 are also input to the BUSC Decoder and sent to the the CBI as the message destination (To Code). The CSB Busop logic outputs MSGOP on the Opcode bus to the CBI to alert it to the pending message transfer.

By this time, the Cache Address Register contains the CPU message. Shortly after Cache Memory asserts Fail, it sets MESS(L) and sends it back to the CPU. This indicates that Cache is prepared to send the message but is still waiting for the CSB.

At this point in the BUSC operation the following conditions are set and waiting for the CBI to indicate (with DSEL) that access to the CSB has been granted:

a. The CPU has set BUSC(L) low and sent Cache Memory the message on the CAB and the message opcode on the PDB;

b. Cache Memory has stored the message in the Cache Address Register and transmitted the MSGOP code to the CBI;

c. Cache Memory has set the Fail signal high to the CPU and the Busop and To Code to the CBI;

d. Cache Memory has set AWAKE(L) low when it went to the WFB state.

e. Cache Memory has set the Message in progress (MESS) status signal low.

When the CSB becomes available and the CBI asserts DSEL, Cache Memory immediately puts the CPU message on the bus through the CBI. DSELCT causes the State Machine to shift to the CSHFRE state and to output the DONE status signal. This signal is input to the Fail logic, where it drops the Fail and MESS(L) status signals to the CPU. This signals the completion of the message transfer to the CPU.

3-113. Read Message from CBI Operation

Messages to the CPU from modules on the CSB are input on the ADATA Bus to Cache Memory, and then to the CPU via the CDB. Recall that CSB messages consist of three parts: the upper and lower words, and the From Code. The BUSC(L) process for an incoming message does not cause Fail to go

high or request the CBI to get onto the CSB. The CPU separately issues the bus commands to read the message. Typically, for this Read Message operation, the CPU loads PDB bits 4 and 5 to command Cache Memory to first send the upper word, then the lower word, then the From Code.

The CBI sets Send Word Operand (SWOP) and/or Send Address Operand (SAOP) active to signal the arrival of an incoming message. These signals are input to the Interrupt and Busy logic where either will produce MSGINT and BUSY. Cache Memory sends MSGINT to the CPU, causing causes it to issue a Bus Command. It sends BUSY to the CBI as a status indicator to other modules that the CSB is being used for a message transaction. The CPU issues the Read From Code, Read Upper, and Read Lower in any combination during one clock. (Remember that the CPU is a 16-bit processor, thus it can read only half the 32-bit message during each clock cycle.)

Cache Memory also applies MSGINT to the Cache Clock Generation logic to generate CLKFRM and CLKREGA. These signals are sent to the CBI where they clock the incoming message and FROM code into the CBI ADATA latches and FROM register.

When the CPU responds to MSGINT with BUSC(L), Cache Memory sends ENREGA to the CBI to enable the output of the Message From register. The CBI sends the From Code output of this register to the CMA where it is applied to the Data Set Mux.

The CPU may read a From Code, a MSGUP or MSGLO word in any sequence, and it may send any one or all of these message operation commands. Which code the CPU sends and in what sequence is entirely dependent upon the microcode. Each message operation is initiated by BUSC(L) as an operation separate from any other message operation. At the time the CPU sends BUSC(L) to Cache Memory, it also puts the Message Opcode on the PDB. Cache Memory decodes the PDB in the BUSC Decoder and, for example, if the code is MSGUP, it applies the MSGUP to the Data Set Mux in the CMA. The upper word of the message from the CBI is output from the Data Set to the CPU on the CDB. When the CPU completes this message operation, it may repeat the process for another message operation on the next clock cycle. When the CPU completes its message operation, it drops BUSC(L), allowing the Cache to proceed with other operations.

3-114. Check and Be Memory Operations

While in either the WFB or CSHFRE states, Cache Memory constantly monitors the CSB for activity. If it detects activity, it goes to the CHCK state to compare the address on the CSB with the addresses of Cache data. If a valid match exists, the block is dirty, and the CSB transaction is a Main Memory read, Cache Memory will abort the Main Memory read and send the data to the requesting module. If a match exists but the block is clean, Cache marks its block Invalid. (Assume Cache is in the WFB state.)

Read Block Private (RBP) prompts the start of a Check operation. These signals indicate a bus transaction by a module (other than Cache Memory) on the CSB. When the Cache receives RPB WNB, it generates a WASANY status signal. This signal is input to the State Machine where it causes Cache Memory to shift through the CSHFRE state to the CHCK state (taking two clocks to make the change in states).

The CHCK status signal output from the State Machine is applied to the ADATA/CAR Mux where it selects the ADATA Bus input. The address on the ADATA Bus is then compared to the contents of the A and B Set TAG RAMs. If a Miss occurs, CHCK is reset on the next clock cycle, the Cache returns to CSHFRE and back to WFB one clock later.

If a Hit occurs and the CSB transaction is a write to Main Memory, CHCK is input to the Status Update logic where it marks the block Invalid. Cache Memory then returns to the CSHFRE state on the next clock, and back to the WFB state one clock later.

If a Hit occurs and the CSB transaction is a read from Main Memory, RBP or RBS from the CSB generates WASRD in Cache Memory. This status signal is combined with Hit Dirty (HITDTY) and CHCK in the Abort logic to generate ABORTTB. ABORTTB is sent to the CBI to interrupt the Main Memory transaction and to the State Machine to set Cache Memory to the Be Memory (BEMEM) state. The BEMEM state progresses through a write to CSB operation in the same manner as in the SD state.

The CPU sets the signal LOCK high when it is reading Cache Memory data. When LOCK is high no other module may access Cache data. If the Cache is LOCKed when a Dirty Hit occurs during a Check operation, and the CSB transaction is a read from Main Memory, LOCK is input to the State Machine where it prevents CHCK, HITDTY, or WASRD from setting Cache to the BEMEM State. It does not affect the ABORT logic, however, which remains asserted to prevent the requesting module from reading invalid data from Main Memory.

NOTE

As long as the Cache is LOCKed and WASANY is high, the State Machine oscillates clock-to-clock between CSHFRE and CHCK. When the CPU writes to Cache Memory and LOCK is cleared, Cache proceeds to CHCK and then to BEMEM.

LOCKED is also input to the Status Update logic. When asserted, it causes the status of the block being checked to remain unchanged. When LOCKED is not high, it combines with CHCK to generate CHCKOK. CHCKOK allows the blocks to be marked Invalid. When CHCKOK is asserted, it causes the status of the block NOT involved in the CSB transaction to be marked Least Recently used Block (LRB).

When Cache Memory changes from the CHCK State to BEMEM, it sets up conditions similar to the SD State within the CAC for data transfer to the CSB. The BEMEM status signal from the State Machine is input to the ADATA/CAR Mux to select the ADATA Bus for input to the Data Sets. This status signal is also input to the BUS Decoder to force the TOENB output. Cache Memory sends TOENB to the CBI where it enables the CBI to use the requesting module's address as the data block destination. BEMEM also goes to the ENREG, AWAKE, and WORD COUNT logic where it performs the same functions as the SD status signal in the SD State. After the Word Count reaches 3, Cache Memory returns to CSHFRE.

MAIN MEMORY

SECTION IV

This section describes the Main Memory Module of the HP 3000 Series 64 Computer. As any major module, it accesses the rest of the computer via the Central System Bus (CSB), and has a Common Bus Interface (CBI) PCA to communicate with the CSB. See Figure 4-1.

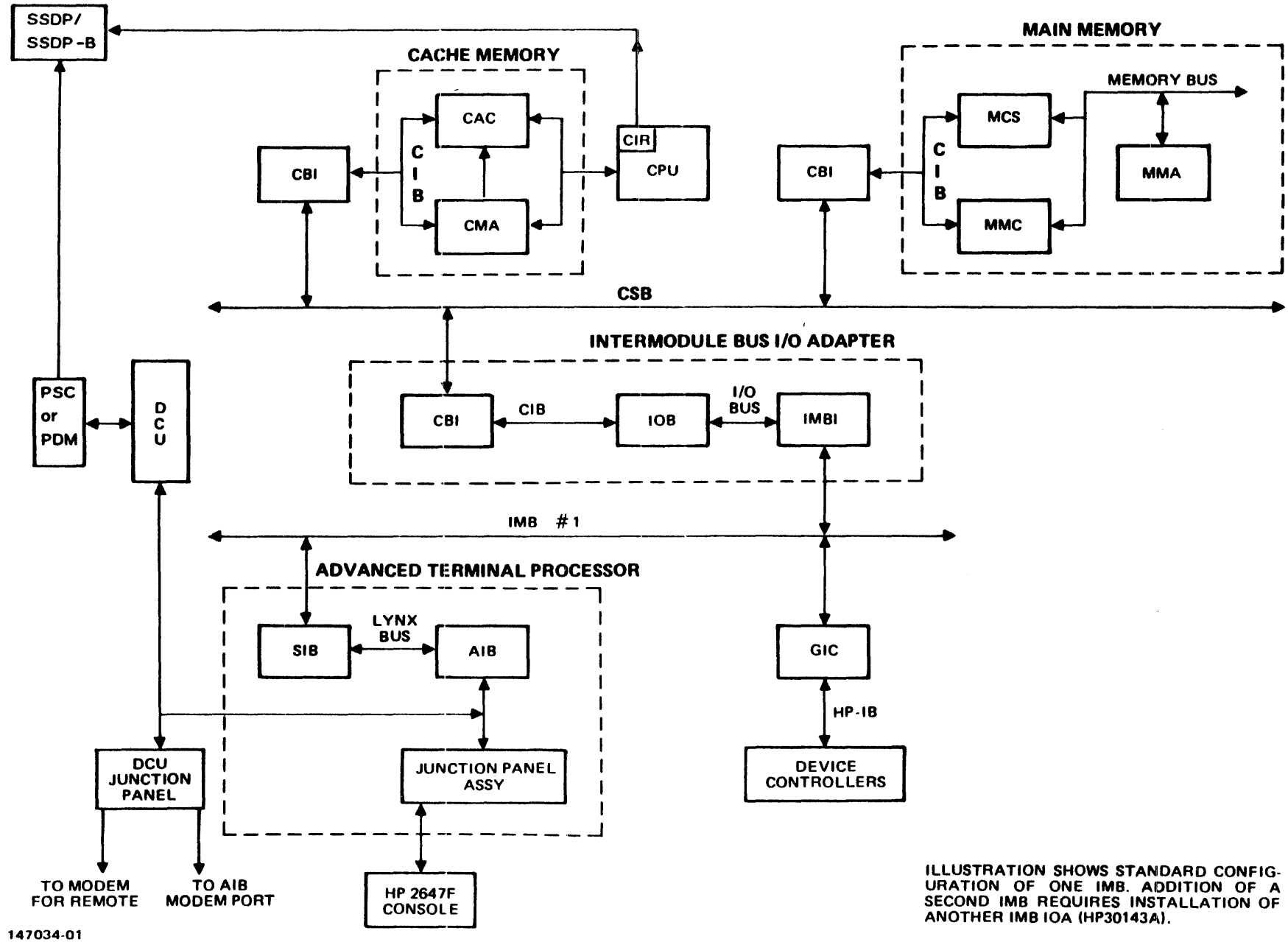


Figure 4-1. Major Modules of the Computer

4-1. INTRODUCTION

The Main Memory Module provides the computer's data storage. Its PCAs include Main Memory Arrays (MMAs), a Main Memory Control (MMC), Memory Correction and Storage (MCS), and the Memory Backplane (MBP).

The MBP has 11 slots. Three hold CBI, MMC and MCS PCAs. The remaining eight slots are reserved for MMAs (the computer will work with only one MMA, but is normally configured with at least two).

4-2. Error Detection and Correction

Single-bit data errors are automatically detected, logged and corrected. All double-bit and some multiple-bit errors are detected and logged. Power failure will not cause data errors; a power-down procedure ensures that data in the Cache Memories is transmitted to Main Memory before the power loss is effective. Main Memory is protected by battery-backup for at least 15 minutes, depending on the total computer load.

4-3. Data Format

Data is transmitted from Main Memory to other major computer modules via the CSB. Data is transmitted in blocks, a block consisting of eight 16-bit words. (A block is also said to consist of four 32-bit "double words".) See Figure 4-2.

During CSB transmission, each word is joined by a parity bit. Within Main Memory, the 32-bit double word structure is retained, but the two parity bits are replaced by seven "syndrome" (error detection) bits. See Figure 4-2.

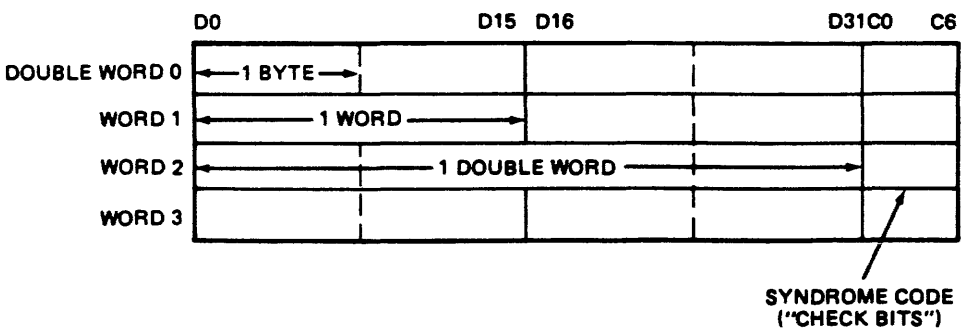


Figure 4-2. Data Block Format Within Main Memory

4-4. Addressability

The MCS PCA accepts address bits ADATA 00-31. Each address corresponds to one data word in memory. See Figure 4-3.

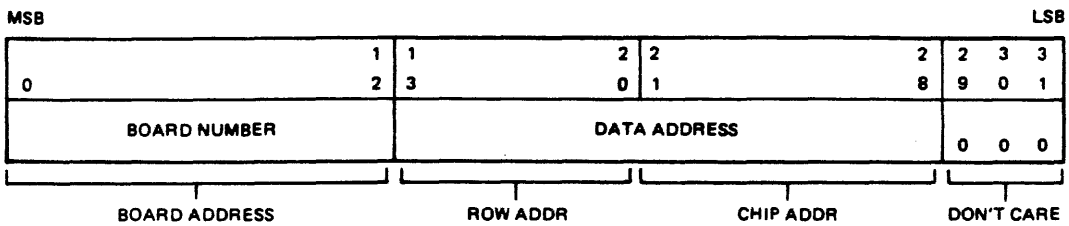


Figure 4-3. Memory Address Format

4-5. Component Technology

Transistor-Transistor Logic (TTL) and Emitter-Coupled Logic (ECL) are used in Main Memory. ECL is used in the circuits where data is transmitted to or from the CSB, and in the clock and timing circuits. The speed of ECL components is responsible in part for the increased memory storage capacity of the computer. The refresh and memory array are Schottky and low power Schottky TTL.

4-6. PHYSICAL COMPONENTS

Main Memory consists of the MMC and MCS PCAs and from one to eight MMA PCAs, all of which plug into the MBP. (Basic configuration requires at least two MMAs). See Figure 4-4.

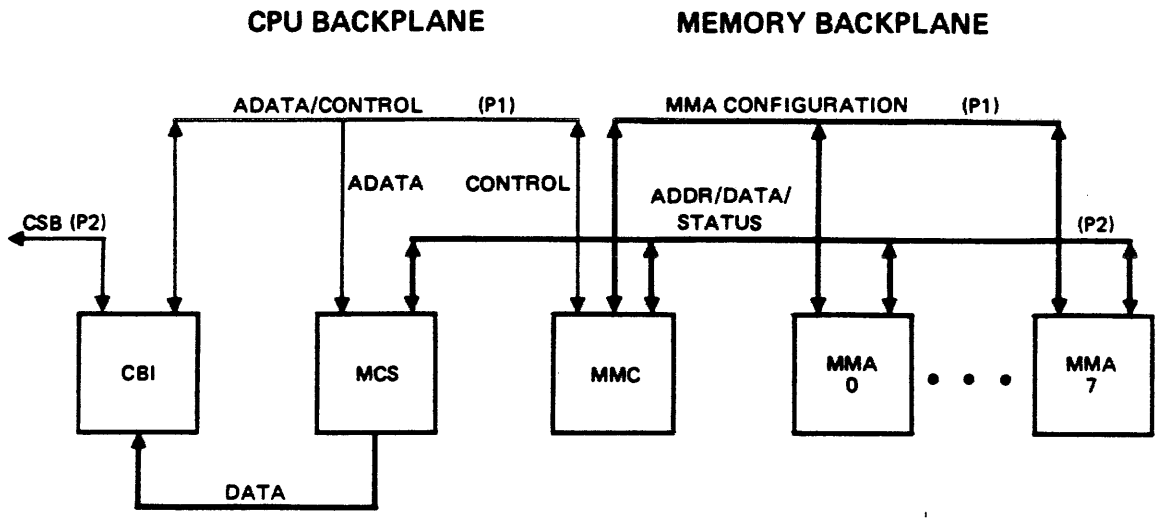


Figure 4-4. CBI/Main Memory Interface

As shown in Figure 4-4, a CBI is required to interface Main Memory to the CSB. Control, status, and timing signals are transmitted to and from the MMC PCA and the MCS PCA. These PCAs process the signals and transmit the required data, address and control signals to the Main Memory Arrays where the data is stored in Random Access Memory (RAM).

4-7. Main Memory Control PCA

The MMC PCA receives control signals from the computer and processes them for the memory array. Refresh pulse and address counters are generated on this PCA. Control for memory read and write sequences is also generated on this PCA.

4-8. Memory Correction and Storage PCA

Data going to and from the memory array is transmitted via the MCS PCA. Data is stored in an eight-word register file during a write operation if memory is doing a refresh. Seven syndrome bits are generated to detect errors in the data that occur during read and write operations. Two parity bits are generated for data transmitted to the CBI. The MCS PCA has error-correction circuitry to locate and correct single-bit errors in data reads from RAM, and also contains error logging and message logic, memory status registers, and the address registers.

4-9. Main Memory Array PCAs

The MMA PCAs contain dynamic RAMs used for data storage. Each MMA holds 1 Mbyte of memory. Each has control logic for the RAMs, address decoders, and memory size indication for the CPU.

4-10. MEMORY FUNCTIONS AND OPERATING CHARACTERISTICS

a. Data Writes. Data with an address is sent to the memory in a block of four double (32-bit) words. A 7-bit check word is generated for each double word and stored with the data.

b. Data Reads. An address, with parity, is supplied to memory. The data block in that location is read one 39-bit word at a time and checked against the syndrome bits.

c. Message responses. This function allows the internal workings of the memory to be interrogated for status indications or logged errors.

4-11. Access Speed

Memory reads require 12 system clocks, and writes 13 system clocks to complete. The read timing includes one clock for address on the CSB, seven memory clocks to access the data, and four clocks for data on the CSB. Writes require one clock for address on the CSB, four for data, and eight for memory to complete the write.

4-12. Access Bandwidth

Peak memory bandwidth is 17.8 MBytes/s (16 bytes in twelve 75-nanosecond clocks) without refresh. Refresh could cause memory to delay reads or writes by up to ten clocks every 15 to 30 us. This gives an effective total memory bandwidth of 16.8 to 17.3 MBytes.

4-13. Message Speed

Message commands (Send Word) to memory require five clocks: one for the message on the CSB, three for memory to handle the message, and one for memory to return a message, if required.

4-14. MEMORY OPERATIONS

There are two types of Main Memory operations: messages (known as Send Word Operations, SWOPs), and accesses (known as Read Block Private, RBP, or Write Old Block, WOB). If a response to a message or access is required, memory automatically returns the information to the requesting module.

The Memory Message State Machine constantly monitors the state of the SWOP BUSOP line. This opcode causes memory to interpret the contents of the ADATA lines as a message to memory. (The ADATA lines are located on the Common Interface Bus, CIB, which is the cable that connects the Main Memory Module to its CBI PCA.) Similarly, the Memory Access State Machine monitors the state of the RBP and WOB BUSOP lines. For these operations, memory begins an access (read or write) cycle.

4-15. Messages

There are three message formats: Read/Write Status, Read/Write Logging Ram, and Read Register File Data. (If more than one format is specified at one time, invalid functions will be performed.) CBI status can be cleared in any of the read or write formats.

The Read Logging RAM format, in addition to returning Logging RAM information, allows memory to return seven check bits. These are generated on one of the 32-bit double-words stored in the register file at the location specified by RFADDR0-2. Location 7 contains the current message, and locations 0-6 contain random information unless the last command to memory was a write. In that case, locations 0-3 contain the four double-words written into Main Memory during that write.

The Memory Message State Machine activates whenever a SWOP BUSOP is sent over the CSB to Module 7. The contents of the CSB data lines while the BUSOP is present are interpreted as the message to memory. The message state machine will automatically cycle through four states (not including Idle) on the next four system clocks, as follows:

State 0. Idle.

State 1. Message (MSG) - Information on the CSB Data lines is stored on the MCS in the register file at address 7, and interpreted by memory.

State 2. Awake (AWK) - If Bit 7, 8, or 9 is set, memory will request the CSB to return a message to the requesting module.

State 3. Read State (RSTATE) - The registers selected by Bit 7, 8, or 9 in State 1 are gated to the Memory's CBI PCA No. 7.

State 4. Write State (WSTATE) - If a write, enabled by Bits 10 or 11, is to take place, the appropriate registers are loaded.

Once State 3 is entered, the MMC waits for the CBI to be selected on the CSB before going back to idle. If select is delayed, the data chosen during State 2 is stored on the CBI, until select goes active. The Message State Machine goes to idle on the first clock following select. If no read option is specified it is not necessary for the CBI to be selected; the memory goes idle independent of select.

Tables 4-1 through 4-3 show the three formats for messages to and from Main Memory.

Table 4-1. Read/Write Status Message Format

BIT NUMBER	TO MEMORY	FROM MEMORY
00	X	ILLEGAL BUSOP
01	X	DATA PARITY ERROR
02	X	CONTROL PARITY ERROR
03	X	ERROR BUSOP
04	X	NACK ERROR
05	X	ADDRESS/DATA PARITY ERROR
06	X	ACK ERROR
07	READ STATUS	MEMSIZE00
08	0	MEMSIZE01
09	0	MEMSIZE02
10	WRITE STATUS	MEMSIZE03
11	0	MEMSIZE04
12	CLEAR CBI	MEMSIZE05
13	X	MEMSIZE06
14	X	MEMSIZE07
15	X	MEMSIZE08
16	X	MEMSIZE09
17	X	MEMSIZE10
18	X	MEMSIZE11
19	X	MEMSIZE12
20	X	MEMSIZE13
21	X	BOARDNUM0
22	X	BOARDNUM1
23	X	BOARDNUM2
24	MULTIPLE BIT ERROR	MULTIPLE BIT ERROR
25	SINGLE BIT ERROR	SINGLE BIT ERROR
26	NO ARRAY CARD	NO ARRAY CARD
27	WRITE TIMEOUT	WRITE TIMEOUT
28	DISABLE ERROR LOGGING	DISABLE ERROR LOGGING
29	DISABLE SYNDROME WRITE	DISABLE SYNDROME WRITE
30	DISABLE ERROR CORRECTION	DISABLE ERROR CORRECTION
31	SHUTDOWN	SHUTDOWN

Table 4-2. Read/Write Logging RAM Message Format

BIT NUMBER	TO MEMORY	FROM MEMORY
00	X	ILLEGAL BUSOP
01	X	DATA PARITY ERROR
02	X	CONTROL PARITY ERROR
03	X	ERROR BUSOP
04	X	NACK ERROR
05	X	ADDRESS/DATA PARITY ERROR
06	X	ACK ERROR
07	0	X
08	READ LOGGING RAM	X
09	0	X
10	0	X
11	WRITE LOGGING RAM	CHECKBIT0
12	CLEAR CBI	CHECKBIT1
13	X	CHECKBIT2
14	X	CHECKBIT3
15	RFADDR0	CHECKBIT4
16	RFADDR1	CHECKBIT5
17	RFADDR2	CHECKBIT6
18	LOG RAM ADDR0	LOG RAM ADDR0
19	LOG RAM ADDR1	LOG RAM ADDR1
20	LOG RAM ADDR2	LOG RAM ADDR2
21	LOG RAM ADDR3	LOG RAM ADDR3
22	LOG RAM ADDR4	LOG RAM ADDR4
23	LOG RAM ADDR5	LOG RAM ADDR5
24	LOG RAM ADDR6	LOG RAM ADDR6
25	LOG RAM ADDR7	LOG RAM ADDR7
26	LOG RAM ADDR8	LOG RAM ADDR8
27	LOG RAM ADDR9	LOG RAM ADDR9
28	LOG RAM DATA0	LOG RAM DATA0
29	LOG RAM DATA1	LOG RAM DATA1
30	LOG RAM DATA2	LOG RAM DATA2
31	LOG RAM DATA3	LOG RAM DATA3

Table 4-3. Read Register File Data Message Format

BIT NUMBER	TO MEMORY	FROM MEMORY
00	X	REGISTER FILE DATA00
01	X	REGISTER FILE DATA01
02	X	REGISTER FILE DATA02
03	X	REGISTER FILE DATA03
04	X	REGISTER FILE DATA04
05	X	REGISTER FILE DATA05
06	X	REGISTER FILE DATA06
07	0	REGISTER FILE DATA07
08	0	REGISTER FILE DATA08
09	READ REGISTER FILE DATA	REGISTER FILE DATA09
10	0	REGISTER FILE DATA10
11	0	REGISTER FILE DATA11
12	CLEAR CBI	REGISTER FILE DATA12
13	X	REGISTER FILE DATA13
14	X	REGISTER FILE DATA14
15	RFADDR0	REGISTER FILE DATA15
16	RFADDR1	REGISTER FILE DATA16
17	RFADDR2	REGISTER FILE DATA17
18	X	REGISTER FILE DATA18
19	X	REGISTER FILE DATA19
20	X	REGISTER FILE DATA20
21	X	REGISTER FILE DATA21
22	X	REGISTER FILE DATA22
23	X	REGISTER FILE DATA23
24	X	REGISTER FILE DATA24
25	X	REGISTER FILE DATA25
26	X	REGISTER FILE DATA26
27	X	REGISTER FILE DATA27
28	X	REGISTER FILE DATA28
29	X	REGISTER FILE DATA29
30	X	REGISTER FILE DATA30
31	X	REGISTER FILE DATA31

4-16. Memory Accesses

There are two types of Main Memory accesses: reads and writes. Writes are accomplished by the Write Old Block (WOB) BUSOP. Reads are accomplished by the Read Block Private (RBP) BUSOP. During accesses, memory does not distinguish between "clean" (unmodified) and "dirty" (modified) data blocks; it just performs the read or write.

4-17. MEMORY READS. The Access State Machine is in the Idle State until read operation and read address signals are received. Bits 15-28 of ADATA go to the Memory Address Buffer and from there go to the MMAs. (See Figure 4-5). The READOP signal goes to the MMC PCA. The Access State Machine then enters State 0.

State 0. The read address passes through the CBI and MCS to be decoded on the MMA PCA. The READOP is clocked into the MMC. If the array is idle (no refresh in progress), the state machine goes to State 1. If the array is being refreshed, the state machine remains in State 0 until the refresh is finished.

State 1. An MSTART signal from the MMC is sent to the MMAs and the appropriate MMA is enabled after the address is decoded on the MMAs.

State 2. The MMC remains in State 2 until the data words D0, D1, and D2 are about to be read from the RAMs and latched on the MMA PCA. This takes three clocks. MBUSY is a control signal on the MMC that indicates when all four data words have been latched. This occurs on the same clock that latches D3.

State 3. Data word 3 is read from the RAMs and latched on the MMA PCA. MBUSY goes high just prior to State 3, putting the Access State Machine into the single-step mode. This means that steps 4-8 may or may not be performed on sequential clocks.

State 4. D0 is read from the latches on the MMA, then pipelined across the Memory Backplane (MBP) to the MCS PCA. The MMA uses ABUSY(L) to indicate to the MMC that the first double word will be available in 2.5 clocks. This signal becomes MBUSY on the MMC. The D0 syndrome bits are checked and any errors detected. Data is latched in the CBI registers. If a single-bit error is detected, the data is corrected on the internal bus, and the logging RAM marked with the RAM address. If a multi-bit, or No Array Card error occurs, the error is logged. DATA BUSOP, which normally transmits the data, is replaced with ERROROP, which causes a system interrupt.

State 5. D1 follows the same procedure as D0 followed in State 4.

State 6. D2 follows the same read procedure.

State 7. D3 follows the same read procedure.

State 8. The Access State Machine returns to the Idle State.

See Figures 4-5 and 4-6 for the Read Address and Read Data Path flows.

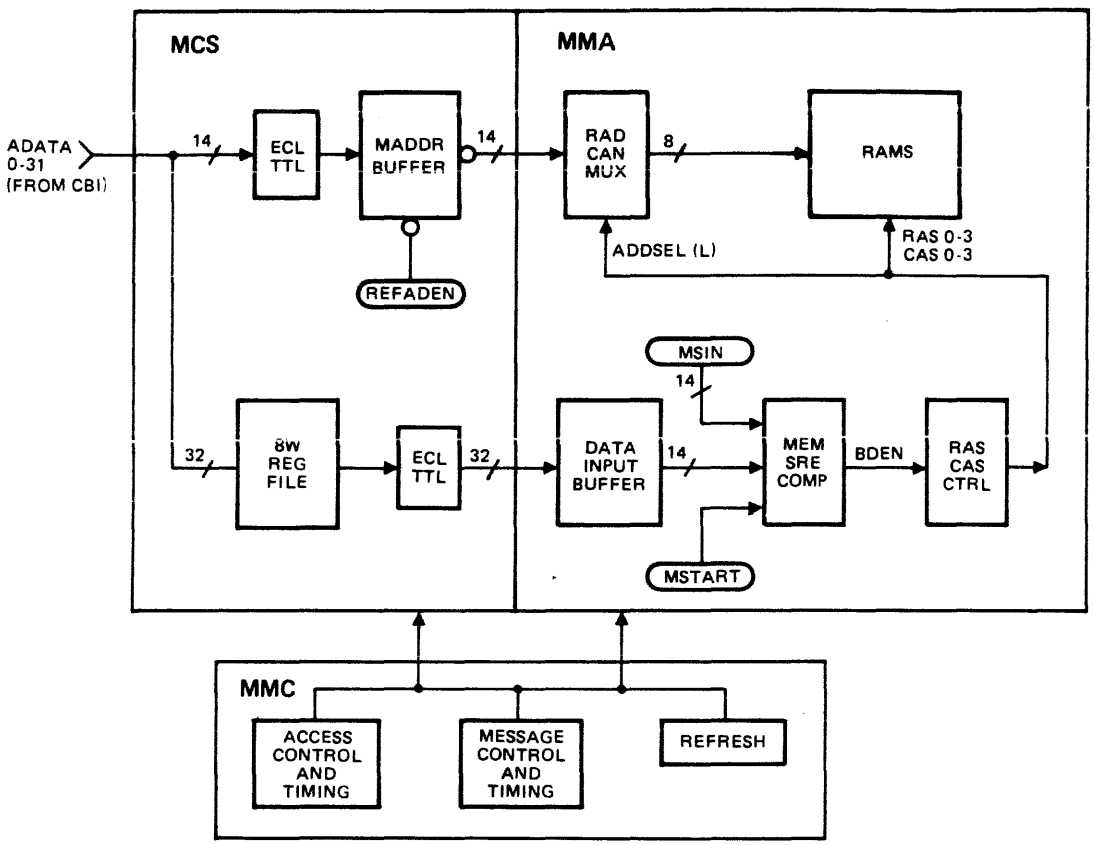


Figure 4-5. Read and Write Address Path

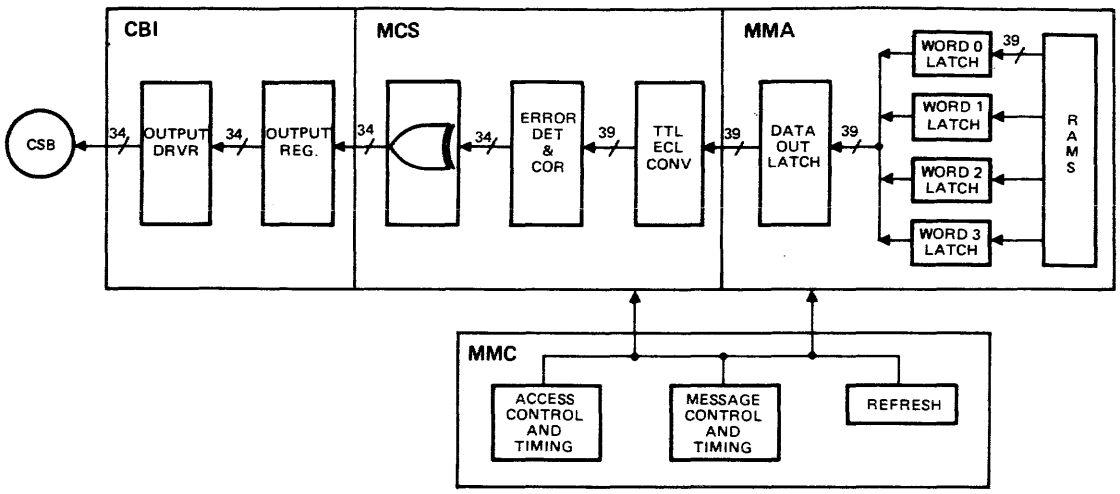


Figure 4-6. Read Data Path

4-18. MEMORY WRITES. In the case of memory writes, the Access State Machine is in the Idle State until a WRITEOP together with a RAM address is received. The address goes to the Register Files and to the MMA. The Access State Machine is forced to State 0.

State 0. The address is decoded on the MMA. D0 is loaded into the Register Files, and transmitted to the MMA if the array is idle. If the array is being refreshed, the MMC State Machine stays in State 0 until the refresh is complete, while the data is loaded into the Register Files.

State 1. D0 and its check bits are written into the array.

State 2. D0 and its check bits are held in the array.

State 3. D1 and its check bits are written into the array.

State 4. D1 and its check bits are held in the array.

State 5. D2 and its check bits are written into the array.

State 6. D2 and its check bits are held in the array.

State 7. D3 and its check bits are written into the array.

State 8. D3 and its check bits are held in the array.

See Figures 4-5 and 4-7 for the Write Address and Write Data paths.

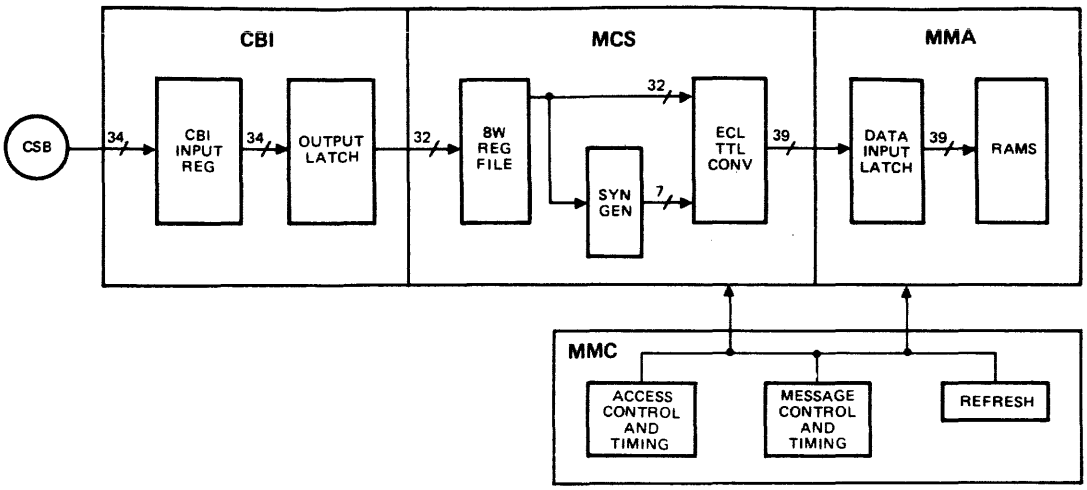


Figure 4-7. Write Data Path

4-19. Operating Modes

There are two basic memory operating modes: Single Step and Free Run. The normal operating state is called Single Step, as operating sequences are controlled by system syncs.

Free Run is used for read, write, or refresh operations, which cannot be interrupted for several clocks once they have been initiated. The MMC and MCS are capable of single step operation, but the MMA is not. The MMA, once activated, must not be stopped because the dynamic RAMs will lose their data if they are not refreshed every 2 ms. The MMC and MCS are capable of generating their own clocks from the system sync, which can be turned on and off by the DCU, or from an internal continuous sync which never stops as long as power is supplied. If the system sync stops, or when the Main Memory is in the Free Run Mode, these continuous clocks run the memory at the same system clock rate (75 ns).

Main Memory will ignore any attempts to abort a write. When the write begins, it must be carried to completion. Failure to complete the access will leave memory in an intermediate write state which will require system attention.

4-20. Refresh Cycles

For dynamic RAMs to retain data integrity they must be refreshed periodically. This is accomplished by supplying the RAMs with a refresh address and a timing pulse. The MMC contains a timer and address counter for this purpose. The timer or refresh oscillator is temperature-controlled such that a refresh request is made every 15 to 30 us depending on the ambient temperature. At that time one row of each RAM in the system is refreshed.

The refresh circuit is synchronized with the access state machine. Therefore, if an access is in progress when a refresh is needed, the refresh will be delayed until the MMAs have finished their access. Similarly an access starting during a refresh will be delayed by the amount of time it overlaps the refresh cycle. This could be as long as 10 clock cycles.

4-21. Power Fail Sequence

When power begins to fail, a power fail sequence is initiated that puts memory into a battery backup state. In this backup state, memory cannot be accessed, although it can respond to messages. The +5 volt supply is provided by a battery-operated DC-to-DC converter and refresh is done automatically. Therefore, when power fails, nothing in memory is lost. When memory is brought out of the backup state, normal operation can resume. The signals SHUTDWN and TTLPON initiate the power fail sequence.

4-22. SHUTDOWN. The SHUTDWN signal is a control bit stored on the MCS that puts memory in a "protect" mode. This bit can be set as a result of a message from the CPU or by the DCU shifting it in through the shift string. Immediately after SHUTDWN goes high, the MMC state machine goes into memory protect, preventing memory access. One refresh cycle later the system is in backup mode.

When the CPU receives a power fail warning signal, it halts activities and saves the state of the machine. Once this is done, the CPU sends a SHUTDWN message to Main Memory. After power returns and SHUTDWN goes low, one refresh cycle is required for the system to allow access again.

4-23. TTLPON. The TTL Power On signal is active whenever DC power is within a designated threshold region. It goes low when DC power drops below this threshold. When TTLPON goes low, memory goes into backup immediately, putting the MMC state machine into memory protect mode. When TTLPON goes high, and if SHUTDWN is low, the system will come out of backup after one refresh clock.

4-24. Error Detection and Correction

Main Memory corrects all single-bit errors and detects all double-bit errors. A seven-bit check code is generated with each 32-bit data word. During a read, the 39 bits are used to generate a 7-bit syndrome code that determines whether the data is correct or in error. For a single-bit error, the syndrome code is decoded to show which bit is incorrect and must be corrected. Since parity for each of the two double words has already been generated, the appropriate parity bit must also be complemented.

The seven-bit syndrome generator produces a unique code for each of the 32 data and 7 check bits that could be in error and unique codes for specific groups of double-bit error pairs.

4-25. Error Logging

Error correction, detection, and logging occurs during the normal cycle time. If a single-bit error occurs, not only is the error corrected, but it is also logged and the single-bit error status bit is set. For a multiple-bit error, a bit is set in the logging RAM and the multiple-bit error status bit is set but the data is not modified. The CSB BUSOP transmitted with the data from the read in the latter case is ERROP, rather than DATA, as normally used.

The error logging array consists of four 1K X 1 static RAMs on the MCS PCA. For each dynamic RAM in the Main Memory, there is a corresponding location or bit in the logging array; if a one is stored in that location, then a single-bit error occurred in that dynamic RAM. The additional locations in the logging array are used for storing information about double-bit errors and some multiple-bit errors.

The address and chip selects for the RAMs consist of three bits for PCA number, two bits for word number, and seven syndrome bits. Two of the syndrome bits are used to generate one of the four RAM chip selects during the logging process so that only one location in one of the RAMs is written into at a time.

4-26. MEMORY DIAGNOSTICS AND TEST STRATEGY

Diagnostic and test capability were important considerations in the design of Main Memory. The memory has been designed for ease of fault location. Status bits are provided to store error conditions. Control bits modify the operation of the memory system, and diagnostic shift strings are used where possible. Controls are available to allow microdiagnostics and software diagnostics to isolate PCA failures.

4-27. Status Conditions and Control Information

Main Memory has several status bits that store error conditions and several control bits which modify the way memory control works. The error conditions saved are WTIMOUT, NOACARD, SNGERR, and MULTERR.

DEC, DSW, and DEL are the control bits available. These bits can be set/cleared by sending messages to memory or by the DCU shift strings. They enable the diagnostics to disable parts of memory, which, in turn, allows other parts of the hardware to be more thoroughly tested. DEC, DSW, and DEL are defined as follows:

a. Disable Error Correction (DEC)

DEC, by not correcting any errors that may be in memory, allows diagnostics to check that there are errors that memory is correcting. This signal also prevents any error conditions, such as multiple-bit error, from halting the system.

b. Disable Syndrome Write (DSW)

DSW allows the diagnostic to artificially introduce errors into the array to check the error detection and correction hardware as well as the error logging control and RAM. This is accomplished by preventing the check bits stored on the MMA from being overwritten when the data corresponding to these bits is modified. By first writing a block of data with DSW low, then writing the same block of data with DSW high and one bit complemented in each double word of the block, the hardware corrects the complemented bit when the block is read back. The error is also logged in the Logging RAM.

c. Disable Error Logging (DEL)

DEL allows the diagnostic to turn off error logging to prevent the logging RAM from being modified while it is being tested.

4-28. Diagnostic Testing

Memory can be tested three ways: by shift strings from the Diagnostic Control Unit (DCU) PCA, by microdiagnostics, and by software diagnostics.

a. Diagnostic Shift Strings

The designs of the MMC and MCS were implemented using shift registers to store data, control information, and other key signals. When problems occur, it is possible for the DCU to shift out this information for examination.

b. Microdiagnostics

The microdiagnostics step through the various levels of the memory hardware. The following tests are performed: memory receive/send message test, memory read test and write test, logging RAM read/write test, memory address test, memory syndrome/error correction test, and memory abort test.

c. Software Diagnostics

A software diagnostic tests all memory control functions. It forces single-bit error detection and correction and multiple-bit error detection. It tests all of the memory array and tries to log errors into the Logging RAMs. Before and after execution of the diagnostic, the Logging RAMs are read and may be displayed.

4-29. MEMORY LOGGING

The memory error logging facility allows the operator to examine the error history of memory. The facility consists of:

a. Memory error logging system process (MEMLOGP)

b. Memory error log analysis program (MEMLOGAN)

c. Memory error logging interval update program (MEMTIMER)

Memory error logging functions independently of standard system logging. Operator interface has no affect on memory logging.

4-30. Memory Error Logging System Process (MEMLOGP)

MEMLOGP runs under HP's Multiprogramming Executive (MPE) operating system. Once initiated, MEMLOGP interrogates the MCS to obtain the latest information.

MEMLOGP is activated during the initialization phase of MPE. If MEMLOGP cannot be activated during initialization, another attempt cannot be made until the system is brought up again.

MEMLOGP attempts to open the system Memory Error Disc File (MEMLOG.PUB.SYS). A new file is created if one does not exist. If MEMLOG exists, the file will be opened without altering the information in the file. The file remains open as long as the system remains up. During the periods

when MEMLOGP is accessing the file, it will lock and unlock the file as necessary to avoid multiple access. The log analysis program (MEMLOGAN) similarly locks and unlocks the file during access periods.

If an operational error is encountered by MEMLOGP, the process will display an error message and terminate. The message is:

ST/<TIME>/MEMORY LOGGING ERROR#<ERRNUM>.LOGGING STOPPED

The range and definitions for <ERRNUM> are:

1-10	Internal MEMLOGP errors
1	FLOCK error on MEMLOG file
2	FUNLOCK error on MEMLOG file
20-500	File system errors involving MEMLOG file. All file errors encountered by MEMLOGP are fatal to the process and cause it to terminate.

Once the MEMLOG file has been opened, MEMLOGP periodically interrogates the error logging array. If errors have occurred, the MEMLOG file is updated. The array is interrogated and updated when MEMLOG is first activated and thereafter approximately once each hour. MEMLOGP performs the following operations:

a. Reads the appropriate MEMLOG record from disc.

b. Scans the error logging array for errors.

c. Updates the error counter in the MEMLOG record for each location where an error occurred.

d. Resets the error logging array to a no-error condition.

e. Writes the updated MEMLOG record to disc.

4-31. Memory Error Logging Interval Update Program (MEMTIMER)

MEMTIMER is a utility that allows the user to modify the time interval between memory log updates. The default is 60 minutes. This interval provides the average installation with an adequate log of the memory.

MEMTIMER alters the current interval to a new value and ends the current interval. This causes MEMLOGP to update the memory log file immediately, then again after the new interval.

A new interval can be specified by the PARM parameter of the RUN command. There is no user dialogue with MEMTIMER. The PARM value is given as a positive integer greater than 0 which represents the number of seconds between log file updates. To begin memory logging at 10-second intervals, the following RUN command would be used:

```
:RUN MEMTIMER; PARM=10
```

To return logging to the default interval of 60 minutes, the following would be entered:

```
:RUN MEMTIMER;PARM=3600
```

Three error conditions are detected by MEMTIMER. If the PARM parameter of the RUN command is equal to or less than zero, MEMTIMER will end after sending the message:

```
** INVALID PARM (DELAY) VALUE **
```

The current interval will then remain unchanged.

If MEMLOGP has been terminated, MEMTIMER will terminate after sending the message:

```
** MEMORY LOGGING PROCESS NOT ACTIVE **
```

In that case there is no timing interval update.

If MEMLOGP is currently updating the memory log file, the following message will appear:

```
** MEMLOGP TIMER ENTRY NOT FOUND **
```

When this message appears, the interval will be updated. MEMTIMER should be run again to ensure that MEMLOGP recognizes the updated interval immediately.

The default timing interval becomes the current timing each time the system is brought up. Therefore, if a non-default timing interval is desired, MEMTIMER must be run after each initialization of the computer.

4-32. Memory Error Log Analysis Program (MEMLOGAN)

MEMLOGAN is a utility that reads and interprets the error information logged and kept in the MEMLOG file. Because of the security placed on the MEMLOG file by MPE, MEMLOGAN cannot be read from any group.account other than PUBlic.SYStem, unless the RELEASE command was entered for MEMLOG by the system manager.

The default output device for MEMLOGAN is the terminal. You can direct the output to another device (such as LP) by using a :FILE equation referencing the formal file designator OUT, as follows:

```
:FILE OUT;DEV=LP
```

There is no user interaction with MEMLOGAN. However, certain MEMLOG file handling operations are available through the PARM parameter of the RUN command. The following PARM values are recognized by MEMLOGAN:

PARM=0. Causes the current contents of MEMLOG to be printed on the output device. The contents of the file will not be changed. This is the default value.

PARM=1. Causes the current contents of MEMLOG to be printed on the output device, after which the file is reset to a no-error state. All previously logged errors are deleted from the log file.

NOTE

When a computer is initialized for the first time or the memory size is changed, MEMLOGAN should be run with PARM=1 as soon as the computer is up and running. This will ensure a clean MEMLOG file and that subsequent error conditions are valid.

PARM=2. Causes the current contents of MEMLOG to be printed on the output device, after which the file is purged. (This is the only way to purge MEMLOG; normally only the system manager will do this.)

4-33. OUTPUT. MEMLOGAN output will vary according to whether the MEMLOG file has been updated, and if so, whether errors were detected. If no errors have been logged, MEMLOGAN will display a message like the following:

```
LOGGING STARTED     -DATE:
LAST LOG UPDATE     -DATE:

***NO ERRORS LOGGED***
```

4-34. ERRORS. If an error is detected by MEMLOGAN, the program will print an error message. A typical error message looks like the following:

```
LOGGING STARTED        - DATE:          TIME:
FIRST ERROR LOGGED     - DATE:          TIME:
LAST ERROR LOGGED      - DATE:          TIME:
LAST LOG UPDATE        - DATE:          TIME:
TIMING INTERVAL          0:00:01

-------------------------------------------------
|     ADDRESS     |       ERROR TYPE        | ERROR |
|-----------------------------------------  |
| BOARD | WORD    |   TYPE   BIT  CHIP      | COUNT |
-------------------------------------------------
|   0   |   1     |   DATA    7   U1504     |  154  |
-------------------------------------------------

END OF PROGRAM
```

4-35. OPERATOR ENTRY. The following commands will produce a line printer copy of the log file:

```
:HELLO FIELD.SUPPORT
:FILE OUT;DEV=LP
:RUN MEMLOGAN.PUB.SYS
```

I/O SYSTEM

SECTION V

The HP 3000 Series 64 Computer I/O System includes two major hardware entities: Intermodule Bus I/O Adapters, and Intermodule Bus I/O channels. As shown in Figure 5-1, the Intermodule Bus I/O Adapter (IMB IOA) includes the I/O Buffer (IOB), Intermodule Bus Interface (IMBI), and Common Bus Interface (CBI) PCAs. I/O channels are represented in Figure 5-1 by the General I/O Channel (GIC) and the Advanced Terminal Processor (ATP).

This section describes the general nature of the I/O System, then describes the Intermodule Bus I/O Adapters and the Intermodule Bus I/O channels.

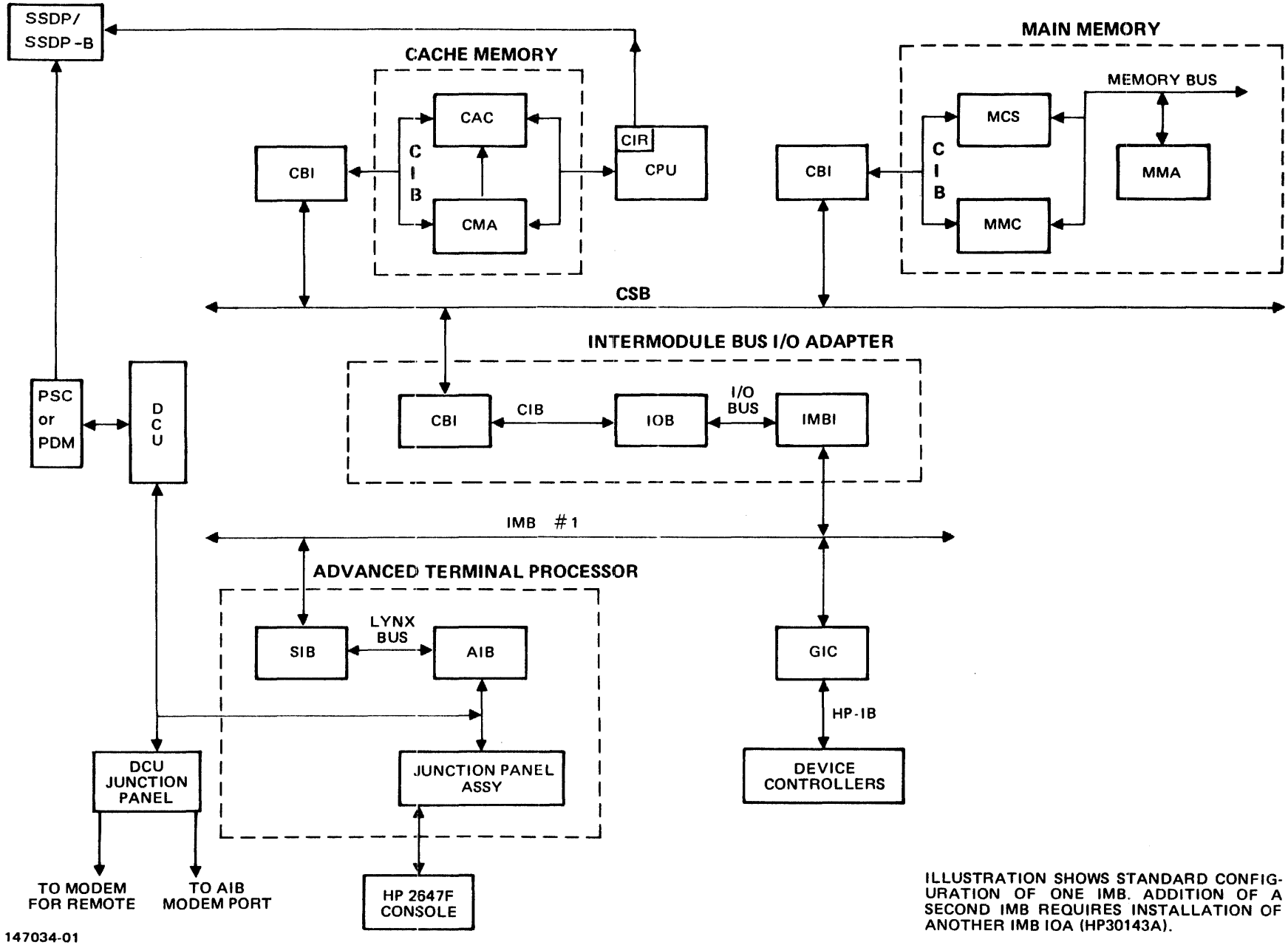


ILLUSTRATION SHOWS STANDARD CONFIGURATION OF ONE IMB. ADDITION OF A SECOND IMB REQUIRES INSTALLATION OF ANOTHER IMB IOA (HP30143A).

Figure 5-1. Major Modules of the Computer

5-1. GENERAL NATURE OF THE I/O SYSTEM

The purpose of a computer is to input, process, and output information. Under the Multiprogramming Executive (MPE) operating system, the information may be created and used by MPE, by compilers, by user programs, or by the users themselves. MPE handles the information in groups called files. A file is a collection of information identified by a name recognized by MPE.

MPE uses media such as disc, diskettes, and tape for storing the information. On any of these media, a file may contain MPE commands, system or user programs, or data, in any combination. Within a file, all information is organized into units of related data called logical records that for most applications are similar in form, purpose and content. The records in the file can be arranged in almost any order: alphabetically, numerically, chronologically, or by subject matter. The logical record is the smallest group of data that MPE can address. Programs, however, can recognize data blocks within each record (a block is four double words). In fact, programs can recognize and manipulate words, bytes, and even individual bits. (A word consists of two bytes; a byte consists of eight bits.) Data is physically transferred between Main Memory (or a Cache Memory) and the peripheral device on which the file resides. On disc and magnetic tape files, a block consists of one or more logical records; for files on other media, a block normally is equivalent to one logical record.

To summarize the interrelation of files, logical records, and blocks: a file is a collection of records treated as a unit and recognized by a name; a logical record is a collection of information treated as a unit, residing in a file; and a block is a group of one or more logical records transmitted to or from a file by an I/O operation. The purpose of the I/O System, then, is to perform physical I/O operations for MPE. The user normally does not interact directly with the I/O system, only indirectly through the file system. Normally, I/O operations are invisible to the user.

5-2. INTERMODULE BUS I/O ADAPTER (IMB IOA)

The IMB IOA interfaces the Central System Bus (CSB) with the Intermodule Bus (IMB), handling all data and message transfers between them. The IMB IOA includes IOB, IMBI, and CBI PCAs.

Functionally, the IMB IOA converts synchronous, block-sized transfers on the CSB to asynchronous, word-sized IMB-compatible transfers, and vice versa. A block consists of eight 16-bit words (sometimes also referred to as four 32-bit "double words.") The IMB IOA also converts IMB Channel Service Requests and Interrupt Requests into messages to the CPU, and interprets commands from the CPU. See Figure 5-2.

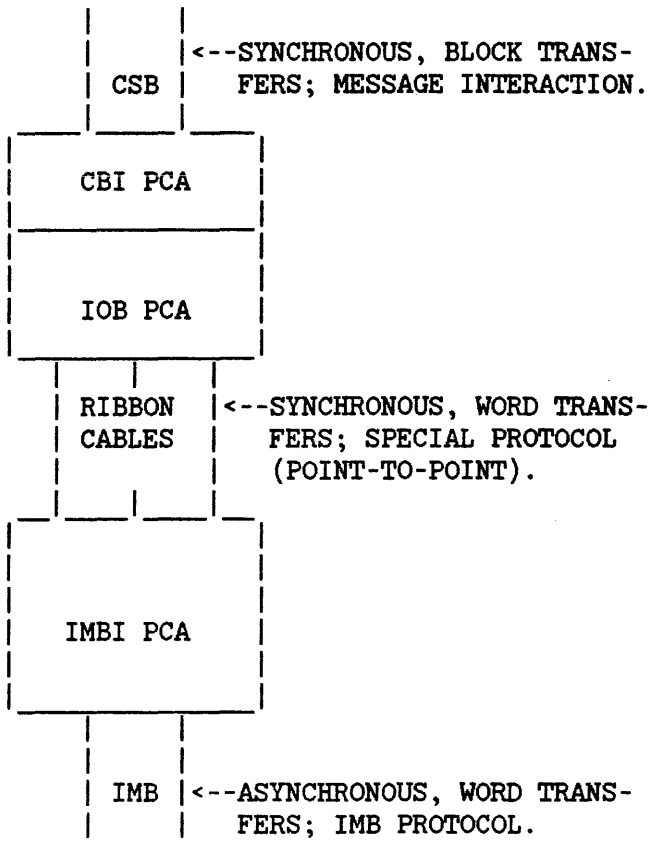


Figure 5-2. CSB/IMB Transfers via the IMB IOA

The IMB IOA has the following capabilities:

a. Support of IMB Memory and I/O commands from CPU to IMB

b. Ability to check addresses on the CSB for the IOB PCA's Cache Memory

c. Ability to (re)initialize to a known state

d. Ability to do an orderly power-down

The following features provide those capabilities:

a. A Cache Memory for fast Main Memory request handling.

b. Messages from the CPU are translated to IMB commands, and IMB commands are translated to messages to the CPU.

c. The Diagnostic Control Unit (DCU) can initialize the IMB IOA during power-up.

d. The CPU can command the IOB PCA to flush its Cache Memory to Main Memory and reinitialize for powering down. The IOA also translates PFW and PON signals to TTL for the IMB.

The following paragraphs describe the functions of the CBI, IOB, and IMBI PCAs.

5-3. CBI PCA Functions

The CBI PCA is the communication link between the CSB and the IOB PCA. It contains two registers (A and B) that buffer data, messages, and addresses from the CSB. It also contains logic to decode CSB commands, and bus accessing logic that does priority checking and prevents CSB access conflicts. Other CBI PCA registers keep track of the sources of check addresses and messages so the IOB PCA knows where to send the requested data and message responses.

The CBI has two ports: one connects to the CSB; the second connects to a cable called the Common Interface Bus (CIB) which leads to the IOB PCA. (The CIB also exists on the CPU backplane.)

5-4. IOB PCA Functions

The IOB PCA has two major functions:

a. It handles IMB memory requests.
b. It transfers messages between the IMBI PCA and the CPU.

To handle the first job, the IOB PCA has its own four-block Cache Memory. (This Cache Memory should NOT be confused with the Cache Memory of the CPU and Cache Memory Module.) When the IMBI PCA sends a memory request on behalf of the IMB, the IOB PCA first checks its Cache Memory for a copy. If it doesn't find one, it requests the data from Main Memory. While that is happening, all other Cache Memories on the CSB are checking to see if they can supply the data block. If one of them can because its copy is dirty/valid, the request to Main Memory is aborted, and the Cache Memory supplies the data to the IOB.

If the IOB PCA does have a copy of the requested data in its own Cache Memory, it supplies it to the IMBI PCA. Since the IMBI PCA requests one word at a time, and a block of data contains eight words, the IOB PCA's Cache Memory will have the requested word seven-eighths of the time (assuming sequential accesses, as most IMB operations are).

To handle messages, the IOB PCA alerts the IMBI PCA that a message is coming from the CPU and passes it (after translating it from ECL to TTL levels) when the IMBI PCA is ready for it. Messages in this direction are interpreted as IMBI commands. These messages transfer across the CSB when a Send Word Operation (SWOP) is received. The IMBI always returns a message response back to the CPU reporting completion status of the command. For transfers in the opposite direction, the IOB PCA picks up the message from the IMBI PCA and relays it to the CPU. These unsolicited messages occur as a result of an IMB interrupt. No response is required for unsolicited messages.

5-5. IMBI PCA Functions

The IMBI PCA controls the movement of information between the IMB and the CSB. The IMBI decides, on a priority basis, whether to service memory requests on the IMB, execute commands for I/O channels on the IMB received on the CBI from the CPU, or to send unsolicited messages to the CPU to report assertion of Interrupt Request (IRQ) or Channel Service Request (CSRQ2). The IMBI initiates these operations by signaling the IOB PCA. In addition, the IMBI handshakes addresses and data on the IMB, acting as an IMB "memory subsystem" slave. When the IMBI is processing CPU commands (messages) it is the IMB master.

The IMB is an asynchronous bus, meaning there are no synchronized clocks between PCAs or on the IMB itself. In contrast, the PCAs connected to the CSB have 75-ns clocks. The IMBI thus synchronizes the I/O system to the 75-ns computer clock, so that IMB events cannot cause synchronization faults.

If a power failure is imminent, the IMBI PCA stops servicing IMB memory requests. In addition, all Cache Memories on the CSB are "flushed" (their dirty/valid contents are sent to Main Memory) prior to the loss of the Power On (PON) signal. Special circuitry on the IMBI ensures that all attachments to the IMBI are reset until DC power reaches operating levels. The circuitry also preserves IMB operations if IMBI cables are removed, or if power drops in the IOB PCA.

5-6. IMB/CSB SIGNALS

Signals that interface the IMB with the CSB may be described in three groups:

a. IOB to/from CBI
b. IOB to/from IMBI
c. IMBI to/from channel controllers

The IOB/CBI signals define the Common Interface Bus (CIB), which physically consists of a cable between the two PCAs. (The CIB also exists on the CPU backplane.)

The IOB/IMBI signals define the I/O Bus (IOB), which physically comprises two frontplane ribbon cables between the two PCAs.

The IMBI/channel controller signals define the IMB, which physically comprises most of the I/O Bay card cage backplane.

5-7. I/O SYSTEM ARCHITECTURE

5-8. IOB PCA Block Level Description

The following paragraphs describe the major components of the IOB PCA. See Figure 5-3.

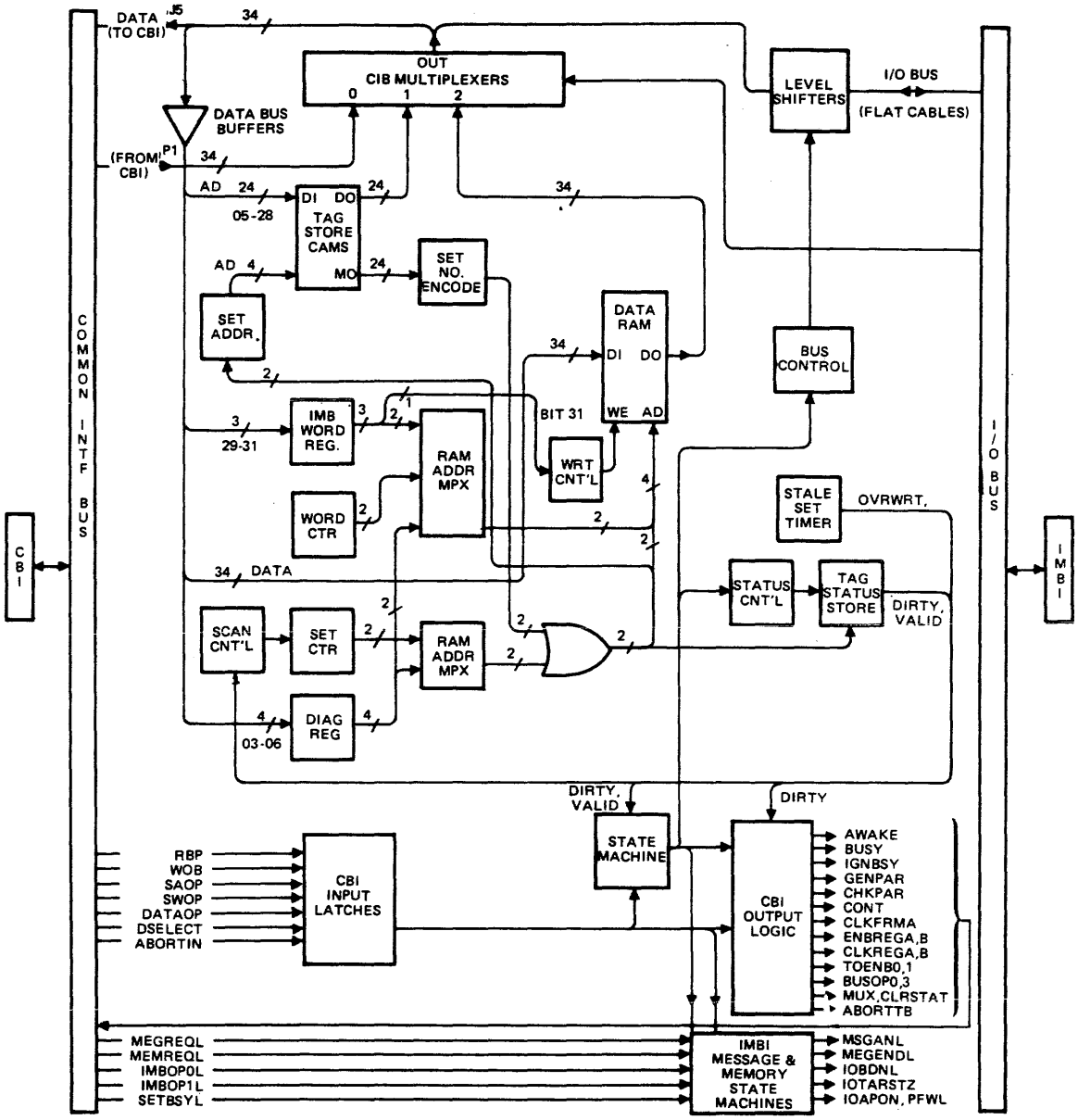


Figure 5-3. IOB PCA, Simplified Block Diagram

5-9. CIB MULTIPLEXERS. The CIB Multiplexers gate data onto the main Address/Data Bus. "CIB" is renamed "DATA" as it leaves the PCA to conform to the naming standards of the CBI PCA. The CIB is used for communications between the IOB, CBI, and IMBI PCAs. The inputs to the Multiplexers are:

a. Addresses/Data from the CBI
b. TAG Data
c. RAM Data

As the CIB feeds both the CBI PCA and the IMBI PCA, the possible data paths are:

a. From CBI PCA to IMBI PCA (usually for messages)
b. From TAG STORE to CBI PCA (to send memory addresses)
c. From IMBI PCA to CBI PCA (to send messages & addresses)
d. From DATA RAM to CBI PCA (for memory writes)
e. From DATA RAM to IMBI PCA (for IMBI memory reads)

Data to and from the IMBI is converted by ECL/TTL bidirectional level shifters on the IOB.

5-10. DATA BUFFERS. The output of the Data Buffers is called Buffered CIB (BCIB). The BCIB comprises a unidirectional bus that supplies addresses and data to the rest of the IOB.

5-11. TAG STORE CAMS. The Tag Store Content Addressable Memories (CAMs) work in the opposite manner as do Random Access Memories (RAMs). When data is input to a CAM, it outputs the address where that data is located. On the IOB PCA, the CAMs serve a second purpose of storage and comparison of Tags (portions of addresses that describe where a block in the Cache Memory belongs in Main Memory) with incoming addresses. The CAMs automatically compare incoming address with their contents, and output the block number (0 to 3) when matches are found. (The IOB can hold 4 blocks.)

The CAMS output 24 Match lines. These lines are encoded into two-bit set addresses by the Set Number Encoder Logic. These two bits identify which of the four data blocks are being addressed.

5-12. DATA RAMS. The Data RAMs represent the "buffer" of the I/O Buffer (IOB) PCA. They consist of ten 16X4 Register Files that store 16x34 bits (six bits are unused). This is the storage needed for four 4x34-bit blocks (32 data bits plus two parity bits equals 34.)

The Bus Control Logic controls the data flow to and from the Data Bus.

5-13. IMB WORD REGISTER. This register stores the lower three bits of addresses from the IMBI PCA. During an IMBI memory access, the Word Register bits point to the indicated data word in the block.

5-14. WORD COUNTER. The Word Counter is a two-bit counter that points to the correct word in the block during data transfers between the IOB PCA and Main Memory.

5-15. SET COUNTER/SCAN CONTROL. When the IOB is in the Idle state, its Cache Memory is scanned to see if there are any empty or overwritable blocks for use on the next IMBI memory request.

The Set Counter is a two-bit rollover counter which does the scanning. The Scan Control Logic starts and stops the Set Counter.

5-16. DIAGNOSTIC REGISTER. This is a four-bit register containing bits 03-06 of the Buffered CIB. It is used only during diagnostics to access a location in the Data RAM, the Tag CAM, or in the Tag Status Store.

5-17. RAM ADDRESS MULTIPLEXERS. These select the proper address lines to the RAMs from either the Word Counter or the IMB Word Register for the lower two bits; and from the Set Counter or the CAM set encoder for the higher two bits. The contents of the Diagnostic Register are also multiplexed into the RAM Address Multiplexer.

5-18. STALE SET TIMER. This timer keeps track of how long a data block has been unused in the IOB PCA's Cache Memory. After 2-4 microseconds, it marks an unused block available for overwriting, on the assumption the data in it will not be needed soon.

5-19. CACHE BLOCK STATUS. This consists of a 256X4 Read Only Memory (ROM) with inputs from the state machine, the previous Tag Status, and the new Tag Status. The Tag Status Store is a 16x4 bit RAM. The Tag Status bits and their meanings are:

a. OVERWRT - block can be overwritten
b. DIRTY(L) - block has been modified
c. VALID(L) - block contains valid data

5-20. CBI INPUT LATCHES/OUTPUT LOGIC. These two circuitries contain the logic the IOB PCA uses to interface with the CBI PCA.

5-21. STATE MACHINE. The State Machine is the IOB PCA's controller. There are five active states, plus Idle. The active states are:

a. SNDADR (Send Address)
b. SNDATA (Send Data)
c. RCVDATA (Receive Data)
d. CHECK (Check Cache Memory for match with CSB address)
e. BEMEM (Be Memory to the requesting device)

5-22. MESSAGE/MEMORY STATE MACHINE. IOB interfacing with the IMBI is handled by the Message/Memory State Machine. Its inputs include signals from the IMBI, the CBI Input Latches, and the State Machine. Its outputs are the interface signals to the IMBI.

5-23. IOB PHYSICAL DESCRIPTION. Figure 5-4 shows the physical outline of the IOB PCA. It includes labels for the connector pins and pin assignments, and locators for the major test points.

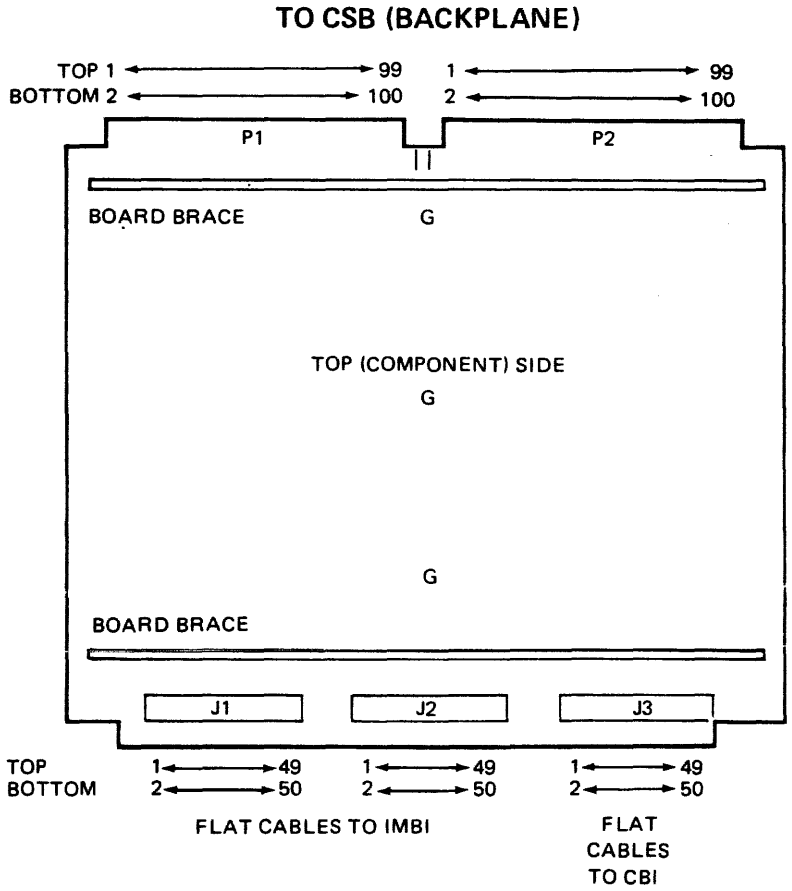


Figure 5-4. IOB PCA Physical Outline

5-24. IMBI PCA Block Level Description

The following paragraphs describe the IMBI PCA. See Figure 5-5.

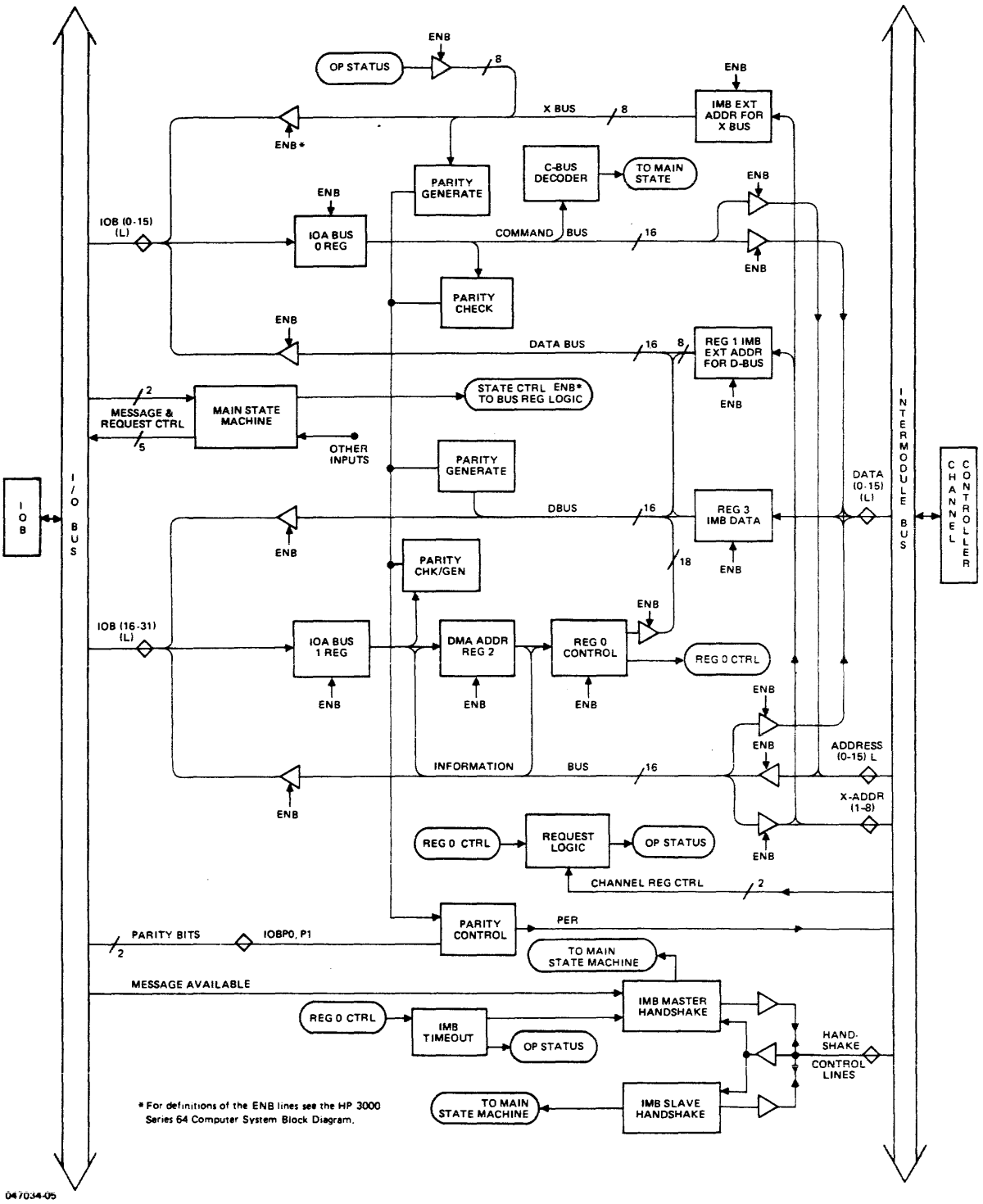


Figure 5-5. IMBI PCA, Simplified Block Diagram

5-25. X-BUS. The X-Bus is an eight-bit, tri-state data bus over which the IMB extended address bits are transferred to the IOB PCA during memory operations. During register accesses, operation status is sent over the X-BUS. Since the X-BUS is used only for sending to the IOB, it has a parity generator, but no parity checker. The X-BUS drives only IOBUS bits 08-15; bits 00-07 default to 0.

5-26. D-BUS. The D-Bus is a 16-bit, tri-state data bus over which the IMB data bits are transferred to the IOB PCA during memory operations or operations with Register 3. Registers 0 and 1 are enabled onto the D-BUS when accessed during register operations. The D-BUS has a parity generator, but no checker, since the D-BUS is only for data going to the IOB PCA. The D-BUS drives IOBUS(16-31) during register operations and both words of the IOBUS during memory writes.

5-27. I-BUS. The I-Bus is a 16-bit, tri-state data bus over which data for IMB commands and IMBI register loads is transferred from the IOBUS 1 Buffer Register. The I-BUS goes directly to Registers 0 and 2. Register 2 is gated onto the I-BUS during Register 2 operations or when sending an IMB memory address to the IOB PCA during memory operations. In the idle state, the IMB address bits are enabled onto the I-BUS to load Register 2 for memory accesses. During register operations, IMB write commands, or IMB memory read operations, the I-BUS is gated onto the IMB data lines. For a read of Register 2, Register 2 is gated onto the I-BUS and the I-BUS is gated onto IOBUS(16-31). The I-BUS has a parity generator and parity checker circuit, since data flows both directions between the IOBUS and IMB.

5-28. C-BUS. This is a 16-bit data bus driven by the IOBUS 0 Buffer Register. It contains the command word when serving the CPU, and data during IMB memory read operations. It is gated onto the IMB address lines during commands, and onto the IMB data lines during IMB memory reads. It has a parity checker. Logic on this bus decodes the CPU commands.

5-29. REGISTER 0, TEST AND CONTROL REGISTER. This register is used to initialize the IMBI PCA, enable test features, and control the interrupt logic. It is loaded from the I-Bus but read from the D-Bus.

5-30. REGISTER 1, EXTENDED MEMORY ADDRESS REGISTER. This register contains the upper eight bits of the IMB 24-bit memory address. It is loaded from the IMB extended address bits. There are two identical "copies" of Register 1. One copy drives the X-BUS for memory operations. When IMBI Register 1 is read by command, the copy on the D-BUS is enabled.

5-31. REGISTER 2, MEMORY ADDRESS REGISTER. This is a 16-bit, tri-state register used to store the lower 16 bits of the IMB memory address. It is loaded from and enabled onto the I-BUS, since parity for the IMB address must be generated at the same time it is stored to Register 2, to be ready for the IOB PCA when initiating a memory operation.

5-32. REGISTER 3, IMB DATA REGISTER. This is a 16-bit, tri-state register loaded from the IMB data lines with information for the IOB PCA. During IMB memory writes, it is loaded from the IMB with the data to be written to memory. During IMB write commands or writes to IMB Register 3, it is loaded with data the IMBI PCA drives onto the IMB data lines. During IMB read commands, Register 3 is loaded with data supplied by the I/O channels.

5-33. PARITY CONTROL LOGIC. This logic controls parity generation, checking, and reporting on the IMBI PCA. It stores the IOBUS parity bits when the IOBUS data registers are loaded. It checks parity and sets status bits for parity errors on commands. It checks parity on memory read data and asserts a parity error signal on the IMB. It drives the IOBP0(L) and IOBP1(L) parity bits when the IMBI PCA sends information to the IOB PCA.

5-34. MAIN STATE MACHINE. This is a 12-state machine which controls IMBI operating sequences. It monitors signals from the IOB PCA, the IMB, and other logic, to determine which sequence of states to perform. The machine has four sequences: unsolicited message; IMBI command; IMB command; and IMB memory operation. It also generates control signals to communicate with the IOB PCA and other IMBI PCA logic blocks, and generates signals to control IMBI registers and data paths.

5-35. IMB TIMEOUT LOGIC. This logic generates a timeout signal and status bit if the IMB Master Handshake Logic gets no response after conducting an IMB handshake for more than 1023 clock periods. This signal is used to terminate the handshake.

5-36. IMBI RESET LOGIC. This logic drives the reset lines for all hardware on the PCA. It asserts reset signals when either IOARST(L) or PON goes low. It is also responsible for generating a three-clock reset pulse after the IMBI PCA goes idle following a write to Register 0 with bit 22 set (clear IMB).

5-37. POWER ON CIRCUIT. This circuit ensures that the IMB PON signal remains low for any of the following conditions:

a. IMBI PCA power supplies are off.

b. The IMBPON signal from the IOB PCA is false.

c. The IMBI/IOB interface cables are disconnected.

d. IOB PCA power is off.

5-38. INTERRUPT LOGIC. This logic signals the Main State Machine when an unsolicited message is sent to the CPU. It provides for setting and clearing the request masks for the IMB IRQ and CSRQ2 signals, and for synchronizing these signals to the clock. It ensures that the request masks are cleared when an unsolicited message is transmitted to the CPU.

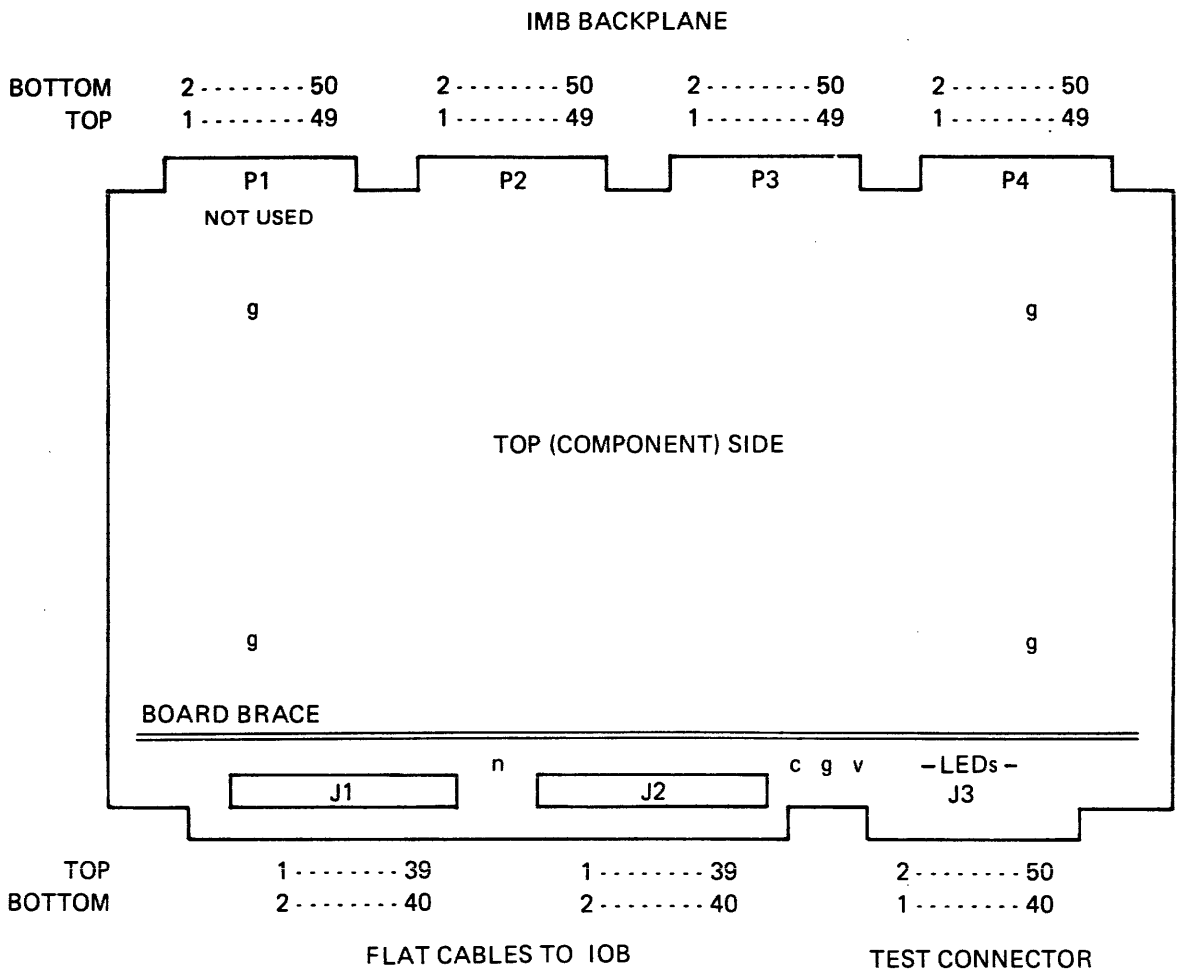
5-39. IMB MASTER HANDSHAKE LOGIC. This logic is responsible for obtaining the IMB when there is a Send Word command from the CPU for the IMBI PCA, and then for conducting the handshaking of that command on the IMB under control of the Main State Machine.

5-40. IMB SLAVE HANDSHAKE LOGIC. This logic is a set of two state machines which control the IMB data and address handshakes when the IMBI PCA is servicing an IMB memory request. Completion of the handshake is synchronized and reported to the Main State Machine.

5-41. DATA NOT VALID LOGIC. This logic recognizes and reports assertion of the IMB DNV signal during a master handshake by the IMBI PCA.

5-42. I/O BUS REGISTERS. These two 16-bit registers take data from the IOB PCA during memory reads or Send Word message transmissions. IOB00(L)-IOB15(L) are loaded into the Bus 0 Register which is enabled onto the C-Bus; IOB16(L)-IOB31(L) are loaded into the Bus 1 Register which is enabled onto the I-BUS.

5-43. IMBI PCA PHYSICAL DESCRIPTION. Figure 5-6 shows the physical outline of the IMBI PCA. It includes labels for connector pins and pin assignments, and locators for the major test points. The status of major registers is indicated by the 15 LEDs located between connector J3 and the PCA brace. Table 5-1 illustrates the arrangement, labeling and function of each LED.



Test Points

c = Clock: High during first half of state time (rising edge triggers state changes). 50% duty cycle, Schottky TTL signal.

g = Common: binding posts connected to board logic common.

n = -5.2 volt test point between J1, J2.

v = +5.0 volt test point by J3, primarily for probe power.

Figure 5-6. IMBI PCA LEDs and Test Points

Table 5-1. IMBI PCA LED Definitions

- - - - - - - - J3 LED Arrangement - - - - - - - -

I D	X 1	R 1	R 2	M 1	M 2	L	W	S 1	S 2	S 3	I N	C M	I M	B R

LABEL	Signal	On if and only if the IMBI is . . .
ID	IFTL	In the IDLE state.
XI	XFTL	trying to send unsolicited message (X1,X2 state).
R1	R1FTL	requesting message from IOB (R1 state).
R2	R2FTL	checking parity, content of message (R2 state).
M1	M1FTL	executing an IMB command (M1 or M2 state).
M2	M2FTL	asserting IMB command handshake lines (M2 state).
L	LFTL	performing an IMBI register operation (L state).
W	WFTL	sending response message to CPU (W state).
S1	S1FTL	sending memory address to IOB (S1 state).
S2	S2FTL	waiting for completion of data portion of memory operation (S2FTL) with IOB.
S3	S3FTL	completing IMB memory handshake (S3 state).
IN	INTL	going to enter X state soon, as there is a valid reason to send an unsolicited message to the CPU.
CM	CSRQMFL	enabled to recognize and report assertion of IMB CSRQ2L signal (channel program request mask).
IM	IRQMFL	enabled to recognize and report assertion of IMB IRQL signal (interrupt request mask).
BR	MYBRQFL	asserting the IMB BRQL signal to gain control of the IMB to send a command (only used if a CPP is installed).

5-44. IMB IOA SYSTEM-LEVEL CHARACTERISTICS

The following paragraphs describe the operation of the IMB IOA at the computer system level, and describe the affects of CSB operation command messages to the IMB IOA. In general, the CPU controls the IMB IOA by sending two types of message commands across the CSB: Send Word Operands (SWOPs), and Send Address Operands (SAOPs). SWOPs are interpreted as commands to the IMBI PCA; SAOPs are interpreted as diagnostic commands to the IOB PCA.

5-45. Execution of Send Word Operation Commands

When the CPU issues a SWOP command to the IMB IOA, it is received by the CBI PCA. The CBI notifies the IOB PCA by passing on the SWOP. The IOB PCA then notifies the IMBI PCA that a message is available by asserting MSGAV(L). The word "message," as used here, is defined as one of several Send Word commands which have been latched in the CBI PCA. These messages take the form of formatted IMBI commands. The commands set up the channel to take the appropriate action. When the IMBI PCA is ready to receive the message, it initiates a handshake with the IOB PCA to get the message word from the CBI PCA. The IMBI PCA checks parity on the message, then executes it. Next, the IMBI generates a response message and sends it back through the IOB and CBI to the CPU. The IOB and IMBI then return to their idle states.

The IMB IOA is set BUSY on the CSB from the time the CBI PCA receives the SWOP message until the response message goes over the CSB. This prevents any other messages from coming in and overrunning the message buffer on the CBI PCA.

The following are the Send Word messages for the IMB IOA: IMBI Register Read, IMBI Register Write, IMB I/O Global Read commands, IMB I/O Global Write commands, IMB I/O Addressed Read commands, IMB I/O Addressed Write commands.

The following are the Send Word responses for the IMB IOA: IMBI Register Operation Response, IMB Command Response, Unsolicited Message.

5-46. IMB Memory Requests

The IMBI PCA works as "memory" on the IMB, performing the protocol side of the request (the IMB Slave handshake). The only operations performed are reads and writes.

The steps in servicing a memory write are:

a. Send the memory address and opcode to the IOB PCA.

b. Complete the IMB handshake while sending the data word to the IOB PCA.

The steps in servicing a memory read are:

a. Send the memory address and opcode to the IOB PCA.

b. Wait for the IOB to provide the data, while completing the IMB address handshake.

c. Assert the data word onto the IMB and complete the data handshake. If there is a parity error detected on data from the IOB, the IMBI asserts a Parity Error signal on the IMB before completing the handshake.

5-47. Servicing CPU Commands

Commands from the CPU receive highest-priority treatment from the IMBI PCA. A 32-bit message from the CPU contains 15 bits of address, 16 data bits and one read/write bit. The IMBI PCA translates the CPU message into address, opcode, and data for an IMB I/O command. CPU commands addressed to Channel 0, Device O are register operations for the IMBI itself. Once the IMBI sends an IMB I/O command, it acts as the IMB master.

The steps in servicing CPU commands are:

a. When the CBI PCA receives the message, the IOB latches the message and relays signal to the IMBI.

b. The IMBI takes control of the IMB so it can begin a master handshake.

c. The IMBI receives the message from the CBI PCA, via the IOB PCA.

d. The IMBI checks parity on the message. If there is a parity error, the command will not be executed; the IMBI will jump to step g.

e. The IMBI asserts the command on the IMB and the handshake sequence is begun.

f. One of the following will terminate execution of the command:

   (1). A global command handshake is terminated by the IMBI itself after 1 microsecond.

   (2). An addressed IMB I/O command is completed when the channel responds.

   (3). If the handshake is not complete within 1,024 clock cycles, the IMBI terminates it with a timeout error.

g. The IMBI sends a message back to the CPU indicating error or special status conditions relating to execution of the command. For read commands, the data from the channel is included.

5-48. UNSOLICITED MESSAGES

The IMBI PCA alerts the CPU (via the IOB and CBI PCAs) to incoming Interrupt Requests and Channel Service Requests by sending it an unsolicited message. The ability to send unsolicited messages is controlled by message mask bits in register 0 of the IMBI PCA. There is a separate mask bit for each of the IMB's IRQ and CSRQ2 signals. They are enabled and disabled independently. When set, these mask bits cause the IMBI to initiate an unsolicited message transmission to the CPU whenever IRQ or CSRQ2 is asserted on the IMB. When an unsolicited message is transmitted to the CPU, both message masks are turned off by the IMBI, disabling other such messages until the bits are reenabled by writes to register 0. If the message is not sent (i.e., if the message is held back by the IMB due to other higher-priority operations), the mask bits are not cleared, and the message will be retransmitted after the other operations have been performed. This mechanism is the IMBI's lowest-priority option. In general, if the signal's mask is enabled and the signal is asserted, the IMBI will attempt to send a message to the CPU, reporting the IMB request. Unsolicited messages do not require a response from the CPU, but prompt the CPU to investigate the cause of the interrupt.

5-49. EXECUTION OF SEND ADDRESS COMMANDS

The CPU uses the Send Address Bus operation to query the internal logic on the IOB PCA, for diagnostic purposes only. Thus the IOB always intercepts the corresponding Send Address Operation (SAOP) command and generates its own response rather than passing them to the IMBI PCA.

The IOB PCA sends the CPU a response to each SAOP, even though the SAOP may not contain valid data. This is because the IMB IOA must always respond to an incoming message.

5-50. INITIALIZATION AND RESET

During power-up and power-fail (and sometimes during normal operation) the IMB IOA must be reset to a known valid state. There are several ways to do this:

a. During power-up, the DCU automatically manipulates the shift strings on the IOB to set the Initialize bit. The IOB and IMBI PCAs initialize after a minimum of four system clock cycles.

b. The same operation can be done by the console operator while in the DCU Maintenance Mode, either by entering the Reset command or by modifying the string on the IOB PCA and sending it four clocks. The Reset command resets the whole computer; modifying the IOB PCA shift string resets only the IMB IOA. (Refer to Section VI for a description of the DCU Maintenance Mode commands.)

c. During normal operation or during a power down/power fail, the CPU can force the IOB PCA to reinitialize itself by sending it a SAOP command. That makes the IOB "flush" its Cache Memory (send the contents to Main Memory).

d. The IMBI and all channels on the IMB can be reset by issuing a Send Word command to the IMBI register #0 with bit 22 set. This method does not affect the IOB.

The IMB IOA is considered initialized when the following is true:

a. All state machines on the IOB and IMBI PCAs are in their idle states.

b. All data blocks in the IOB's Cache Memory are marked Invalid.

c. The IMB is inactive, i.e., there are no pending Interrupt Requests, no pending Channel Service Requests, no bus requests, and there is no IOB traffic.

5-51. IMB MEMORY ACCESSES

As previously described, the IOB acts as "memory" to the IMBI; i.e., all Main Memory transfers to the IMBI pass through the IOB's Cache Memory. Generally, the IOB's Cache contain the last four blocks of data written to or read from Main Memory during a Direct Memory Access (DMA) transfer. If during an IMBI memory read operation, the IOB's Cache Memory has a copy of the required data, it will supply it to the IMBI. If the IOB Cache does not have a copy of the data, it will get it either from Main Memory or the CPU's Cache Memory.

5-52. Hit Access

A "hit access" is an instance when the IOB PCA Cache Memory contains a valid copy of the data requested. The IOB PCA supplies the requested data block to the IMBI PCA, or allows the IMBI to write into the block, depending on the operation. This operation usually takes place in two cycles, and accesses of this type account for approximately 87 percent of memory accesses. (The "hit ratio" is approximately 87 percent.)

The opposite of a hit access is a "Nohit" access, meaning the IOB Cache Memory doesn't have a copy of the requested data. There are two kinds of Nohit access: accesses to the First Word in the Block (FWIB), and accesses to any other word in the block (Not FWIB).

5-53. Nohit Access - Not First Word In Block

When the IMBI PCA sends a memory request for a block not in the IOB PCA Cache Memory, the IOB delays the request while it goes to Main Memory for the data. After the data is brought in, the request cycle is restarted, and the second access results in a "hit."

5-54. Nohit Access - First Word In Block

When the IOB receives a Nohit Access from the IMBI, resulting in a data block transfer to the IOB PCA's Cache Memory, a check is made to see if the First Word in Block (FWIB) is the one requested. If that is the case, the IOB allows the IMBI access to the word while simultaneously bringing the block from Main Memory into the IOB's Cache Memory. This cuts about five clock cycles from the waiting time for a memory request, depending on the size of the data.

A FWIB is more common than a Not FWIB. This is because most I/O transfers are Direct Memory Accesses (DMAs) of large segments of data. When these transfers cross data block boundaries, the access is a Nohit FWIB access to the IOB PCA. For a typical 1-kbyte transfer, 63/64ths of the Nohit accesses will be FWIB.

5-55. IMB IOA ACCESS PRIORITY ON THE CSB

When the I/O System wants to talk with another CSB module, its IMB IOA must first gain access to the CSB itself.

Interrupt requests to the CPU are initiated by the IMB IOA by I/O commands (messages) from the CPU.

Data transfer priorities are based on a CSB "round robin" approach, to share CSB time. Refer to Section II of the manual.

5-56. INTERMODULE BUS I/O CHANNELS

The preceding paragraphs of this section dealt with the IMB IOA portion of the computer's I/O System. The remaining paragraphs describe the second major portion, the IMB I/O channels. By the time a message or data block from another CSB module reaches an IMB I/O channel, it has propagated through the IMB IOA, as shown in Figure 5-1.

5-57. IMB FUNCTIONAL DESCRIPTION

The IMB carries the communication between the IMBI PCA and the individual device controllers. The IMBI acts as the bus arbiter; it has continuous access to the bus and relinquishes it only on request. To access memory, the I/O channels must request the bus through a priority structure described in Paragraph 5-74.

The IMB has separate address and data paths which complete data transfers using an IMB Master/Slave handshake protocol. When a channel is initiating communication, it is a Master; when receiving it is a Slave. The IMBI and IOB PCAs appear as "memory" to the channel initiating a memory transfer; thus the IMBI appears as a Slave to that channel. Conversely, when the IMBI initiates I/O commands to the IMB channels, the IMBI is Master and the channels are Slaves.

The IMB contains 66 signal lines and 12 priority lines. The signals can be grouped in three categories:

a. Information. This group includes the address bus lines, data bus lines, and bus operation codes.

b. Bus Control. These signals control access to the bus.

c. System Status. These signals include Interrupt Request, Channel Service Request, Data Not Valid, Power Fail Warning, Power On, Parity Error, and System Reset.

5-58. IMB CONFIGURATIONS

The I/O System supports two types of I/O channels: General I/O Channels (GICs) and Advanced Terminal Processors (ATPs). Supported device interfaces include those for the 2619 terminal family, the Intelligent Network Processor (INP), and the ATP's Asynchronous Interface Board (AIB) PCA.

The computer's standard configuration supports one IMB, which, electrically, accommodates up to 15 I/O channels. There may be practical limits to this number, however; refer to the HP 3000 Computer System Configuration Guide, Part Number 5953-0664.

As shown in Figure 5-7, a second IMB can be added to the computer. If added, IMB No. 2 requires a second IOB PCA, a second IMBI PCA, and (not show in Figure 5-7) a second CBI PCA. (These three PCAs are known collectively as the IMB IOA, as previously described.)

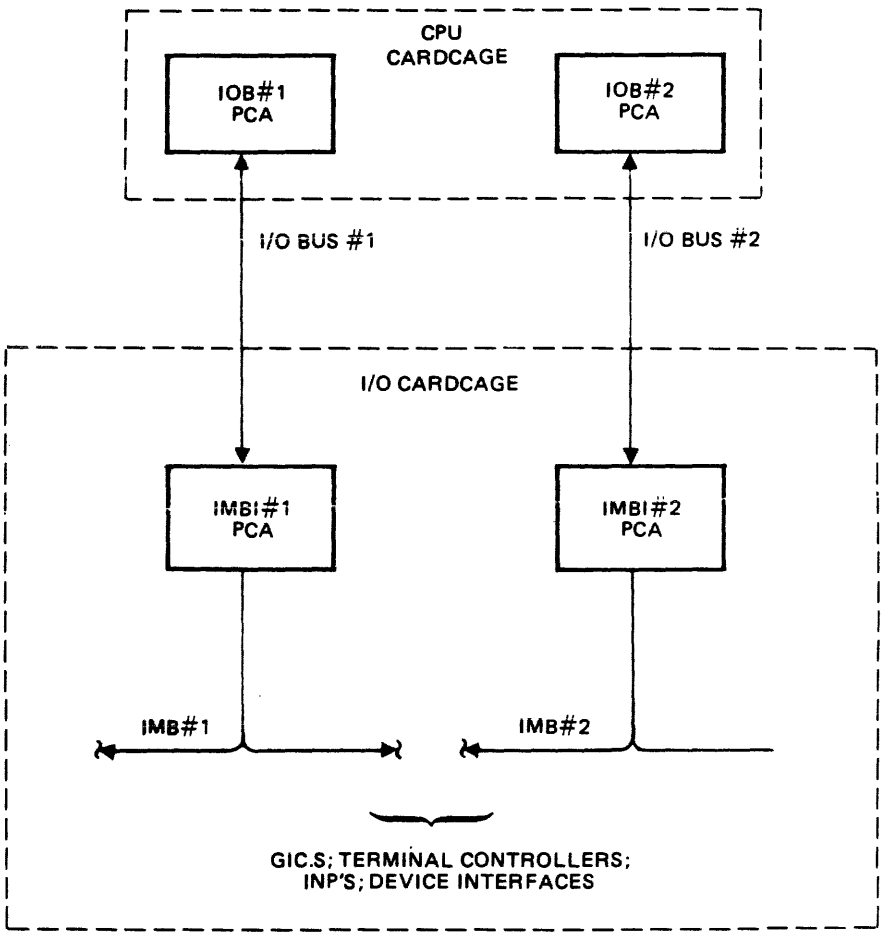


Figure 5-7. Basic I/O Structure

Figure 5-8 shows the assignments for the slots in the I/O Bay card cage. The "PROC" and "BIC" labels in the top row, above slots 1, 2, 9, and 10, are intended for a future product enhancement. As that enhancement is not yet available, those slots can be used by any device interface PCAs, with one IMPORTANT RESTRICTION: if the computer has two INPs, EITHER slot 1 OR slot 2 can be used by one of the INP interface PCAs, but NOT slot 1 AND slot 2. Likewise, EITHER slot 9 OR slot 10 can be used by an INP interface PCA, but NOT BOTH.

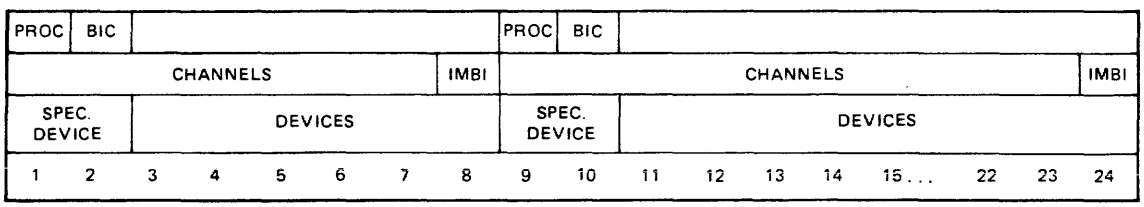


Figure 5-8. I/O Bay Card Cage Slot Assignments

Figures 5-9 and 5-10 show recommended slot assignments for a one-IMB and two-IMB system.

SLOT	PCA	CHANNEL #	"TO" DEVICE
24	IMBI		
23	GIC	2	MAG TAPE
22	GIC	3	SYSTEM DISC
21-13	GIC or DEV. INTF.	4-15	OTHER DISCS; INPs; MAG TAPES; PRINTERS
12	SIB	1	AIB
11-4	AIB		ASYNCH. TERMINALS

Figure 5-9. Recommended Slot Assignments, One-IMB System

IMB #1

SLOT	PCAs	CHANNEL #	"TO" DEVICE
24	IMBI #1		
23	GIC	2	MAG TAPE
22	GIC	3	SYSTEM DISC
21-18	GIC	4-15	OTHER DISCS; TAPES;
17	SIB	1	AIB
16-9	AIB		ASYNCH. TERMINALS

IMB #2

SLOT	PCAs	CHANNEL #	"TO" DEVICE
8	IMBI #2		
7-1	GIC; DEV. INTSs.	1-15	DISCS; TAPES; INPs; PRINTERS, ETC.

Figure 5-10. Recommended Slot Assignments, Two-IMB System

5-59. FUNDAMENTALS OF I/O CHANNELS

Once information intended for an I/O device propagates through the IMB, it reaches an I/O channel. Typical I/O channel hardware consists of a Device Controller and a General I/O Channel (GIC) PCA. See Figures 5-11 and 5-12 for a simplified block diagram and physical outline of the GIC PCA.

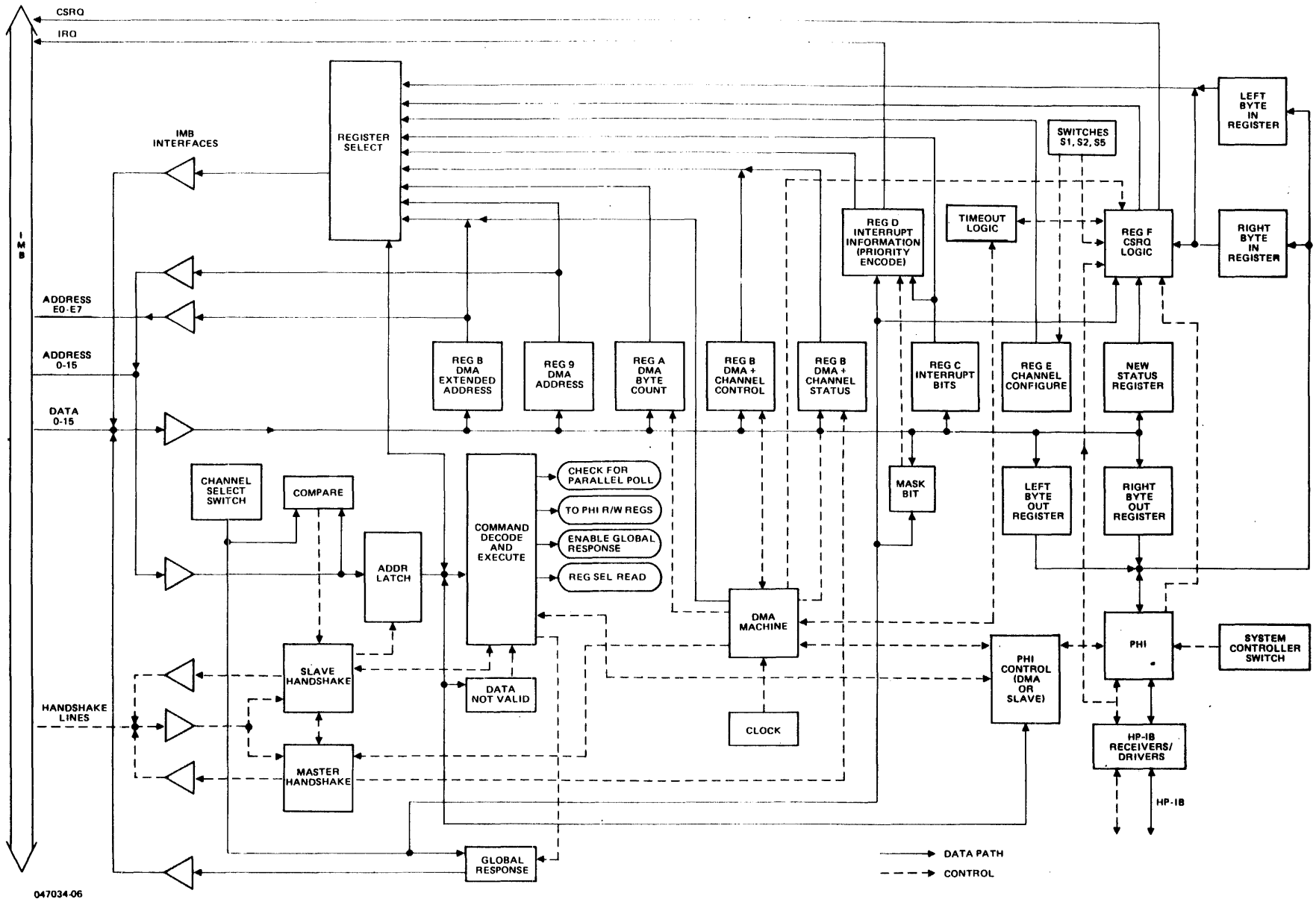


Figure 5-11. GIC PCA Simplified Block Diagram

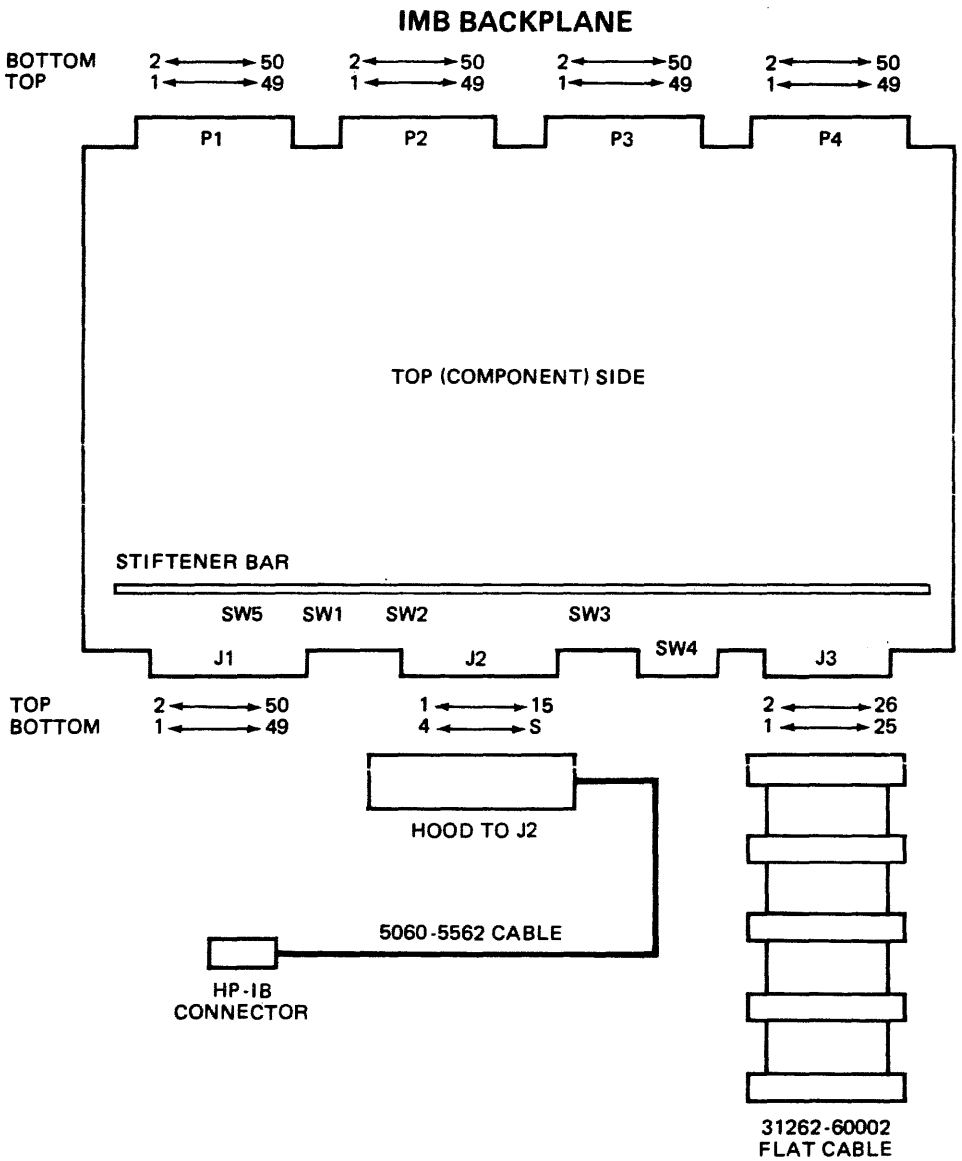


Figure 5-12. GIC PCA Outline

A typical device controller consists of one or more PCAs. It (they) can be housed in the CPU or I/O Bay(s) of the computer, or within the cabinet of the device itself. The device controller can drive only one peripheral or may drive several slave units of the same peripheral.

Each device controller has a four-word entry (numbered 0-3) in the Device Reference Table (DRT). The second word (No. 1) contains a pointer to a data area associated with that entry. The data area consists of an Interrupt Linkage Table (ILT); one or more Device Information Tables (DITs), depending on how many units the device controller is driving); and an I/O program area. Along with various other information, the Driver Linkage Table (DLT) contains Code Segment Table (CST) and Segment Transfer Table (STT) values for defining the location of the driver routines associated with that particular device controller. The DIT contains information relevant to one physical I/O device and is configured differently for each type of device. In each case, however, the third word of this table points to an entry in the I/O Queue (IOQ) when a request is being made.

A device's DRT number contains nine bits. The format is MM CCCC DDD, where MM=IMB Module No.; CCCC=Channel No.; and DDD=Device No.

The references for the IMB IOAs are:

Logical	Physical
0	1
1	2

The following is an example of calculating the DRT number for a device. Determine the DRT number for the System Disc Drive, located on IMB No. 1, Channel 2, Device 0.

```
Formula:(128 X Logical Module #) + (8 X Channel #) + Device #

                0          +          16      +    0  =  16
```

The DRT can reside in any bank and begin at any location as long as there is room for the complete table in one bank. The base location of the table will be contained in absolute locations 8 and 9. These locations contain the bank number and DRT base address, respectively. The absolute location of a DRT entry is calculated by adding the module channel device number to the base offset number (bank number and base address).

The IOQ is a single table containing a fixed number of entries having a fixed number of words per entry. If there are no I/O requests pending in the system, none of the DIT entries will be pointing to the IOQ. In this case, all entries of the IOQ are unused and the second word of each entry points to the first word of the next entry. Thus all unused entries are linked together. If that file system makes a request to use Unit 1 of the Device Controller, the I/O system will unlink the first free entry in the IOQ and fill it with information pertaining to the request (including buffer address and logical device number). Assume that the next request is for Unit 2, followed by a second request for Unit 1. This second request for Unit 1 causes the first word of the initial request to point to the next unused entry, which is then filled with information pertaining to the second request. Eventually the IOQ will contain a queue of requests for Unit 1, a separate queue for Unit 2, and so on, plus a linked list of free entries.

Next, an I/O driver is executed to initiate the request. An I/O program will then be run on a device, using the request parameters given in the IOQ. When the request is fulfilled, the IOQ entry is returned to the free list. The IOQ only establishes the priority of requests for each device on a

first-in, first-out basis. Questions of priority in executing I/O drivers are resolved by the Dispatcher. Once several Device Controllers are running I/O programs, priority conflicts are resolved by hardware service priority.

NOTE

The device number associated with a particular DRT entry defines a Device Controller, and not necessarily an actual physical device. Also, some controllers identified by one device number are capable of driving several physical devices. Individual identification of physical devices is made by logical device numbers. The logical device number is the value used by the file system in requesting I/O, and the I/O system software performs the logical-to-physical device number translation.

5-60. SOFTWARE I/O

Even though GIC I/O tasks are handled with channel programs whose instructions specify I/O channel operations, software is still responsible for managing the I/O activity. Software I/O instructions are provided for this purpose, and to allow software to control interrupts, and start or halt channel programs, determine which channel addresses are occupied, and initialize or reset the I/O hardware. In addition, there may be unusual circumstances that require direct communication between software and a channel or device. (This description applies to GICs, but not necessarily to ATPs. ATPs use specially-designed "control programs.")

Some of the I/O instructions are global, as they affect and interact with all I/O channels simultaneously. Other instructions are addressed to a specific channel. There are no software I/O instructions intended exclusively for direct transactions with a device.

5-61. ADDRESSED I/O INSTRUCTIONS

There are nine I/O instructions: SIOP, HIOP, INIT, SMSK, RMSK, STRT, DUMP, RIOA, and WIOA. All use the module number of the IMB IOA addressed. The instructions support the multiple-IMB structure of the computer.

5-62. Initialize Channel (INIT)

This instruction prepares a channel for use by initializing the channel hardware and memory locations dedicated to that channel.

5-63. Start I/O Program (SIOP)

This instruction expects that the the Top of Stack (TOS) contains the 16-bit address of the first channel instruction of the channel program. TOS-1 contains the channel and device numbers.

This instruction does not begin execution of the channel program immediately; it merely causes the channel to request service. If the channel program for the device is not inactive, then the instruction is not executed, and the CPU indicators are set to CCG.

5-64. Halt I/O Program (HIOP)

This instruction expects that the TOS contains a channel and device number. It will halt execution of the channel program for the selected device if the channel program is in the Wait State.

With its channel program halted, a device may not request any new system interrupts, but any interrupts pending at the time the program halts will be serviced.

The program might not halt immediately, as there are places in the channel program where it is not advisable (during DMA transfers, for example). In general, a halt will not complete until a Wait instruction is encountered, or the program halts itself.

5-65. Read I/O Adapter (RIOA)

This instruction performs an IMB read, specified in the TOS. Either global or addressed commands may be done. The programer specifies the channel number, device number, I/O command, and register number. The data returned by the operation is saved on the TOS after the command specification are popped. The primary purpose of this instruction is to allow software to read registers on the I/O channels.

5-66. Write I/O Adapter (WIOA)

This instruction performs an IMB write, specified in the TOS. Either global or addressed commands can be done. The programmer specifies the channel and device numbers, I/O command, and register number (for the Write Data command), and the data to be sent. The entire specification is popped when the command completes successfully.

5-67. Set Mask (SMSK)

The four words on the TOS are stored into absolute memory locations 001A-001D. Each word is broadcast to all I/O channels on the IMB IOAs that were marked "present" during the initial power-on and cold load sequence.

Each IMB IOA has its own mask word and each bit position of the word corresponds to its channel number. When the bit is at a logical 0, the channel controller may not request an external interrupt for the device. If the bit is at a logical 1, an interrupt may be requested. The TOS is decreased by four words when the instruction is complete.

5-68. Read Mask (RMSK)

This instruction reads the four absolute memory locations 001A-001D and pushes them onto the stack. Each memory location represents a mask for each IMB IOA. There are always four words for the RMSK, even if there are fewer than four IMB IOAs in the system. Each bit of each word is to disable/enable the channel controller or to request external interrupts for the device. When the instruction is complete, the four TOS words will contain the masks for each of the IMB IOAs.

5-69. Start/Dump

A warmstart, dump, or load is executed on the module/channel/device specified in TOS Bits 7-15. The device can be either a disc or a magnetic tape for Start, but must be a disc for Dump. Start performs a load, but IMB IOA mapping is skipped. Dump does a memory dump by copying all CPU register contents to a reserved area of Main Memory starting at !0301. Then it performs a load without initializing memory. This is done by means of MPE's "soft dump" facility, which resides on the System Disc.

5-70. IMB CHANNEL PROGRAM CHARACTERISTICS

Each channel on the IMB can support up to eight devices (numbered 0 through 7), each device having a dedicated channel program controlling its I/O operations. These programs reside in Main Memory, and consist of specialized channel instructions that are completely unrelated to the machine instructions of the CPU. All eight programs for a given channel may be running simultaneously, even though actual execution at any time is dedicated to only one of those programs. When service to an HP-IB device is suspended, execution transfers to another program based on device demand and priority. This priority is established during a parallel poll response from the device requesting service. At that time, execution will begin (resume) on the lowest numbered device requesting service, according to the IMB priority as described in the next paragraph.

5-71. IMB PRIORITIES

5-72. IMB Direct Memory Access (DMA) Transfer

Channels requiring the IMB for a Direct Memory Access (DMA) get it based on their physical distance from the IMBI PCA in the card cage. The slot position with the highest priority is the one closest to the IMBI. As the distance from the IMBI increases, priority decreases.

5-73. Channel Priority

Any combination of device controllers can assert Channel Service Requests (CSRQs) simultaneously. When that happens, a "round-robin" priority scheme takes effect. The channel that has most recently been serviced will not be serviced again until all channels of lower priority are serviced.

5-74. "Round-Robin" Device Priority

Once channel priority has been established, a secondary "round-robin" scheme is established for devices on each individual channel. The following is an example of how the scheme works.

DEVICES	0	1	2	3	4	5	6	7
Last device to be serviced				X				
PPOL devices						X		X
Device requiring HIOP/SIOP		X			X		X	

The "round-robin" scheme asks the following questions:

Q: Who requires attention but is not PPOL?
A: 1,4,6 (only the first one is detected)

Q: Who is PPOL?
A: 5,7 (all are detected)

Q: Who was last serviced?
A: 3

Q: Who wants service?
A: 1,5,7

Q: Therefore, who gets serviced first?
A: 5 (since 3 was last serviced)

5-75. IMB CHANNEL PROGRAM INTERPRETATION

Interaction of the main CPU software with a device running under channel program control is limited to alteration of the program and starting/halting the program execution through the SIOP and HIOP machine instructions. That is, the main software does not interpret or execute the channel instructions, and the consequences of a channel program must generally be brought to the attention of software by an interrupt from the channel hardware.

An IMB channel requires that Channel Instruction execution be performed by the CPU which contains special microcode. This microcode can fetch, decode, and execute channel instructions without changing the existing software state of the machine. The microcode is entered when a channel requests service through assertion of the CSRQ2 line on the IMB. This microcode, like the system microcode, is stored in WCS.

5-76. IMB CHANNEL PROGRAM MANAGEMENT

Each of the eight devices on a channel has a four-word DRT entry in Main Memory. This entry is initialized by software prior to starting a channel program. As several programs can exist for one device, software sets the DRT entries and selects the program for the I/O task at hand.

Channel program execution is enabled with the SIOP machine instruction. When the channel program interpreter (CPU microcode) detects the request to start, it will set bits 0, 1 of word 3 of the DRT entry to 1,0 and begins program execution. The program then runs until halted by the program itself, by an abort condition or by an HIOP machine instruction. The HIOP instruction sets bits 0, 1 of word 3 to 0,1 and sets a bit in the channel to cause the device to request service for the change in program status. The CPU will service the request by terminating the channel program for the device and clearing bits 0,1 in word 3. An HIOP does not halt the program immediately, but sets a request to halt the program. The channel program will complete the current instruction and possibly execute a few more, but it is guaranteed to halt by the next WAIT instruction. There could also be a delay if the program which issued HIOP is of lower priority than other devices requesting channel service. Software cannot assume a program is no longer being serviced until bits 0 and 1 in DRT word 3 are both 0. If a program halts or aborts itself, it clears the DRT bits immediately. Thus, bit 0 of word 3 gives the running status, and bit 1 indicates a change is running status is pending.

Word 0 of the DRT entry is the channel program pointer and indicates the address in Main Memory of the next channel instruction to be executed. The pointer is updated by the program interpreter at certain times during channel program execution.

Each device is assigned a region in Main Memory which serves as a source of information during channel program execution. This area is called the Channel Program Variable Area (CPVA). It contains information about interrupts and aborts. Word 1 of the DRT entry is the absolute address of the CPVA.

5-77. CHANNEL INSTRUCTION SET

Channel instructions fall into three categories: channel functional, interactive, and non-interactive. The channel functional group permits the program to exercise the most basic functions of the I/O channel and the HP-IB. All data transfers and control functions may be effected with these instructions, providing the programmer has some knowledge of the HP-IB protocol and the channel hardware. HP-IB interface commands can be sent with the Command HP-IB instruction, and data transfers may be controlled with DMA. The primary application of this instruction group includes direct control of HP-IB devices with no programmatic assistance.

The non-interactive instructions provide a link between various segments of the program and allow orderly communication with software. These instructions produce no interaction with the device and can be used with any device irrespective of its protocol.

The interactive instructions are used with devices that obey the computer's protocol. I.e., transactions that can be accomplished with several instructions from the Channel Functional group are combined into a fixed sequence under a single interactive instruction. These sequences assume that the device recognizes not only an HP-IB primary address, but also a second address which contains a directive or command modifier.

All interactive instructions send a secondary address message to the device immediately following the address to talk or listen. These secondary addresses direct the device to interpret the subsequent data in a particular manner. The HP-IB standard defines the secondary address format as X11SSSSS where X is a don't care bit and SSSSS refers to the application-dependent bits of the message. Protocol uses the left-most S bit to distinguish the operation as either data or control. In this context, a control operation is one which indicates or affects the status of the device or its channel program.

0110SSSS = Data Secondary
0111SSSS = Control Secondary

The channel instructions that fall into the control category have their SSSS values predefined. Two of the control secondaries may be identical, but the operations are distinguishable as the device will be addressed to talk in one case and listen in the other.

CHANNEL INSTRUCTION SET CATEGORIES

INSTRUCTION	TYPE
Relative Jump	Non-Interactive
Interrupt	"
Wait	"
Read	Interactive
Write	"
Device Specified Jump	"
Identify	"
Read Control	"
Write Control	"
Clear	"
Read Modify Write	Channel Functional
Read Register	"
Write Register	"
Command HP-IB	"
Execute DMA	"
Write Relative Immediate	"
CRC initialize	"
CRC Compare	"

CHANNEL PROGRAM INSTRUCTIONS

Hex Code	Instruction	No. of Words	Comments
0000	Relative Jump	2	
01XX	Interrupt	2	018X-with halt 010X-with run
020X	Wait	2	
03XX	Read	5	If data chained, 00X0, where X=
04X0	Write	5	Blocks left
05XX	DSJ	2+XX	
0600	Identify	2	
07XX	Read Control	5	
08XX	Write Control	5	
09XX	Clear	2	
0A0X	Read Modify Write	2	
0B0X	Read Register	2	
0C0X	Write Register	2	
0D0X	Command HP-IB	5	
0E00	Execute DMA	5	
0FXX	Write Relative Immediate	2	
10XX	CRC Initialize	2	
11XX	CRC Compare	2	

5-78. I/O REQUESTS, INTERRUPTS AND PRIORITIES

The I/O interrupt system provides a framework for devices and channels to request machine program interrupts and channel program service. This paragraph describes device/channel interaction for HP-IB-compatible channels (e.g., the GIC), although other types of interface protocol is permitted. All channels follow the IMB protocol for requesting service.

Devices make service requests to channels via parallel polls. The channel determines device service, sorts reasons and priorities for requesting, and requests service on the IMB on behalf of its devices. I/O requests for external interrupts are made on the IMB via the IRQ line, and channel program service requests are made via the CSRQ line. The CPU obtains information for CSRQ using the "round-robin" channel priority scheme described in Paragraph 5-74.

5-79. Request Facilities

		Priority
		High
CSRQ	Channel Request For CPU Service	|
	Transparent To Software, Non-Maskable	|
		|
IRQ	Channel Request For Interrupt	|
	New Environment For Software	|
	Maskable At Channel Or System Level	Low

5-80. PARALLEL POLL. When no other operations are being performed on the HP-IB, the channel puts the bus into the parallel poll state (asserting ATN and EOI simultaneously). All devices must assert a logical one for an affirmative response. Although the HP-IB protocol allows any device to be assigned to any lines for polling, the DIO poll assignment determines which device will be serviced according to the device priority scheme previously described. When the GIC PCA detects a service Parallel Poll it generates a CSRQ2 to the IMBI PCA. The CSRQ2 signal is passed on to the CPU as an unsolicited message.

HP-IB device number 0 has the highest polling priority. It asserts its response on DIO Line 8. The polling and priority structure of HP-IB devices is arbitrated by the GIC PCA. It works as described in Paragraph 5-74.

When the GIC PCA detects a Service Parallel Poll, it generates a CSRQ2 on behalf of the polling device to the IMBI PCA. The CSRQ2 signal is then passed to the CPU as an unsolicited message by the IOB and CBI PCAs. If a response algorithm is required, the CPU provides it.

5-81. CSRQ. CSRQ is an IMB signal that channels assert to request channel program service on behalf of the interrupting device. A Processor Switch is provided on the IMB channel assemblies to select between CSRQ1 and 2. The switch should be set to the "B" (CSRQ2) position for normal operation.

The following conditions cause a channel to assert CSRQ:

a. Termination of a DMA transfer
b. An interrupt condition in the HP-IB interface logic
c. A request to start or halt a device's channel program
d. An affirmative parallel poll response on the HP-IB

Software cannot prevent a CSRQ from being recognized and serviced by the CPU, although some control is provided for selecting the reason a channel may assert CSRQ. Conditions a, b, and c are not requests from devices, but are generated by the GIC on behalf of a device.

CSRQ may not be the only reason the CPU was interrupted. Some testing is done to determine if CSRQ is the highest priority interrupt at the time (there are also IRQ, parity errors, and power fail interrupts). If a channel program for a device is halted when a request occurs for it, then the channel will look for the request from other devices. Servicing of CSRQ is transparent to the software environment.

An Obtain Service Information (OBSI) I/O Command is issued by the CPU to determine which device requires channel program service. During a Service Poll, each channel controller assigned to the IMBI will assert a logical 1 on its IMB data line if it is requesting channel service. The CPU control program selects the highest-priority device (using the "round-robin" priority scheme described in Paragraph 5-74), and issues an OBSI command through the IMBI to the channel device controller. The controller then returns its channel service status, and the CPU takes any necessary action.

5-82. IRQ. The Interrupt register is an addressible channel register (#C) which has one bit per device which is set to queue an interrupt request for that device. If any bit is set in the Interrupt register when the Interrupt Mask Bit is set by an SMSK command, the IMB IRQ line is asserted to request a CPU external interrupt. The output of the Interrupt register is priority encoded and the device number is sent back as data of the OBII (Obtain Interrupt Information) IMB command. When the mask bit is cleared by SMSK, no IRQ can be asserted. When an interrupt request for a device has been recognized, its bit in the interrupt register will be cleared by another write to register C. Device 0 has the highest interrupt priority. Note that devices cannot cause IRQ; channel program processors and diagnostic software are the only means intended to be allowed to set an interrupt request for a device. Devices directly generate CSRQ only.

DIAGNOSTIC THEORY AND USE

SECTION VI

This section describes the diagnostic theory of the HP 3000 Series 64 Computer. It also describes the operating functions of the Diagnostic Control Unit (DCU), and the user interface with the diagnostics themselves.

6-1. DIAGNOSTIC OBJECTIVES

The diagnostics have three objectives:

a. to provide PCA-level failure isolation,

b. to provide fault-locating tests customers can use,

c. to provide remote system diagnosis.

The diagnostics are designed to isolate failures to three or fewer PCAs, with an anticipated success ratio of 80 percent. That is, it is expected that in four cases out of five, the diagnostics will isolate failures to three or fewer PCAs. The diagnostics do not require specialized training on the part of the operator, and most can be run in remote mode.

6-2. DIAGNOSTIC CAPABILITIES

The diagnostic capabilities are controlled by two PCAs: the DCU and the Writable Control Store (WCS). They are overviewed in the following paragraphs.

6-3. Diagnostic Control Unit Overview

The DCU provides access to all PCAs in the CPU Bay card cage. Each CPU Bay card cage PCA contains shift registers that can be connected to form one large circular register ("shift string"). The DCU, at the operator's command, can halt the system and shift out data from a PCA, examine and/or modify it, then return it.

The DCU also controls the computer clocks; it can clock the entire system or any group of PCAs. The shift-stringing and clock-control functions allow the DCU to load/read the WCS and the Look Up Table (LUT), load/read memory, initialize the computer, perform maintenance and display functions, and load microcode.

The DCU has two major diagnostic roles. First, it is used to test a specific kernel of CPU hardware. Secondly, it enables the isolation capabilities of a group of tests called Fault Locating Diagnostics (FLDs).

In the first role, the Kernel Hardware Diagnostic is used to verify the kernel of hardware which loads and executes the FLDs. The Kernel Hardware Diagnostic consists of a set of DCU commands stored on flexible disc.

In the second role, aiding the FLDs in isolating failures, the DCU accesses hardware nodes that would otherwise not be accessible to microcode. This is done by embedding DCU commands in the FLDs themselves.

6-4. Writable Control Store Overview

The computer's microprograms are loaded from Random Access Memory (RAM) on the WCS PCA. To change a microprogram, the operator loads the WCS (via the DCU) from a flexible disc. This means microdiagnostics are executed without specialized test equipment.

The microdiagnostics isolate most failures with more certainty than provided by higher-level tests. Implementation, however, requires large amounts of microcode, much more than can be loaded into a conventional Read Only Memory (ROM) store. With WCS, the code is loaded and executed in sections, "overlays", each section 15 kbytes to 25 kbytes.

6-5. DIAGNOSTIC LEVELS

The diagnostics test the computer in ascending order of instruction levels. The levels are:

Level 1. Kernel Hardware verification
Level 2. Fault Locating Diagnostics (FLDs)
Level 3. Software diagnostics

Levels 1 and 2 test the PCAs in the CPU and Cache Memory, Main Memory, and I/O System Modules. These levels isolate failures to three or fewer PCAs (with an expected success rate of 80 percent). Level 3 contains CPU, Memory, and peripheral software diagnostics. The following paragraphs describe each level.

6-6. Kernel Hardware Verification, Level 1

Level 1 includes two diagnostics: a DCU self test, and a Kernel Hardware Diagnostic (KHD). The DCU self test resides in microprogram on ROM on the DCU PCA. It is started by a command entered at the console, loaded from flexible disc or by pressing a Self Test Switch on the PCA. The diagnostic provides Pass/Fail indications for the DCU hardware.

The KHD tests the kernel of CPU hardware necessary to execute the next higher level of diagnostics, the FLDs. The KHD consists of a set of DCU commands stored on flexible disc. Its tests are described in the following paragraphs.

6-7. WCS TEST. The WCS is tested using the DCU to force certain CPU signals to access the WCS array. Reads and writes of data patterns then test the functioning of the arrays. When the KHD detects a failure, it passes the WCS address of the failing bits to the DCU for display at the console.

6-8. MICROSEQUENCING AND DATA PATH TESTS. This portion of the KHD tests the basic CPU microsequencing and data flow. The DCU forces microcode into the CPU one-line-at-a-time, then checks to determine if the line produced the intended results.

The functions tested include: Rank 1-to-Rank 2 transfers, Control Store Address Register (CSAR) incrementing, V Bus inputs, jump logic, literals, ALU functioning, R Register and S Register decoding, Store field decoding, ALU Function Field decoding, system stop conditions, DEBUG command execution, and diagnostic freezes.

6-9. Fault Locating Diagnostics, Level 2

The FLDs have the chief responsibility for isolating failures to three or fewer PCAs.

There are six FLD sections:

a. Basic CPU checks
b. Preliminary CPU and Cache Memory checks
c. Final CPU and Cache Memory checks
d. Main Memory and I/O checks
e. Cold load loopback checks
f. I/O map test

The FLDs test on a functional circuit basis. A functional circuit consists of several logic gates connected to do a job. With the knowledge of which circuit is being tested and how it is physically partitioned across the PCAs, the deduction can be made as to which PCAs most likely contained the failing circuitry. The FLDs then rank the suspected PCAs in 1-2-3 order, and pass the information to the DCU which displays it at the console.

6-10. Software Diagnostics, Level 3

As previously described, the FLDs test the hardware in functional blocks. Thus they may occasionally fail to detect a short within a block. The magnetic tape-based software diagnostics aid in analyzing such problems.

The Main Memory Module is tested by a software diagnostic that interrogates the data arrays, syndrome generators, and the error-correction and logging circuits. The diagnostic isolates array failures to the IC level. This diagnostic is executed on the Diagnostic Utility System (DUS).

6-11. THE DCU PCA

The DCU PCA is a microprocessor-based diagnostic interface to the computer. It is physically located in CPU Bay Card Cage Slot No. 1. This enables it to access all PCAs connected to the card cage backplane.

The DCU has two primary functions: (a) it provides operator keyboard control of system startups; (b) it helps isolate hardware failures to three or fewer PCAs (using the diagnostics previously described).

By using serial shift strings and individual PCA clock control, the DCU can retrieve, modify, and restore virtually all of the registers, flags, state machine elements, and memory arrays on each PCA in the CPU Bay card cage. Communication with the operator is established via two serial data ports. One is for the local console; the other is for a modem for connection with a remote console.

The DCU also controls the Power System Controller (PSC). Via the PSC, the DCU can monitor AC and DC power, log failures, and take action (e.g., order a power fail shutdown). In addition, the DCU contains a battery-backed-up counter which keeps track of time during power failures.

6-12. DCU HARDWARE

The DCU PCA contains a Z80 microprocessor, 56 kbytes of Read Only Memory (ROM), 8 kbytes of static Random Access Memory (RAM), a PSC/PDM interface, a one-hour elapsed time counter, an RS-422 port interfacing to the system console, an RS232-C port for remote console hook-up, System Clock control signals to the CPU PCAs, a clock burst generator to control the System Clock, shift registers, and serial inputs for up to 28 PCAs. See Figure 6-1.

The DCU PCA interfaces to the rest of the computer via 28 signal lines. The lines go to the PCAs in the CPU Bay card cage, and are used to retrieve information such as status or contents of registers.

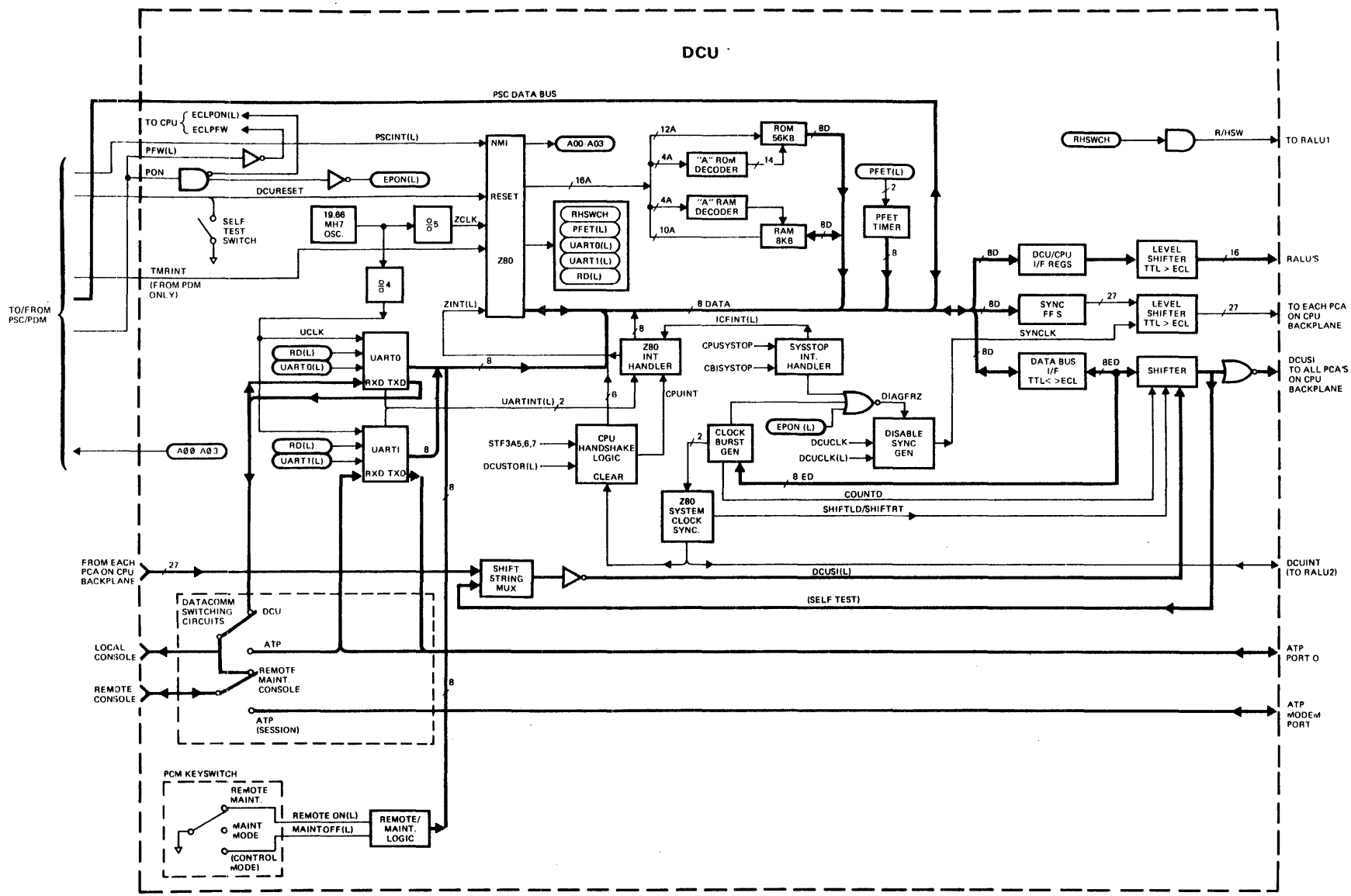


Figure 6-1. DCU PCA Simplified Block Diagram

6-13. DCU FUNCTIONAL DESCRIPTION

The following paragraphs describe the major logical circuitry of the DCU.

6-14. The Microprocessor

The most important component on the DCU PCA is the Z80 microprocessor. Most of the other ICs on the PCA either support the Z80 or are controlled by it.

The Z80 is an eight-bit processor that runs on a four-Mhz, single-phase clock. It has 16 address lines, eight bidirectional data lines, five main control lines, and two "interrupt" inputs. Instructions (which must be stored externally) are executed by stepping through a small set of operations: opcode fetch, memory read or write, I/O port read or write, and interrupt acknowledge. Depending on the instruction, execution requires from 4 to 23 clock cycles. Ten is the average number; thus the Z80 executes approximately 400k instructions per second.

Control of the other DCU PCA circuitry is achieved mainly through the used of I/O instructions. The microprocessor can monitor signals, read shift strings, etc., by treating these circuits as input ports. It can also manipulate control lines, turn syncs on and off, and send messages to the console by executing output instructions to other ports.

6-15. The ROM and RAM

The Z80's 16 address lines give it access to 64K bytes of memory. This is divided into 56 kbytes of ROM and 8 kbytes of RAM.

The ROM contains the program that runs the Z80. The RAM is used for general purpose data storage. Shift strings, console keyboard inputs, and microcode being transferred from tape to WCS, are examples of data saved or buffered in RAM. The first kilobyte is battery-backed-up; information that must be preserved during power failure is kept there.

6-16. Interrupts

Events that can interrupt the microprocessor include:

a. AC power too high or low,
b. DC power too high or low,
c. A character is received from the console or the operating system,
d. A System Stop occurs,
e. The CPU requests information via the DCUSTOR signal,
f. PDM one-second timer (32460B only).

The Z80 has two inputs for interrupts: a software-maskable interrupt (Int(L)) (not used on the 32460A; it is the interrupt signal from the watchdog timer in the 32460B) and a non-maskable interrupt (NMI(L)). The latter cannot be ignored as the PSC/PDM uses that line to input shutdown sequences. When the Z80 receives an interrupt, it automatically executes a firmware service routine.

6-17. Syncs

The DCU can issue system clock signals to any combination of PCAs connected to the CPU Bay card cage backplane. This allows diagnosis of failures on those PCAs. The DCU controls the clocks by manipulating its SYNC lines. The SYNC lines are sent to each PCA connected to the backplane. Each PCA uses its SYNC line to generate its own clock cycles.

6-18. Shifter

The shifter, in conjunction with the shift string multiplexer and the shift control lines, allows access to the registers and state machine elements in the CPU, as well as the contents of WCS, LUT, and Main Memory. This accessibility is made possible by the use of serial shift strings. Each CPU PCA has many of its registers and state machines built of shift register ICs. These are all joined to form a string, and both ends are connected to the DCU. To read a shift string, the DCU joins the ends of the string together, then reads them eight bits at a time. If the DCU needs to modify the contents of any register, it does so while the contents are "on board" the DCU, then shifts the string back to its PCA.

6-19. Clock Burst Generator

This circuit is used to generate one to 255 system clock cycles. It is used when the DCU needs to send the computer a certain number of clocks, as for example when the operator enters a CK or US command while in Maintenance Mode (the commands are described later in this Section). It is also used in the shifting process; to read eight bits at a time, the DCU sends the PCA eight clocks in a burst.

6-20. DCU Interface with the CPU

There are instances when the DCU needs to send information to the Multiprogramming Executive (MPE) operating system without stopping the computer and shifting the data into a register. For example, when the contents of the DCU Event Log are transferred to the system disc.

To facilitate this interface, the DCU has 16 data lines that go to the S Bus Multiplexer on the RALU PCAs, three data lines to the CTLB PCA, and two handshake lines. The data is transferred as follows: The CPU puts a command on the three lines to the DCU and pulses its handshake line (DCUSTOR). This pulse is used to latch the command in a register and to interrupt the Z80. The Z80 processes the interrupt, places the data on the 16 data lines, and pulses its handshake line (DCUINT), which signals the CPU that the information is available.

6-21. DCU OPERATING MODES

The DCU has four modes of operation: initialization, control, maintenance, and idle. Each is described in the following paragraphs.

6-22. Initialization Mode

The Initialization Mode is entered when the system is powered-up, either at initial power-on or at auto-restart after a power failure. In this mode, the DCU initializes itself, does a self test on the DCU and shift strings, and checks for a power failure. If the system is being initially powered-up, the DCU then enters the Control or Maintenance Mode, depending on the position of the Control/Maintenance Switch.

If a power failure occurs, the DCU loads microcode into the WCS to check the current load device status. (The device number is saved in the DCU battery-backed-up RAM.) If the disc is ready, the DCU loads and executes a microprogram which loads the system microcode from the load device, and restarts MPE. If the disc is not ready, the DCU does a one-minute test on the WCS and LUT, then rechecks to see if the disc is ready. If the disc still isn't ready, the DCU displays a message indicating the disc was not accessible and enters the Control or Maintenance Mode (those modes are described later in this section). The DCU will keep checking the disc for 10 minutes to see if it is accessible for an auto-restart. Once the disc is accessible to the computer, the operator can enter the Auto Restart command and the computer will resume its power fail recovery routine. The computer will not auto-restart without operator intervention if more than 10 minutes has passed.

If the computer is auto-restarting (after a power failure), the DCU enters the Idle Mode once the restart is complete. If a power failure had not occurred, the DCU would have displayed a Control or Maintenance Mode prompt.

6-23. Operator Interface

Operator interface in the Initialization Mode depends on whether the computer is in initial power-on or in auto-restart after a power failure.

In the former case, after the operator powers-on the computer, the DCU does a self test. On successful completion, it displays "DCU SELF TEST COMPLETE". The DCU then writes the default Start, Load, and Dump devices in the system status banner at the top of the console screen. The default values are: Load=0,2,1; Start=0,3,1; Dump=0,2,1. The DCU then displays a prompt, depending on which mode is enabled.

If the DCU Self Test fails, the suspected PCAs are identified and displayed at the console, if possible; if not possible (e.g., the DCU cannot communicate with the console), an error code is generated on the DCU LEDs.

In the case of an auto-restart following a power failure, the DCU performs the self test and displays the results at the console, as above. It then attempts a power fail auto restart from the START device number stored on battery-backed-up RAM. Whenever the LOAD, START, or DUMP device numbers are changed by the operator, the new numbers are written into the RAM. Should battery power be lost, the DCU reverts to the default values stored in ROM.

When the self test is complete, the DCU displays the message, "LOADING INIT/IDENT MICROCODE". The DCU then initializes WCS and loads a microprogram and the appropriate LOAD, START, or DUMP device number into WCS. This microprogram then performs an identify on the device and displays "LOADING DEVICE MICROCODE." The DCU then loads a device-dependent microprogram into WCS and performs a read on that device. It then displays "LOADING SYSTEM MICROCODE". The microprogram reads the system microcode from the device into WCS. Locations in WCS for the program currently running are reserved to prevent the program from writing over

itself. Once the system microcode is loaded into WCS the DCU starts a WCS microprogram. An MPE message is displayed (Coolstart, Reload, etc.) when the microcode is loaded after initial power on.

6-24. Control Mode

Control Mode allows the operator to macro-run the computer, cold load or warm start it, dump it, control the status display, display the DCU Log, control console speed, and display all Control Mode commands.

To enter Control Mode, the Control/Maintenance Switch must be in the Control position. When the operator types CNTL-B, the DCU sends the prompt C>. The operator then enters one of the Control Mode commands.

CAUTION

When you are finished using the Control or Maintenance Mode, use an EX[IT] Command before leaving the computer. The EX[IT] command will put the DCU in IDLE mode. If left in the Control or Maintenance Mode, MPE cannot output to the console, and the computer will eventually hang up.

The Control Mode commands are listed next in alphabetical order of their TWO-LETTER OPERATOR KEYBOARD ENTRYS. (Thus "LG" for Log appears BEFORE "LO" for Load.) The commands appear in the following paragraphs in the order they are listed here.

Operator Entry	Definition
AR	Auto Restart
DI	DIsplay
DU	DUmp
HA	HAlt
HE	HElp
LG	Log
LO	Load
RU	RUn
SP	SPeed
ST	STart
SW	SWitch

6-25. AUTO RESTART (AR). This command allows the operator to try to bring-up the computer after a Restart attempt fails due to a disc drive problem. After the disc drive problem is corrected, the operator can enter the AR command to try to continue the Restart.

6-26. DISPLAY (DI). This command is used to write the system status display banner to the system console. The status display is updated only when the DCU is in Control or Maintenance Mode.

To write the display, the operator enters DI. The display consists of one line, in inverse video, at the top of the system console. The status display is protected by an automatic Memory Lock, so a manual Memory Lock should not be used by the operator when the display is on.

The status display has six fields: Run/Halt, the octal display of the Start, Dump and Load settings, the WCS and Memory Breakpoint Enabled Flags, and the Overtemp Flag. The Load Register, which also serves as a diagnostic switch register, is 16 bits wide. The Start and Dump Registers are eight bits wide.

Display example:

```
RUN  START 0,2,1  DUMP 0,3,1  LOAD 0,3,1
```

6-27. DUMP (DU). This command dumps the system from the indicated (or saved) dump device. Performing the dump is done by halting the system using the HALT command, then using the DUMP command. If the system is not halted, the operator will be asked if the session can be aborted. The DCU does not generate any messages on successful completion, but error messages are generated if the dump is not successful. This command keeps the system in the Control Mode upon completion.

a. Setting the dump device address and doing a system dump:

C>DU[MP] [:]= <imb>,<channel>,<unit>

imb = DUMP device imb # (0-3) - bits 7,8. Default, 0.
channel = DUMP device channel # (0-15) - bits 9-12. Default, 2.
unit = DUMP device unit # (0-7) - bits 13-15. Default, 0.

b. System dump from the saved device:

```
C>HALT (RETURN)
C>DU[MP] (RETURN)
 .
 .     dump routine interaction
 .
```

6-28. HALT (HA). This command performs a macro-halt on the computer. When halted, the DCU remains in the Control Mode.

6-29. HELP (HE). This command lists all the commands valid in the Control Mode, as listed in these paragraphs.

6-30. LOG (LG). This command displays the contents of the DCU Status Log, indicating power line transients and dropouts, temperature status, and most recent DC voltages and currents, or it displays the DCU Event Log, which shows the last 128 computer events and the time each occurred. The

power system status log is for 32460A Computers only. No power system events are logged for the 32460B. The LG ST command for the 32460B will refer users to the SSDP-B LEDs that display power system status.

C>LG [<type>]
type = type of LOG to display
ST[ATUS] - display current hardware status
EV[ENT] - display event log

```
STATUS LOG

        DCU STATUS LOG

        AC POWER STATUS

PEAK AC VOLTS: PHASE 1 =  [<VOLTS>]  PHASE 2 =  [<VOLTS>]  PHASE 3 = [<VOLTS>]
OVERVOLTAGE TRANSIENT:  00000

LINE DROPOUT COUNT =  000

TEMPERATURE STATUS =  < 40  C

DC POWER SUPPLY STATUS
CPU BAY

            PS1          +5    5.04 V  45 A
                        +12   11.96 V   9 A
                        -12   12.04 V   2 A
            PS2          -5    5.00 V  95 A
            PS3          -5    5.00 V  80 A
            PS4          +5 B  5.00 V  20 A
                        +28   27.94 V   5 A
            PS5          -2    2.10 V 110 A

I/O BAY

            PS6          +5    5.01 V  90 A
            PS7          +5    5.01 V  80 A
```

DCU EVENT LOG
The following events are logged in the DCU logging RAMs:

- RUN
- SELF TEST
- LOAD
- START
- DUMP
- HALT
- POWER (ON/FAIL/AUTO-RESTART)
- OVERTEMP WARNING
- MODE CHANGE (MAINT-> CONTROL, ETC.)
- SYSTEM HALT
- POWER SUPPLY FAULT (DC OVER/UNDER VOLTAGE, OVERCURRENT, AC OVER/UNDER VOLTAGE)

```
    DCU EVENT LOG
DATE      TIME  EVENT
11/14/80  06:00 POWER ON
11/14/80  06:05 MAINTENANCE MODE ENTERED
11/14/80  06:25 OVERTEMPERATURE WARNING
11/14/80  06:50 SYSTEM HALTED
11/14/80  07:25 SYSTEM LOADED
  (and so on up to 128 events)
```

6-31. LOAD (LO). This command loads the MPE operating system from the load device. Loading requires that the operator first halt the computer, using the HALT command. If the system is not halted, the operator will be asked if it can be. The DCU does not generate any messages on successful completion of the LOAD command, but error messages are displayed if the load does not complete successfully.

a. Setting the load device address and loading MPE:

```
C>LO[AD] [:]= <imb>,<channel>,<unit>

    imb     = LOAD device imb # (0-3) - bits 7,8
    channel = LOAD device channel # (0-15) - bits 9-12
    unit    = LOAD device unit # (0-7) - bits 13-15
The default is 0,2,0.
```

b. Loading from the current load device:

```
C>HALT
C>LOAD
LOADING INIT/IDENT MICROCODE
LOADING DEVICE MICROCODE
LOADING SYSTEM MICROCODE
            ..... screen is cleared
```

6-32. RUN (RU). This command performs a macro-run on the computer. In this mode (Control), once the macro-run is done the computer exits the mode and returns the console to MPE.

6-33. SPEED (SP). There are two SPEED commands. One controls the DCU/Console speed only; the second also controls MPE's console speed. The two versions use the same syntax.

a. DCU/Console speed only:

```
C>SP[EED] <inspeed>,[<outspeed>]
  inspeed = input speed (10, 14, 15, 30, 60, 120, 240, 480, 960 character/sec)
  outspeed = output speed
```

b. DCU and MPE speed:

```
:SPEED <inspeed>,<outspeed>

CHANGE SPEED AND INPUT "MPE"
MPE
```

6-34. START (ST). This command loads MPE from the indicated (or saved) device. The operator must first used the HALT command to halt the system, then use START to warm start it. If the system is not halted, the operator will be asked if the session can be aborted. The DCU does not generate any messages upon successful completion, but error messages are displayed if the start is unsuccessful.

a. Setting the start device address and warm starting the system:

C>ST[ART] [:]= <imb>,<channel>,<unit>

imb = START device imb # (0-3) - bits 7,8
channel = START device channel # (0-15) - bits 9-12
unit = START device unit # (0-7) - bits 13-15
Default = 0,3,0

b. Warm starting from the current start device:

C>HALT (RETURN)
C>ST[ART] (RETURN)

6-35. SWITCH (SW) REGISTER. This command allows the operator to alter the contents of the 16-bit Switch Register. Note that the load device number is also altered when the Switch Register is altered.

a. Setting the entire register:

C>SW[ITCH] [:] = <switch>
switch = switch register contents

b. Setting individual bits:

C>SW<n> [:] = <state>
n = bit to set or clear (0=msb, 16=lsb)
state = state (0 or 1)

6-36. Maintenance Mode

CAUTION

When you are finished using the Control or Maintenance Mode, use the EX[IT] Command before leaving the computer. If left in the Control or Maintenance Mode, MPE cannot output to the console, and the computer will eventually hang-up.

Maintenance Mode allows the operator to perform hardware functions including micro-halting, micro-stepping, and micro-running the system; reading or altering shift strings, memory, WCS, or LUT; and displaying Maintenance Screens.

To enter Maintenance Mode, the Control/Maintenance Switch must be in the Maintenance position. When the operator types CNTL-B, the DCU sends the Maintenance Mode prompt (M>), and the DCU enters the Maintenance Mode. Control Mode commands may also be entered in the Maintenance Mode.

The Maintenance Mode commands are listed next.

NOTE

The commands are listed in alphabetical order of their TWO-LETTER OPERATOR KEYBOARD ENTRIES. e.g., the Micro-Step command is found as "US"; the Modify String command is found as "MS". The ABORT command is a special case; the operator keyboard entry for ABORT is CNTL-Y. In the following paragraphs, the commands appear in the order they are listed here.

Operator Entry	Definition
BA	BAse for numeric display
BY	remote mode disconnect (BYe)
CK	Clock(s)
DC	DCU control lines
DM	Display Memory
DS	Dump all board Strings to diskette or cassette
ED	Execute Diagnostic
EH	Enable Hard stop mode for system
EK	(sub-command of "TK"; see "TK")
ES	Enable Soft system error handling
EX	EXit maintenance mode
FL	Start the Fault Locating diagnostic system
HE	HElp facility
LL	List LUT
LM	List Memory
LS	List String
LW	List WCS
ML	Modify LUT
MM	Modify Memory
MS	Modify String
MW	Modify WCS
NX	(sub-command of "TK"; see "TK")
RL	Reset DCU Log
RM	Remote
RS	Reset Shift Strings
RX	Reset DCU
SC	SCreen
SY	SYnc
TK	Text (and execute) Kernel Diagnostic
TL	Text LUT
TW	Text WCS
UH	Micro-Halt

UP	UPdate maintenance display
UR	Micro-Run
US	Micro-Step
WS	Walk Stack
ZR	Perform self-test on remote hardware
ZS	DCU self test

6-37. BASE (BA). The BASE command sets the numeric default base for all numbers input or displayed. This applies to constants, register names, or expressions to be evaluated containing constants and/or register names, and screen displays.

BA[SE] [:] = <base>
base = B (binary)
O (octal)
D (decimal)
H (hex)

Numbers to be specified in a specific base are designated by using the following conventions:

= decimal; #32 = 32 decimal
% = octal; %32 = 32 octal (26 decimal)
! = hexidecimal; !32 = 32 hex (50 decimal)

6-38. BASE NUMBER CONVERSION. The operator can use the BASE command to convert numbers from one base to the default base by typing the number and its base.

<expression><base>
<expression> = number or numeric expression

The DCU then displays the converted number.

<expression><base> = <converted number>

M> BA = H	user sets default base to HEX
M>256D	user types number to convert
256D = 100	DCU displays conversion in HEX

6-39. BASE EXPRESSION EVALUATION. For all BASE commands which allow expressions, either a numeric value followed by the base or a numeric expression can be entered. In evaluating a numeric expression, the BASE command uses the following order of operations:

< logical shift left (first)
& logical AND
@ logical OR
$ logical XOR
/ division
* multiplication
- subtraction
+ addition (last)

The order of evaluation can be altered by using parentheses (). Registers, variables, and strings can be used in expressions. Expressions can be used only where explicitly allowed. All other entries must be numeric values. When HEX numbers are entered, precede values with A-F as the most significant digit with a 0 (e.g., enter the HEX number FFFF as 0FFFFH).

```
M>BA = H

M>1000H + (100H + 25D)*4

1000H + (100H + 25D)*4 = 1464

M>RA=RB+100H
  RA=100

M>V0=1
  V0=1

M>V0=V0+1
  V0=2

M>V0=V0$V0
  V0=0

M>V5=CTLB.0:6
  V5=3F

M>CTLB.0:6
  CTLB.0:6=3F

M>CTLB.0:6%1
  CTLB.0:6=7E
```

6-40. CLOCK (CK). This command allows the operator to clock the computer by striking the RETURN Key. To exit the CLOCK command, strike any other key, except ";". (The ";" key is reserved for stringing commands in a single line.)

If repeated clocks are not wanted, and the CLOCK command is the last command in a string, the operator must ensure that there is at least one space between <clocks> and the RETURN.

6-41. DCU CONTROL (DC). The DCU CONTROL command sets the DCU control lines DCUSHFT, DCULOAD, FRZENB(L), HRDSTP, or DIAGHOLD.

DC <ctl>

ctl = control word

bit

0 DCUSHFT (1=set,0=clear)
1 DCULOAD (1=set,0=clear)
2 FRZENB(L) (0=set,1=clear)
3 HRDSTP (1=set,0=clear)
4 DIAGHOLD(1=set,0=clear)

Use this command to control internal/external clocking. The clock control bit (HRDSTP) is saved by the DCU firmware and remains set or cleared until changed by another DC command. To properly use this feature, micro-halt the computer and send DC 0CH. This sets HRDSTP and clears the other control lines.

6-42. DISPLAY MEMORY (DM). The DISPLAY MEMORY command lists large blocks of memory on the console. It is displayed in both octal and ASCII. The field width must be set from one to eight words for tables.

```
M>DM <address>[,<count>[,width]]
   address = starting address ([bbbb.]aaaa)
         bbbb. = memory bank
         aaaa  = address in bank
   count   = number of words to dump
   width   = number of words dumped in a line (Default = 8)
```

6-43. EXECUTE DIAGNOSTIC (ED). This command causes the DCU to execute the diagnostic loaded into WCS beginning at the optional starting address. The DCU returns to the Maintenance Mode after completing the diagnostic.

```
M>ED [<addr>]
        addr = diagnostic starting address in WCS
          default = start at WCS addr 0
```

Diagnostics in WCS have the ability to display error messages set up by diagnostic programs and can tell the DCU to load the next diagnostic.

6-44. LIST LUT (LL). The LIST LUT command lists the indicated word of the Look Up Table.

LL <address>

address 0-0FFFFH <may be entered in any valid base>

default - causes the next word in LUT to be displayed.

The listing is formatted into the following fields:

```
LUT
 ADDRESS     ENTRY.PT    SRF.0:3    NIRSUB     DSPL       DBORSM
H0000       H0000       B000       B0         B000       B0
 --------    --------    --------   -------    --------   ------

 XC          PARITY      -INDR      XXX
B00         B0          B0         B000
 --------    --------    --------   --------
```

6-45. LIST MEMORY (LM). This command lists the block of memory containing the indicated address. (A block contains eight 16-bit words.)

LM [<address>

address 0-0FFFFFFFH — may be in entered in any base and may be an expression containing a register name (e.g., LM DB+6)

default - causes the next block to be displayed

The listing is formatted into the following fields:

```
MEM
 ADDRESS    WORD0      WORD1      WORD2      WORD3      WORD4
H00000000  H0000      H0000      H0000      H0000      H0000
 --------   --------   --------   --------   --------   --------

 WORD5      WORD6      WORD7      SOURCE
H0000      H0000      H0000      H0000
 --------   --------   --------   --------
```

6-46. LIST SHIFT STRING (LS). The LIST STRING command lists the shift string for the identified PCA. The string is formatted into the various registers unique to that PCA. Upon exiting, the SYNC line to the PCA displayed remains enabled so the clock command will generate clocks to it.

LS <PCA name>

Valid PCA Names:

```
VBUS  CBI3  SKSP  DCU3  MCS   RAL1  WCS0  WCS1
CAC   CBI5  CMA   DCU0  IOB1  MMC   RAL2
CBI1  CBI7  CTLA  DCU1  IOB2  OPT   RAL3
CBI2  CIR   CTLB  DCU2  IOB3  RAL0  SKSP
```

6-47. LIST WCS (LW). The LIST WCS command lists the indicated word of WCS using the following format:

```
WCS
 ADDRESS   WCS.0:16  WCS.16:16 WCS.32:16 WCS.48:16
 H0000     H0000     H0000     H0000     H0000
 --------  --------  --------  --------  --------
```

LW [<address>]
address 0-0FFFFH <may be entered in any valid base>

default - causes the next word of WCS to be displayed.

NOTE

Remember that the commands are listed in alphabetical order of their TWO-LETTER OPERATOR KEYBOARD ENTRIES. If you are looking for Micro-Step, for example, you should refer to "US".

6-48. MODIFY LUT (ML). The MODIFY LUT command allows the operator to modify the indicated LUT word.

ML [<address>]
address 0-0FFFFH <may be entered in any valid base>

default - causes the next word of LUT to be displayed for modification.

6-49. MODIFY MEMORY (MM). The MODIFY MEMORY command allows the operator to modify any of the 16 bytes of memory in the block of memory containing the indicated address. The listing uses the same display format as the LM command.

MM [<address>][,F[LUSH]]

address 0-0FFFFFFFFH

default - causes the next block of memory to be displayed for modification.

FLUSH - causes value input to be flushed from Cache to Main Memory, then read from Main Memory

6-50. MODIFY STRING (MS). This command allows the operator to modify any of the register fields in the shift string for the indicated PCA. The SYNC line to the PCA remains enabled so the CLOCK command can generate clocks to the PCA.

Single fields may be skipped during modification by using the TAB key. Digits in a field may be skipped during modification by using the space bar.

6-51. MODIFY WCS (MW). The MODIFY WCS command allows the operator to modify the indicated word of WCS. The format is the same as that for the LIST WCS command.

MW [<address>]

address 0-0FFFFH

default - causes the next word of WCS to be displayed for modification.

6-52. RESET DCU LOG (RL). This command allows the operator to clear the DCU Status or Event Logs.

M>RL [<LOG>]

LOG = type of log to reset (default = EVENT LOG)

EV = event log
ST = status log

6-53. REMOTE (RM). This command tests the remote link and allows the operator to establish a remote connection.

M>RM

For the remote link-up to work, several conditions must be
met: the remote console must be connected to the DCU via the junction panel; the REMOTE/CONTROL/MAINT. Switch must be set to REMOTE; communications with the remote console must have been established; and the speed of the local and remote consoles must be the same.

6-54. RESET SHIFT STRINGS (RS). This command initializes all PCA shift strings.

6-55. RESET DCU HARDWARE (RX). This command is equivalent to powering-on after a power failure, without doing an auto-restart. All DCU internal status bytes and buffer pointers are initialized (except the DCU Event Log).

6-56. SCREEN (SC). The SCREEN COMMAND allows the user to control the display when using the REGISTER DISPLAY commands. There are three SCREEN commands: FIRMWARE (default), SOFTWARE, and an extended register display. Screens are displayed in default bases unless the base is altered with the BASE command. The default bases are Hex code for FIRMWARE and Octal for SOFTWARE.

M>SC[REEN],[<type>]

type = FIRM for firmware display
= SOFT for software display

The display formats are shown below.

```
                    FIRMWARE MAINTENANCE DISPLAY

     RUN    URUN        UPDATE:OFF          BREAKPOINTS:  MB  WB

RA  0800   BKX3  FF02   SP0A  0000   NIR  0000  CPX1  0000   STAT
RB  0800   BKX4  FF03   SP1B  001C   CIR  0000  CPX2  0000   X
RC  F800   BKX5  FF04   SP2B  0008
RD  1C00   BKX6  FF05   SP3B  0000   CTRS 0000  PERF  0000
RE  07FF   BKX7  FF06   SP4A  0000   REGN 0800
1800                                              CSAR  0000   R2AD
RG  0800   BNKP  FF00   YRGA  0000   F1   00    RAC   00     R3AD
RH  00FF   BNKD  AAAA   YRGB  0000   F2   00
           BNKS  0000                 F3A  00    RRGA  0000   NP2A
SR    00                  OPA   0000   F4A  00    SRGA  0000   UBA
ESR   00                  OPB   0000   F5B  00
                                                  RRGB  0000   NP2B
NAMER 00   PDB   0000                   FSS  00    SRGB  0000   UBB
ENAMR 00   CAR   00000000               TFF  01
                                                                RAR
```

```
                    SOFTWARE MAINTENANCE DISPLAY

RUN   URUN          UPDATE:OFF          BREAKPOINTS:  MB   WB

RA  0000      DB    0000.0000      CPX'  0000   bndv tclk msgint
RB  0000      DL          0000     CPX2  0000   ics  disp syshlt
RC  0000      Q           0000
RD  0000      SM    0000.0000      NIR   0000
RE  0000      Z           0000     CIR   0000
RF  0000
RG  0000      PB    0000.0000      STAT  0000   mitroc
RH  0000      P           0000     X     0000
                    PL          0000
SR    00
ESR   00      P-PB        0000
```

NOTE

The bank is displayed to the left of the decimal point and the address is displayed to the right of the decimal point for DB, SM, and PB.

CPX1, CPX2 & STAT have certain bits decoded to the right of the display:

Register	Bit	Flag	Meaning if Inverse	Meaning if Normal
CPX1	2	bndv	Bounds violation	No bounds violation
	6	tclk	Timer interrupt	No timer interrupt
	9	msgint	Message interrupt	No message interrupt
CPX2	3	ics	ICS flag set	ICS flag not set
	7	disp	Dispatcher flag set	Disp flag not set
	10	syshlt	System halted	System not halted
STAT		m	Privileged mode	User mode
		i	Interrupts enabled	Interrupts disabled
		t	User traps enabled	User traps disabled
		r	Right stackop pending	Right stackop not pending
		o	Overflow flag set	Overflow flag not set
		c	Carry flag set	Carry flag not set
		nn	-	Segment number

6-57. SYNC (SY). This command allows the operator to selectively enable SYNC signals to any combination of PCAs.

M>SY <PCA id>[,<PCA id>]...[<PCA id>][,<ctl>] RETURN

PCA id = any valid PCA

ctl = SET to set syncs or CLEAR to clear syncs

If a few PCAs are to be synchronized, the PCA names may be entered in a single line terminated by the SET CTL command. For larger groups of PCAs, enter names on separate lines, then enter SET CTL. None of the syncs are enabled until the SET command is entered. To clear all syncs, enter the clear command.

```
M> SY PCA#1,PCA#2,...PCA#n,SET

        or

M> SY PCA#1,PCA#2,...PCA#m
M> SY PCA#m+1,.......PCA#n
M> SY SET            enable syncs

        or

M> SY CLEAR          disable syncs
```

6-58. TEXT KERNEL DIAGNOSTIC (TK). This command causes the DCU to text the diagnostic tape into the DCU's RAM. The program on the tape must be written in DCU Kernel Diagnostic language. After it is loaded, it begins execution. The Diagnostic Mode Flag is set to indicate Kernal

Diagnostic Mode, which disables SYSSTOP handling by the DCU and allows the diagnostic itself to handle the SYSSTOP. Additionally, the EK command allows re-execution of the currently loaded page of Kernel Diagnostic.

M>TK [<file>]
Text and execute Kernel test
file = file number to text into RAM
default = text first file

M>NX
loads the next Kernel Diagnostic file on the tape

M>EK
execute the currently loaded Kernel diagnostic in DCU RAM. On completion, return to Maintenance Mode.

6-59. TEXT LUT (TL). The TEXT LUT command reads the data in the indicated file into the LUT.

TL [<file>]
file = file number containing the LUT data
default = read 1st file
0 = rewind tape

6-60. TEXT WCS (TW). The TEXT WCS command reads the data in the indicated file into WCS.

TW [<file>]
file = file number containing the WCS data
Default = read 1st file
0 = rewind tape and read 1st file

6-61. MICRO-HALT (UH). This command micro-halts the system by setting DIAGFRZ to turn off clocks and clearing all SYNC signals.

6-62. UPDATE (UP). The UPDATE command allows the operator to control updating the Maintenance Screen displays when clocking or micro-stepping the computer, or when registers are altered.

```
M> UP[DATE] <flag>,<flag>, .... ,<flag>

            flag = update control

              ON        - updating turned on
              OF[F]     - updating turned off
              ST[RING]  - update string displays
              SC[REEN]  - update screen displays
              AL[L]     - update all displays (this is the
                          default)
              LA[ST]    - update last step of US or CK
              EA[CH]    - update each step of US or CK
              NO[NE]    - update none of the CK or US steps
```

6-63. MICRO-RUN (UR). This command allows the operator to micro-run the system by sending SYNC signals to all PCAs. This command does not set the Run/Halt flip-flop to Run.

6-64. MICRO-STEP (US). The MICRO-STEP command generates the indicated number of clocks to the PCAs. It also updates the last PCA string displayed after each clock, after all clocks, or does not update clocks at all, depending on the state of the Update Flag set by the UPDATE command.

US <clocks>
clocks 1-255 generate 1-255 clocks

This command allows the user to repeat-clock the system by striking the (RETURN) key. To exit the MICRO-STEP command, strike any other key, except ";". (The ";" key is reserved for stringing commands in a single line.)

If repeated clocks are not desired and the U-STEP command is the last command in a string, ensure that there is at least one space between <clocks> and the (RETURN).

6-65. WALK STACK (WS). The WALK STACK command allows the operator to trace Stack Markers from 'Q' back to the first marker (delta-Q = 0). The command is designed to handle both ICS and non-ICS stacks.

M> WS [<count>]

count = maximum number of markers to trace (1-255)

```
M> WS

            STACK MARKERS

    ADDRESS      X-REG   DELTA-P   STATUS  DELTA-Q
000000.037064  000000    000000   000000   000000
```

6-66. DCU SELF TEST (ZS). DCU Self Test performs tests on ROM (checksums), RAM, UARTs (wrapped and cross-coupled), terminal (accessiblity), DCU shift string hardware, power fail clock, and PSC/PDM self test. Upon exit, the system is reset, and the DCU code is restarted. Due to the destructive nature of this command, a confirmation message is issued to the user prior to execution of the command.

```
M>ZS        DCU self test

OK TO RESET SYSTEM?Y

DCU SELF TEST COMPLETE
```

6-67. Idle Mode

Idle Mode is entered when the operator exits the Control Mode via one of the terminating commands (RUN/LOAD, etc.), or exits either the Control or Maintenance Mode via the EXIT command.

In Idle Mode, the DCU continually executes a self test, monitors the power supplies, checks to determine if the operator has typed CNTL-B to enter the Control or Maintenance Mode, and checks to determine if the system itself has set the SYSSTOP line to enter the Maintenance Mode.

The Sysstop Handler is enabled when the DCU is in the Idle Mode. The Sysstop Handler examines the CPX1 and CPX2 Registers to determine the source of the interrupt, then reads the appropriate shift strings and displays the source(s) of the interrupt at the console. The Sysstop Handler then returns the computer to the Control or Maintenance Mode (depending on the position of the Control/Maintenance Mode Switch).

On detecting a SYSSTOP interrupt, the Sysstop Handler displays "SYSSTOP INTERRUPT" and one or more of the following messages:

a. CPU TIMER - the CPU has an instruction hung such that more than a maximum number of clocks has passed.

```
CPU TIMEOUT
M>              ...note MAINTENANCE MODE prompt
```

b. WCS PARITY INTERRUPT - the system writable control store has a parity error-- usually a hardware failure.

```
WCS PARITY ERROR
M>
```

c. MICROCODE HALTS - a hardware failure has forced the microcode to do a 'panic stop' in the microcode.

```
DIAG STOP ERROR
M>
```

d. CBI ERRORSUPTS (one or more of the following) - the CBI has detected an invalid condition--it can either be caused by a band sending CBI or by a bad receiving CBI.

```
HDWE ERROR CBI1
HDWE ERROR CBI2
HDWE ERROR CBI3
HDWE ERROR CBI5
HDWE ERROR CBI7
M>
```

e. BREAKPOINT INTERRUPTS - a memory or WCS breakpoint previously set in maintenance mode mode has been reached.

```
MEM BREAKPOINT AT xxxx.xxxx

WCS BREAKPOINT AT xxxx

M>
```

f. LUT PARITY ERROR - the system microcode Lookup Table has a parity error-- usually a hardware failure.

```
LUT PARITY ERROR
M>
```

g. CACHE ERRORS (either of the following) - the cache hardware has detected an invalid condition.

```
CAC ERROR
CMA ERROR
M>
```

h. SYSTEM HALT - system halt conditions are system microcode halts. They are different from SYSSTOPS.

```
SYSTEM HALT <nnnn> - <description>
```

nnnn	description
0001	SST VIOL in SEG # 1
0002	ABSENT SEGMENT ON ICS
0003	ABSENT or TRACE on SEG # 1
0004	STACK OVERFLOW on ICS
0005	CST length = 0
0006	CHANNEL PROG TIMEOUT
0007	BOOTSTRAP CHECKSUM ERROR
0010	BOOTSTRAP CHNL PROG ABORT
0011	PSEB INSTRUCTION WHILE (QI-18) < 0
0012	MODULE SEND MESSAGE TIMEOUT
0013	INCORRECT MODULE RESPONDING
0014	CHANNEL NOT SYSTEM CONTROLLER
0015	NON-RESPONDING IOB MODULE
0016	NON-RESPONDING CHANNEL
0017	CHANNEL 0 RESPONDING
0020	NO CSRQ OR EXT INTP ON MESSAGE INTP
0021	NOT ABLE TO PUT AS CONTROLLER IN CHARGE
0022	RECEIVE MESSAGE TIMEOUT
0023	I/O ERROR, PARITY/TIMEOUT
0024	UNKNOWN SYSHALT

i. MULTIPLE BIT MEMORY ERRORS - multiple bit errors (which are not correctable) have been detected in system memory.

```
MULTI-BIT ERROR
M>
```

j. UNEXPECTED DEBUG COMMAND IN MICROCODE - usually results from attempting to run diagnostics without the 'ED' command. A special diagnostic microcode command (DEBUG) has been encountered and the DCU is not prepared to handle it.

```
UNEXPECTED DEBUG
M>
```

k. INVALID ADDRESSES - usually an access to nonexistent memory--one or more of the following:

```
INVALID ADDRESS--MODULE 1
INVALID ADDRESS--MODULE 2
INVALID ADDRESS--MODULE 3
INVALID ADDRESS--MODULE 5
INVALID ADDRESS--MODULE 7
INVALID ADDRESS--CAC
```

l. CONTINUOUS DCUSTOR ERROR - series 64 is generating continuous DCUSTOR interrupt to the DCU. Somehow the Series 64 is in an abnormal state and the DCU had to disable this interrupt line.

```
CONTINUOUS DCUSTOR ERR
M>
```

6-68. Running Remote Diagnostics

With the exception of DCU Self Test (ZS), a remote operator with an HP 2642, 2645, or 2647 terminal can run any diagnostic available to the local operator. To complete the remote hook-up, the two operators must follow this procedure:

a. The operators first ensure their terminals are set for the same Baud rate (either 300 or 1200).

b. The local operator sets the Control/Maintenance Switch to Maintenance, then enters "RM" at the keyboard. The terminal banner now reads "Remote Enabled".

c. The remote operator sets the DA/VO Switch on his modem to VO. He then dials the number of the local modem. The local modem answers with a high-pitched tone.

d. The remote operator sets the DA/VO Switch on his modem to DA, then places the receiver on the modem. The two modems are now connected; the local console banner reads "Remote Established", the modem DTR LED lights, and the Remote LED on the System Status and Display Panel lights.

e. The remote operator must use "TELL" to send messages to the local console. Otherwise the computer will try to interpret the messages as commands.

NOTE

This procedure may vary slightly, depending on the type of phone and modem used.

SERIES 64 (32460A) POWER AND POWER CONTROL

SECTION VII

The HP 3000 Series 64 (32460A) Computer power system includes an isolation transformer, seven power supplies, a Power Control Module (PCM), a Power System Controller, and a battery backup pack. This section describes those components.

7-1. POWER SYSTEM CONTROLLER

The Power System Controller (PSC) PCA acts as the interface between the power supplies and the rest of the computer. It has two primary jobs: (a) it monitors and controls the supplies to ensure the computer has valid power levels; (b) it aids the Diagnostic Control Unit (DCU) in diagnosing and troubleshooting power supply failures. The PSC gathers information from the power supplies and sends it to the DCU to have problems diagnosed. (The DCU was described in Section VI.) See Figure 7-1.

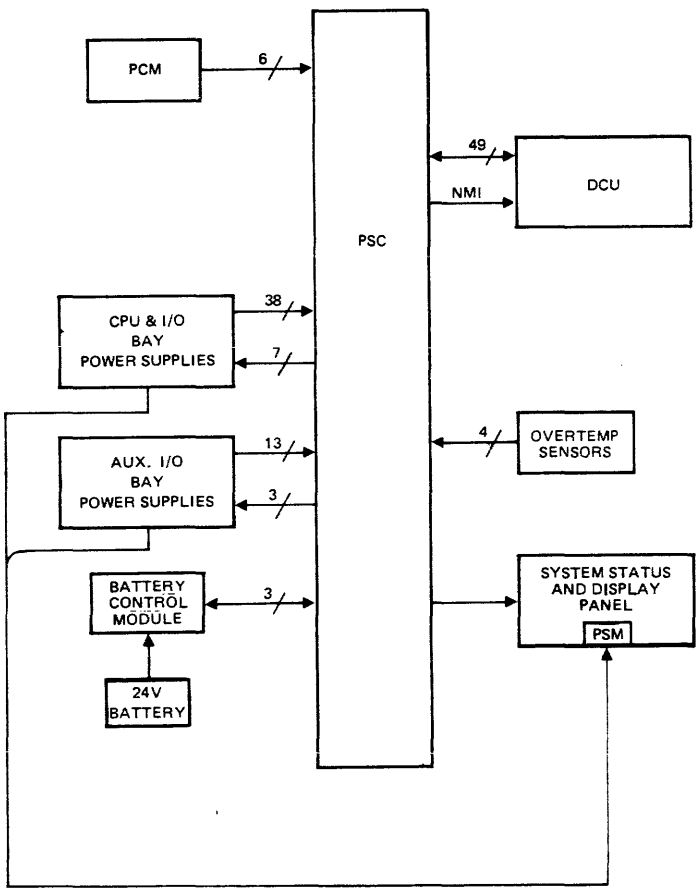


Figure 7-1. Relationship of PSC and Power Supplies

The PSC is not an autonomous, independent PCA. It must be connected to the DCU to function. (The computer will operate without the PSC, but this is not recommended, as serious power supply problems could go undetected until it is too late for the computer to protect data before power is lost.)

The PSC monitors AC and DC voltages, DC current, and overtemperature conditions, as shown in Figure 7-1. It has LEDs that allow the operator to display individual voltages and currents. The PSC, at the direction of the DCU, will shut down computer operations if a voltage goes out of tolerance or an overtemperature condition develops. Additionally, the PSC provides Line, Run/Halt, Battery, Overtemp, and Remote signals to the System Status and Display Panel (SSDP).

7-2. PSC HARDWARE

The PSC has analog and digital circuitry. As depicted in Figure 7-1, the circuitry on the left of the PCA is analog; the circuitry on the right of the PCA is digital.

7-3. PSC/DCU Communication Circuits

Commands from the DCU enter the PSC via a four-bit Address Bus, A0-A3. See Figure 7-2. Data is transmitted and received via the eight-bit bidirectional PSC Data Bus.

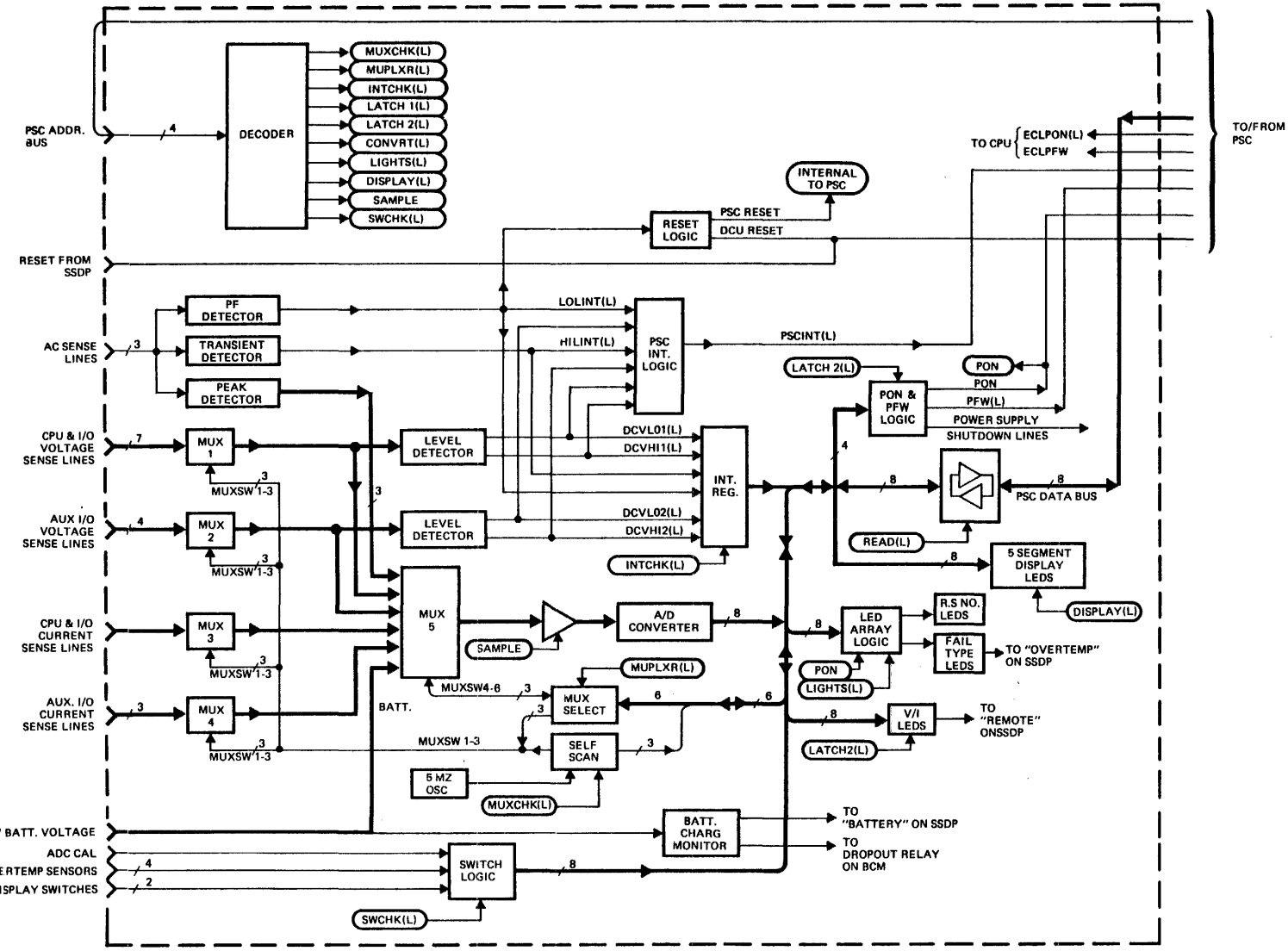


Figure 7-2. PSC Simplified Block Diagram

Interrupts from the PSC to the DCU are transmitted via the PSCINT(L) line. This line, which is non-maskable, requires the DCU's immediate attention, as it follows these signals on the PSC:

DCVLO1(L) Goes active when any CPU or I/O Bay power supply voltage falls below minimum levels.

DCVHI1(L) Goes active when any CPU or I/O Bay power supply voltage rises above maximum levels.

DCVLO2(L) Goes active when any Auxiliary I/O Bay power supply voltage falls below minimum levels.

DCVHI2(L) Goes active when any Auxiliary I/O Bay power supply voltage rises above maximum levels.

LOLINT(L) Goes active when any RMS AC voltage falls below minimum levels.

HILINT(L) Goes active when the voltage of any AC phase rises above maximum levels. The HILINT(L) circuitry detects line spikes as brief as 500 ns. This routine was removed for DCU date codes >= 2302.

7-4. Inputs From Power Supplies

Power supply voltage and current sense lines enter the PSC via Multiplexers 1 through 4. See Figure 7-2. Multiplexer (Mux) 1 handles the Voltage Sense lines for the CPU and I/O Bays; Mux 2 handles the Voltage Sense lines for a planned optional bay. From the multiplexers, the sense lines enter a level detector which checks for out of tolerance highs or lows. If the level is too high or too low, a PSCINT(L) signal is sent to the DCU.

Multiplexers 3 and 4 handle Current Sense lines from the power supplies. These lines do not propagate through any level-detection circuitry. (Over or undercurrents are detected and controlled by each power supply individually.) All voltage and current levels are inputted to Mux 5 where they can be sampled by the DCU.

7-5. DC Levels

The PSC has two operating modes: Self Scan and DCU Read. DCU Read occurs anytime the DCU reads information from the PSC. At any other time, the PCA is in Self Scan.

In Self Scan, the PSC constantly scans its inputs from the power supplies for out of tolerance levels. If an out of tolerance level is detected, a DC Voltage Low (DCVLO) or DC Voltage High (DCVHI) signal is sent to the PSC Interrupt logic and the PSC Interrupt Register. The address of the supply that sent the out of tolerance level is frozen in the Mux Select logic. The Interrupt logic interrupts the DCU by activating PSCINT(L). The DCU then interrogates the Interrupt Register to determine the type of interrupt, and reads the address of the faulty supply from the Mux Select logic. The DCU also lights the appropriate PSC LEDs to identify the faulty supply.

7-6. AC Voltage Sense

AC Voltage Sense lines enter the PSC from three isolation transformers in the PCM. These lines go through three levels of detection: power fail, transient detection, and peak level detection. Transient detection was removed for DCU date codes >=2302.

7-7. POWER FAIL DETECTION. Power fail detection is done by rectifying the AC input voltage and sending it to a low voltage level comparator. If the AC voltage from an isolation transformer secondary drops below 216 VAC, an AC Low Level Interrupt (LOLINT(L)) signal is sent to the Interrupt Register. PSCINT(L)) is sent to the DCU. The DCU then interrogates the Interrupt Register to find the cause of the interrupt.

7-8. AC TRANSIENT DETECTION. Transient detection takes the rectified input voltage and compares it with a maximum acceptable level. If the voltage is over 254 VAC, an AC High Level Interrupt (HILINT(L)) is generated and sent to the PSC Interrupt logic and the Interrupt Register. This routine was removed for DCU date codes >=2302.

7-9. AC PEAK LEVEL DETECTION. This circuitry measures the peak AC line voltage at the secondary of the main isolation transformers. This circuitry does not generate an immediate interrupt to the DCU, but is placed as an input to Mux 5 to be read by the DCU.

7-10. System Power and Overtemperature Failures

The following paragraphs describe the actions of the PSC hardware when a power failure or overtemperature condition occurs. All the LEDs mentioned remain lighted as long as battery backup power is available.

7-11. OVERTEMPERATURE CONDITIONS. The system has two sets of overtemperature sensors designed for either "low" (30 degrees C) or "high" (40 degrees C) overtemperature conditions. When a "low" switch opens, the following happens:

a. Overtemp LED on front display lights.

b. Overtemp message is sent to system console.

c. Console "beeps" every 10 seconds.

When a "high" overtemp switch opens, the following occurs:

a. Overtemp LED on front display lights.

b. Overtemp message is sent to system console.

c. Console "beeps" once each second.

d. The operator has one minute to take action. The choices are to either shut down the computer or ignore the warning.

e. After one minute, PFW(L) goes active. Ten ms later, all power supplies except the battery charger/backup supply are shut down via their Remote Shutdown lines from the PSC. At this time, power to the overtemp LED is lost and the LED turns off.

f. The system will not restart until the overtemp switches close and AC power is turned off and back on.

7-12. DC POWER SUPPLY FAILURE. When DCVLO1(L) or DCVLO2(L) goes active, or when a DC power supply fails during power-on, the following happens:

a. PFW(L) goes active immediately.

b. An LED on the PSC lights indicating which supply is faulty. If more than one supply or voltage is faulty, the lighted LED indicates which failed first.

c. An LED on the PSC lights indicating the type of failure (e.g., DC Undervoltage).

d. An LED on the SSDP lights indicating which supply is faulty (for parallel supplies, one LED represents the pair).

7-13. AC LINE FAILURE. When LOLINT(L) goes active, the DCU microprocessor computes the time remaining before the outputs of the DC supplies will fall. If the AC voltage does not return to normal in that time, PFW(L) is asserted and an LED on the PSC PCA is lit indicating an AC failure. If the AC line does return (as it would in the case of a temporary line dropout) before the outputs of the DC supplies start decaying, PFW(L) remains high and the system is notified that a non-fatal AC line failure occurred. This routine has been removed for DCU date codes >=2302.

7-14. AC LINE OVERVOLTAGE. When HILINT(L) goes active, the PSC determines if the overvoltage condition is a line transient or is an RMS overvoltage failure. If it is a line transient, it is reported to the system. If an RMS or continuous overvoltage condition is present, an overvoltage message is sent to the system console, PFW(L) is immediately asserted and an LED on the PSC is lit indicating the type of failure. After 10 msec, all DC power supplies except the battery charger/backup supply are shut down. This routine has been removed for DCU date codes >=2302.

7-15. PSC Battery Status/Monitor Circuit

The PSC Battery Status/Monitor Circuit checks the battery charging/discharging current and voltage. An LED on the SSDP indicates battery status. This LED flashes quickly when the battery is discharging, flashes slowly when it is charging, and turns off when the battery is fully-charged.

In the battery backup mode, the battery monitor circuit disconnects the battery from the backup supply if the battery discharges to a level of 20.0VDC. This helps prevent the battery from becoming permanently damaged due to excessive discharging.

7-16. PSC LED DISPLAY

Two switches control the operation of the PSC LED display. When Switch S1 (DISPLAY ON/OFF) is pressed, the display shows the output voltage of the first power supply in the list which follows. When Switch S2, (DISPLAY CYCLE) is pressed, the display shows the output current of that supply. Each time S2 is subsequently pressed, the display advances to the next voltage or current. For multiple-output supplies, each output is displayed before advancing to the next supply.

In addition to the numeric value, two LEDs light. One indicates whether the number displayed is a voltage or a current; the other indicates which supply is being monitored (this LED will also light if the power supply fails). The LEDs are numbered 1 through 11. The first seven corresponding supplies are:

```
PS 1.  63312F/P88 (+5.0V  @  50A)    (CPU Bay)
           "      (+12V   @  10A)        "
           "      (-12V   @  10A)        "
   2.  62605M/P71 (-5.2V  @ 100A)        "
   3.  62605M/P71 (-5.2V  @ 100A)        "
   4.  63312F/Q29 (+28.2V @  15A)        "
           "      (+5VB   @  22A)        "
   5.  62605M/P70 (-2.0V  @ 120A)        "
   6.  62605M/P71 (+5.0V  @ 100A)    (I/O Bay)
   7.  62605M/P71 (+5.0V  @ 100A)        "
```

LEDs 8 through 11 are available for power supplies for a planned optional bay.

To turn off the display, press S1.

7-17. PSC CONNECTION REQUIREMENTS

The following paragraphs provide specifications for the physical connections between the PSC and the rest of the computer. Also provided are pin assignments for the external connectors.

7-18. Connections to Power Supplies

a. Voltage Sense Lines:

- o Input impedance greater than 10k ohms.
- o Single-ended inputs; ground is common and is referenced from the PSC power input ground connection.
- o Intended to be connected to power supply sense lines at either supply end or backplane end.

b. Current Sense Lines:

- o Input impedance greater than 10M ohms.
- o Differential inputs; max. differential input voltage = 100 mV.
- o Maximum common mode voltage = +8 V.
- o Minimum common mode voltage = -12 V.
- o Connections are made to current monitor outputs of power supplies. Wires should be twisted pairs to minimize differential mode noise.

c. Shutdown Lines:

Inputs go to anodes of common-cathode diode array to prevent a faulty supply from pulling all other supplies down.

d. AC Sense Lines:

- Connected to three AC transformers (one per phase). Transformers are fused on the secondary side because of low current drawn.
- AC sense transformers each have a 187/250 VAC primary, 11:1 turns ratio, and are capable of supplying 10 mA rectified output current without significant loading of secondary winding.

7-19. SSDP Interface

Signal Name	Current Requirements
BATTERY1STATUS	-2 mA (OFF), +1 mA (ON)
OVERTEMP	-2 mA (OFF), +2 mA (ON)
CPU RUN/HALT	-2.5 mA (HALT), +1 mA (RUN)
REMOTE	-2 mA (OFF), +1 mA (ON)
+5V	-770 mA
+5VB#1	-120 mA
+15VC	-30 mA
-5.2V	+160 mA

7-20. PSC Pin Assignments

J1 -- PSC/DCU Test Fixture Interface (50-pin ribbon cable connector)

Pin	Signal	Pin	Signal
1.	A00	26.	WR(L)
2.	D0	27.	A13
3.	A01	28.	
4.	D1	29.	A14
5.	A02	30.	
6.	D2	31.	A15
7.	A03	32.	RESET(L)
8.	D3	33.	
9.	A04	34.	PSCINT(L)
10.	D4	35.	TIMERINT(L)
11.	A05	36.	
12.	D5	37.	DBUSENAB(L)
13.	A06	38.	PSCENAB(L)
14.	D6	39.	ROMDISAB(L)
15.	A07	40.	CPUR/H
16.	D7	41.	PFW(L)
17.	A08	42.	PON
18.	M1(L)	43.	
19.	A09	44.	
20.	MREQ(L)	45.	GND
21.	A10	46.	CLOCK
22.	IORQ(L)	47.	GND
23.	A11	48.	GND
24.	RD(L)	49.	GND
26.	A12	50.	GND

J2 --SSDP Interface (16-pin ribbon cable)

Pin	Signal
1.	+5V
2.	+5V
3.	+5V
4.	OVERTEMP
5.	GND
6.	GND
7.	GND
8.	GND
9.	+15C
10.	+5VB#1
11.	-5.2V
12.	
13.	
14.	CPURUN/HALT
15.	REMOTE
16.	BATTERY1STATUS

J3 -- Power Cable (9 pins)

Pin	Signal
1.	-12V
2.	Analog Ground
3.	+5VB1 (from CPU Bay Battery Backup Supply)
4.	+12V
5.	+15VC (From 63312F/P88 Bias Supply)
6.	+5V
7.	
8.	-5.2V
9.	Digital Ground

J4 -- Reserved for planned optional bay.

J5 -- PSC/PCM Interface (9-pin connector)

Pin	Signal
1.	DC Enable/Disable Switch #1
2.	AC Sense Transformer (phase A)
3.	AC Sense Transformer (phase A)
4.	DC Enable/Disable Switch #1
5.	DC Enable/Disable Switch #2
6.	AC Sense Transformer (phase B)
7.	AC Sense Transformer (phase C)
8.	AC Sense Transformer (phase C)
9.	AC Sense Transformer (phase B)

J6 -- Power Supply Voltage Sense & Shutdown Lines (20-pin connector)

Pin	Signal
1.	Keying plug
2.	-12V Sense
3.	+5V Sense (63312F/P88)
4.	-5.2V Sense
5.	-2V Sense
6.	+12V Sense
7.	+5V#1 Sense (62605M/P71)
8.	+28VB
9.	Shutdown (PS#4: 63312F/Q29)
10.	+5VB Sense
11.	Battery Dropout Relay
12.	Shutdown (PS#2: 62605M/P71)
13.	Shutdown (PS#3: 62605M/P71)
14.	Battery Charging Current Sense
15.	Shutdown (PS#1: 63312F/P88)
16.	Low Overtemp Sensor/Switch Line (CPU & I/O Bays)
17.	High " " " " " " "
18.	Shutdown (PS#7: 62605M/P71)
19.	Shutdown (PS#5: 62605M/P70)
20.	Shutdown (PS#6: 62605M/P71)

J7 -- Power Supply Current Sense Inputs (20-pin connector)

Pin	Signal		
1.	63312F/P88	(+12V)	Current Sense (High)
2.	"	"	" (Low)
3.	"	(-12V)	" (High)
4.	"	"	" (Low)
5.	62605M/P71	(-5.2V#1)	" (High)
6.	"	"	" (Low)
7.	"	(-5.2V#2)	" (High)
8.	"	"	" (Low)
9.	62605M/P71	(+5V#1)	" (High)
10.	"	"	" (Low)
11.	62605M/P70	(-2V)	" (High)
12.	"	"	" (Low)
13.	63312F/Q29	(+5VB)	" (High)
14.	"	"	" (Low)
15.	62605M/P71	(+5V#2)	" (High)
16.	"	"	" (Low)
17.	Keying plug		
18.			
19.	63312F/P88	(+5V))	" (High)
20.	"	"	" (Low)

High and Low refer to the high- and low-potential lines of each current-sense pair.

J8 -- Reserved for planned optional bay.

J9 -- Current Limit Reference Lines for CPU and I/O Bays; 20-pin connector.

Pin	Signal			
1.	+5.0 Iref	63312F/P88	(CPU)	PS#1
2.	+12 Iref	"	"	"
3.	-12 Iref	"	"	"
4.	-5.2 Iref	62605M/P71	"	PS#2
5.	-5.2 Iref	"	"	PS#3
6.	+5B Iref	63312F/Q29	"	PS#4
7.	+28B Iref	"	"	"
8.	-2.0 Iref	62605M/P70	"	PS#5
9.	+5.0 Iref	62605M/P71	(I/O)	PS#6
10.	+5.0 Iref	"	"	PS#7
11.				
12.				
13.				
14.				
15.				
16.	Keying Plug			
17.	63312F/Q29	(+28V)	Current Sense	(High)
18.	"	"	"	(Low)
19.				
20.				

J10--CPU and I/O Bay Power Supply Test Interface
(50-pin ribbon cable connector)

Pin	Signal	Power Supply (Location and Designation)		
1.	Ground			
2.	Ground			
3.	+5.0 VS	63312F/P88	(CPU)	PS#1
4.	+12 VS	"	"	"
5.	-12 VS	"	"	"
6.	-5.2 VS	62605M/P71	"	PS#2,3
7.	+5B VS	63312F/Q29	"	PS#4
8.	+28B VS	"	"	"
9.	-2.0 VS	62605M/P70	"	PS#5
10.	+5.0 VS	62605M/P71	(I/O)	PS#6,7
11.	+5.0 Ih	63312F/P88	(CPU)	PS#1
12.	+5.0 Il	"	"	"
13.	+5.0 Iref	"	"	"
14.	+12 Ih	"	"	"
15.	+12 Il	"	"	"
16.	+12 Iref	"	"	"
17.	-12 Ih	"	"	"
18.	-12 Il	"	"	"
19.	-12 Iref	"	"	"
20.	-5.2 Ih	62605M/P71	"	PS#2
21.	-5.2 Il	"	"	"
22.	-5.2 Iref	"	"	"
23.	-5.2 Ih	"	"	PS#3
24.	-5.2 Il	"	"	"
25.	-5.2 Iref	"	"	"
26.	+5B Ih	63312F/Q29	"	PS#4
27.	+5B Il	"	"	"
28.	+5B Iref	"	"	"
29.	+28B Ih	"	"	"
30.	+28B Il	"	"	"
31.	+28B Iref	"	"	"
32.	-2.0 Ih	62605M/P70	"	PS#5
33.	-2.0 Il	"	"	"
34.	-2.0 Iref	"	"	"
35.	+5.0 Ih	62605M/P71	(I/O)	PS#6
36.	+5.0 Il	"	"	"
37.	+5.0 Iref	"	"	"
38.	+5.0 Ih	"	"	PS#7
39.	+5.0 Il	"	"	"
40.	+5.0 Iref	"	"	"
41.	Shutdown	63312F/P88	(CPU)	PS#1
42.	Shutdown	62605M/P71	"	PS#2
43.	Shutdown	62605M/P71	"	PS#3
44.	Shutdown	63312F/Q29	"	PS#4
45.	Shutdown	62605M/P70	"	PS#5
46.	Shutdown	62505M/P71	(I/O)	PS#6
47.	Shutdown	62605M/P71	"	PS#7
48.				
49.				
50.				

J11 -- Reserved for planned optional bay.

7-21. AC AND DC POWER

The remainder of Section VII describes the AC and DC power components.

7-22. AC Input Specifications

The computer works with the following AC input voltages:

Type Service	Maximum Input Current	Where Used
3 Phase,208V,4wireY+GND	24A/Phase	USA Standard
3 Phase,380V,4wireY+GND	13A/Phase	Europe/Icon Std
3 Phase,415V,4wireY+GND	12A/Phase	United Kingdom

Input Tolerance: +4%, -10%
Power Factor: 0.6 to 0.8 typ
Inrush Current: 250 A per phase peak
Isolation Transformer: 3 @ 5KVA each
Input Frequency: 50 Hz or 60 Hz

7-23. DC Output Specifications

The computer has seven DC power supplies having the following maximum outputs:

+12.0 V @ 10 A (to CPU and I/O)
+ 5.0 V @ 200 A (to I/O)
+ 5.0 V @ 50 A (to Memory)
+ 5.0 V @ 25 A (+5B Battery backup, to Memory, CPU, I/O)
- 2.0 V @ 115 A (to CPU and Memory)
- 5.2 V @ 200 A (to CPU and Memory)
-12.0 V @ 10 A (to CPU and I/O)

7-24. AC Power Physical Characteristics

A PCM and a three-phase, box-shielded isolation transformer are the main AC power components. The PCM and the transformer are mounted in the bottom of the I/O Bay. Cable harnesses distribute AC power throughout the CPU and I/O Bays.

AC power uses an earth ground/logic common connection which provides isolation between the cabinet metal and logic common. This is compatible with safety regulations because isolation transformers are used in power inputs.

7-25. DC Power Physical Characteristics

Seven power supplies and a battery backup pack provide DC power. The DC system also includes a bus bar for the CPU Bay, bus bars for the I/O Bay, and several cable harnesses for power distribution.

7-26. AC Power Functional Characteristics

In the United States, the computer must be connected to 208 VAC, 60-Hz, three-phase power. In other countries, the computer can be connected to 415 VAC-Y, 50-Hz, three-phase, or 380 VAC-Y, 50-Hz, three-phase power.

Most AC power components are contained on the PCM, as shown in Figure 7-3. The PCM has a 50-Ampere, three-pole circuit breaker. The isolation transformers minimize computer susceptibility to line-generated noise and minimize the transfer of computer-generated noise onto the power lines. Fault protection and distribution from the transformer secondaries is provided on the PCM by a three-pole, 20-Ampere circuit breaker supplying the power supplies, and a 5-Ampere fuse supplying the cabinet fans. The main 50-Ampere breaker also contains an auxiliary contact for disconnecting the battery backup. Three step-down, low power, isolation transformers are connected across the secondary of the 20-Ampere breaker to provide a means for the PSC PCA to monitor the line voltage. All outputs from the PCM are through high-current, quick-disconnect connectors.

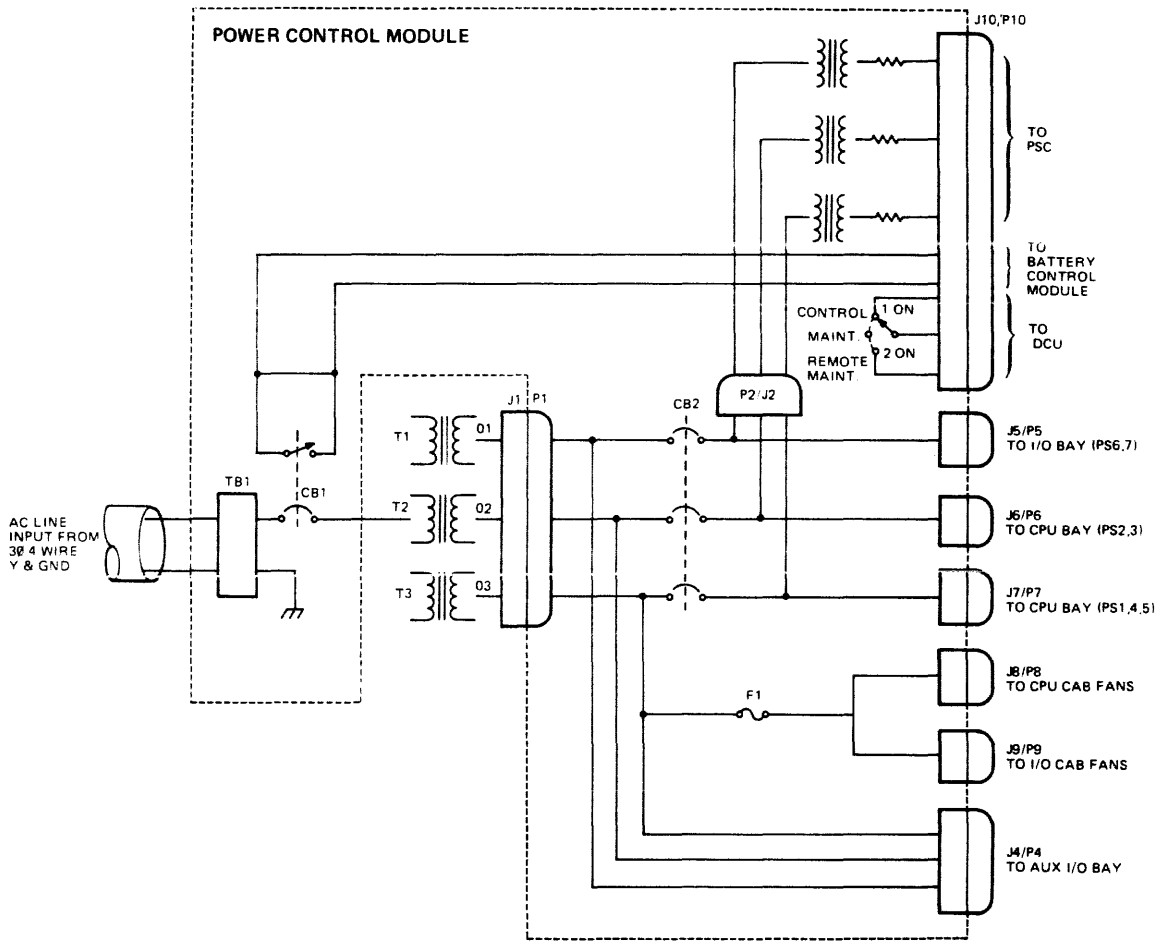


Figure 7-3. AC Power Distribution

7-27. DC Power Functional Characteristics

The seven DC power supplies are shown in Figure 7-4. Their maximum outputs are:

HP 63901F	+ 5.0 V @ 50 A
(one)	+12.0 V @ 10 A
	-12.0 V @ 10 A
	+15.0 V @ 50 mA
HP 63902F	+28.8 V @ 15 A
(one)	+ 5.0 V @ 25 A
HP 62970M	- 2.0 V @ 115 A
(one)	
HP 62971M	+ 5.0 V @ 100 A
(two)	
HP 62971M	- 5.2 V @ 100 A
(two)	

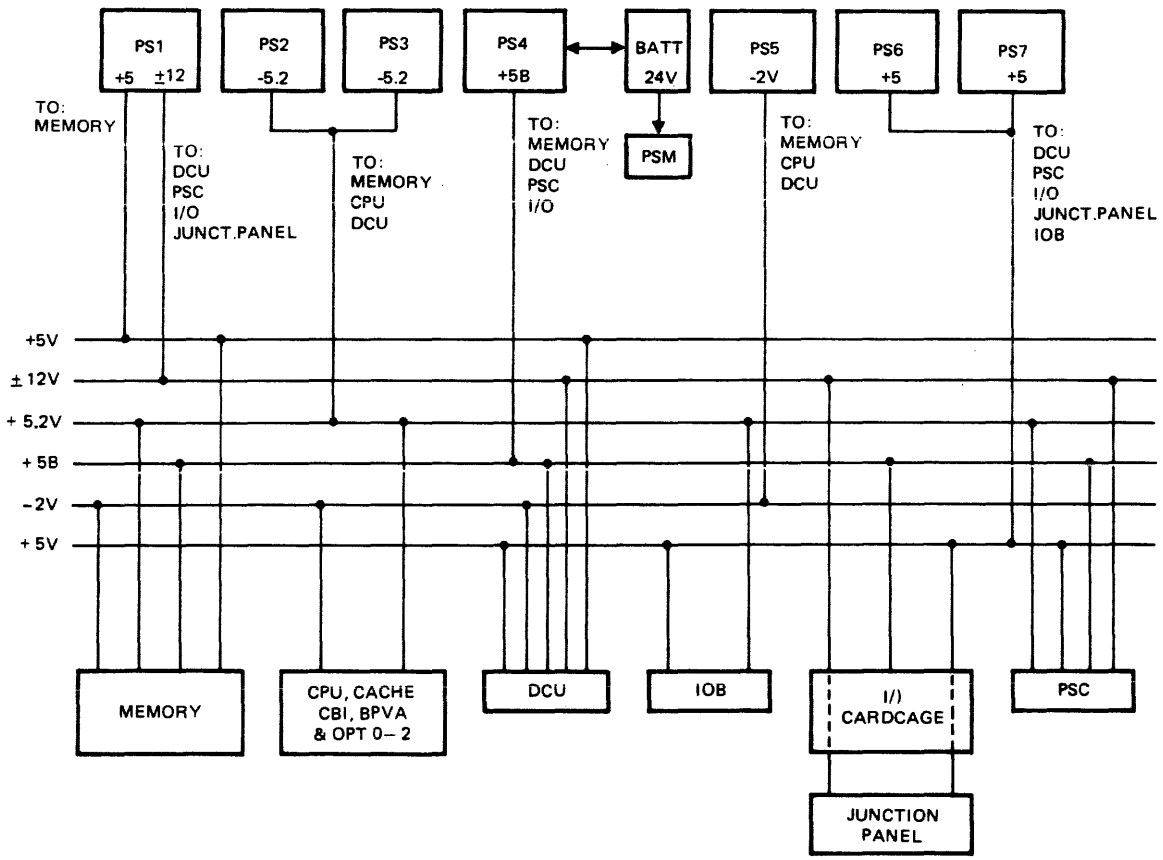


Figure 7-4. DC Power Distribution

The two +5 volt units in the I/O Bay are connected in parallel to provide current-sharing. For the same reason, the two -5.2 volt units in the CPU Bay are connected in parallel.

The +/- 12 volt, +5 volt, and +5 volt B power is distributed with cables. The -2 volt and -5.2 volt power is distributed via a six-layer bus bar. By alternating the bus bar layers between voltages and grounds for each of the three power supplies (two at -5.2 volts and one at -2.0 volts) susceptibility to external noise is kept to a minimum. The bus bar has tabs for connections to the power supplies and to distribute the load on the Memory and CPU Backplanes. The +5 volt supplies for I/O are also connected by bus bars. To provide stability and central distribution of current to the PCAs, power supply sense lines are connected at central locations on the backplanes.

7-28. Power Overload Protection

In addition to circuit breakers and fuses, the DC outputs of the power supplies are protected against short circuits by current-limiters. On the high-current outputs (+5.0 Volts, -2.0 Volts, -5.2 Volts), if a short circuit occurs for longer than a second, the power supply shuts itself down. Table 7-1 shows the calibration factors used to determine the current from a given voltage.

Table 7-1. Power Supply Calibration Factors

POWER SUPPLY	FUNCTION	PSC J10 PINS HIGH - COMMON	CALIBRATION FACTOR
PS1 63901F	+ 5V Set Vol.	3 - 1,2	- -
PS1 63901F	+ 5V Curr. Ref.	13 - 12	1.25mV/Amp .8Amp/mV
PS1 63901F	+ 5V Curr. Mon.	11 - 12	1.25mV/Amp .8Amp/mV
PS1 63901F	+12V Set Vol.	4 - 1,2	- -
PS1 63901F	+12V Curr. Ref.	16 - 15	4.75mV/Amp .21Amp/mV
PS1 63901F	+12V Curr. Mon.	14 - 15	4.75mV/Amp .21Amp/mV
PS1 63901F	-12V Set Vol.	5 - 1,2	- -
PS1 63901F	-12V Curr. Ref.	19 - 18	4.75mV/Amp .21Amp/mV
PS1 63901F	-12V Curr. Mon.	17 - 18	4.75mV/Amp .21Amp/mV
PS2&3 62971M	- 5.2V Set Vol.	6 - 1,2	- -
PS2 62971M	- 5.2V Curr. Ref.	22 - 21	0.75mV/Amp 1.33Amp/mV
PS2 62971M	- 5.2V Curr. Mon.	20 - 21	0.75mV/Amp 1.33Amp/mV
PS3 62971M	- 5.2V Curr. Ref.	25 - 24	0.75mV/Amp 1.33Amp/mV
PS3 62971M	- 5.2V Curr. Mon.	23 - 24	0.75mV/Amp 1.33Amp/mV
PS4 63902F	+ 5B Set Vol.	7 - 1,2	- -
PS4 63902F	+ 5B Curr. Ref.	28 - 27	2.35mV/Amp .43Amp/mV
PS4 63902F	+ 5B Curr. Mon.	26 - 27	2.35mV/Amp .43Amp/mV
PS4 63902F	+28.8V Set Vol.	8 - 1,2	- -
PS4 63902F	+28.8V Curr. Ref.	31 - 30	2.5mV/Amp .4Amp/mV
PS4 63902F	+28.8V Curr. Mon.	29 - 30	2.5mV/Amp .4Amp/mV
PS5 62970M	- 2.0V Set Vol.	9 - 1,2	- -
PS5 62970M	- 2.0V Curr. Ref.	34 - 33	0.56mV/Amp 1.79Amp/mV
PS5 62970M	- 2.0V Curr. Mon.	32 - 33	0.56mV/Amp 1.79Amp/mV
PS6&7 62971M	+ 5.0V Set Vol.	10 - 1,2	- -
PS6 62971M	+ 5.0V Curr. Ref.	37 - 36	0.75mV/Amp 1.33Amp/mV
PS6 62971M	+ 5.0V Curr. Mon.	35 - 36	0.75mV/Amp 1.33Amp/mV
PS7 62971M	+ 5.0V Curr. Ref.	40 - 39	0.75mV/Amp 1.33Amp/mV
PS7 62971M	+ 5.0V Curr. Mon.	38 - 39	0.75mV/Amp 1.33Amp/mV
PS1 63901F	Shutdown	41 - 1,2	- -
PS2 62971M	Shutdown	42 - 1,2	- -
PS3 62971M	Shutdown	43 - 1,2	- -
PS4 63902F	Shutdown	44 - 1,2	- -
PS5 62970M	Shutdown	45 - 1,2	- -
PS6 62971M	Shutdown	46 - 1,2	- -
PS7 62971M	Shutdown	47 - 1,2	- -

7-29. POWER SUPPLY ADJUSTMENTS

In the following descriptions, the power supplies are referred to by the designators used in Tables 7-1 and 7-2. Table 7-2 shows voltage and current operating limits for each supply.

Table 7-2. Power Supply Operating Limits

Power Supply #	Nominal Voltage	Operational Limits Lower	Operational Limits Upper	Production Adjustments Curr. Lim.(mV)	Production Adjustments Volts (lower/upper)
PS1	+5.00	+4.90V	+5.10V	58.75/76.00	+4.998/+5.002
PS1	+12.00	+11.9V	+12.10V	44.18/49.82	+11.95/+12.05
PS1	-12.00	-11.9V	-12.10V	44.18/49.82	-11.95/-12.05
PS2	-5.230	-5.175V	-5.275V	ABOVE 65.00	-5.215/-5.225
PS3	-5.230	-5.175V	-5.275V	58.00/62.00	-5.255/-5.265
PS4	+5.00	+4.90V	+5.10V	53.20/58.80	+4.998/+5.002
PS4	+28.80	+28.00V	+28.90V	34.20/41.80	+28.75/+28.85
PS5	-2.100	-2.08V	-2.12V	57.00/67.31	-2.11/-2.09
PS6	+5.05	+4.80V	+5.10V	ABOVE 65.00	+4.995/+5.005
PS7	+5.05	+4.80V	+5.10V	42.30/47.70	+5.045/+5.055

7-30. Voltage Adjustments

Table 7-2 lists the tolerances for the power supplies in a fully-configured computer. The Production Adjustments columns show the limits set by the factory before the computer is shipped.

The settings of the -5.2 volt and -2.0 volt supplies (PS2, 3, and 5) are critical as they supply ECL voltages which allow very little latitude. Before being installed in an empty system, the power supply outputs should be adjusted to nominal voltages with their current limits set to minimum. With the system unloaded and AC power on, all supply voltages should be checked using a calibrated DC voltmeter. All voltages should be nominal as measured at the appropriate pins on PSC PCA Connector J10, listed previously. All power supplies (except parallel units PS2/3 and PS6/7) require setting a voltage adjust potentiometer while monitoring the voltage at the power supply sense leads or the appropriate pins on PSC PCA Connector J10.

When adjusting power supplies, it is recommended that the "Disable Shutdown" line on the PSC be tied to ground. This will disable PON removal and AC power will not have to be recycled to restore PON.

7-31. Current Limit Adjustments

Adjustments to power supply current limit controls should be done with the computer unloaded or lightly-loaded. This results in the quickest adjustment for the parallel supplies (PS2/3 and PS6/7). If current is being drawn from both supplies simultaneously, voltage-balancing for the current sharing transfer is much harder. All the 62605M supplies should have their Latch OFF/Foldback Reset Switch set to Foldback Reset when adjustments are being made, then switched to Latch OFF when the adjustments are complete.

Adjustments to power supply current limits are made by monitoring the current reference terminals on the power supplies or at PSC PCA Connector J10. The Current Limit Potentiometers are adjusted for the Reference Settings listed in Table 7-2.

7-32. BATTERY BACKUP

Battery backup supplies +5 volts to the Main Memory arrays when AC power is supplied to the computer. It also supplies +5 volts when the AC power is interrupted, the length of time depending on the size of the load. A computer with 8 Mbytes of memory and 20 Intelligent Network Processors (INPs), drawing 22 Amperes, will receive battery backup power for approximately half an hour. A computer with 8 Mbytes of memory but no INPs, drawing 13 Amperes, will receive power approximately 50 minutes.

Backup consists of a 63902F dual DC-to-DC converter, a Battery Control Module PCA, and a battery assembly consisting of 12 five-ampere-hour cells. When AC power is normal, the first DC-to-DC converter section of PS4 supplies 28.8 VDC to the battery assembly and the second DC-to-DC converter, via the path shown as a solid line in Figure 7-5. When the AC line is down, the battery assembly supplies power via the dashed line shown in Figure 7-5.

The voltage supplied to the battery and the +5B DC-to-DC converter depends on the ambient temperature. For optimum life of the battery, the first stage converter section should be set to 28.8 VDC at 20 degrees C. As the ambient temperature changes, the charging voltage will be modified by approximately -40 mV per degree C.

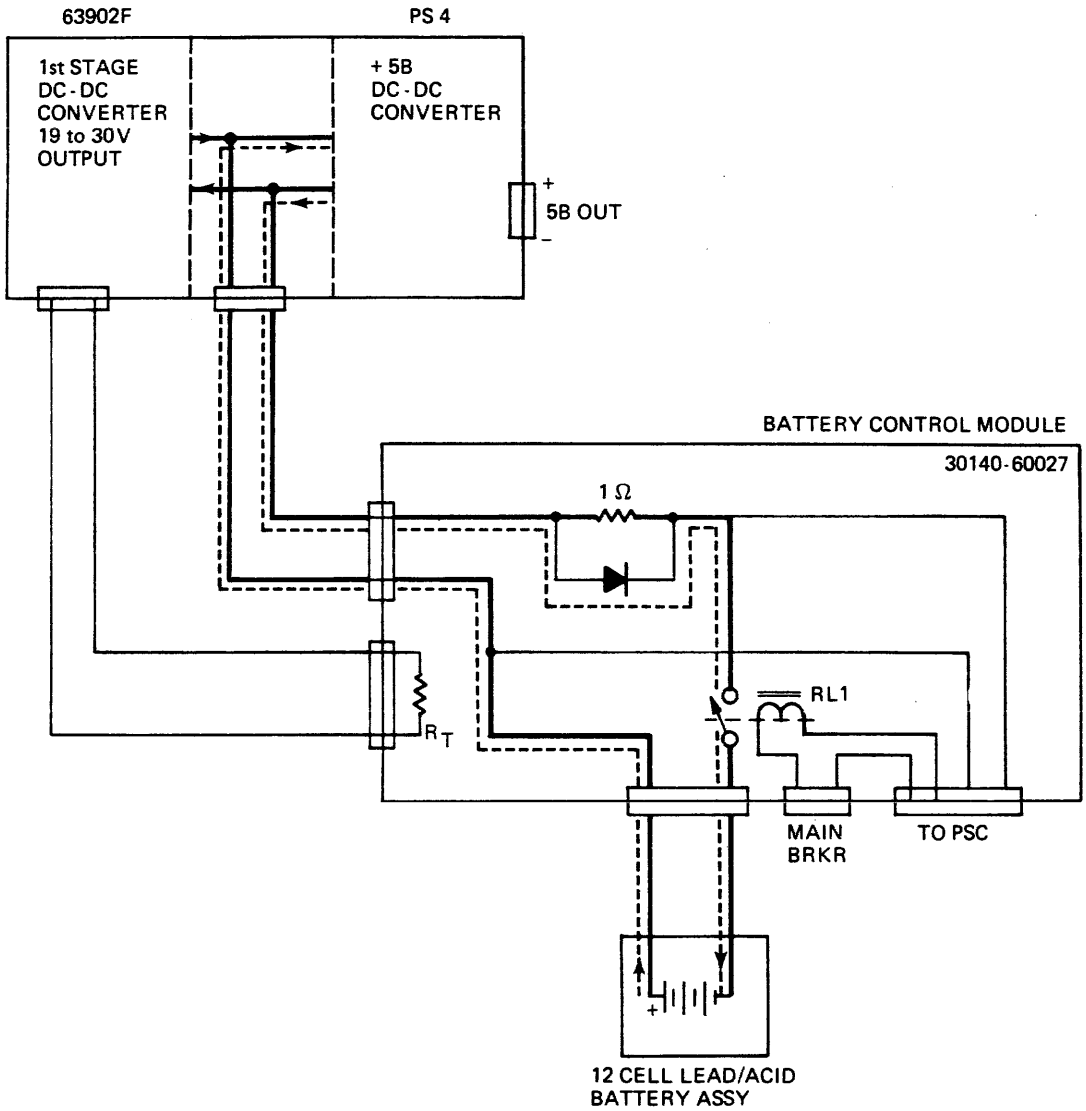


Figure 7-5. Battery Backup System

7-33. INDIVIDUAL POWER REQUIREMENTS

Table 7-3 lists the DC power requirements for individual components of the computer. Table 7-4 lists the volt-ampere requirements for the CPU and I/O Bays.

Table 7-3. DC Requirements

PCA	P	X	-5.20	-2.00	+5.00	+12.00	-12.00	+5.00B	+24V	#PCAs
DCU	C	P	2.63	3.10	3.00	.200	.10	.20	.00	1
PSC	C	P	.15	.00	1.00	.500	.25	.09	.00	1
PDM	C	P	.00	.00	.00	.00	.00	1.00	.20	1
RALU	C	A	31.68	12.80	.00	.00	.00	.00	.00	4
CIR	C	A	7.57	2.48	.00	.00	.00	.00	.00	1
VBUS	C	A	7.07	3.06	.00	.00	.00	.00	.00	1
SKSP	C	A	5.80	3.39	.00	.00	.00	.00	.00	1
WCS	C	A	16.30	3.60	.00	.00	.00	.00	.00	2
WCSN	C	P	5.00	3.50	5.00	.00	.00	.00	.00	1
CTLA	C	A	7.17	2.90	.00	.00	.00	.00	.00	1
CTLB	C	A	7.00	3.53	.00	.00	.00	.00	.00	1
CAC	C	A	7.23	4.29	.00	.00	.00	.00	.00	1
CMA	C	A	9.36	2.05	.00	.00	.00	.00	.00	1
CBI	C	A	24.84	11.84	.00	.00	.00	.00	.00	4
IOA	C	A	13.90	6.86	4.00	.00	.00	.00	.00	2
MMC	M	P	2.42	3.55	.12	.00	.00	.53	.00	1
MCS	M	P	7.00	6.05	.67	.00	.00	.19	.00	1
MMA	M	A	.64	.00	42.10	.00	.00	15.41	.00	8
INP	I	P	.00	.00	25.00	2.50	1.50	4.20	.00	10
SIB	I	A	.00	.00	6.00	.00	.00	.00	.00	1
AIB	I	A	.00	.00	20.00	.00	.00	.00	.00	4
PINT	I	A	.00	.00	3.00	.20	.12	.00	.00	2
GIC	I	A	.00	.00	29.40	.21	.00	.00	.00	7
IMBI	I	A	.00	.00	9.00	.00	.40	.00	.00	1
J12D	I	P	.00	.00	.82	.10	.09	.00	.00	1
J4M	I	P	.00	.00	10.40	1.76	2.00	.00	.00	8
Totals			155.76	73.00	159.51	5.47	4.46	20.62	.20	66

P - Denotes module in which PCA is used (CPU, MEM, I/O).
X - Denotes whether data is Actual or Predicted.
J12D- Denotes a 12 direct connect ATP junction panel.
J4M - Denotes a 4 modem ATP junction panel.
PINT- Denotes the IMBI-2619 printer interface.

24 Volts is used on the Series 64 (32460B) PDM PCA only.

Table 7-4. Volt-Ampere Requirements for CPU and I/O Bays

Ref	Item	Loc	VA		Surge Current		DC Power Outputs	
			Max	Typ	Max	Typ	Max	Typ
PS5	62970M	CPU	850	650	35 A		-2.0V @ 115A	-2.0V @90A
PS2	62971M	CPU	2000	1700	35 A		-5.2V @ 100A	-5.2V @90A
PS3	62971M	CPU	2000	1700	35 A		-5.2V @ 100A	-5.2V @90A
PS1	63901F	CPU	1300		80 A		+5V@50A 12@10A	+5V@40A +12V@3A
PS4	63902F	CPU	1000		80 A		+5B@25A +28.2@15A	+5VB @20A
PS6	62971M	I/O	2000	1500	35 A		+5V @ 100A	+5V @80A
PS7	62971M	I/O	2000	1500	35 A		+5V @ 100A	+5V @80A
	Fans	CPU	440	350	-		-	-
	Fans	1 I/O	125	100	-		-	-

TOTAL VA: 6000 VA 150A/PHASE SURGE

SERIES 64 (32460B) POWER AND POWER CONTROL

SECTION VIII

The HP 3000 Series 64 (32460B) Computer power system uses six power modules (the term modules refers to supplies, arranged in four module sets, to provide DC power. In addition, the power system uses an AC unit, three ferro-resonant transformers, a Power Distribution Monitor , and a battery backup pack. This section describes those components.

8-1. POWER MODULE SETS

The six power modules are arranged in four module sets. As shown in figure 8-1, the sets are identified A through D.

Module Set A contains two parallel modules, A1 and A2. They provide -5.225V to the Main Memory, CPU, and Cache Memory PCAs.

Module Set B consists of one module which delivers the battery backup voltage (see figure 8-1). This module provides 5.05V to the memory array on the Main Memory PCAs. During a power failure, the battery backup will provide power to the memory arrays (MMAs), MMC, MCS, INP, DCU, and PDM for at least 15 minutes.

Module Set C also consists of one module. It provides -2.1V, +12V, and -12V to the CPU and Cache Memory PCAs.

Module Set D contains two parallel modules, D1 and D2. They provide 5.05V to the Main Memory and I/O PCAs. (If an optional Auxiliary I/O Bay is present, it contains a fifth set, Module Set E; its characteristics are identical to set D's.)

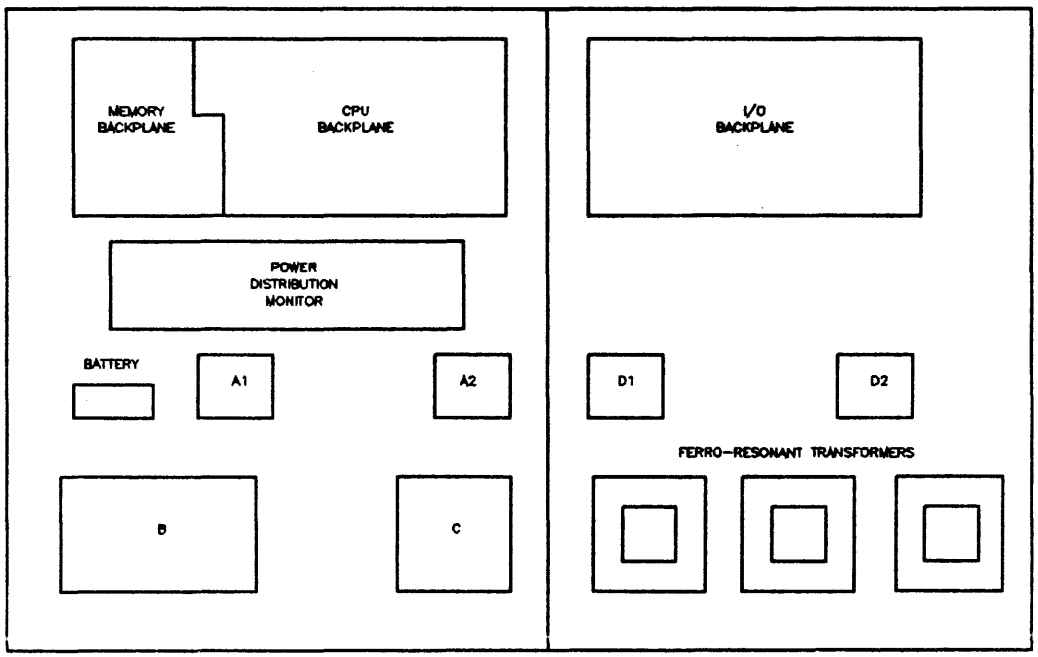


Figure 8-1. Series 64 (32460B) Power Module Organization

8-2. POWER DISTRIBUTION MONITOR

The Power Distribution Monitor (PDM) PCA acts as the interface between the power modules and the rest of the computer. It has three primary jobs: (a) it monitors and controls the modules to ensure the computer has valid power levels; (b) it aids the Diagnostic Control Unit (DCU) in diagnosing and troubleshooting power module failures, and (c) it redistributes +12V, -12V and the battery backed up +5VB voltages. The PDM gathers information from the power modules and sends it to the DCU to have problems diagnosed. (The DCU was described in Section VI.) See Figure 8-2.

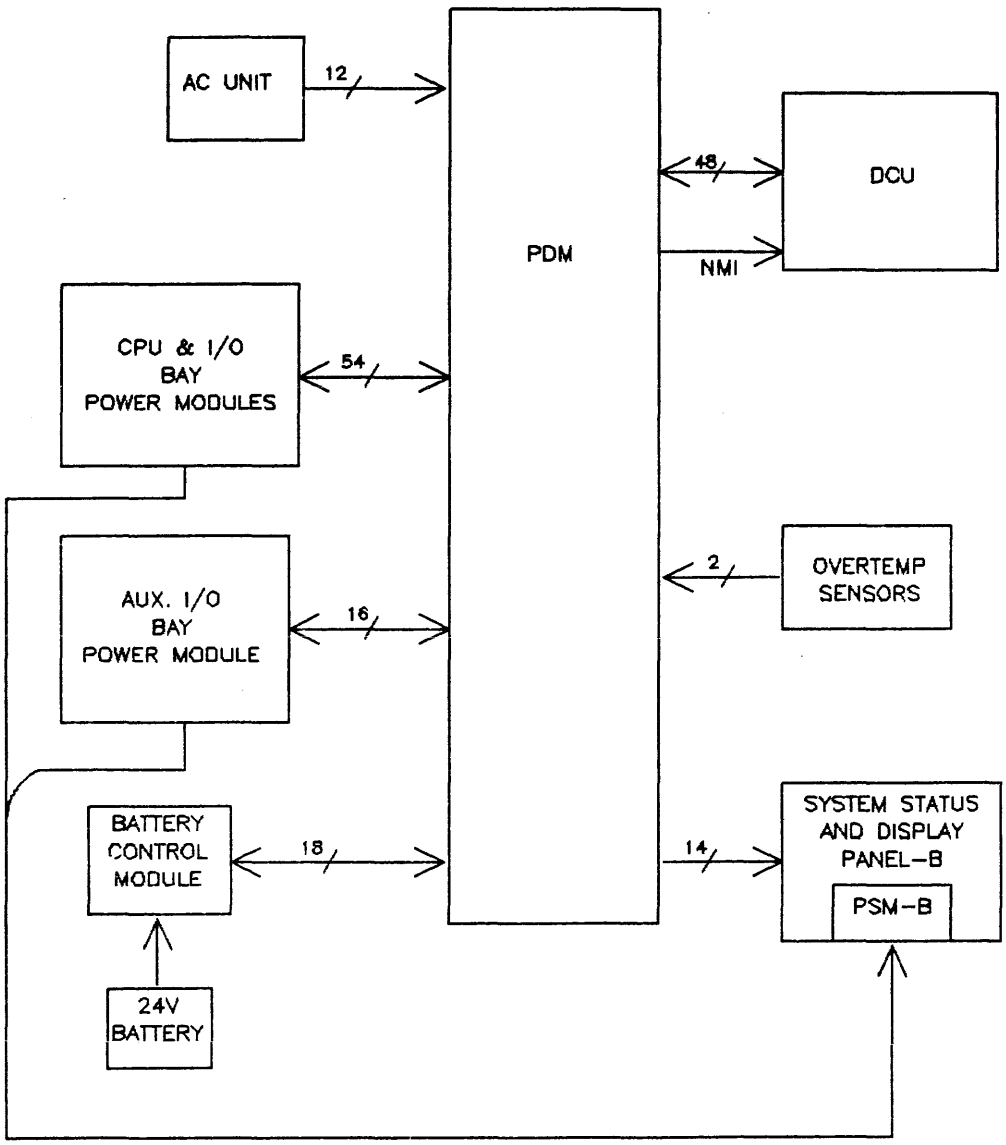


Figure 8-2. Relationship of PDM and Power Modules

The PDM is an independent PCA for most of its functions and requires minimal control from the DCU.

NOTE

The computer will not operate without the PDM installed. Power is distributed from the power modules through the PDM to the individual PCAs in the system. Without the PDM, the PCAs will receive no power. However, the flat cable which connects the DCU to the PDM can be disconnected; if this cable is disconnected, the H LED on the System Status and Display Panel (SSDP-B, where B references the Series 64 32460B) will light to show that the DCU and the PDM are not communicating.

The PDM monitors AC and DC voltages, DC current, and overtemperature conditions, as shown in Figure 8-2. The DCU, at the direction of the PDM, will shut down computer operations if a voltage goes out of tolerance or an overtemperature condition develops. Additionally, the PDM provides Module Status, Battery Status, Battery Voltage, Overtemp, AC Unit failure, System Reset, DCU-to-PDM Status, and Remote signals to the SSDP-B. See Figure 8-3.

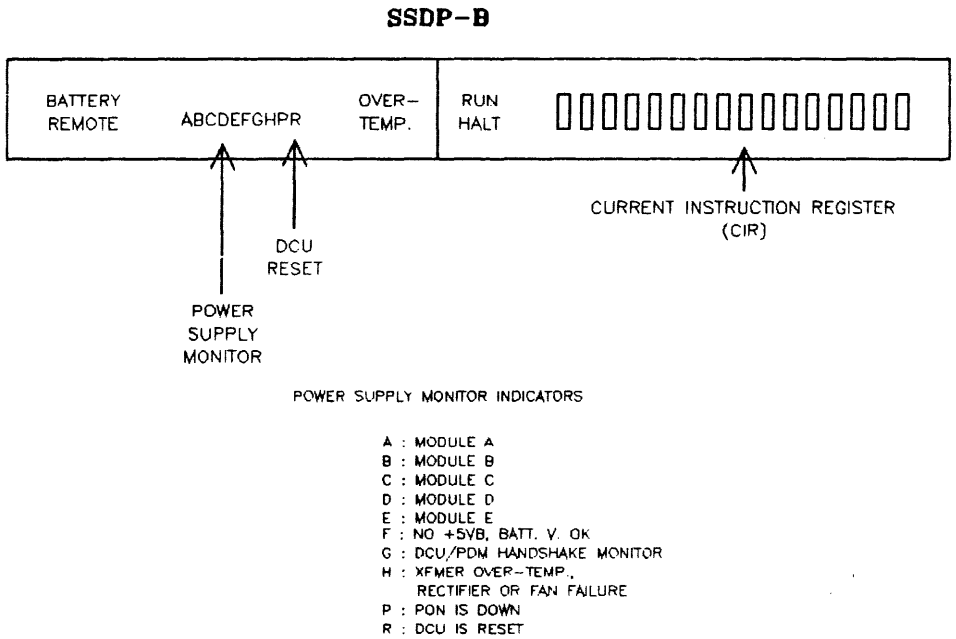


Figure 8-3. System Status and Display Panel (SSDP-B)

8-3. PDM HARDWARE

The PDM consists of 10% analog and 90% digital circuits which monitor AC and DC power for the Series 64 (32460B). The following paragraphs describe the functions of these circuits.

8-4. PDM/DCU Communication Circuits

Commands from the DCU enter the PDM via a four-bit Address Bus, A0-A3. See Figure 8-4. Data is transmitted and received via the eight-bit bidirectional PDM Data Bus.

Interrupts from the PDM to the DCU are transmitted approximately every 1 second via the TMRINT line. This interrupt causes the DCU to trigger a Watch Dog Timer. The Watch Dog Timer expects the DCU and PDM to handshake with every interrupt from the TMRINT line. If this request-response relationship between the DCU and the PDM is broken for more than 10 seconds, the G LED on the SSDP-B lights to show that the PDM and DCU are not communicating.

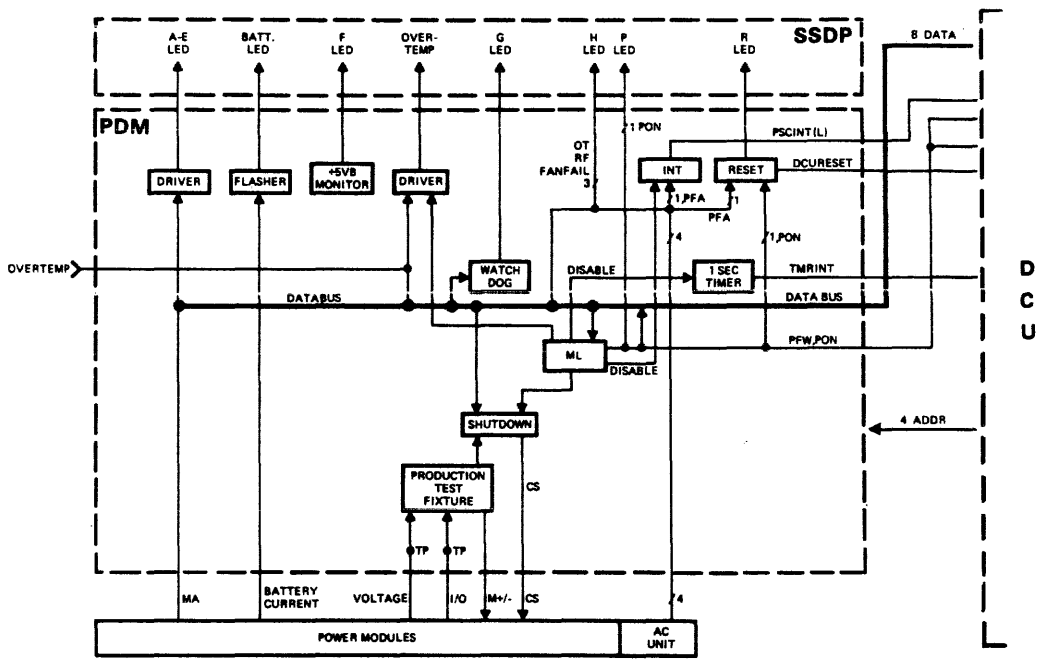


Figure 8-4. PDM Simplified Block Diagram

8-5. DC Monitoring - Module Alarms

Module Alarms (MAs) are the open collector signal sent from each power module to the PDM. When the power modules are functioning normally, the MA signal associated with each module is low. When a failure occurs (overvoltage, undervoltage, or overtemperature) in a power module, the MA associated with the failed module goes high. The PDM detects this alarm, interrupts the DCU, and lights the LED on the SSDP-B that corresponds to the failed module set.

The status of each power module in a module set is represented by 4 LEDs located on each module. These LEDs are configured as follows:

- 0 ON (This green LED is lighted when the module is working properly.
- 0 OV (Overvoltage. This LED is red.)
- 0 UV (Undervoltage. This LED is red.)
- 0 OT (Overtemperature. This LED is red.)

When a failure occurs, an LED on the SSDP-B will point to the failed module set. By looking at the LEDs on the module of the failed module set, the failed module in the module set and the nature of the failure can be easily determined.

8-6. AC Monitoring

The following paragraphs describe the actions of the PDM when there is an AC power failure, a fan failure, or a rectifier failure.

8-7. AC POWER FAILURE. AC power failure monitoring is done by rectifying the three phase AC power to the system and then monitoring this voltage on the high voltage DC input bus. The nominal voltage on this bus is 300 V DC. When the DC voltage on the bus drops below 250 V DC, the PFA (Power Fail Alarm) signal is set high, which causes the NMI (Non-Maskable Interrupt) signal to be sent to the DCU. The DCU enters a power fail service routine to determine the cause of the interrupt.

8-8. FAN FAILURE, RECTIFIER FAILURE, AND RECTIFIER OVERTEMP. When any of these conditions occurs, the associated AC Unit alarm is sent to the PDM. These alarms are called FANFAIL, RFA, and ROT, respectively. The PDM lights the H LED on the SSDP-B to show the failure. When the H LED is lit, the problem could be any of the following:

a. Internal AC Unit breaker tripped. This breaker can trip accidentally when the system is in motion.

b. Fan power to the CPU bay and/or I/O bay is missing. Check AC Unit plugs routing this power. Check transformer plugs to ensure that they are tightly connected.

c. Possible thermal overload within AC Unit. A thermal overload will trip the AC Unit internal breaker. It may be necessary to replace the AC Unit.

d. Rectifier failure generally reflects a malfunctioning AC Unit. It may be necessary to replace it.

8-9. System Power and Overtemperature Failures

The following paragraphs describe the actions of the PDM hardware when a power failure or overtemperature condition occurs. All the LEDs mentioned remain lighted as long as battery backup power is available.

8-10. OVERTEMPERATURE CONDITIONS. Two sets of overtemperature sensors are provided in the top of both the CPU and I/O bays. One set of sensors sends a signal when the exhaust air temperature exceeds 40 degrees C (which is approximately 30 degrees C ambient temperature). The other set of sensors sends a signal when the exhaust air temperature exceeds 50 degrees C (which is approximately 40 degrees C ambient temperature). The DCU reads the status of these overtemperature sensors once per second and takes appropriate actions as follows:

a. When the exhaust air temperature exceeds 40 degrees C, but is less than 50 degrees C, the banner

```
EXHAUST TEMP > 40C
```

flashes on the system console and the console "beeps" every 10 seconds.

b. When the exhaust air temperature exceeds 50 degrees C the banner

```
EXHAUST TEMP > 50C
```

flashes on the system console and the console "beeps" once per second.

The operator has one minute to take action. The choices are to shut down the computer or to ignore the warning.

c. After one minute, the banner

```
OVERTEMP SHUTDOWN
```

flashes on the system console.

e. 15 seconds after the shutdown banner appears on the system console, the HWOTSD (hardware overtemperature shutdown) signal causes a mass shutdown of all power modules except the battery charger and the off battery converter. The OVERTEMP LED on the SSDP-B is lighted just before the overtemperature shutdown is performed. This LED is battery driven and does not turn off when the modules are shut down.

f. The system will not restart until the overtemp switches close and AC power is turned off and back on.

8-11. DC POWER MODULE FAILURE. When a DC power module fails, either during power-on or during normal system operation, the following happens:

a. The failed module sends a module alarm to the PDM.

b. The PDM lights the LED on the SSDP-B that corresponds to the module set of the failed module.

c. The modules DO NOT shut down. The LEDs on the modules will indicate which module has failed and the type of failure that has occurred.

NOTE

When only one module fails in a module set where multiple modules are connected in parallel, it is common for the failed module to have no lighted LEDs and the functioning module to have its undervoltage LED lighted.

8-12. AC LINE FAILURE. When a drop in the AC voltage causes the voltage on the high voltage DC bus to fall below 250 V DC. The following steps are performed by the DCU:

a. Disable the NMI.

b. Wait 300 microseconds, then check the PFA signal.

(1.) If PFA is now low, this is a false alarm due to noise. Enable the NMI and exit the power fail service routine.

(2.) If PFA is high, this is a real power failure. Go to step c below.

c. Set PFW (Power Fail Warning) to low.

d. Wait 5 milliseconds.

e. Set PON (Power-On) to low.

f. Both the DCU and the PDM are reset until power is restored.

8-13. PDM Battery Status/Monitor Circuit

The PDM Battery Status/Monitor Circuit checks the battery charging/discharging current and voltage. An LED on the SSDP-B indicates battery status. This LED flashes quickly when the battery is discharging, flashes slowly when it is charging, and turns off when the battery is fully-charged.

In the battery backup mode, the battery monitor circuit disconnects the battery from the backup supply if the battery discharges to a level of 20.0VDC. This helps prevent the battery from becoming permanently damaged due to excessive discharging.

8-14. PDM CONNECTION REQUIREMENTS

The following paragraphs provide specifications for the physical connections between the PDM and the rest of the computer. Also provided are pin assignments for the external connectors.

8-15. Connections to Power Modules

a. Voltage Sense Lines - 10k Ohm current limiting resistors only.

c. Module Alarm (MA) - This open collector signal from the power modules to the PDM informs the PDM when an overvoltage, undervoltage, or overtemperature has occurred.

d. Converter Shutdown (CS) - This 7 to 10 milliampere signal from the PDM to the power modules shuts down the corresponding module when the signal is applied.

e. Current Output Sense (IO) - This analog signal form the power modules to the PDM indicates the output current of the power module and is scaled to a value of +5V. This signal goes to the current test points and to the production test fixture for the PDM. 10K Ohm internal resistance.

f. Voltage (V) - The voltages in the system are connected to test points and the production test fixture for the PDM through 10K Ohm current limiting resistors.

g. Up-margin/Down Margin (+M/-M) - These TTL signals from the PDM to the power modules are used to change the corresponding voltages Up or Down by 5%. These lines are used exclusively in the production test fixture for the PDM.

8-16. SSDP-B Interface

Signal Name	Current Requirements
BATTERY	-2 mA (OFF), +1 mA (ON)
OVERTEMP	-2 mA (OFF), +2 mA (ON)
CPU RUN/HALT	-2.5 mA (HALT), +1 mA (RUN)
REMOTE	-2 mA (OFF), +1 mA (ON)
A-H,P,R	10 mA each LED (ON)
CIR	400 mA (ON)

8-17. PDM Pin Assignments

J1 -- PDM/DCU Interface (50-pin ribbon cable connector)

Pin	Signal	Pin	Signal
1.	System Ground	26.	System Ground
2.	System Ground	27.	D4
3.	A00	28.	D5
4.	A01	29.	D6
5.	A02	30.	D7
6.	A03	31.	MREQ
7.	A04	32.	System Ground
8.	System Ground	33.	IOREQ
9.	A05	34.	RD
10.	A06	35.	WR
11.	A07	36.	RESET
12.	A08	37.	System Ground
13.	A09	38.	NMI
14.	System Ground	39.	TMRINT
15.	A10	40.	ROMDISAB
16.	A11	41.	PSCENAB
17.	A12	42.	System Ground
18.	A13	43.	DBUSENAB
19.	A14	44.	PON
20.	System Ground	45.	PFW
21.	A15	46.	CPUR/H
22.	D0	47.	System Ground
23.	D1	48.	
24.	D2	49.	System Ground
26.	D3	50.	System Ground

J2 --SSDP-B Interface (16-pin ribbon cable)

Pin	Signal
1.	R LED
2.	A LED
3.	B LED
4.	OVERTEMP
5.	C LED
6.	D LED
7.	E LED
8.	F LED
9.	G LED
10.	H LED
11.	P LED
12.	
13.	
14.	CPURUN/HALT
15.	REMOTE
16.	BATTERY LED

J3 -- Module B and Charger Interface (20 pin connector)

Pin	Signal
1.	Charger Converter Shutdown
2.	Battery Connect
3.	System Ground
4.	System Ground
5.	Charger A
6.	Charger Current Output
7.	Battery Current Monitor
8.	+5VBB Source
9.	+5VBB Source
10.	System Ground
11.	System Ground
12.	System Ground
13.	Battery Connect (RTN)
14.	KEYING PLUG
15.	System Ground
16.	Module B Converter Shutdown
17.	Module B Module Alarm
18.	Module B +5 Current Output
19.	Battery Voltage Monitor
20.	

J4 -- AC Unit Interface (20 pin connector)

Pin	Signal
1.	Low Overtemp Switch
2.	High Overtemp Switch
3.	
4.	
5.	Battery Connect (RTN)
6.	System Ground
7.	System Ground
8.	KEYING PLUG
9.	Battery Connect
10.	Fan Fail
11.	Rectifier Overtemp
12.	Rectifier Failure Alarm
13.	
14.	System Ground
15.	System Ground
16.	
17.	Chassis Ground
18.	System Ground
19.	Power Fail Alarm
20.	

J5 -- SSDP-B Power (4-pin connector)

Pin	Signal
1.	+5V
2.	-5.2V
3.	+5VB
4.	System Ground

J6 -- Module E Interface (20-pin connector)

Pin	Signal
1.	E1 Converter Shutdown
2.	E1 Module Alarm
3.	System Ground
4.	E2 Converter Shutdown
5.	E2 Module Alarm
6.	
7.	
8.	
9.	
10.	E2 +5 Current Output
11.	E1 +5 Current Output
12.	
13.	E +5 Up Margin
14.	E +5 Down Margin
15.	System Ground
16.	System Ground
17.	KEYING PLUG
18.	
19.	System Ground
20.	System Ground

J7 -- Module C Interface (20-pin connector)

Pin	Signal
1.	C1 Converter Shutdown
2.	C1 Module Alarm
3.	System Ground
4.	
5.	
6.	System Ground
7.	KEYING PLUG
8.	C1 -2 Current Output
9.	
10.	
11.	C1 -12 Current Output
12.	C1 +12 Current Output
13.	C -2 Up Margin
14.	C -2 Down Margin
15.	C +12 Up Margin
16.	
17.	C +12 Down Margin
18.	C -12 Up Margin
19.	C -12 Down Margin
20.	System Ground

J8 -- Module A Interface (20-pin connector)

Pin	Signal
1.	A1 Converter Shutdown
2.	A1 Module Alarm
3.	KEYING PLUG
4.	A2 Converter Shutdown
5.	A2 Module Alarm
6.	System Ground
7.	
8.	
9.	
10.	A2 -5.2 Current Output
11.	A1 -5.2 Current Output
12.	
13.	A -5.2 Up Margin
14.	A -5.2 Down Margin
15.	System Ground
16.	System Ground
17.	System Ground
18.	System Ground
19.	System Ground
20.	

J9 -- Module D Interface (20-pin connector)

Pin	Signal
1.	D1 Converter Shutdown
2.	D1 Module Alarm
3.	System Ground
4.	D2 Converter Shutdown
5.	D2 Module Alarm
6.	System Ground
7.	
8.	
9.	System Ground
10.	D2 +5 Current Output
11.	D1 +5 Current Output
12.	
13.	D +5 Up Margin
14.	D +5 Down Margin
15.	System Ground
16.	System Ground
17.	System Ground
18.	System Ground
19.	System Ground
20.	

J10--Production Test Interface (16 pin ribbon cable connector)

Pin	Signal
1.	A1 Shutdown Input
2.	A2 Shutdown Input
3.	
4.	B Shutdown Input
5.	C1 Shutdown Input
6.	C2 Shutdown Input
7.	Charger Shutdown Input
8.	
9.	D1 Shutdown Input
10.	D2 Shutdown Input
11.	
12.	Protected +5VBB
13.	E1 Shutdown Input
14.	E2 Shutdown Input
15.	
16.	+5VB Shutdown Input (Shuts down B and D1 simultaneously)

J11 -- Production Test Interface (50 pin ribbon cable connector)

Pin	Signal
1.	System Ground
2.	System Ground
3.	
4.	D1 +5V Current Output
5.	D2 +5V Current Output
6.	
7.	D +5V Up Margin
8.	D +5V Down Margin
9.	
10.	A1 -5.2 Current Output
11.	A2 -5.2 Current Output
12.	
13.	A -5.2 Up Margin
14.	A -5.2 Down Margin
15.	B -12V
16.	C -2V
17.	B +5V
18.	A -5.2V
19.	C +12V
20.	C -12V
21.	E +5V
22.	B +12V
23.	D +5V
24.	System Ground
25.	System Ground
26.	System Ground
27.	Charger Current Output
28.	B +5 Current Output
29.	Battery Current
30.	C1 -2 Current Output
31.	C1 +12 Current Output
32.	C1 -12 Current Output
33.	
34.	
35.	
36.	
37.	C -2 Up Margin
38.	C -2 Down Margin
39.	C +12 Up Margin
40.	C +12 Down Margin
41.	C -12 Up Margin
42.	C -12 Down Margin
43.	
44.	E1 +5V Current Output
45.	E2 +5V Current Output
46.	
47.	E +5V Up Margin
48.	E +5V Down Margin
49.	
50.	Battery Voltage Sense

J12 -- +12S and -12S (4 pin connector)

Pin	Signal
1.	+12S
2.	System Ground
3.	System Ground
4.	-12S

J13 -- +5B (6 pin connector)

1,2,3,4,5,6	+5VB

J14 -- +12V, -12V, and +5B output (9 pin connector)

Pin	Signal
1,2,3	+5VB
4,7	+12V
6,9	-12V
8	+5V (E Current Output, Aux. I/O)

J15 -- COMMON (3 pin connector)

Pin	Signal
1.	System Ground
2.	System Ground
3.	System Ground

J16 -- +5V, -5.2V, and -2V input (5 pin connector)

Pin	Signal
1.	
2.	
3.	+5V
4.	-5.2V
5.	-2V

J17 -- +12V and -12V input (12 pin connector)

Pin	Signal
1,4,7,10	+5VB
2,5,8,11	System Ground
3,6,9,12	-12V

J18 -- +12V, -12V, and +5B output and AUX I/O voltage monitor input (9 pin connector)

Pin	Signal
1,2,3	+5VB
4,7	+12V
6,9	-12V
8	+5 (E Current Output, AUX I/O Bay)
5	System Ground

8-18. AC AND DC POWER

The remainder of Section VIII describes the AC and DC power components.

8-19. AC Power Physical Characteristics

An AC Unit and three ferro-resonant transformers are the main AC power components. The AC Unit and the transformer are mounted in the bottom of the I/O Bay. Cable harnesses distribute AC power throughout the CPU and I/O Bays.

The power system uses an earth ground/logic common connection which provides isolation between the cabinet metal and logic common.

8-20. AC Input Specifications

The computer works with the following AC input voltages:

Type Service	Maximum Input Current	Where Used
3 Phase,208V,4 wireY+GND	24A/Phase	USA Standard
3 Phase,380V,3 or 4 wireY+GND	13A/Phase	Europe/Icon Std
3 Phase,415V,3 or 4 wireY+GND	12A/Phase	United Kingdom

NOTE

The input A.C. power users a 5 wire WYE to provide co the Series 64 (32460A) and the Series 64 (32460B). How wire into the system connects to a terminal lug and is no hardwired connections outside the U.S. the neutral line The transformers therefore provide a 208V 3 phase delta a 120/208V WYE source.

Input Voltage Tolerance (phase to phase):
+10% to -10%
Input Frequency Tolerance:
50 Hz nominal, 47.5Hz to 52.5Hz (+-5%)
60 Hz nominal, 57Hz to 63Hz (+-5%)
Power Factor:
0.95 typ
Inrush Current:
208V line, 500A peak, 1 cycle duration
380V line, 325A peak, 1 cycle duration
415V line, 300A peak, 1 cycle duration

Line Current: The input A.C. phases are equally balanced due to the ferro-resonant isolation transformers.

8-21. AC Power Functional Characteristics

In the United States, the computer must be connected to 208 VAC, 60-Hz, three-phase power. In other countries, the computer can be connected to 415 VAC-Y, 50-Hz, three-phase, or 380 VAC-Y, 50-Hz, three-phase power.

The AC system consists of an AC Unit and 3 ferro-resonant transformers These transformers generate an approximately square wave voltage, which is then rectified and used as the high voltage DC source. This 300 V DC source is then distributed to all the DC power modules in the system. The AC Unit also acts as the source of 220VAC for all the system fans. Three connector ports, one from each bay (CPU, I/O, and Auxiliary I/O), are available to deliver and supply fan power to each of the bays in the system. The AC Unit has another port which connects to the PDM to detect AC Unit overtemperature, AC Unit rectifier failure, AC Unit power failure, and CPU or I/O bay fan failure.

8-22. DC Power Physical Characteristics

Six power modules and a battery backup pack provide DC power. The DC system also includes a bus bar for the CPU Bay, bus bars for the I/O Bay, and several cable harnesses for power distribution (see figure 8-5).

Each of the modules in a multiple module module set is connected in parallel to produce its assigned voltage and to provide current-sharing.

Heavy duty rigid bus bars are used to distribute the high current sources. Wire harnesses are used to distribute the lower current sources. The distribution for module sets A and C uses a 'U' shaped laminated bus bar approximately 0.3 inches thick, 3 inches deep, and 23 inches wide.

High currents sources are modules A,C, and D (also E, if attached). Module B is a low current source.

8-23. DC Output Specifications

The computer has six DC power modules having the following maximum outputs:

Module Set	Voltage Delivered	Number of Supplies in Set	
A	-5.2 @ 200A	2	
B	+5B @ 30A	1	
C	-2.1V @ 115A +/-12V @ 10A	1	
D	+5.1V @ 200A	2	
E	+5.1V @ 200A	2	(For Auxiliary I/O bay)

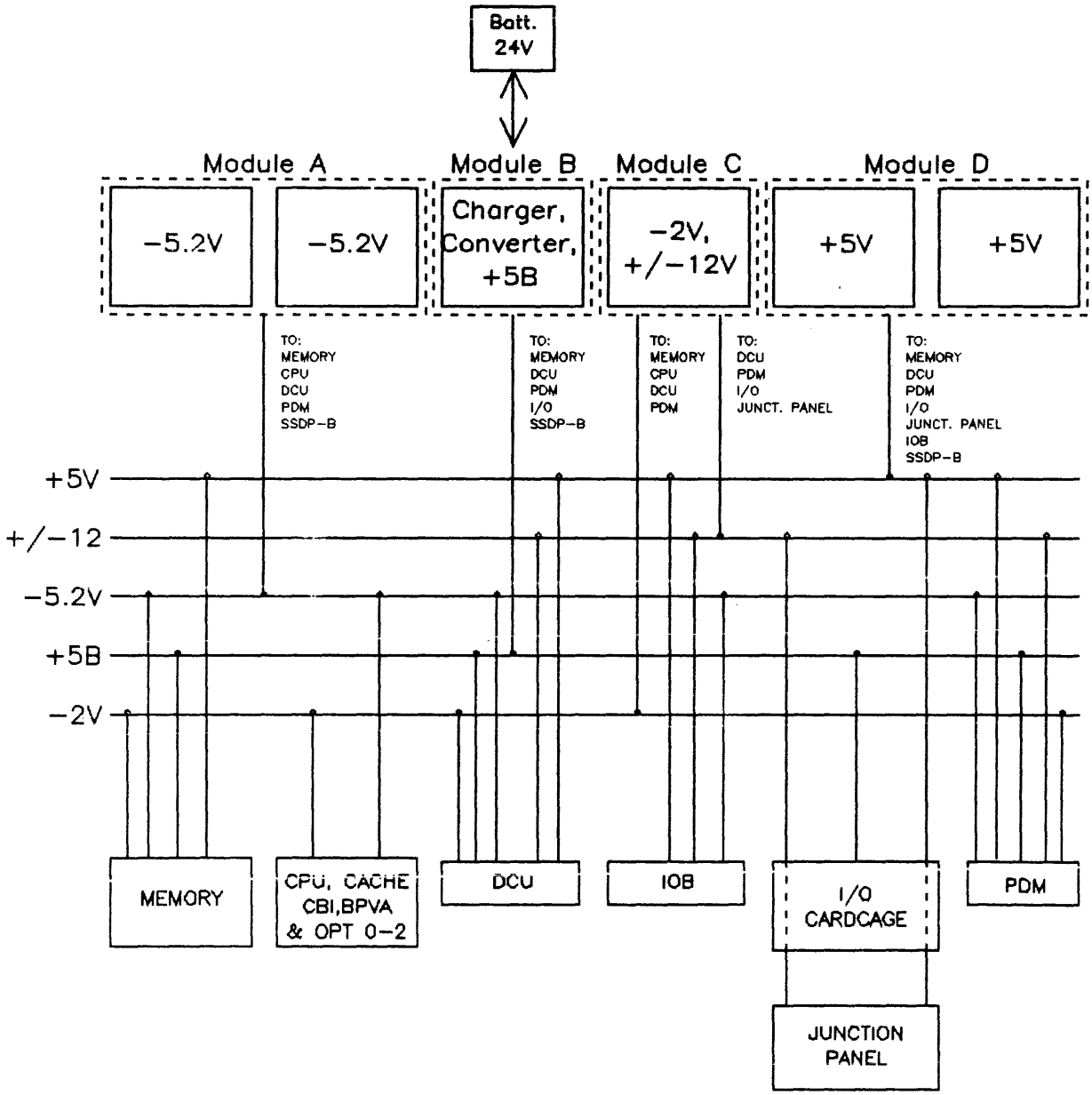


Figure 8-5. DC Power Distribution

8-24. Power Overload Protection

The DC outputs of the power modules are protected against short circuits by current-limiting resistors. These resistors have a value of 10k Ohms.

8-25. Power Module Operating Limits

Module Set of Supply	Nominal Voltage	Operational Limits Lower	Operational Limits Upper	Voltage Latch Off Under	Voltage Latch Off Over
A	-5.225V	-5.175V	-5.275V	-3.8V	-6.8V
B	+4.95V	+4.90V	+5.40V	+3.5V	+6.6V
B	+28.5V	+28.2V	+28.8V	+15V	+33V
C	-2.10V	-2.08V	-2.12V	-1.2V	-2.8V
C	+12.0V	+11.8V	+12.2V	+9.5V	+14.7V
C	-12.0V	-11.8V	-12.2V	-9.5V	-14.7V
D & E	+5.10V	+5.05V	+5.15V	+3.6V	+6.6V

8-26. BATTERY BACKUP

Battery backup supplies +5 volts to the Main Memory arrays when normal AC power is supplied to the computer. It also supplies +5 volts when the AC power is interrupted, the length of time depending on the size of the load. A computer with 8 Mbytes of memory and 20 Intelligent Network Processors (INPs), drawing 22 Amperes, will receive battery backup power for approximately half an hour. A computer with 8 Mbytes of memory but no INPs, drawing 13 Amperes, will receive power approximately 50 minutes.

The battery backup system is one module composed of two interlinked parts. These parts are the battery charger and the off battery converter. The battery charger, under normal AC line conditions, supplies output current over a voltage range of 20VDC to 28.8VDC. The maximum output current of the charger is approximately 15 Amps. A minimum of of 5 Amps is available for charging the 24VDC sealed lead acid battery array. The remaining current is supplied to the off battery converter. When AC power is lost, the battery charger will no longer function and the battery array begins to supply power to the off battery converter.

The off battery converter supplies its nominal 4.95VDC output to the logic circuits which require memory backup. The off battery converter always supplies its output voltage; it does not matter whether its input voltage is supplied from the AC line or by battery power. If the off battery converter is being supplied by battery power, the converter will only maintain its output voltage for a limited time. If the battery discharges to a level of 20.0VDC, the battery will disconnect from the converter until AC power is restored. This helps prevent the battery from becoming permanently damaged due to excessive discharging.

The battery array consists of 12 five-ampere-hour cells and is rated at 24 V nominal.

8-27. DC POWER REQUIREMENTS FOR PCAs

The DC requirements for the Series 64 (32460B) PCAs are the same as for the Series 64 (32460A). See Table 7-3 for details.

GENERAL SPECIFICATIONS

APPENDIX

A

This appendix contains quick-reference specifications for the HP 3000 Series 64 Computer. Included are physical parameters and power requirements, as well as general specifications for the CPU and Cache Memory Module, the Main Memory Module, and the I/O System Module.

A-1. CABINET DIMENSIONS AND WEIGHT

Cabinet dimensions and weight are as follows:

Height: 47.9" (121.8 cm)

Depth: 26.4" (66.9 cm)

Width: 68.8" (174.8 cm)

Weight: 32460A 1100 lbs (500kg)
32460B 1200 lbs (515kg)

A-2. POWER REQUIREMENTS

The power requirements are shown below. For a detailed power description, refer to Sections VII and VIII.

o Series 64 (32460A) AC System

Power Line Input Voltage:

Type Service	Maximum Input Current	Where Used	Recommended
3 Phase, 208V, 4wireY+GND	24A/Phase	USA Standard	Std Option
3 Phase, 380V, 4wireY+GND	13A/Phase	Europe/Icon Std	Option 015
3 Phase, 415V, 4wireY+GND	12A/Phase	Europe/Icon Std	Option 016

Input Tolerance: +6%, -10%
Power Factor: .6 to .8 typical
Inrush Current: 150 A per phase peak Isolation Transformers: 3 at 5KVA each

o Series 64 (32460A) DC System

Seven modular power supplies having the following maximum output capabilities:

+12 V at 10 A (to CPU and I/O)
+5 V at 200 A (to I/O)
+5 V at 50 A (to Main Memory)
+5 V at 25 A (battery backup)
-2.1 V at 115 A (to CPU and Main Memory)
-5.2 V at 200 A (to CPU and Main Memory)
-12 V at 10 A (to CPU and I/O)

o Series 64 (32460B) AC System

Type Service	Maximum Input Current	Where Used
3 Phase,208V,4 wireY+GND	24A/Phase	USA Standard
3 Phase,380V,3 or 4 wireY+GND	13A/Phase	Europe/Icon Std
3 Phase,415V,3 or 4 wireY+GND	12A/Phase	United Kingdom

NOTE

The input A.C. power users a 5 wire WYE to provide co
the Series 64 (32460A) and the Series 64 (32460B). How
wire into the system connects to a terminal lug and is no
hardwired connections outside the U.S. the neutral line
The transformers therefore provide a 208V 3 phase delta
a 120/208V WYE source.

Input Voltage Tolerance (phase to phase):
+10% to -10%
Input Frequency Tolerance:
50 Hz nominal, 47.5Hz to 52.5Hz (+-5%)
60 Hz nominal, 57Hz to 63Hz (+-5%)
Power Factor:
0.95 typ
Inrush Current:
208V line, 500A peak, 1 cycle duration
380V line, 325A peak, 1 cycle duration
415V line, 300A peak, 1 cycle duration

Line Current: The input A.C. phases are equally balanced due to the ferro-resonant isolation transformers.

o Series 64 (32460B) DC System

Six power modules arranged in four module sets have the following maximum output capabilities:

Module Set	Voltage Delivered	Number of Modules in Set	
A	-5.2 @ 200A	2	
B	+5B @ 30A	1	
C	-2.1V @ 115A +/-12V @ 10A	1	
D	+5.1V @ 200A	2	
E	+5.1V @ 200A	2	(For Auxiliary I/O bay)

A-3. ENVIRONMENTAL

Environmental specifications are as follows:

o TEMPERATURE	
Non-operating	-40C to 75C
Operating	20C to 25.5C (68 to 78 degrees F)
Maximum/ Minimum	30 degrees C (86 degrees F)/15 degrees C (55.4 degrees F)
o HUMIDITY	
Recommended Operating	40 to 60%, no condensation
Maximum/ Minimum	80%/30% (48 hrs maximum)
o VIBRATION/SHOCK	
Unpacked	5-55Hz, 19mm P-P
Packaged	1 g Repetitive Impact
Shock, unpackaged	30 g, 1 ims

A-4. CPU AND CACHE MEMORY MODULE

The CPU and Cache Memory Module specifications are listed below. For a detailed description of the module, refer to Section III.

o CPU

ECL technology
75 ns micro-instruction cycle time
Dual 16-bit ALU architecture
16-bit Bank Register for addressing up to 4 Gbytes of memory
8-kbyte Writable Control Store

o CACHE MEMORY

8-kbyte RAM, two-set associative
145-ns effective memory access time
95 percent estimated hit ratio

A-5. MAIN MEMORY MODULE

Main Memory Module specifications are listed below. For a detailed description of the module, refer to Section IV.

- o Fault tolerance: single-bit error detection and correction; double-bit detection.
- o 1 Mbyte per Main Memory Array (MMA) PCA; two MMAs minimum recommended configuration; eight MMAs maximum.
- o Error log: four 1 kbyte X 1 static RAMs.
- o Memory contents retained during power fail for minimum of 15 minutes, depending on total computer load.
- o Maximum 17.8 Mbytes per second transfer rate; block data transfers requiring 12 system clocks (900 ns).

A-6. I/O SYSTEM MODULE

I/O System Module characteristics are listed below. For a detailed description of the module, refer to Section V.

- o Uses Intermodule Bus I/O Adapter (IMB IOA) to interface I/O channels with other Central System Bus (CSB) modules.
- o Capability of up to two IMBs.
- o Uses most HP-IB-compatible peripherals.

A-7. HARDWARE CONFIGURATION

The hardware is contained in two interconnected bays: a CPU Bay and an I/O Bay. The CPU Bay houses all PCAs for the CPU, Cache Memory, Main Memory, Diagnostic Control Unit, Common Bus Interfaces, and the I/O Buffer PCA. It also holds DC power supplies and cooling fans.

The I/O Bay houses the Intermodule Bus Interface PCA, and PCAs for the General I/O Channels, Advanced Terminal Processor, Intelligent Network Processor, and the interface PCAs for peripheral devices. It also holds DC power supplies, cooling fans, and a multi-purpose junction panel for the peripherals.

Figure A-1 shows the general arrangement of the bays and card cages for the Series 64. Figure A-2 show the same view for the Series 64B. Figure A-3 shows the PCA slot assignments in the CPU Bay for both systems.

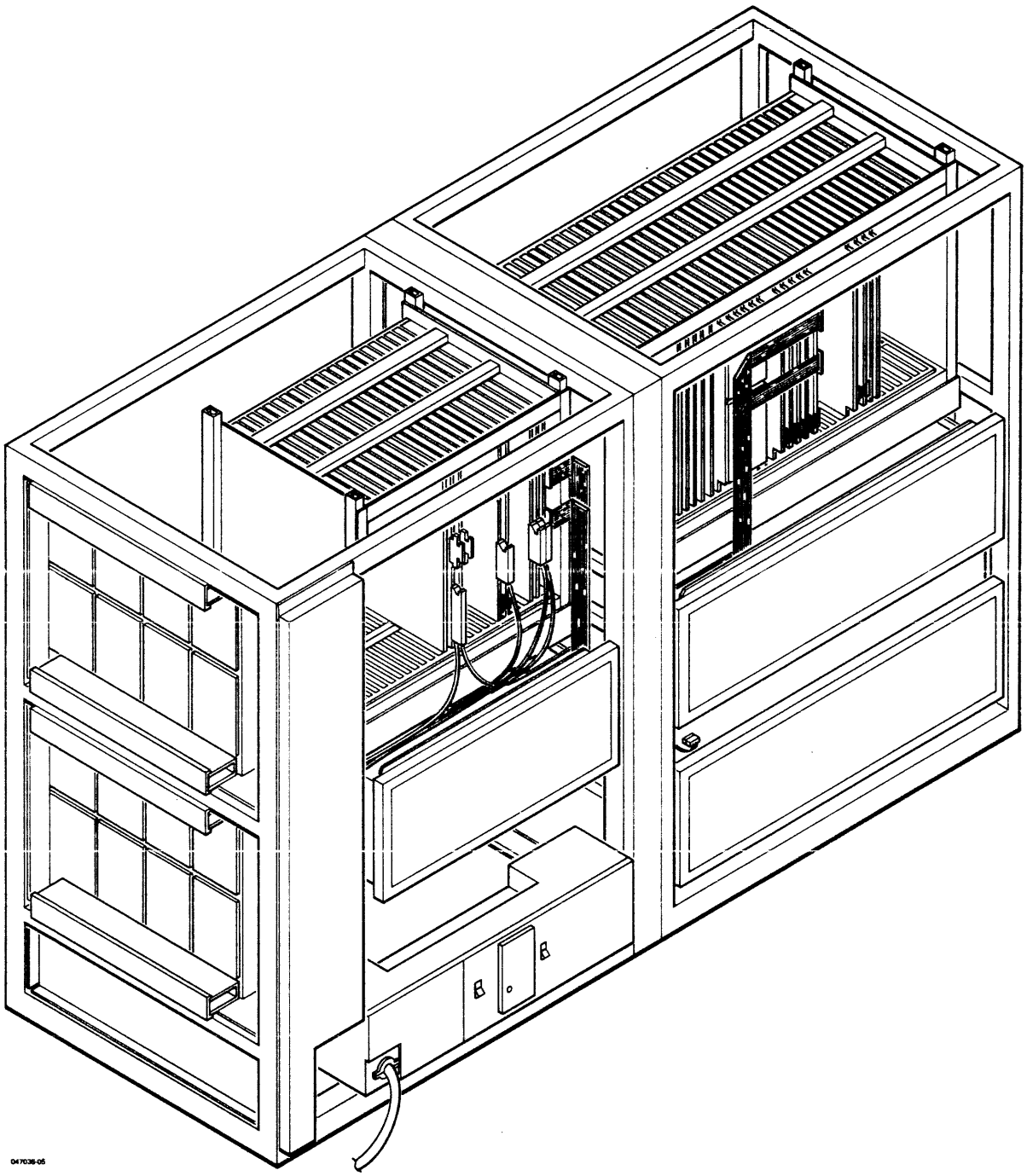


Figure A-1. CPU and I/O Bays, Series 64 (32460A), Rear View

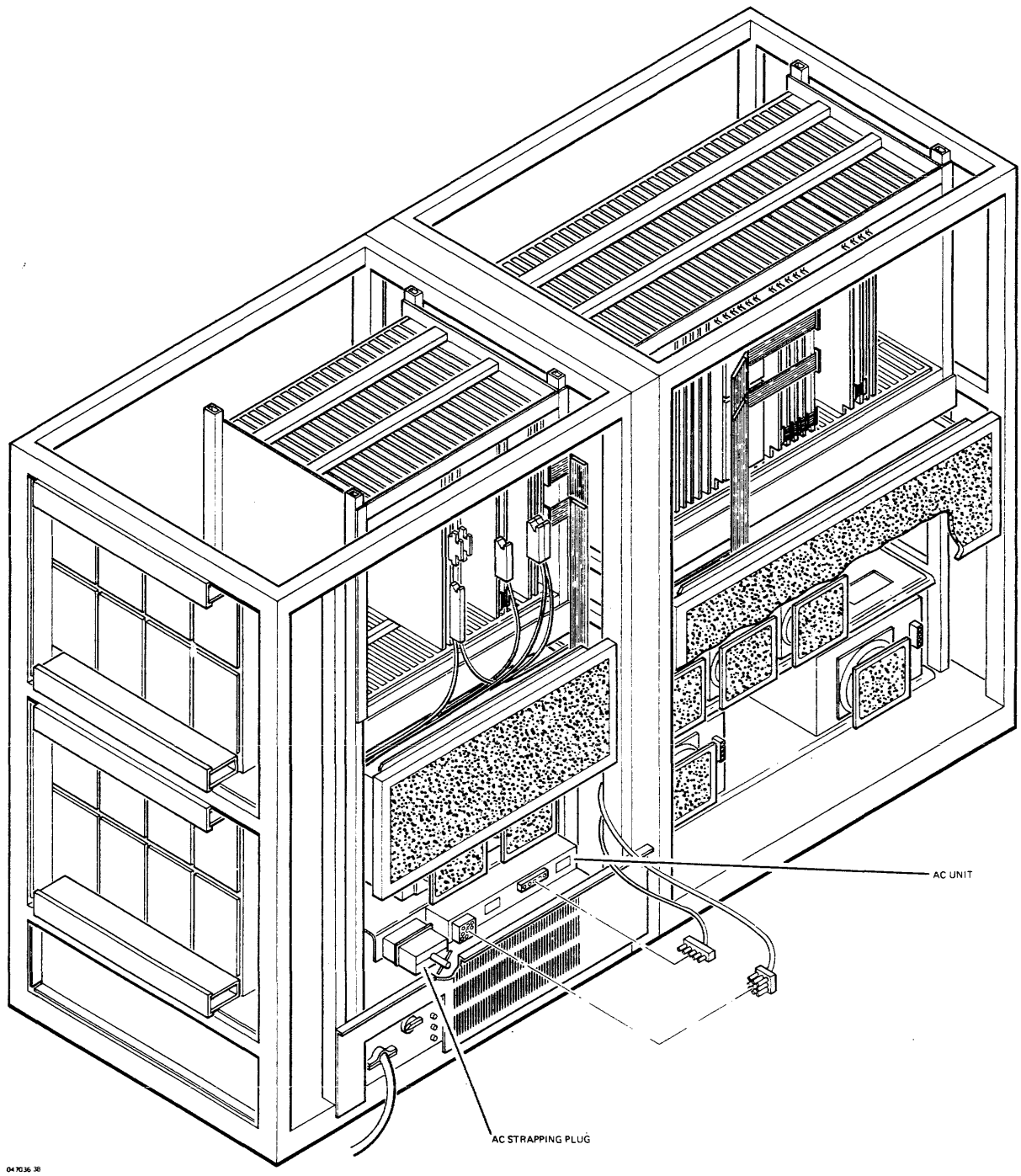


Figure A-2. CPU and I/O Bays, Series 64 (32460B), Rear View

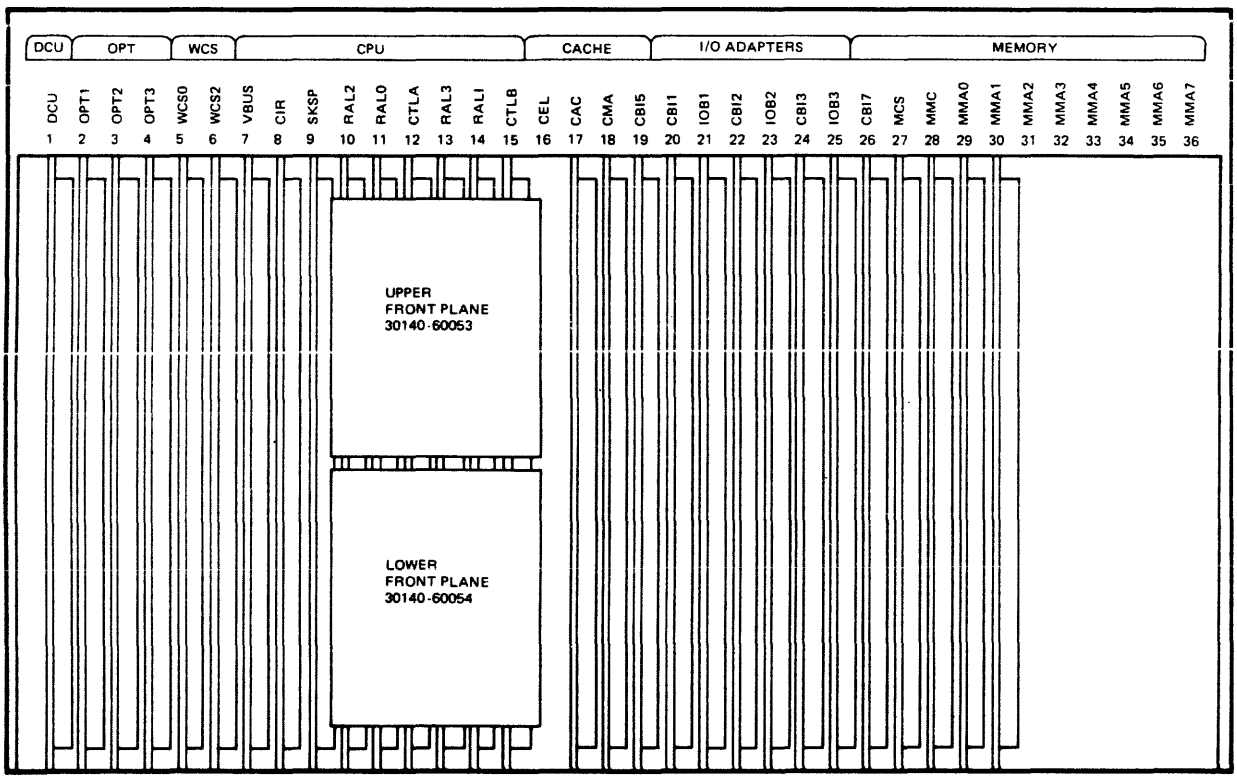


Figure A-3. Slot Assignments in CPU Bay (both systems)

GLOSSARY OF ABBREVIATIONS AND ACRONYMS

APPENDIX B

This Appendix is a glossary of abbreviations and acronyms used in this manual. It also includes "buzz words" such as "hit ratio" which are used in a specialized sense in connection with the HP 3000 Series 64 Computer. Each abbreviation, acronym, or buzz word is defined; in some cases, a short description of the word and/or its use is included.

The list is intended as a quick-reference. Each expression is also defined the first time it is used in each section of the manual. The list does not include abbreviations such as CPU, ALU, I/O, PCA, AC, or DC, which are believed to be commonly understood, but it does include such words as Word and Block, to remind the reader of their exact meanings.

ATP: Advanced Terminal Processor

BCM: Battery Control Module (a PCA)

Block: Eight 16-bit Words, or four 32-bit Double Words

Byte: Eight bits (half a Word)

CAB: Cache Address Bus

CAC: Cache Array Controller (a PCA)

Cache Memory: A major part of the CPU and Cache Memory Module. It is a small (8 k-byte), high-speed memory, used to speed-up CPU access to data by eliminating the CPU need to constantly send requests to Main Memory.

NOTE

The Input/Output Buffer (IOB) PCA has a Cache Memory of its own. When reading in Section V about the IOB PCA, take care not to confuse its Cache with the Cache of the CPU and Cache Memory Module.

CAM: Content Addressable Memory. A CAM does the opposite job of the more commonly-known Random Access Memory (RAM). When data is input to a CAM, it outputs the address where the data is stored.

CBI: Common Bus Interface (a PCA)

CIB: Common Interface Bus (do not confuse with CBI)

Clean: A block of data that has not been modified since it was copied into cache from main memory is said to be clean.

CMA: Cache Memory Array (a PCA)

CSAR: Control Store Address Register

CSB:	Central System Bus
CSOR:	Control Store Output Register
DCU:	Diagnostic Control Unit (a PCA)
Dirty:	A block of data which contains any word that has been modified since it was copied into cache from main memory is said to be dirty.
DMA:	Direct Memory Access
Double Word:	Two 16-bit Words
DUS:	Diagnostic Utility System
ECL:	Emitter-Coupled Logic
FLD:	Fault-Locating Diagnostic
GIC:	General I/O Channel (a PCA)
Hit Ratio:	The percentage of time(s) the information the CPU wants is immediately available in the Cache Memory.
ICB:	Intra-Cache Bus
IMB:	Intermodule Bus
IMB IOA:	Intermodule Bus I/O Adapter. The IMB IOA includes an I/O Buffer (IOB) PCA, an Intermodule Bus Interface (IMBI) PCA, and a Common Bus Interface (CBI) PCA.
IMBI:	Intermodule Bus Interface (a PCA)
IOB:	I/O Buffer (a PCA)
KHD:	Kernel Hardware Diagnostic
(L):	When a signal name is followed by "(L)", the signal is active (true) when Low.
MA:	Module Alarm
MCS:	Memory Correction and Storage (a PCA)
MMA:	Main Memory Array (a PCA)
MMC:	Main Memory Controller (a PCA)
PCM:	Power Control Module (a PCA)
PDB:	Processor Data Bus
PSC:	Power System Controller (a PCA)

SSDP:	System Status and Display Panel
WCS:	Writable Control Store
Word:	Sixteen bits (two Bytes)