DEC 7000/10000 AXP VAX 7000/10000 Platform Technical Manual

Order Number EK—7000A—TM.001

This manual is a reference for Digital service engineers. It describes the platform architecture, the LSB bus, and the power, cabinet control, and cooling systems of the DEC 7000 AXP, DEC 10000 AXP, VAX 7000, and VAX 10000 systems.

digital equipment corporation
maynard, massachusetts

First Printing, August 1993

Alpha AXP	DECUS	VAXBI
AXP	DWMVA	VAXELN
DEC	OpenVMS	VMScluster
DECchip	ULTRIX	XMI
DEC LANcontroller	UNIBUS	The AXP logo
DECnet	VAX	digital

Contents

Chapter 3 Power, Cabinet Control, and Cooling Systems

Preface

Intended Audience

This manual is a reference for Digital service engineers. It describes the platform architecture, the LSB bus, and the power, cabinet control, and cooling systems of the DEC 7000 AXP, DEC 10000 AXP, VAX 7000, and VAX 10000 systems.

Document Structure

This manual has three chapters and two appendixes, as follows:

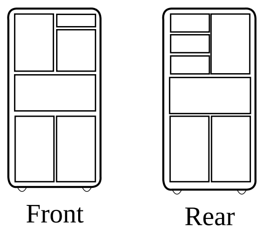
- **Chapter 1, Platform Overview,** gives you a brief description of the system platform.
- **Chapter 2, LSB Bus,** provides information on the system bus.
- **Chapter 3, Power, Cabinet Control, and Cooling Systems,** describes these platform systems.
- The **Appendixes** give in- depth information on topics covered in the manual. Appendix A describes the cabinet control system cables. Appendix B provides information on system power requirements.

Conventions Used in This Document

Terminology. Unless specified otherwise, the use of "system" refers to either a DEC 7000 AXP or VAX 7000 system. The DEC 7000 AXP systems use the Alpha AXP architecture. References in text use DEC 7000 to refer to DEC 7000 AXP systems.

Book titles. In text, if a book is cited without a product name, that book is part of the hardware documentation. It is listed in Table 2 along with its order number.

Icons. The icons shown below are used in illustrations for designating part placement in the system described. A shaded area in the icon shows the location of the component or part being discussed.



Addresses. All addresses in this document are specified in hexadecimal values, unless otherwise indicated.

Coding conventions. The LSB description in Chapter 2 uses code fragments to describe the operation of several aspects of the LSB architecture. The following conventions are observed:

- The letters B and C are used to represent arbitrary cycles in time. The text accompanying the code typically indicates whether the cycle is arbitrary or of a specific type (for example, a command cycle).
- For scalar signals, bit subscripts are used to indicate the value of the signal in other cycles relative to the reference cycle. For example, STALL<C- 2> refers to the value of the STALL signal two cycles before cycle C.
- For vector signals, word subscripts are used to indicate the value of the signal in other cycles relative to the reference cycle. For example, REQ[C- 2]<5:0> refers to the value of REQ<5:0> two cycles before cycle C.

Table 1 lists the operators used in the code fragments along with their meanings.

Table 1 Operators Used in Code Fragments

Operator	Result
AND	Bitwise AND
AND (unary)	AND of all bits in operand
EQL	Equal —1 if operands are equal
GEQ	Greater- than- or- equal —1 if operand 1 is greater than or equal to operand 2
GTR	Greater- than —1 if operand 1 is greater than operand 2
NOT (unary)	Bitwise inverse
OR	Bitwise OR
OR (unary)	OR of all bits in operand
SL0	Shift left and fill with zeros —shift operand 1 left by the number of bits specified by operand 2 and fill with zeros
SL1	Shift left and fill with ones —shift operand 1 left by the number of bits specified by operand 2 and fill with ones
SR0	Shift right and fill with zeros —shift operand 1 right by the number of bits specified by operand 2 and fill with zeros
SR1	Shift right and fill with ones —shift operand 1 right by the number of bits specified by operand 2 and fill with ones
SXT	Sign extend —extend by replicating the most significant bit
XOR	Bitwise exclusive OR
&	Concatenate —operand 1 is the most significant bits; operand 2 is the least significant bits
#	Indicates base of operand; for example, 0001#2 is a binary operand
<>	Encloses bit subscripts
[]	Encloses word subscript

Documentation Titles

Table 2 lists the books in the DEC 7000/10000 and VAX 7000/10000 documentation sets. Table 3 lists other documents that you may find useful.

Table 2 DEC 7000/10000 and VAX 7000/10000 Documentation

Title	7000 Systems Order Number	10000 Systems Order Number
Installation Kit	EK–7000B–DK	EK–1000B–DK
Site Preparation Guide	EK–7000B–SP	EK–1000B–SP
Installation Guide	EK–700EB–IN	EK–100EB–IN
Hardware User Information Kit	EK–7001B–DK	EK–1001B–DK
Operations Manual	EK–7000B–OP	EK–1000B–OP
Basic Troubleshooting	EK–7000B–TS	EK–1000B–TS
Service Information Kit—VAX 7000	EK–7002A–DK	EK–1002A–DK
Platform Service Manual	EK–7000A–SV	EK–1000A–SV
System Service Manual	EK–7002A–SV	EK–1002A–SV
Pocket Service Guide	EK–7000A–PG	EK–1000A–PG
Advanced Troubleshooting	EK–7001A–TS	EK–1001A–TS
Service Information Kit—DEC 7000	EK–7002B–DK	EK–1002B–DK
Platform Service Manual	EK–7000A–SV	EK–1000A–SV
System Service Manual	EK–7002B–SV	EK–1002B–SV
Pocket Service Guide	EK–7700A–PG	EK–1100A–PG
Advanced Troubleshooting	EK–7701A–TS	EK–1101A–TS
Reference Manuals		
Console Reference Manual	EK–70C0B–TM	
KA7AA CPU Technical Manual	EK–KA7AA–TM	
KN7AA CPU Technical Manual	EK–KN7AA–TM	
MS7AA Memory Technical Manual	EK–MS7AA–TM	
I/O System Technical Manual	EK–70I0A–TM	
Platform Technical Manual	EK–7000A–TM	
Upgrade Manuals		
KA7AA CPU Installation Card	EK–KA7AA–IN	
KN7AA CPU Installation Card	EK–KN7AA–IN	
MS7AA Memory Installation Card	EK–MS7AA–IN	
KZMSA Adapter Installation Guide	EK–KXMSX–IN	
DWLAA Futurebus+ PIU Installation Guide	EK–DWLAA–IN	
DWLMA XMI PIU Installation Guide	EK–DWLMA–IN	
DWMBB VAXBI Installation Guide	EK–DWMBB–IN	
H7237 Battery PIU Installation Guide	EK–H7237–IN	
H7263 Power Regulator Installation Card	EK–H7263–IN	
BA654 DSSI Disk PIU Installation Guide	EK–BA654–IN	
BA655 SCSI Disk and Tape PIU Installation Guide	EK–BA655–IN	
Removable Media Installation Guide	EK–TFRRD–IN	

Table 3 Related Documents

Title	Order Number
General Site Preparation	
Site Environmental Preparation Guide	EK–CSEPG–MA
System I/O Options	
BA350 Modular Storage Shelf Subsystem Configuration Guide	EK–BA350–CG
BA350 Modular Storage Shelf Subsystem User's Guide	EK–BA350–UG
BA350-LA Modular Storage Shelf User's Guide	EK–350LA–UG
CIXCD Interface User Guide	EK–CIXCD–UG
DEC FDDIcontroller 400 Installation/Problem Solving	EK–DEMFA–IP
DEC LANcontroller 400 Installation Guide	EK–DEMNA–IN
DEC LANcontroller 400 Technical Manual	EK–DEMNA–TM
DSSI VAXcluster Installation and Troubleshooting Manual	EK–410AA–MG
InfoServer 150 Installation and Owner's Guide	EK–INFSV–OM
KDM70 Controller User Guide	EK–KDM70–UG
KFMSA Module Installation and User Manual	EK–KFMSA–IM
KFMSA Module Service Guide	EK–KFMSA–SV
RRD42 Disc Drive Owner's Manual	EK–RRD42–OM
RF Series Integrated Storage Element User Guide	EK–RF72D–UG
TF85 Cartridge Tape Subsystem Owner's Manual	EK–OTF85–OM
TLZ06 Cassette Tape Drive Owner's Manual	EK–TLZ06–OM
Operating System Manuals	
Alpha Architecture Reference Manual	EY–L520E–DP
DEC OSF/1 Guide to System Administration	AA–PJU7A–TE
DECnet for OpenVMS Network Management Utilities	AA–PQYAA–TK
Guide to Installing DEC OSF/1	AA–PS2DA–TE
OpenVMS Alpha Version 1.5 Upgrade and Installation Manual	AA–PQYSB–TE
VMS Upgrade and Installation Supplement: VAX 7000–600 and VAX 10000–600 Series	AA–PRAHA–TE
VMS Network Control Program Manual	AA–LA50A–TE
VMSclusters and Networking	
HSC Installation Manual	EK–HSCMN–IN
SC008 Star Coupler User's Guide	EK–SC008–UG
VAX Volume Shadowing Manual	AA–PBTVA–TE
Peripherals	
Installing and Using the VT420 Video Terminal	EK–VT420–UG
LA75 Companion Printer Installation and User Guide	EK–LA75X–UG

Chapter 1

Platform Overview

This chapter provides an overview of the power and packaging for DEC 7000/10000 and VAX 7000/10000 systems. The following sections are included in this chapter:

- DEC 7000/VAX 7000 Cabinets
- DEC 10000/VAX 10000 Cabinets
- Cabinet Components
- LSB Card Cage
- Power, Cabinet Control, and Cooling Systems
- Plug- In Units

1.1 DEC 7000/VAX 7000 Cabinets

The DEC 7000/VAX 7000 cabinet variants are listed in Table 1- 1.

Table 1-1 DEC 7000/VAX 7000 Cabinets

Number	Description
H9F00–AA	System cabinet, 120/208 V, 60 Hz
H9F00–AB	System cabinet, 380–415 V, 50 Hz
H9F00–AC	System cabinet, 202 V, 50–60 Hz
H9F00–BA	Expander cabinet, 120/208 V, 60 Hz
H9F00–BB	Expander cabinet, 380–415 V, 50 Hz
H9F00–BC	Expander cabinet, 202 V, 50–60 Hz

The system cabinet contains an LSB card cage, power system, cabinet control system, cooling system, and plug- in units. The expander cabinet contains the same components less the LSB card cage. A DEC 7000/VAX 7000 system can consist of a system cabinet alone or a system cabinet with one or two expander cabinets.

1.2 DEC 10000/VAX 10000 Cabinets

The DEC 10000/VAX 10000 cabinet variants are listed in Table 1- 2.

Table 1-2 DEC 10000/VAX 10000 Cabinets

Number	Description
H9F00–CA	System cabinet, 120/208 V, 60 Hz
H9F00–CB	System cabinet, 380–415 V, 50 Hz
H9F00–CC	System cabinet, 202 V, 50–60 Hz
H9F00–DA	Expander cabinet, 120/208 V, 60 Hz
H9F00–DB	Expander cabinet, 380–415 V, 50 Hz
H9F00–DC	Expander cabinet, 202 V, 50–60 Hz
H9F00–AE	Battery cabinet, left
H9F00–AF	Battery cabinet, right

The system cabinet contains an LSB card cage, power system, cabinet control system, cooling system, and battery plug- in units. The expander cabinet contains a power system, cabinet control system, cooling system, I/O plug- in units, and disk plug- in units. The battery cabinet contains battery plug- in units for the expander cabinet. A minimum DEC 10000/VAX 10000 system consists of the system cabinet, one expander cabinet, and one battery cabinet. A second expander cabinet and battery cabinet can be added.

1.3 Cabinet Components

Table 1- 3 lists the cabinet components that are described in this manual.

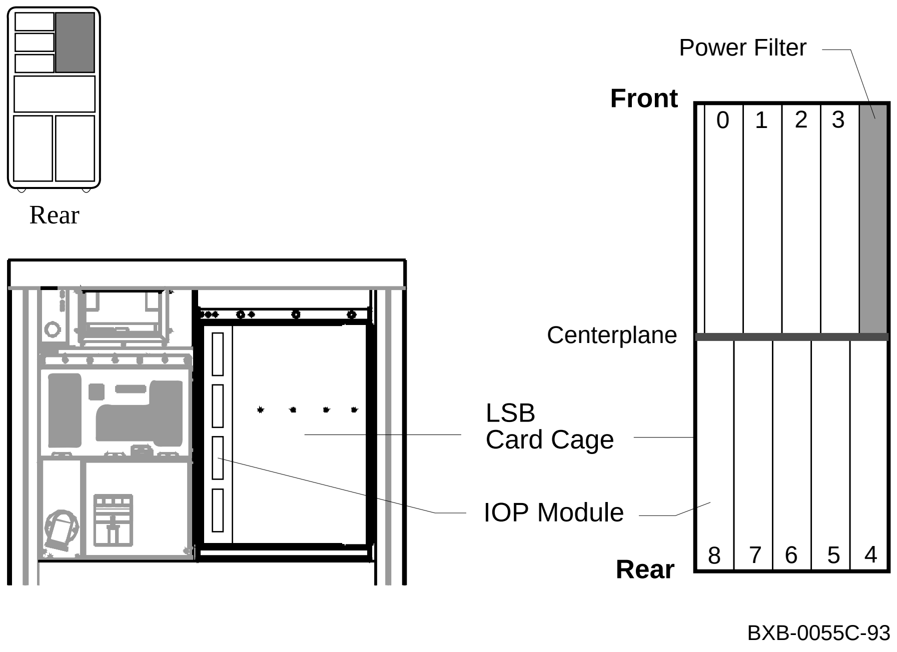
Table 1- 3 Cabinet Components

Component	Description
LSB	System bus and centerplane enclosure in which the processor, memory, and I/O port modules reside.
DC distribution box	Connects the AC service line, 48V DC power regulators, uninterruptible power supply batteries, and the cabinet control logic module.
Power regulators	Generate 48V DC from the AC line or from a 48V battery power source.
Plug- in units	Contain I/O buses, disk and tape storage, or batteries.
Control panel	Provides indicators and switches for controlling and monitoring cabinet operation. It also provides the cable connection for the console device. System cabinet only.
Cabinet control logic	Contains the logic to operate the power and cooling systems. Each system and expander cabinet has a cabinet control logic (CCL) module.
Blower	Cooling airflow for the cabinet.

1.4 LSB Card Cage

The LSB card cage, shown in Figure 1- 1, has nine slots grouped around a centerplane. One slot is dedicated to the IOP module; the other eight slots hold processor, memory, or filler modules. The LSB bus is described in Chapter 2.

Figure 1- 1 LSB Card Cage



1.5 Power, Cabinet Control, and Cooling Systems

The power, cabinet control, and cooling systems are identical in the system and expander cabinets. The power system consists of an AC input box, a DC distribution box, one or more power regulators, the cabinet control logic (CCL) module, and batteries (optional in DEC 7000/VAX 7000 systems; included in DEC 10000/VAX 10000 systems). The power system is shown in Figures 1- 2 and 1- 3. The cooling system consists of a blower and the cabinet control logic module. The blower is shown in Figure 1- 4. The power, cabinet control, and cooling systems are described in Chapter 3.

Figure 1- 2 Power System — DEC 7000/VAX 7000 Systems

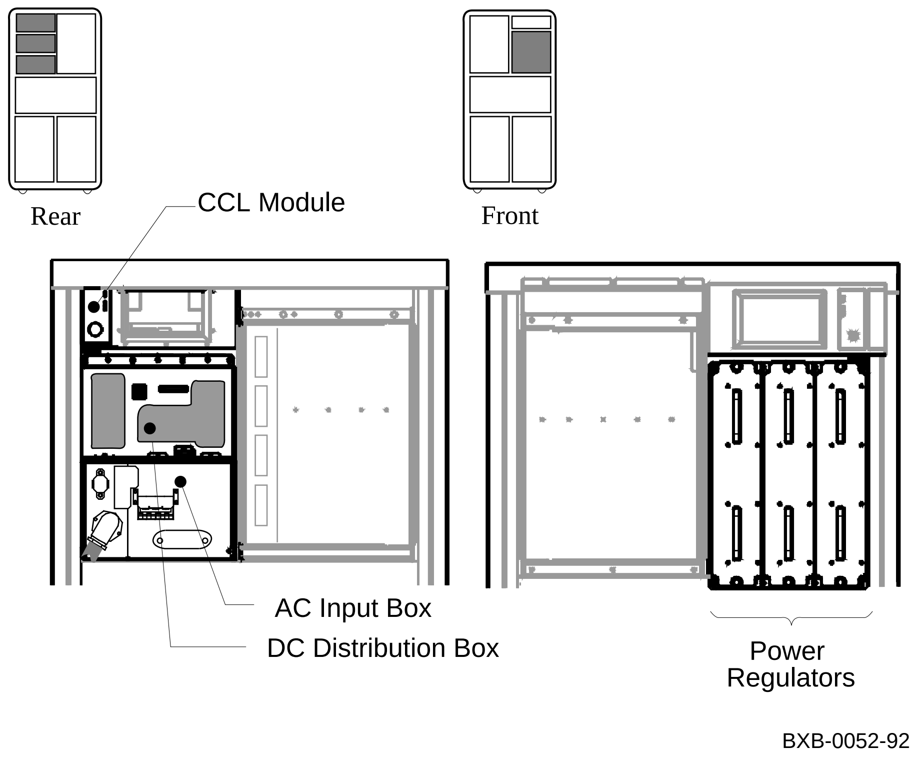


Figure 1-3 Power System — DEC 10000/VAX 10000 Systems

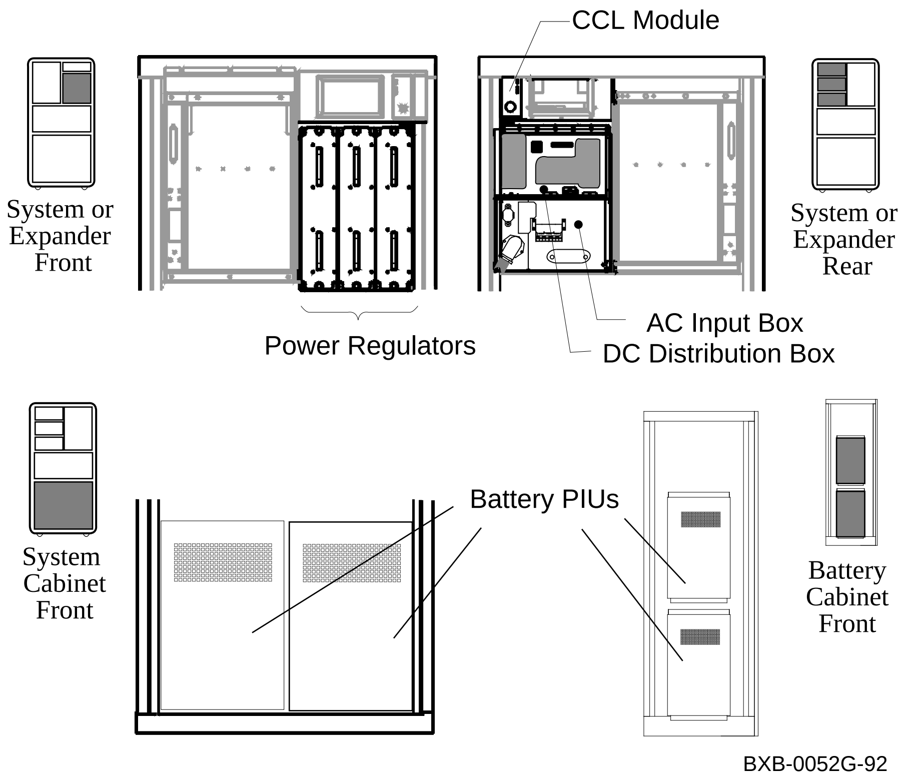
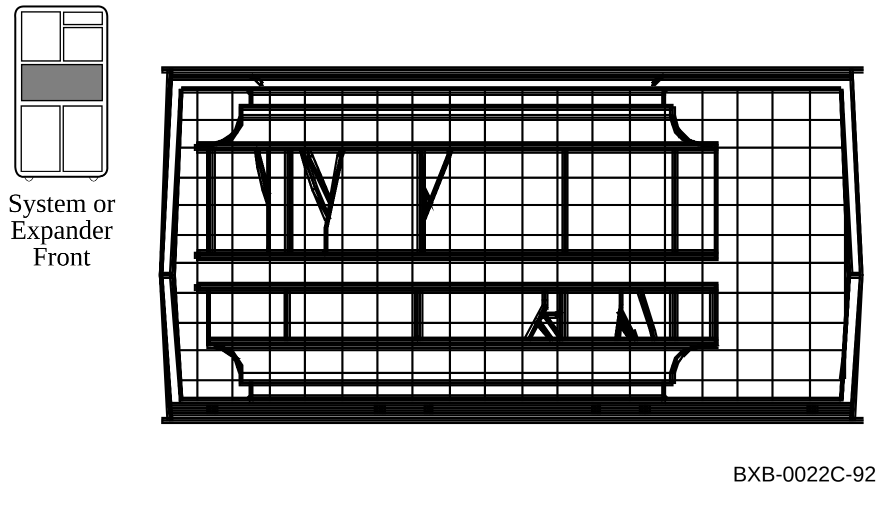


Figure 1-4 Cooling System Blower



1.6 Plug- In Units

I/O buses, disks, and batteries are housed in the system and expander cabinets in plug- in units (PIUs). PIU quadrants are designated for specific types of PIUs; the location of these quadrants is shown in Figure 1- 5. The PIUs that may be located in each quadrant are listed in Table 1- 4.

Figure 1- 5 PIU Quadrants

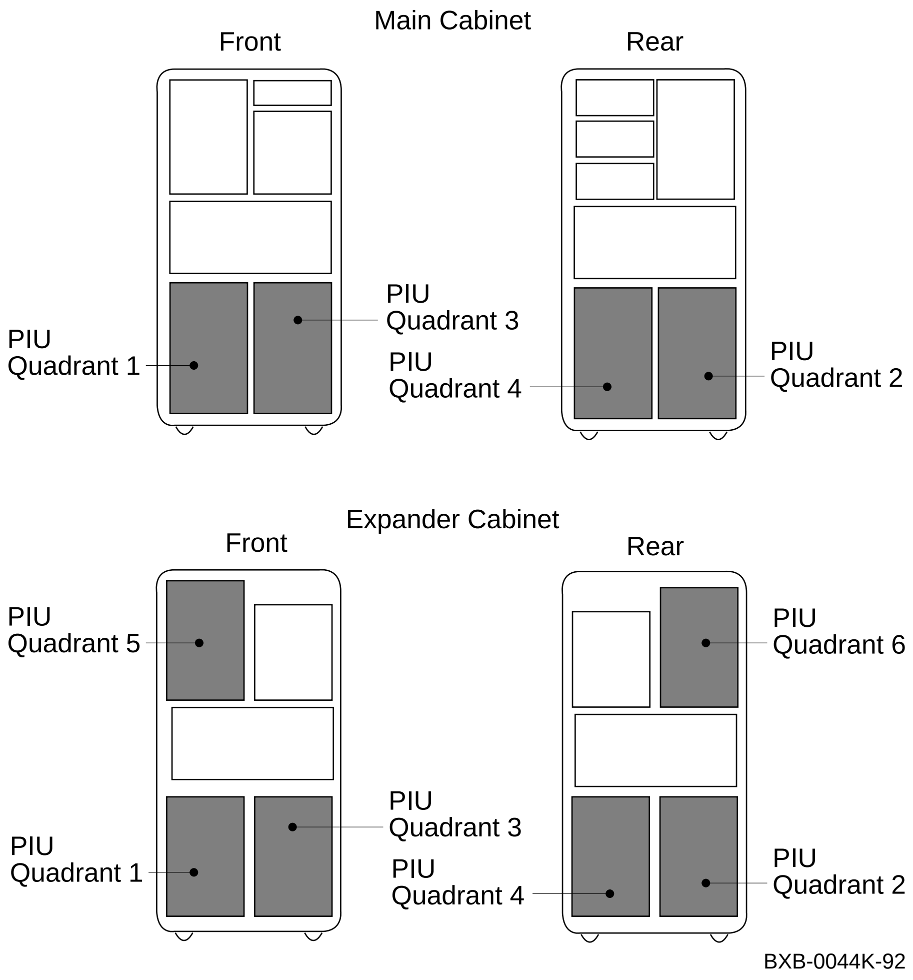


Table 1-4 PIU Quadrant Restrictions

PIU	Quadrant	Restrictions
XMI	1 and 2 or 3 and 4	If a VAXBI is connected, the XMI PIU must be in quadrants 1 and 2. Requires two PIU quadrants.
Futurebus+	2 or 4	Must be in the rear of the cabinet. Requires one PIU quadrant.
VAXBI	3 and 4	Must be in the same cabinet as the XMI to which it is connected. Requires two PIU quadrants.
DSSI disk	Any	For proper airflow, arrow on rear panel must point toward blower. Requires one PIU quadrant.
SCSI disk and tape	Any	For proper airflow, arrow on rear panel must point toward blower. Requires one PIU quadrant.
Battery	1 and 2 or 3 and 4	In DEC 7000/VAX 7000 systems, can be located in quadrants 3 and 4 only. Requires two PIU quadrants.

Chapter 2

LSB Bus

The LSB is the system bus for the DEC 7000/10000 and VAX 7000/10000 systems. The LSB bus is defined by the protocol that a node on the bus must follow, the electrical environment of the bus, the backplane, and the logic used to implement the protocol.

This chapter describes the LSB bus and includes these sections:

- Overview
- Arbitration
- Bus Cycles
- Bus Transactions
- Memory Bank Mapping
- Addressing
- Cache Memory
- Registers
- Console and Initialization
- Errors

2.1 Overview

The LSB is a limited length, nonpended, synchronous bus. It is 128 bits wide and uses distributed arbitration.

All transactions on the LSB bus occur in a set of fixed cycles relative to an arbitration cycle, and as many as three transactions can be in progress at one time. To accomplish this, the bus arbitrates on a dedicated set of control signals; arbitration may be overlapped with data transfer. Data and address are multiplexed on the same set of signals.

The LSB protocol supports writeback caches.

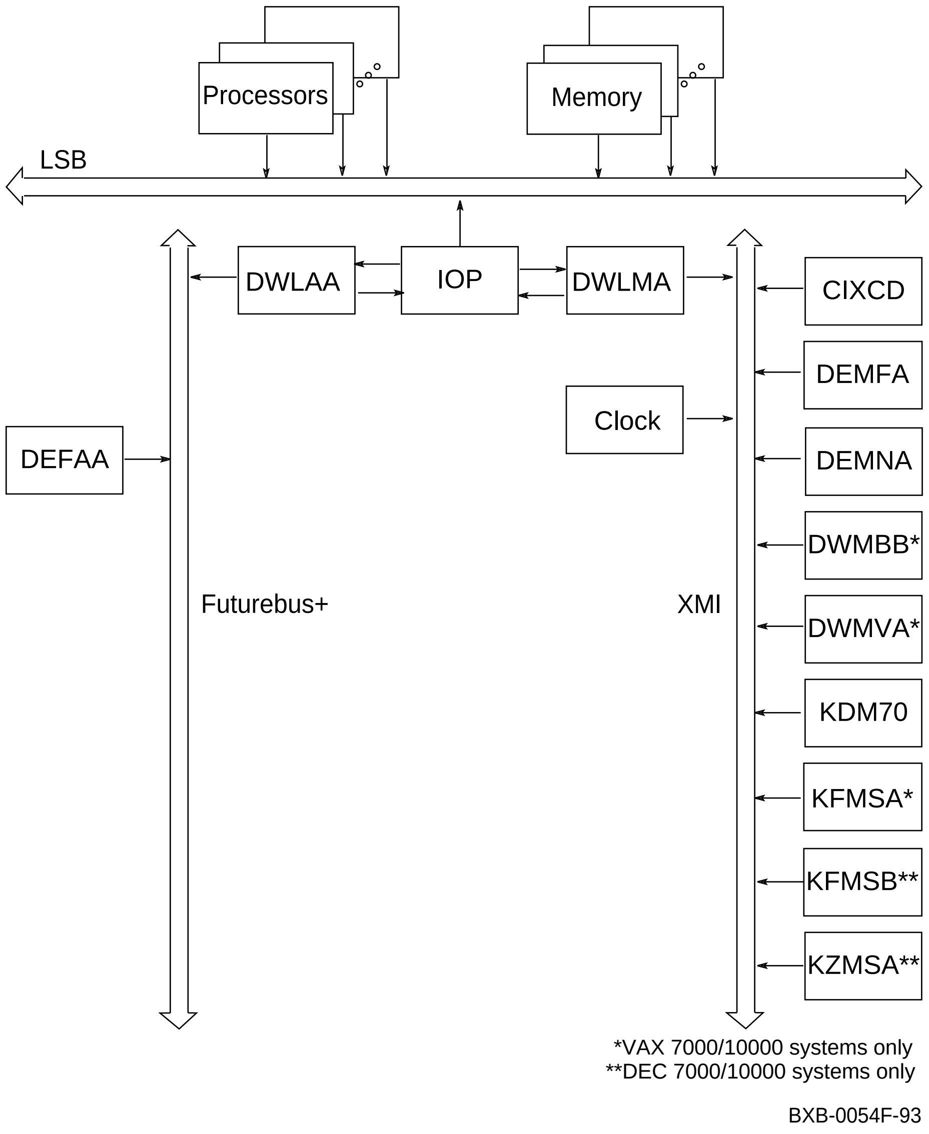
2.1.1 LSB Bus Specifications

Memory transfer length	64 bytes
Data path width	128 bits
Bus cycle time	20 ns
Usable bandwidth	640 Mbytes/sec

A node on the LSB is a module. The LSB can support from one to six processor nodes, one to seven memory nodes, and a single I/O port node, which is the interface to I/O plug- in units.

A block diagram of a typical system using the LSB is shown in Figure 2- 1.

Figure 2- 1 System Block Diagram



2.1.2 Module Identification

The LSB bus has slots for nine nodes. These nodes are identified in Table 2- 1. Each processor or memory module receives NID<2:0> on its backplane connector to identify its slot. The NID<2:0> signals are selectively connected to GND on the backplane to identify the slot, as specified in Table 2- 1.

The I/O port module is always node 8. It occupies a special slot in the backplane that is physically incompatible with the other slots. Since it always occupies the same slot, it does not need NID signals to determine its node number.

Table 2-1 LSB Node Identification

Node Number	Module	NID<2>	NID<1>	NID<0>
0	Processor or memory	Open	Open	Open
1	Processor or memory	Open	Open	GND
2	Processor or memory	Open	GND	Open
3	Processor or memory	Open	GND	GND
4	Processor or memory	GND	Open	Open
5	Processor or memory	GND	Open	GND
6	Processor or memory	GND	GND	Open
7	Processor or memory	GND	GND	GND
8	I/O port	Not applicable	Not applicable	Not applicable

2.1.3 Signals

All LSB data and control signals are implemented in negative logic. Signals are asserted low; that is, when a signal is asserted (TRUE or 1), the open drain driver pulls the line low. In the absence of an assertion, the bus terminator pulls the line high, so the signal is deasserted (FALSE or 0).

Table 2-2 LSB Signal Types

Type	Description
OD	Open drain – May be driven by one or more modules. If not driven, the wire defaults to a deasserted level.
CLK	Clock – Input only. It is driven from outside the LSB environment.
STRAP	Selectively connected to GND on the backplane to provide slot identification to the module.
TTL	Standard TTL level signal that is driven by logic external to the LSB environment.
OC	Open collector – Standard TTL level signal that may be driven by one or more modules. The pullup is supplied by logic external to the LSB environment (CCL module).

Table 2-3 LSB Signals

Signal	Wires	Type	Description
D<127:0>	128	OD	Data and command/address[1]
ECC<27:0>	28	OD	Error correction for data cycles
REQ<9:0>	10	OD	Arbitration request
STALL	1	OD	Responder stall
ERR	1	OD	Error detected by any module
SHARED	1	OD	Data is cached
DIRTY	1	OD	Memory data is stale
CNF	1	OD	Responder confirmation
CA	1	OD	Command/address cycle
LOCKOUT	1	OD	Lock out new requests
CRD	1	OD	Corrected read data
PH0	1	CLK	Clock phase 0 (sine)[2]
PH90	1	CLK	Clock phase 90 (cosine) [2]
NID<2:0>	3	STRAP	Slot identification [3]
LSB_RESET	1	OC	Reset everything
CCL_RESET	1	OC	Initiate reset sequence
BAD	1	OC	Self- test not successful
LOCTX	1	OC	Local console terminal transmit
LOCRX	1	TTL	Local console terminal receive
PSTX	1	OC	Power supply status transmit
PSRX	1	TTL	Power supply status receive
EXP_SEL	2	OC	Expander select
SECURE	1	OC	Secure console
LDC PWR OK	1	TTL	Local disk converter OK
PIU MOD A OK	1	TTL	Plug- in unit module A OK
PIU MOD B OK	1	TTL	Plug- in unit module B OK
RUN	1	OC	System run indicator
CONWIN	1	OC	Console win status

[1] Only D<38:0> are used during command cycles and CSR data cycles.

[2] Separate copies of the PH0 and PH90 signals are distributed to each module.

[3] NID<2:0> are hardwired on the backplane to provide an individual code for each slot.

2.2 Arbitration

The LSB bus uses a distributed arbitration scheme. Processor modules and the IOP module use ten signals, REQ<9:0>, to request the bus. All nodes independently monitor these lines to determine whether a transaction has been requested, and if so, which node wins the right to send a command cycle to start the transaction. The arbitration process ensures that transactions start only at specified times, and that only one node starts a transaction in any cycle. If the bus is idle, a request may be initiated in any cycle. If a transaction is in progress, requests may be made only during every fifth cycle to guarantee that new transactions do not interfere with transactions already in progress.

Arbitration control is pipelined. The arbitration sequence is:

1. A node places an arbitration request by asserting a REQ line.
2. All bus nodes determine which node won the arbitration. Because of the pipelining, other nodes may request the bus during this cycle, but these additional REQ assertions are ignored.
3. The winner of the bus arbitration may start a transaction.

2.2.1 Signals Used in Arbitration

Arbitration is controlled by these signals:

- REQ<9:0>
- CA
- STALL
- ERR
- RESET

REQ<5> is the highest priority request, and REQ<0> is the lowest. REQ<5> and REQ<0> are allocated to the IOP module. REQ<4:1> and REQ<9:6> are allocated to processor modules. The REQ<9:6,4:1> lines are dynamically reallocated among the processor modules using a least-recently- granted algorithm (see Section 2.2.5).

The CA, STALL, and ERR signals also influence arbitration. The arbitration logic monitors these signals to ensure that a new transaction is not started when it could result in cycle overlaps with transactions still in progress. The RESET signal sets priority to the default (power- up) state.

2.2.2 Arbitration Request Process

Nodes can request the bus during any cycle when it is idle or during every fifth cycle when it is in use. To request the bus, a module asserts its assigned REQ line. This section describes an algorithm for determining when a module is permitted to assert its REQ line.

Each node implements an allocation mask, ALLOC<16:0>, that indicates which future cycles are not available for command cycles. This mask is updated to track the allocation of cycles to existing and new transactions, and to account for stalls (STALL cycle) and errors (ERR signal). The higher numbered bits of ALLOC<16:0> represent the cycles farthest into the future.

ALLOC<16:0> is updated at the end of each cycle B using the following formula[1]:

```
ALLOC<16:0>=(ALLOC<16:0> SR0 (NOT STALL<B-2>)) OR             !SHIFT
           ((SXT CA<B-1>) AND (00011110111101111#2
        SL0 STALL<B-2>)) OR                                    !ALLOCATE
           ((SXT (ERR<B-1> OR RESET<B-1>)) AND 11111111111111111#2);
                                                              !RESET
```

This equation shifts the allocation mask right on each non- STALL cycle, sets some of the mask bits during each command cycle, and sets all the mask bits when ERR is asserted.

A module may assert its assigned REQ[2] signal in any cycle C if the following conditions are met:

```
ASSERT_REQUEST<C> =
        MODULE_REQUEST AND
        (CA<C-3> OR                         ! (ignore REQ in CA cycle)
           (REQ[C-3]<9:0> EQL 0))       AND ! no prior request
        (REQ[C-2]<9:0> EQL 0)           AND ! no prior request
        (NOT CA<C-2>)                   AND ! no prior request
        (NOT ASSERT_REQUEST<C-1>)       AND ! don't do it twice
        (NOT ALLOC[C]<2>)               AND ! sync with prev. transactions
        BANK_PERMITTED;                     ! if mem bank is idle
```

REQ[C- 3], REQ[C- 2], CA<C- 2>, and ASSERT_REQUEST<C- 1> block the request, to assure that no additional requests are issued after the first request on an idle bus. (Due to the bus control pipelining, one additional request may occur in the cycle after the first request on an idle bus. The protocol for arbitration grant assures that this request is ignored.) ALLOC is monitored to synchronize requests with bus transactions that are already in progress.

The BANK_PERMITTED condition assures that the cache block to be referenced does not match that of any other transactions currently in progress. (The computation of BANK_ PERMITTED is explained in Section 2.2.6.)

2.2.3 Arbitration Grant Process

When a node wins arbitration, it asserts CA and drives a command on the bus two cycles after it requests the bus. (The module also drives its node ID onto REQ<3:0> to facilitate bus monitoring.)

Specifically, a module may assert CA and drive a command on D<38:0> during any cycle C if the following conditions are met:

```
ARB_WIN = ASSERTED_REQ[C-2]<5,9:6,4:0> GTR (REQ[C-2]<5,9:6,4:0> SR0 1);
ASSERT_CA = (REQ[C-3]<9:0> EQL 0) AND
            (NOT CA<C-2>)AND
            (NOT CA<C-3>)AND
            (NOT CA<C-4>)AND
            ARB_WIN;
```

ARB_WIN is detected in the module that has asserted the highest-numbered REQ signal during cycle C- 2.

REQ[C- 3], CA<C- 2>, CA<C- 3>, and CA<C- 4> block the grant, to assure that no additional command cycles are issued immediately after the first-

[1] See page viii for information about coding conventions.

[2] REQ<3:0> are also asserted during command cycles to facilitate bus monitoring. These equations describe only the REQ assertions used for arbitration.

command cycle on an idle bus, and to minimize the likelihood of bus collisions in the presence of arbitration- related failure modes.

2.2.4 Recognizing Command Cycles

Each node monitors the bus for command cycles. A module must interpret any cycle C as a command cycle if the following conditions are met:

```
COMMAND_DETECTED<C> = CA<C>AND
                      (REQ[C-2]<9:0> GTR 0) AND
                      (NOT CA<C-1>) AND
                      (NOT CA<C-2>) AND
                      (NOT CA<C-3>) AND
                      (NOT CA<C-4>);
VALID_COMMAND_DETECTED<C> = COMMAND_DETECTED<C> AND
                            (XOR D[C]<38:0>)
```

Assertion of the CA signal indicates the presence of a command cycle. The state of REQ and CA during prior cycles is monitored to minimize the likelihood of bus collisions in the presence of arbitration- related failures.

A command cycle is recognized only if odd parity is observed over D<38:0> during the command cycle. If a parity error occurs during the command cycle, the command cycle is ignored.

2.2.5 Processor Arbitration Scheme

All processor modules share the use of REQ<9:6,4:1>. Each of these signals is assigned to only one processor module. (If the maximum number of processor modules is not installed, REQ<9:6,4:1> signals are allocated to those installed according to the algorithm specified here.)

In the absence of errors, the arbitration algorithm always puts the processor that was least recently granted the bus at the highest priority. This is achieved by giving a processor module the lowest priority whenever it wins the bus and moving the other modules up in priority by one.

To track its current priority, each processor module keeps a priority register, PRIO<9:6,4:1>. The PRIO register is updated at the end of each cycle (C in the algorithm) as shown here:

```
CACPU = CA[C] AND                               ! CPU command cycle
        (NOT REQ[C-2]<5>) AND
        (REQ[C-2]<9:6,4:1> GTR 0) AND
        NOT (CA[C-1] OR CA[C-2] OR CA[C-3] OR CA[C-4]);
CASE1 = ERR[C] OR ERR[C-1] OR ERR[C-2] OR RESET; ! abort
CASE2 = CACPU AND                               ! higher CPU won arb
        (REQ[C-2]<5,9:6,4:1> GEQ (PRIO<9:6,4:1> SL0 1)) AND
        NOT CASE1;
CASE3 = CACPU AND                               ! this CPU arbed and
                                                ! won
        (OR (REQ[C-2]<9:6,4:1> AND PRIO<9:6,4:1>)) AND
        NOT (CASE2 OR CASE1);
CASE4 = NOT                                     ! none of the above
        (CASE1<0> OR CASE2<0> OR CASE3<0>);
        PRIO<9:6,4:1> =
        ((SXT CASE1) AND (0001#2 SL0 NID<2:0>))  ! slot ID + 1
        OR
        ((SXT CASE2) AND (PRIO<9:6,4:1> SL0 1))  ! shift PRIO
        OR
        ((SXT CASE3) AND 00000001#2)            ! lowest PRIO
        OR
        ((SXT CASE4) AND PRIO<9:6,4:1>);        ! PRIO unchanged
```

The effect of this logic is to reset the PRIO from the decoded node ID after RESET or error (see Table 2- 4), to increase a processor module's priority whenever a higher priority processor module wins the bus, and to force a module's priority to the lowest level after it wins the bus.

This least- recently- granted priority scheme guarantees that a processor module is not locked out from bus access indefinitely by other processor requests.

Table 2- 4 PRIO Values After RESET

NID<2:0>	PRIO<9:6,4:1>
000	0000 0001
001	0000 0010
010	0000 0100
011	0000 1000
100	0001 0000
101	0010 0000
110	0100 0000
111	1000 0000

2.2.6 Memory Bank Contention

Nodes arbitrate for access to memory banks. A memory bank consists of 144 DRAMs and is the smallest section of memory that can be interleaved.

A node is permitted to arbitrate for memory access only when the node previously accessing the same memory bank has completed its access. The memory bank is computed from the memory address. For purposes of this algorithm, all CSR accesses and Private commands are treated as references to bank 0. Thus, the memory bank contention system effectively prevents interleaving of CSR accesses and Private commands.

To determine whether arbitration is permitted, a node must keep command history information and update it at the end of each cycle C according to the following algorithm:

```
BANK2<5:0> =
    ((SXT CA<C>) AND BANK1<5:0>)) OR  !conditional load
    ((SXT (NOT CA<C>)) AND BANK2<5:0>);
BANK1<5:0> =
    ((SXT CA<C>) AND COMMAND_BANK<5:0>)) OR !conditional load
    ((SXT (NOT CA<C>)) AND BANK1<5:0>);
BANK2_CYCLES_REMAIN<3:0> =           !count down from 13
    ((SXT CA<C>) AND (BANK1_CYCLES_REMAIN<3:0> - (NOT STALL<C>))) OR
    ((SXT ((NOT CA<C>) AND (BANK2_CYCLES_REMAIN<3:0> GTR 0))) AND
        (BANK2_CYCLES_REMAIN<3:0> - (NOT STALL<C>)));
BANK1_CYCLES_REMAIN<3:0> =           !count down from 13
    ((SXT CA<C>) AND (13#10 - (NOT STALL<C>)) OR
    ((SXT ((NOT CA<C>) AND (BANK1_CYCLES_REMAIN<3:0> GTR 0))) AND
        (BANK1_CYCLES_REMAIN<3:0> - (NOT STALL<C>)));
BANK_PERMITTED = NOT                 ! does either active bank match?
    ((((MODULE_REQUEST_BANK<5:0> EQL BANK2<5:0>)
    AND
```

```
(BANK2_CYCLES_REMAIN<3:0> GTR 0))
OR

((((MODULE_REQUEST_BANK<5:0> EQL BANK1<5:0>)
AND
(BANK1_CYCLES_REMAIN<3:0> GTR 0)));
```

This algorithm stores the identity of the preceding two banks that have passed on the bus and counts the number of remaining cycles downward during non- stalled cycles to determine when the transactions have completed. The bank for a new module request is compared against the banks of any transactions still in progress, and the arbitration is suppressed if there is a match.

Although the memory bank scheme can force short delays to arbitration for a specific bank, it cannot cause indefinite lockouts, if the requester maintains a continuous request for one bank until serviced.

2.2.7 IOP Arbitration Priority Scheme

The IOP module uses REQ<0> and REQ<5> to arbitrate for the bus; this means that it arbitrates at either the highest priority or the lowest priority. To avoid lockout of processors, the IOP module limits its use of high priority arbitration.

Specifically, the IOP module may not block any memory bank indefinitely by asserting REQ<5> during each available arbitration opportunity to that bank. To prevent lockouts, after the IOP accesses a memory bank once using REQ<5>, it must use REQ<0> for the next request to that bank. (If the REQ<0> arbitration is lost, the IOP may resume arbitration at REQ<5>.)

2.3 Bus Cycles

An LSB bus cycle is one cycle of the LSB clock. During a bus cycle, one bit is transferred on each of the bus lines. There are three types of LSB cycles:

- Command cycles
- Data cycles
- Null (unused) cycles

A bus transaction consists of one command cycle and four data cycles, separated by ten or more unrelated cycles.

2.3.1 Command Cycle

During a command cycle, the commanding node asserts CA and drives D<38:0> with command and address information. The format of D<38:0> is shown in Figure 2- 2. The commanding node drives its node ID onto REQ<3:0> to provide unambiguous identification of the commanding node for hardware monitoring and debug purposes. Table 2- 5 decodes the command field shown in Figure 2- 2.

Figure 2- 2 Command Cycle Format

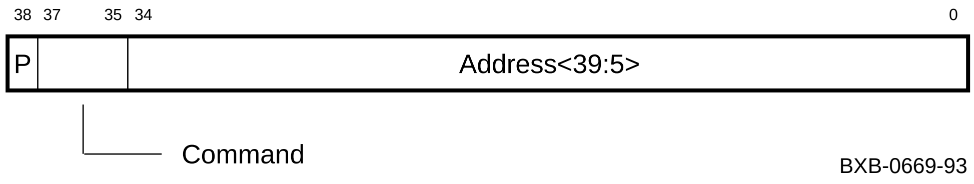


Table 2- 5 Command Field

Command Field	Command
000	**Read** —Read the contents of the specified block
001	**Write** —Write to the specified block
010	**Reserved**
011	**Write Victim**[1] —Write of a dirty[2] cache block to main memory
100	**Read CSR** —Read the contents of a register
101	**Write CSR** —Write to a register
110	**Reserved**
111	**Private** —Local operation (see page 2- 12)

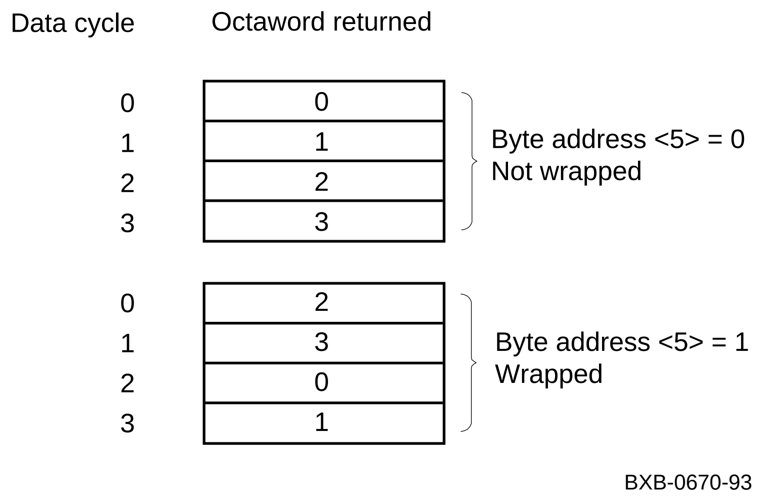
[1]Victim —the only copy of a dirty cache block. This block must be flushed from cache and written to main memory to make room for a new cache block that will occupy the same physical location in cache.

[2]Dirty —a cache block that has been modified.

Data Wrapping

For memory transactions, the address field contains bits <39:5> of the byte address. Byte address bits <39:6> (D<34:1>) uniquely specify the 64- byte cache block to be transferred. Byte address bit <5> (D<0>) specifies 32- byte wrapping as shown in Figure 2- 3.

Figure 2- 3 Data Wrapping



Byte address bit <5> (D<0>) is valid for both Read and Write transactions, and both memory reads and writes are wrapped.

For CSR requests, the address field uniquely specifies the longword CSR to be accessed.

Parity

D<38> is driven with odd parity over D<37:0>. That is, if the number of bits asserted in D<37:0> is even, D<38> is asserted; otherwise it is de- asserted. Modules check D<38> during command cycles to verify that the parity is correct. If the parity is incorrect, the module asserts ERR within four cycles and ignores the command.

D<127:39> and ECC<27:0> contain arbitrary values during command cy- cles.

Private Command

The Private command enables modules to use LSB cycles for local opera- tions. Private transactions have good parity in the command cycle. The cor- responding data cycles are ignored by other modules. Other modules ig- nore the content of Private transactions but treat them as normal transactions for arbitration and timing purposes.

A module using a Private transaction may drive arbitrary data on the bus during data cycles. The module must assert CNF during the appropriate cycle, but it may drive arbitrary information onto the SHARED and DIRTY lines during the shared/dirty cycle. If STALL is driven during the appropriate cycles, the data cycles are delayed as they are in other bus transactions.

2.3.2 Data Cycle

Each transaction has four data cycles. During memory data cycles, D<127:0> contain read or write data. Similarly, ECC<27:0> contain the ECC corresponding to the data in D<127:0>. If the data is sourced from a memory that uses the same ECC coding scheme as the LSB, the ECC information may be passed from that source without intermediate checking and correction.

In general, nodes receiving memory data cycles must check ECC<27:0>. However, if the data destination is a memory covered by the same ECC coding scheme, the data may be passed unchecked.

During stalled data cycles, D<127:0> and ECC<7:0> may hold arbitrary data. In the case of CSR transactions, bad parity is sent. In the case of memory transactions, the data and ECC may be arbitrary, and the ECC code may be bad or good. (The inclusion of the cycle count bits in the ECC code assures that bad data is not mistaken for good data, even in the presence of faults on the STALL line.)

During a Read CSR or Write CSR transaction, D<38:0> are driven in the first data cycle. D<37:0> contain data, and D<127:39> contain arbitrary data. D<38> has odd parity over D<37:0>. (If the number of bits asserted in D<37:0> is even, D<38> is asserted; otherwise it is deasserted.) The ECC<27:0> signals contain arbitrary values during Read CSR and Write CSR data cycles. During the remaining data cycles of a CSR transaction, D<127:0> and ECC<27:0> are not driven.

Nodes receiving CSR data cycles must check D<38> of the first data cycle for correct parity. If the parity is not correct, the module must assert ERR within four cycles.

2.3.3 ECC Coding

Figure 2- 4 shows the ECC scheme for checking and correcting memory data cycles. The 128- bit data path is divided into four longwords, and the ECC field is divided into four corresponding check words. Figure 2- 4 shows the coding scheme for D<31:0> and ECC<6:0>.

Figure 2-4 LSB ECC Coding

ECC		D<31:0>																																Cycle Count	
		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	1	0
6	XOR	X	X	X	X	X	X	X	X																	X	X	X	X	X	X	X	X	X	X
5	XOR	X	X	X	X	X	X	X	X									X	X	X	X	X	X	X	X									X	X
4	XOR	X	X							X	X	X	X	X	X			X	X							X	X	X	X	X	X			X	X
3	XNOR			X	X	X				X	X	X				X	X			X	X	X				X	X	X				X	X	X	X
2	XNOR	X		X			X	X		X			X	X			X	X		X			X	X		X			X	X			X		
1	XOR				X		X		X		X		X		X	X	X				X		X		X		X		X		X	X	X		X
0	XOR	X		X	X		X					X		X	X	X			X			X		X	X	X	X		X				X	X	

BXB-0671-93

The same coding scheme is used for each of the other three longwords. The ECC field for each longword is computed from a logical XOR of all table columns corresponding to ones in the longword; bits <3:2> of the result are then inverted. (Inversion of ECC<3:2> assures that a null bus cycle is not mistaken for an error- free or correctable- error data cycle.) Two cycle count bits are included to facilitate detection of sequencing errors resulting from open circuits on the STALL line. The cycle count bits are 00, 01, 10, and 11, during the first, second, third, and fourth data cycles, respectively. The correspondence of data longwords to ECC fields is shown in Table 2- 6.

Table 2- 6 Correspondence of Data Longwords to ECC Fields

Data Longword	ECC Field
D<31:0>	ECC<6:0>
D<63:32>	ECC<13:7>
D<95:64>	ECC<20:14>
D<127:96>	ECC<27:21>

2.4 Bus Transactions

All transactions use the same format. Each transaction starts with a command cycle and ends with four data cycles. These five fixed cycles are relative to the arbitration cycles. A minimum of ten cycles separates the command cycle from the data cycles. As many as three transactions can be in progress at one time. Figure 2- 5 shows a single transaction on an idle bus.

Figure 2- 5 Single Transaction Timing

```
                   1         2         3         4         5
cycle    01234567890123456789012345678901234567890123456789012

ARB      ....X................................................
CA       ......X..............................................
Command  ......X..............................................
CNF      .........X...........................................
SHR/DIR  ...........X.........................................
STALL    .....................................................
Data     .................XXXX................................
```

Figure 2- 6 shows transaction timing for consecutive interleaved transactions with no STALLs. Note that three transactions are interleaved on the bus.

Figure 2- 6 Interleaved Transaction Timing (Best Case)

```
                   1         2         3         4         5
cycle    01234567890123456789012345678901234567890123456789012

ARB      ....1....2....3....4....5....6.......................
CA       ......1....2....3....4....5....6.....................
Command  ......1....2....3....4....5....6.....................
CNF      .........1....2....3....4....5....6..................
SHR/DIR  ...........1....2....3....4....5....6................
STALL    .....................................................
Data     .................1111.2222.3333.4444.5555.6666.......
```

2.4.1 Signals Used in Bus Transactions

Command/Address (CA)

The CA signal is asserted by the bus commander to indicate that the current cycle is a command cycle. This signal is monitored by all nodes.

Confirm (CNF)

The confirm (CNF) signal is asserted by the node targeted by the transaction, typically the node that will supply data for a Read transaction or accept data for a Write transaction.

For memory transactions, CNF is asserted by the selected memory module. Processors are not required to assert CNF when they assert DIRTY and supply read data from their caches.

For CSR transactions, CNF is asserted by the targeted device. For Write CSR transactions that target multiple modules (for example, a Write to

LIOINTR), more than one responder may assert CNF. Typically, a targeted node is required to assert CNF for a CSR request and is required to supply read data or accept write data. However, for a Write CSR to the LMBPR register, the IOP may indicate a busy condition by failing to assert CNF.

The STxC instruction is used to access the LMBPR register. The lack of CNF assertion signals that the instruction has failed. Operating system software replays the STxC instruction in a loop with a timeout counter to enable detection of hard LSB errors without lockout problems in multiprocessor systems.

In Private transactions, CNF is asserted by the commander.

The CNF signal is typically asserted three cycles after CA, but the assertion may be delayed by STALLs.

SHARED and DIRTY (SHR/DIR)

The SHARED and DIRTY signals are asserted by processors to indicate the state of their cache entry for the specified address. SHARED and DIRTY must not be asserted for CSR transactions. The SHARED and DIRTY signals are typically asserted five cycles after the command cycle, but the assertion may be delayed by STALLs.

These signals are described in Section 2.7.

STALL

The STALL signal is used by any module during any transaction to indicate that the data cycles will be delayed by one or more cycles. Because the delayed data cycles could interfere with bus cycles of other transactions, the occurrence of STALL influences the timing of other transactions. Specifically, assertion of STALL during a specific cycle B causes the insertion of a dead cycle at cycle B+2 into every bus transaction that is active during cycle B+2.

STALL is asserted only during the ninth cycle after CA, which, itself, may be delayed by prior STALLs. That is, STALL cycles are not counted when determining which is the ninth cycle after CA. STALL may be repeated for multiple cycles.

LOCKOUT

A processor module asserts LOCKOUT after it has failed to gain access to a lock variable (for example, if a VAX 7000/10000 byte- update sequence fails repeatedly). When one or more processors assert LOCKOUT, other processors delay new outgoing requests until LOCKOUT is released. This allows the processor or processors asserting lockout to complete their access without interference.

Corrected Read Data (CRD)

The corrected read data (CRD) signal is used by memory modules to signal that the data supplied in response to an LSB Read command had a correctable error when it was retrieved from storage. The node supplying the data performs the correction (for example, by ECC), and the data supplied on the LSB and its ECC are correct. Processor nodes use this signal to initiate an exception to notify the operating system of this event.

2.4.2 Timing Definition

The positions of the various cycles for a transaction can be computed as follows:

```
V_CA_D<B> = VALID_COMMAND_DETECTED<B>; ! as defined earlier

CYCLE_AFTER_CA<15:1> =
        (CYCLE_AFTER_CA<14:1> SR0 (NOT STALL<B-2>)) &
        V_CA_D<B>;
        ((SXT V_CA_D<B>) AND 1000000000#2);

DATA_CYCLE<B+1> = CYCLE_AFTER_CA<15:11> GTR 0;
CNF_CYCLE<B+1> = CYCLE_AFTER_CA<5>;
SHR_CYCLE<B+1> = CYCLE_AFTER_CA<3>;
STALL_CYCLE<B+1> = CYCLE_AFTER_CA<9> OR STALL<B>;
```

2.4.3 Transaction Examples

Figure 2- 7 Simple STALL Timing

```
                   1         2         3         4         5
cycle    01234567890123456789012345678901234567890123456789012

ARB      ....X................................................
CA       ......X..............................................
Command  ......X..............................................
CNF      .........X...........................................
SHR/DIR  ...........X.........................................
STALL    ...............11111.................................
Data     .................-----XXXX...........................
```

Figure 2- 8 CSR Transaction Timing

```
                   1         2         3         4         5
cycle    01234567890123456789012345678901234567890123456789012

ARB      ....X................................................
CA       ......X..............................................
Command  ......X..............................................
CNF      .........X...........................................
SHR/DIR  ...........X.........................................
STALL    .....................................................
Data     .................X---................................
```

Figure 2- 9 Single STALL in Interleaved Transactions

```
                   1         2         3         4         5
cycle    01234567890123456789012345678901234567890123456789012

ARB      ....1....2....3..-..4....5....6......................
CA       ......1....2....3-....4....5....6....................
Command  ......1....2....3-....4....5....6....................
CNF      .........1....2..-..3....4....5....6.................
SHR/DIR  ...........1....2-....3....4....5....6...............
STALL    ...............1.....................................
Data     .................-1111.2222.3333.4444.5555.6666......
```

Figure 2- 10 Best- Case Interleaved Transaction Timing

```
                   1         2         3         4         5
cycle    01234567890123456789012345678901234567890123456789012

ARB      ....1....2....3....4....5....6.......................
CA       ......1....2....3....4....5....6.....................
Command  ......1....2....3....4....5....6.....................
CNF      .........1....2....3....4....5....6..................
SHR/DIR  ...........1....2....3....4....5....6................
STALL    .....................................................
Data     .................1111.2222.3333.4444.5555.6666.......
```

2.4.4 Interrupts

LSB nodes interrupt each other by writing to defined CSRs. Two interrupt mechanisms are provided through registers in the LSB CSR space in processors and the IOP module:

- *Vectored interrupts* are the traditional I/O adapter interrupts. The interrupt service routine is specified as a function of an IDENT vector supplied by the adapter. The operating system loads the value of this IDENT vector into each I/O adapter at system initialization.
- *Non-vectored or implied vector interrupts* have architecturally defined mechanisms for entering the relevant interrupt service routine and therefore do not require any additional information. Interprocessor interrupts, system error interrupts, and corrected read data interrupts are examples of non- vectored interrupts.

The LSB interrupt mechanism can target a maximum of four processors, while the LSB supports a maximum of six processors. Both KN7AA and KA7AA processor modules use the node ID to determine if a node is eligible to become one of the four possible interrupt targets.

2.4.4.1 Vectored Interrupts

The interrupting node sends a vectored interrupt by writing a value to the LIOINTR register. Software or firmware in the interrupted node retrieves the vector from the LILID register in the IOP module and determines the proper interrupt service routine. Four interrupt levels are defined. The mapping of these four LSB interrupt levels to a processor interrupt priority level is implementation specific. The mapping of an I/O bus interrupt level to an LSB interrupt level is described in the *I/O System Technical Manual*.

IOP Module Rules

The IOP module posts interrupts to processors by writing to the LIOINTR register. It follows these rules:

- For each I/O channel in the system, one interrupt can be pending at each interrupt level on the LSB.
- When a processor module performs a CSR Read of an LILID register, the IOP module considers the relevant interrupt to be serviced.
- Multiple processors can be the target for a single interrupt at one level. The processors are specified in the LCPUMASK register.

- If a processor performs a CSR Read of an LILID register for a specified interrupt level and the IOP module considers all the interrupts posted at that level to be serviced, the IOP module returns a value of zero as CSR Read data.

CPU Module Rules

Processor modules field interrupts from the IOP module and respond by reading the LILID registers. They follow these rules.

- At most four interrupts at one level can be pending on the LSB.
- A processor module maintains a record of the number of interrupts at each level that have been posted by the I/O module but not serviced on the LSB.

KN7AA processor modules implement a seven- bit counter for each LSB interrupt level. The counter resets to zero. When a processor's node is referenced by a Write transaction to the LIOINTR register, the counter for the indicated interrupt level is incremented. When a Read transaction to the corresponding LILID register is observed on the LSB (from any source), the corresponding counter is decremented. The relevant interrupt line is asserted when this counter has a non- zero value.

Operation

1. An I/O adapter posts an interrupt at a specified interrupt level.
2. The IOP module assembles the interrupt vector in the queue for the relevant interrupt level; it will later read the appropriate LILID register. The IOP module then issues a CSR Write transaction over the LSB to the LIOINTR register, using the LCPUMASK register to specify the processor or processors to receive the interrupt.
3. The targeted processor is interrupted at an appropriate level. The processor node issues a CSR Read transaction to the LILID register at the relevant interrupt level and gets the interrupt vector.
4. After the CSR Read of LILID is successfully completed, the IOP module considers the interrupt to be serviced.
5. If an interrupt targets more than one processor, the first processor to win the LSB for the CSR Read of LILID receives the relevant IDENT information. If another interrupt at the relevant level is pending, the additional CSR Reads of LILID will return that IDENT information. If no other interrupts are pending at the specified level, the IOP module returns zeros, forcing the processor to take a passive release.

2.4.4.2 Non- Vectored Interrupts

The interrupting node writes a value to a specified location in LSB broadcast space. Software or firmware in the interrupting node then vectors through a pre- defined mechanism in order to enter the interrupt service routine.

2.5 Memory Bank Mapping

LSB memory resides on a minimum of one and a maximum of seven memory modules.

The memory modules each contain one or two banks of memory. Modules of the same size may be interleaved in groups of two or four. Memory addresses sent during a command cycle are broken into four fields that control module selection, word address, bank selection, and interleave for the memory module. The widths and positions of these fields are controlled by the Address Mapping Register (AMR) on the memory module. Each processor module and the IOP module have Memory Mapping Registers, which are duplicate copies of the AMR registers for each memory module slot in the system. These registers are LMMR0 through LMMR7, corresponding to nodes 0 through 7 respectively. The LMMR registers are loaded by software during initialization, based on the values loaded into the corresponding AMR registers on the memory modules.

To make a memory request on the LSB, a node must first determine which memory bank will serve the request. The request must not be initiated until any prior request to that bank has completed. (This requirement makes LSB bandwidth use more efficient by eliminating STALL cycles caused by contention for the same bank of memory. This scheme also simplifies implementation of caches because the minimum time between accesses to the same cache line is substantially longer than it would be otherwise.) Section 2.2.6 describes the algorithm that a requester follows to determine when access to a specific bank is permitted.

CSR and Private commands always map to bank 0.

2.5.1 Memory Mapping Registers

Processor and IOP nodes each implement eight memory mapping registers, LMMR0 through LMMR7. Each memory mapping register is assigned to the slot with the corresponding node ID; that is, LMMR0 corresponds to node 0, LMMR1 corresponds to node 1, and so forth. In most cases, only a subset of the LMMR registers are in use, corresponding to the slots that contain memory modules. The EN bit in an unused LMMR register is set to 0 at power- up and on system reset to disable the register.

2.5.2 Bank Selection

For a command cycle double- hexword address A<34:0> (corresponding to D<34:0> of the command cycle), a bank number is generated using LMMR0 through LMMR7 (labeled LMMR[N] in the code fragment below) according to the following algorithm:

```
BANK<5:0> = 0;! (BANK = OR of 8 inputs)

FOR N FROM 0 TO 7 DO BEGIN! For each of 8 LMMRs

          ! Names for LMMR bits

    MODULE_ENABLED[N] = LMMR[N]<0>;! EN
    MODULE_ADDRESS[N]<14:0> = LMMR[N]<31:17>; !
    INTRLV_ADDRESS[N]<1:0> = LMMR[N]<4:3>; ! IA
    INTRLV_WIDTH[N]<1:0> = LMMR[N]<2:1>;! INT

          ! Decoded mask fields from LMMR
```

```
        MODULE_ADDRESS_MASK[N]<14:0> =! AW
               000000000000000#2 SR1 LMMR[N]<8:5>;

        INTRLV_ADDRESS_MASK[N]<1:0> =! INT
               000#2 SL1 INTRLV_WIDTH[N]<1:0>;

        BANK_MASK[N]<2:0> =! NBANKS
               000#2 SL1 LMMR[N]<10:9>

              ! 3 least sig bits of LMMR address

        LMMR_NUMBER[N]<2:0> = N;

              ! selected if enabled and high bits match
              ! and interleave matches
              ! (If LMMR's are properly programmed, only one bit
              ! in BANK_SELECTED[N] will be set.)

        MODULE_SELECTED[N] = MODULE_ENABLED[N]
                            AND
        ((A<34:19> AND MODULE_ADDRESS_MASK[N]<14:0>) EQL
                MODULE_ADDRESS[N]<14:0>)
                            AND
        ((A<2:1> AND INTRLV_ADDRESS_MASK[N]<1:0>) EQL
                INTRLV_ADDRESS[N]<1:0>);

              ! which of the 1, 2, 4, or 8 banks on the module?

        BANK_IN_MODULE[N]<2:0> =
            ((A<5:1> SR0 INTRLV_WIDTH[N]<1:0>) AND BANK_MASK[N]]<2:0>);

              ! OR the results onto BANK

        BANK<5:0> = BANK<5:0> OR
            ((MODULE_SELECTED[N] SXT 5#10) AND
            (LMMR_NUMBER[N]<2:0> & BANK_IN_MODULE[N]<2:0>));

    END;
```

Table 2-7 describes the BANK<5:0> field, which is based on the value of the INT and NBANKS fields in LMMR.

Table 2-7 Bank Number as a Function of Interleave

Selected	INT	NBANKS	BANK<5:3>	BANK<2>	BANK<1>	BANK<0>
No	x	x	-	-	-	-
Yes	00	00	NID<2:0>	0	0	0
Yes	00	01	NID<2:0>	0	0	A<1>
Yes	00	10	NID<2:0>	0	A<2>	A<1>
Yes	00	11	NID<2:0>	A<3>	A<2>	A<1>
Yes	01	00	NID<2:0>	0	0	0
Yes	01	01	NID<2:0>	0	0	A<2>
Yes	01	10	NID<2:0>	0	A<3>	A<2>
Yes	01	11	NID<2:0>	A<4>	A<3>	A<2>
Yes	10	00	NID<2:0>	0	0	0
Yes	10	01	NID<2:0>	0	0	A<3>
Yes	10	10	NID<2:0>	0	A<4>	A<3>
Yes	10	11	NID<2:0>	A<5>	A<4>	A<3>

2.6 Addressing

The LSB allows for one terabyte of memory to be accessed by a 40- bit memory address. LSB control and status registers are accessed by a 22- bit address. CSRs on I/O buses are accessed through mailbox structures. Figures 2- 11 and 2- 12 show the relationship between processor byte addresses, memory block addresses, and CSR addresses.

Figure 2- 11 Memory Address Bit Mapping

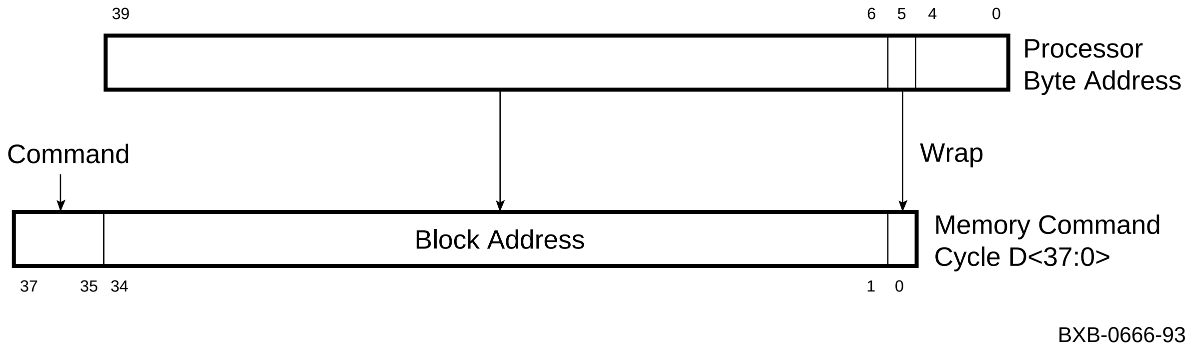
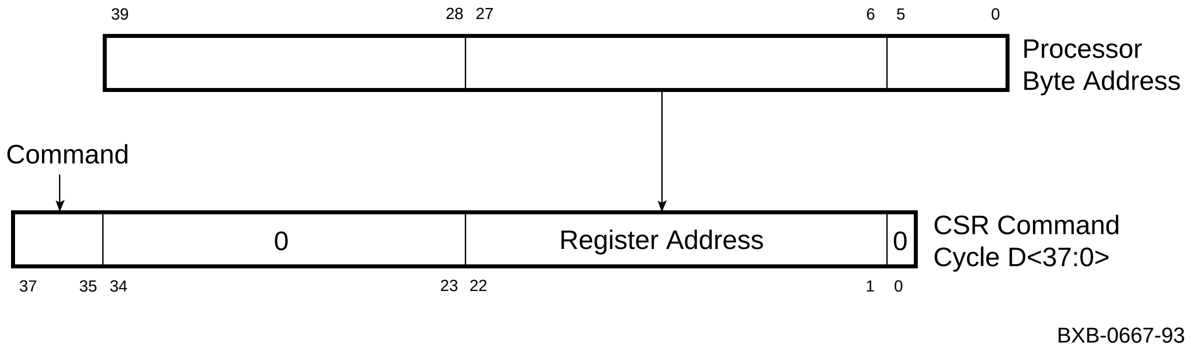


Figure 2- 12 Register Address Bit Mapping



2.6.1 Memory Map

Memory is accessed in 64- byte blocks, addressed with bits<34:0> of a command cycle. Bits<34:1> map to byte address <39:6>, specifying 2^{34} 64- byte blocks; bit<0> maps to byte address <5>, specifying which 32- byte subblock is to be returned first (wrapped), as shown in Figure 2- 13.

Table 2- 8 lists the address of each LSB node.

Figure 2- 13 Wrapping of LSB Data

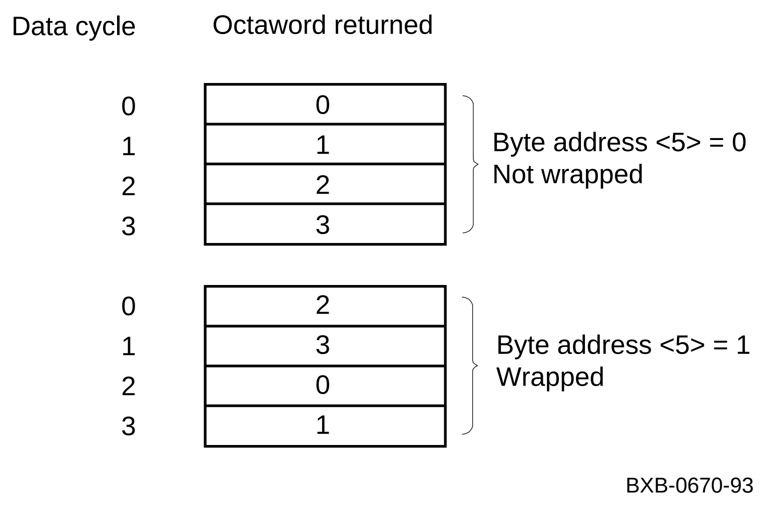


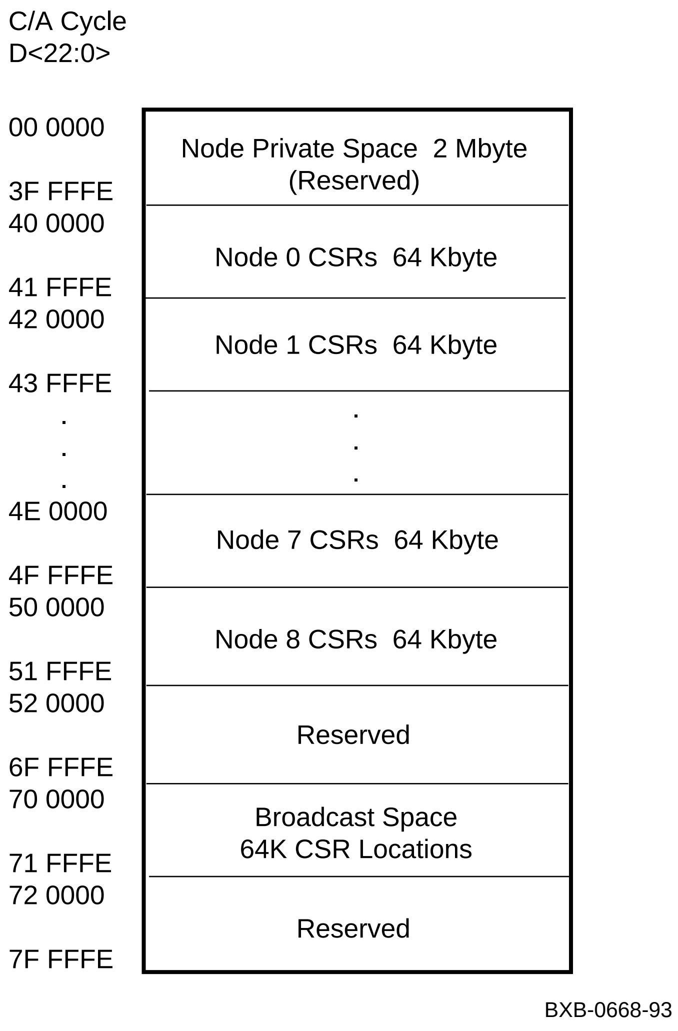
Table 2- 8 LSB Node Base Addresses

Node Number	Module	Base Physical Address (BB) AXP Systems	VAX Systems	C/A Cycle D<22:0>
0	Processor or memory	3 F800 0000	F800 0000	40 0000
1	Processor or memory	3 F840 0000	F840 0000	42 0000
2	Processor or memory	3 F880 0000	F880 0000	44 0000
3	Processor or memory	3 F8C0 0000	F8C0 0000	46 0000
4	Processor or memory	3 F900 0000	F900 0000	48 0000
5	Processor or memory	3 F840 0000	F840 0000	4A 0000
6	Processor or memory	3 F880 0000	F880 0000	4C 0000
7	Processor or memory	3 F8C0 0000	F8C0 0000	4E 0000
8	IOP	3 FA00 0000	FA00 0000	50 0000
Broadcast Space Base (BSB)	—	3 FE00 0000	FE00 0000	70 0000

2.6.2 Register Map

All control and status registers (CSRs) on LSB nodes are, by definition, 32 bits wide and aligned on 64- byte boundaries. (The LMBPR registers are an exception, since they are wider than 32 bits.) CSRs are accessed using the Read CSR and Write CSR commands. Bits D<22:1> of the address field in a CSR Read or CSR Write command cycle specify the register (D<33:23> and D<0> are always zero during CSR command cycles) as shown in Figure 2- 14.

Figure 2-14 Register Address Map



The first two megabytes of CSR space are reserved for local use on each module. References to this region are serviced by resources local to the module, so they are never asserted on the LSB. Nodes may not implement any CSRs visible to the bus in this space or in other reserved spaces.

Broadcast space is used for write- only registers that are written in all nodes in a single bus transaction. This region is used to implement interrupts.

2.6.3 Mailboxes

CSRs on external I/O buses are accessed through mailbox structures in main memory. Read requests are posted in mailboxes, and data is returned in memory with status in the following quadword. Mailboxes are allocated and managed by operating system software. (Successive operations must not overwrite data that is still in use.)

The IOP module services mailbox requests with four mailbox pointer CSRs (LMBPRs) located in the IOP module's nodespace. LMBPR points to a naturally aligned 64- byte data structure constructed by software in memory. This data structure is shown in Figure 2- 15. Table 2- 9 describes the format of the mailbox data structure.

Figure 2- 15 Mailbox Data Structure

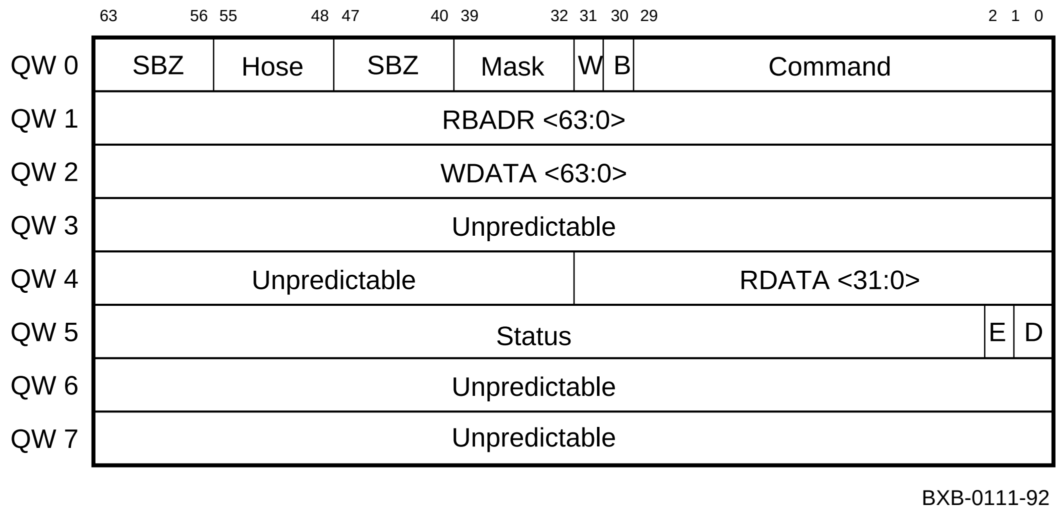


Table 2- 9 Mailbox Data Structure Format

Quad- word	Bits	Name	Description
0	<29:0>	CMD	**Command.** Contains the I/O adapter commands. Commands are adapter specific; see the *I/O System Technical Manual*.
	<30>	B	**Remote bridge access.** When set, the command is a special or diagnostic command directed to the remote side.
	<31>	W	**Write access.** When set, indicates a write on the remote bus.
	<39:32>	MASK	**Disable byte mask.** When a bit is set in the 8- bit field, the corresponding byte of a quadword write transaction on the remote bus is not written.
	<47:40>	SBZ	Should be zero.
	<55:48>	HOSE	**Hose ID.** Specifies the hose. The IOP ignores bits <55:50> and decodes bits <49:48> as follows:
	<63:56>	SBZ	Should be zero.

Bits <49:48>	Hose
00	0
01	1
10	2
11	3

Table 2-9 Mailbox Data Structure Format (Continued)

Quad-word	Bits	Name	Description
1	<63:0>	RBADR	**Remote bus address.** Contains the address of either a CSR in the I/O adapter or a CSR in a remote node on the I/O subsystem bus.
2	<63:0>	WDATA	**Write data.** Contains the data to be written to the address specified in RBADR. If the command is a write, the field is valid; otherwise, it is undefined.
3	<63:0>		Use of this field is undefined, and the contents are unpredictable.
4	<31:0>	RDATA	**Read data.** Contains data from the I/O adapter at the completion of a mailbox transaction. If the command in the CMD filed was a read and the transaction completed successfully (the E bit is clear and the D bit is set), the data is valid. After the completion of a write- type mailbox transaction or a failing transaction, this field is unpredictable. Supported field widths are I/O adapter dependent.
	<63:32>		Use of this field is undefined, and the contents are unpredictable.
5	<0>	D	**Done.** When a mailbox transaction is initiated by software, this bit is clear. To indicate that the mailbox transaction failed, the I/O adapter sets this bit in the Mailbox Status Return packet it sends across the Up Hose to the IOP. The IOP passes the information on when it writes the status to memory.
	<1>	E	**Error.** When a mailbox transaction is initiated by software, this bit is clear. To indicate that the mailbox transaction failed, the I/O adapter sets this bit in the Mailbox Status Return packet it sends across the Up Hose to the IOP. The IOP passes the information on when it writes the status to memory.
	<63:2>	STATUS	**Device-specific status.** Contains information provided by the I/O adapter at the completion of the mailbox transaction. The field is I/O adapter dependent.
6	<63:0>		Use of this field is undefined, and the contents are unpredictable.
7	<63:0>		Use of this field is undefined, and the contents are unpredictable.

2.7 Cache Memory

The LSB uses a write- update cache protocol. Two bus signals, SHARED and DIRTY, enable a node to keep its cache coherent with all other nodes by following the procedures described in this section.

All modules that implement caches monitor the LSB and probe their tag stores during all bus operations that are not Victim Writes or CSR operations. If the result of this probe indicates that a module has a valid copy of the referenced cache line, the module asserts SHARED. Alternatively, on a Write hit, the module can invalidate a cache line.

If the bus operation is a Read transaction and the target cache line is found to have been modified in a module's cache, the module asserts DIRTY. Several modules may assert SHARED in response to a bus operation, but the protocol ensures that only one module can assert DIRTY. If a module asserts DIRTY in response to a bus operation, that module, rather than memory, is responsible for supplying the read data. Memory modules that see DIRTY asserted for a Read transaction are required to suppress the transmission of the read data.

2.7.1 Cache States

Processor modules store the following for each 64- byte cache block:

- A tag consisting of some number of physical address bits
- A VALID (V) bit indicating if this line can be considered
- A SHARED (S) bit indicating if this line may also be resident in another node's cache in the system
- A DIRTY (D) bit indicating if this line has been modified by this node

The allowed combinations of V, S, and D bits are listed in Table 2- 10.

Table 2- 10 Cache States

V	S	D	State of Cache Line Assuming Tag Match
0	X	X	**Not valid miss**.
1	0	0	**Valid for Read or Write.** This cache line contains the only cached copy of the block; the copy in memory is identical to this line.
1	0	1	**Valid for Read or Write.** This cache line contains the only cached copy of the block. The contents of the block have been modified more recently than the copy in memory.
1	1	0	**Valid for Read or Write, but Writes must be broadcast on the bus.** This block might be in another processor's cache. The contents of this block are identical to the copy in memory.
1	1	1	**Valid for Read or Write, but Writes must be broadcast on the bus.** This block might be in another processor's cache. The contents of this block have been modified more recently than the copy in memory.

From the perspective of a processor accessing its cache, a tag probe for a Read transaction is satisfied if the tag matches and the V bit is set. A tag probe for a Write is satisfied if the tag matches, the V bit is set, and the S bit is clear. A cache line is considered dirty (different from other copies) if the D bit is set and is considered clean (same as other copies) if the D bit is clear.

2.7.2 Cache State Changes

The state of any line in a module's cache is influenced by either processor actions or actions of other nodes on the system bus.

2.7.2.1 Processor Actions

Table 2- 11 shows the result of processor actions on the state of a cache line and the bus traffic that follows.

Table 2- 11 Result of Processor Actions on Cache Line

Processor Request	Tag Probe Result[1]	Action on LSB	LSB Response	Next Cache State	Comments
Read	Invalid	Read	$\overline{\text{Shared}}$	$\overline{\text{Shared}}$, $\overline{\text{Dirty}}$	
Read	Invalid	Read	Shared	Shared, $\overline{\text{Dirty}}$	
Write	Invalid	Read	$\overline{\text{Shared}}$	$\overline{\text{Shared}}$, Dirty	
Write	Invalid	Read	Shared	Shared, $\overline{\text{Dirty}}$	Bus Write follows
Read	$\overline{\text{Match}}$ AND $\overline{\text{Dirty}}$	Read	$\overline{\text{Shared}}$	$\overline{\text{Shared}}$, $\overline{\text{Dirty}}$	
Read	$\overline{\text{Match}}$ AND $\overline{\text{Dirty}}$	Read	Shared	Shared, $\overline{\text{Dirty}}$	
Write	$\overline{\text{Match}}$ AND $\overline{\text{Dirty}}$	Read	$\overline{\text{Shared}}$	$\overline{\text{Shared}}$, Dirty	
Write	$\overline{\text{Match}}$ AND $\overline{\text{Dirty}}$	Read	Shared	Shared, $\overline{\text{Dirty}}$	Bus Write follows
Read	$\overline{\text{Match}}$ AND Dirty	Write, Read	$\overline{\text{Shared}}$	$\overline{\text{Shared}}$, $\overline{\text{Dirty}}$	Victim written back
Read	$\overline{\text{Match}}$ AND Dirty	Write, Read	Shared	Shared, $\overline{\text{Dirty}}$	Victim written back
Write	$\overline{\text{Match}}$ AND Dirty	Write, Read	$\overline{\text{Shared}}$	$\overline{\text{Shared}}$, Dirty	Victim written back
Write	$\overline{\text{Match}}$ AND Dirty	Write, Read	Shared	Shared, $\overline{\text{Dirty}}$	Victim written back Bus Write follows
Read	Match	None	None	No change	
Write	Match AND $\overline{\text{Shared}}$	None	None	$\overline{\text{Shared}}$, Dirty	
Write	Match AND Shared	Write	$\overline{\text{Shared}}$	$\overline{\text{Shared}}$, $\overline{\text{Dirty}}$	
Write	Match AND Shared	Write	Shared	Shared, $\overline{\text{Dirty}}$	

[1]An overscore indicates the complement of the state.

2.7.2.2 Bus Actions

Table 2- 12 shows the result of bus actions on the state of a cache line in non- initiating modules and the response that follows.

Table 2- 12 Result of Bus Actions on Cache Line

LSB Operation	Tag Probe Result[1]	Module Response	Next Cache State	Comments
Read	$\overline{\text{Match}}$ OR Invalid	$\overline{\text{Shared}}$, $\overline{\text{Dirty}}$	No change	
Write	$\overline{\text{Match}}$ OR Invalid	$\overline{\text{Shared}}$, $\overline{\text{Dirty}}$	No change	
Read	Match AND $\overline{\text{Dirty}}$	Shared, $\overline{\text{Dirty}}$	Shared, $\overline{\text{Dirty}}$	
Read	Match AND Dirty	Shared, Dirty	Shared, Dirty	Module supplies data
Write	Match and line is interesting	Shared, $\overline{\text{Dirty}}$	Shared, $\overline{\text{Dirty}}$	Module takes the update
Write	Match and line is non- interesting	$\overline{\text{Shared}}$, $\overline{\text{Dirty}}$	Invalid	Module takes the invali- date

[1]An overscore indicates the complement of the state.

NOTE: Writes always clean (make non- dirty) the cache line in both the initiating module and all modules that choose to take the update.

2.7.2.3 Write Operations

Modules perform Write operations under the following conditions:

- **Victims**

  If a cache line is valid and dirty, but the tag does not match the address for the processor request, this line must be written back to memory. Because this is the only up- to- date cache line, and it will be displaced by new data from the processor- initiated read, it must be written to memory.

  To enhance performance, this victim is written back to memory after initiation of the refill that satisfies the original processor request. The victim data must be removed from the cache data store and held in a victim buffer for later transmission on the LSB. While a victim block is in a victim buffer, the module must accept all Read and Write transactions that reference the block, and the block must carry the victim bit with all its implications. Victims are written back to memory with the Write Victim command. Nodes that see a Write Victim command on the bus are allowed to ignore the transaction.

- **Shared Blocks**

  If the response to a tag probe for a processor Write transaction is shared, the write must be sent on the LSB. These writes must be performed with the Write command (Victim bit clear).

2.8 Registers

All LSB registers have the following characteristics:

- All writes are 32 bits long[1]. Byte and word operations are not supported.
- Writes directed at read- only registers may be accepted and acknowledged, but no action is taken, and the value of the register is not affected.

All LSB control and status registers are accessed using the Read CSR and Write CSR commands. Some registers are required in each node, while others are used only in one or two types of modules.

Table 2- 13 defines the bit types used in the register descriptions that follow. Table 2- 14 lists the LSB registers and the type of module in which each is implemented. Each module type also has a number of other registers; see the appropriate technical manual for descriptions of these other registers.

Table 2- 13 Register Bit Types

Bit Type	Description
M	Modify or read/write.
MBZ	Must be zero; always read as zero. Writes to the bit are ignored.
RAZ	Read as zero. Writes to the bit are ignored.
RO	Read only; can only be read by the user. Only hardware can change the value of the bit. User writes to the bit are ignored.
RTC	Read to clear; can only be read by the user. Hardware clears the field after the first read to this register.
R/W	Read/write. Bit can be read and written.
W1C	Write one to clear; can be read by the user. The hardware can change the value of the bit. If the hardware is not updating the bit at the same time, the user can clear the bit by writing a one to it.
W1S	Write one to set; cannot be read by a user. A user may set this bit by writing a one to it. Writing a zero to this bit has no effect.
WO	Write only; can only be written by the user. Reads are undefined.

[1] The only exception to this is a write to a Mailbox Pointer Register (LMBPR0–3). This operation is a write of up to 38 bits. See the description of the LMBPR register on page 2- 45.

Table 2- 14 LSB Registers

				Implemented on		
Mnemonic	Register Name	Address (byte offset)	Access	Proc	Mem	IOP
LDEV	Device	BB[1] + 0000	R/W	X	X	X
LBER	Bus Error	BB + 0040	R/W	X	X	X
LCNR	Configuration	BB + 0080	R/W	X	X	X
LMMR0	Memory Mapping 0	BB + 0200	R/W	X		X
LMMR1	Memory Mapping 1	BB + 0240	R/W	X		X
LMMR2	Memory Mapping 2	BB + 0280	R/W	X		X
LMMR3	Memory Mapping 3	BB + 02C0	R/W	X		X
LMMR4	Memory Mapping 4	BB + 0300	R/W	X		X
LMMR5	Memory Mapping 5	BB + 0340	R/W	X		X
LMMR6	Memory Mapping 6	BB + 0380	R/W	X		X
LMMR7	Memory Mapping 7	BB + 03C0	R/W	X		X
LBESR0	Bus Error Syndrome 0	BB + 0600	RO	X	X	X
LBESR1	Bus Error Syndrome 1	BB + 0640	RO	X	X	X
LBESR2	Bus Error Syndrome 2	BB + 0680	RO	X	X	X
LBESR3	Bus Error Syndrome 3	BB + 06C0	RO	X	X	X
LBECR0	Bus Error Command 0	BB + 0700	RO	X	X	X
LBECR1	Bus Error Command 1	BB + 0740	RO	X	X	X
LILID0	Interrupt Level 0 IDENT	BB + 0A00	RTC			X
LILID1	Interrupt Level 1 IDENT	BB + 0A40	RTC			X
LILID2	Interrupt Level 2 IDENT	BB + 0A80	RTC			X
LILID3	Interrupt Level 3 IDENT	BB + 0AC0	RTC			X
LCPUMASK	CPU Interrupt Mask	BB + 0B00	R/W			X
LMBPR0	Mailbox Pointer 0	BB + 0C00	R/W			X
LMBPR1	Mailbox Pointer 1	BB + 0C00	R/W			X
LMBPR2	Mailbox Pointer 2	BB + 0C00	R/W			X
LMBPR3	Mailbox Pointer 3	BB + 0C00	R/W			X
LIOINTR	I/O Interrupt	BSB[2] + 0000	R/W	X		
LIPINTR	Interprocessor Interrupt	BSB + 0040	R/W	X		

[1] BB is the node space base address (in hex) of the module. See Table 2- 8.

[2] BSB is the broadcast space base address (in hex). See Table 2- 8.

LDEV — Device Register

Address BB + 0000
Access R/W

The LDEV register is loaded during initialization with information that identifies the module.

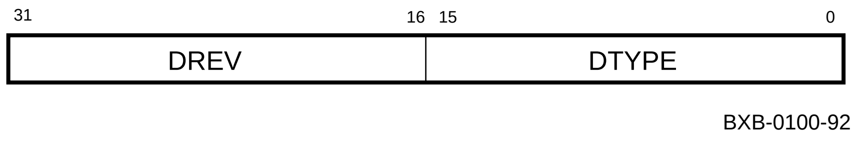


Table 2-15 LDEV Register Bit Definitions

Name	Bits	Type	Description
DREV	<31:16>	R/W, 0	**Device revision.** Identifies the revision level of the module. The value is loaded by hardware or self-test firmware during the node self-test.
DTYPE	<15:0>	R/W, 0	**Device type.** Identifies the type of node. Bit <15> specifies a processor node, bit <14> specifies a memory node, and bit <13> specifies an I/O node. Bits <7:0> contain the device type code.

Node	Device Type (hex)
KN7AA processor	8001
KA7AA processor	8002
MS7AA memory	4000
MS7BB memory	4002
IOP	2000

LBER — Bus Error Register

Address BB + 0040
Access R/W

The LBER register stores the error bits that are flagged when an LSB node detects errors in the LSB operating environment and logs the failing commander ID. The status of this register remains locked until software resets the error bit(s).

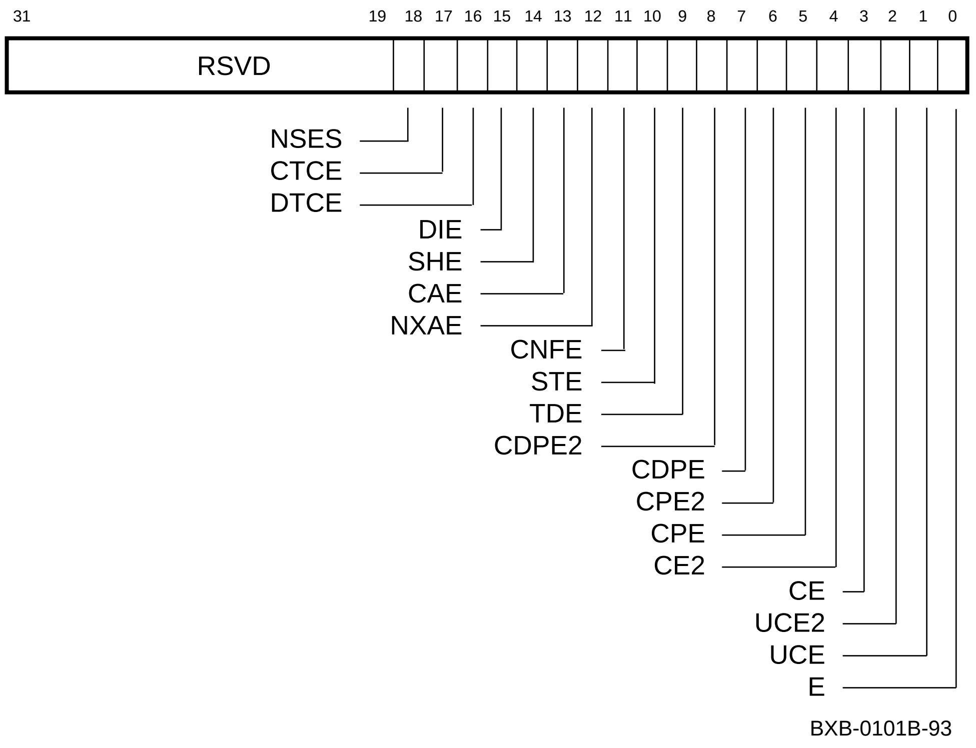


Table 2- 16 LBER Register Bit Definitions

Name	Bits	Type	Description
RSVD	<31:19>	RAZ	**Reserved.** Read as zero.
NSES	<18>	RO, 0	**Node-specific error summary.** Set when a node- specific error condition is reported to a register specified by the implementation (LMERR in processor nodes; MERA in memory nodes; IPCNSE or IPCHST in IOP).
CTCE	<17>	W1C, 0	**Control transmit check error.** Set when an LSB control line is driven incorrectly by a processor or IOP module. When CTCE is set, the module asserts ERR for one cycle. Not implemented on memory nodes.
DTCE	<16>	W1C, 0	**Data transmit check error.** Set when a processor or IOP module detects an error while driving the D<127> and ECC<27:0> lines during a data or command cycle. When DTCE is set, the module asserts ERR for one cycle. Not implemented on memory modules.
DIE	<15>	W1C, 0	**Dirty error.** Set when the processor or memory receives an asserted DIRTY signal during a cycle in which DIRTY signals are not allowed. When DIE is set, the module asserts ERR for one cycle. Not implemented on the IOP module.
SHE	<14>	W1C, 0	**Shared error.** Set if the processor module receives an asserted SHARED signal during a cycle in which SHARED signals are not allowed. When SHE is set, the module asserts ERR for one cycle. Not implemented on memory and IOP modules.
CAE	<13>	W1C, 0	**Command/address error.** Set if the module receives an asserted CA signal during a cycle in which CA signals are not allowed. When CAE is set, the module asserts ERR for one cycle.
NXAE	<12>	W1C, 0	**Nonexistent address error.** Set when the processor or IOP module does not receive confirmation for a command it sent on the LSB. When NXAE is set, the module asserts ERR for one cycle. Not implemented on memory modules.
CNFE	<11>	W1C, 0	**Confirmation error.** Set when the module receives a confirmation signal during a cycle that does not permit confirmation. When CNFE is set, the module asserts ERR for one cycle.
STE	<10>	W1C, 0	**Stall error.** Set when the module receives a STALL signal during a cycle that does not permit stalls. When STE is set, the module asserts ERR for one cycle.
TDE	<9>	W1C, 0	**Transmitter during error.** Set if a CE, UCE, CPE, or CDPE error occurs during a cycle when the module was driving D<127:0>. Result of setting TDE: processor nodes —the module asserts ERR for one cycle; IOP —the module locks command information in the LBECR registers.

Table 2-16 LBER Register Bit Definitions (Continued)

Name	Bits	Type	Description
CDPE2	<8>	W1C, 0	**Second CSR data parity error.** Set when a second parity error occurs while CDPE is set on a CSR data cycle. Not implemented on memory modules.
CDPE	<7>	W1C, 0	**CSR data parity error.** Set if a parity error occurs during a CSR data cycle. When CDPE is set, the module asserts ERR for one cycle and locks D<38:0> in the LBECR registers. Not implemented on memory modules.
CPE2	<6>	W1C, 0	**Second command parity error.** Set if a second parity error occurs on a command cycle while CPE is set.
CPE	<5>	W1C, 0	**Command parity error.** Set when a parity error occurs on a command cycle. When CPE is set, the module asserts ERR for one cycle and locks D<38:0> in the LBECR registers.
CE2	<4>	W1C, 0	**Second correctable data error.** Set when a second correctable ECC error occurs on a data cycle while CE is set.
CE	<3>	W1C, 0	**Correctable data error.** Set if the module detects an ECC error on the LSB. When CE is set, the module asserts ERR for one cycle and locks D<38:0> of the command cycle in the LBECR registers.
UCE2	<2>	W1C, 0	**Second uncorrectable data error.** Set when the module detects a second uncorrectable data error while UCE is set.
UCE	<1>	W1C, 0	**Uncorrectable data error.** Set if the module detects an uncorrectable ECC error on the LSB during a data cycle. When UCE is set, the module asserts ERR for one cycle and locks D<38:0> of the command cycle in the LBECR registers.
E	<0>	W1C, 0	**Error line asserted.** Set whenever the module detects assertion of ERR on the LSB.

LCNR — Configuration Register

Address	BB + 0080
Access	R/W

The LCNR register contains LSB configuration setup and status information.

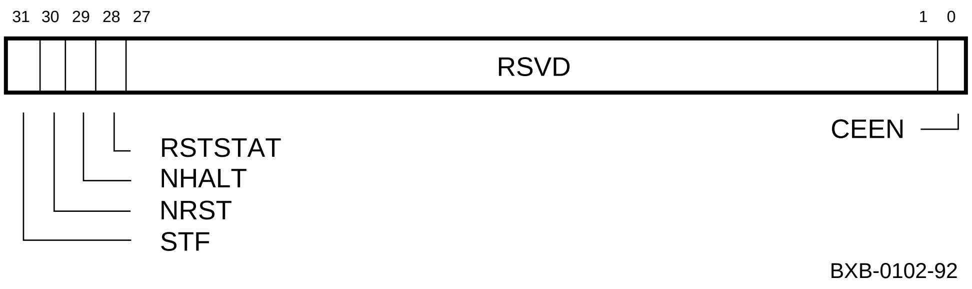


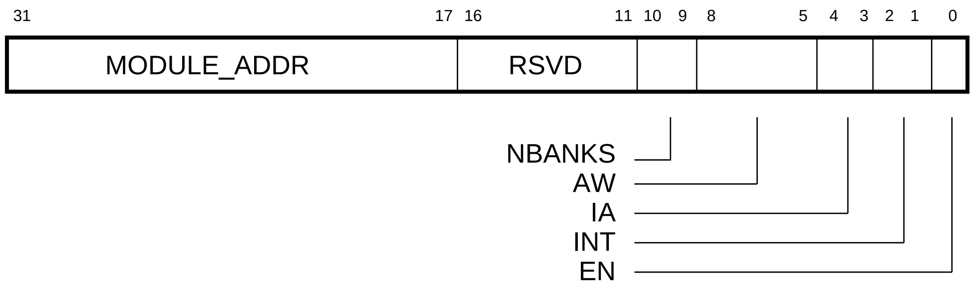
Table 2- 17 LCNR Register Bit Definitions

Name	Bits	Type	Description
STF	<31>	M, 1	**Self-test fail.** When set, indicates that this node has not yet passed self- test.
NRST	<30>	WO, 0	**Node reset.** When set, the node undergoes a reset sequence.
NHALT	<29>	M, 0	**Node halt.** When set, a processor node enters console mode. Not implemented on memory and IOP modules.
RSTSTAT	<28>	W1C, 0	**Reset status.** When set, this bit provides an indication to console software that a processor node should not attempt to become the boot processor but should instead join an already running system. This bit is set when NRST (LCNR<30>) is set. It is cleared with a write of one, at system power- up, or with an LSB RESET command. This bit is not cleared in a reset sequence caused by setting NRST. Not implemented on memory and IOP modules.
RSVD	<27:1>	MBZ, 0	**Must be zero.** Must always be written as zero.
CEEN	<0>	M, 0	**Correctable error detection enable**. When set, enables detection of correctable errors.

LMMR0–7 — Memory Mapping Registers

Address BB + 0200 – BB + 03C0
Access R/W

The LMMR registers define the configuration for all memory modules installed in the system. These registers are copies of the equivalent AMR register in each of the memory modules. Each LMMR register is associated with the LSB module in which the node ID matches the three least significant bits of the LMMR address. That is, LMMR0 is associated with node 0, LMMR1 with node 1, and so forth. The LMMR registers are loaded during system initialization when the memory modules are initialized and configured.



BXB-0104-92

Table 2-18 LMMR Register Bit Definitions

Name	Bits	Type	Description
MODULE_ADDR	<31:17>	M	**Module address.** Specifies the most significant bits of the base address of the memory region spanned by the memory module associated with this register (LMMR0–LMMR7). These bits correspond to bits <39:25> of the byte address, or D<34:20> of the command cycle.
RSVD	<16:11>	RAZ	**Reserved.** Read as zero; writes ignored.
NBANKS	<10:9>	M	**Number of banks.** Specifies the number of individual memory banks (1, 2, 4, or 8) on the memory module associated with this register (LMMR0–LMMR7). This field determines the number of bits of the memory address (0, 1, 2, or 3) that are inserted into the bank number.

LMMR <10:9>	Banks per module	Bits in bank number
00	1	0
01	2	1
10	4	2
11	8	3

Name	Bits	Type	Description
AW	<8:5>	M	**Address width.** Specifies the number of valid bits in MODULE_ADDR (LMMR<31:17>), starting from the most significant bit. The remaining bits of MODULE_ADDR are ignored.
IA	<4:3>	M	**Interleave address.** Specifies which interleave, within a group of interleaved modules, is served by the module associated with this register (LMMR0–LMMR7).
INT	<2:1>	M	**Interleave.** Specifies the number of memory modules interleaved with this module (1, 2, or 4). This value determines the number of bits in the INT field (0, 1, or 2, starting from the least significant bit) that are compared to the least significant bits of the memory address.

LMMR <2:1>	Modules interleaved	Address bits compared
00	1	0
01	2	1
10	4	2
11	Reserved	Reserved

Name	Bits	Type	Description
EN	<0>	M	**Enable.** When set, indicates that the module associated with this register (LMMR0–LMMR7) is installed and it is a memory module.

LBESR0–3 — Bus Error Syndrome Registers

Address BB + 0600 – BB + 06C0
Access RO

The LBESR registers contain the syndrome computed from the LSB Data and ECC fields received during the cycle in which an error is detected. The syndrome is the bit- by- bit difference between the ECC check code generated from the received data and the ECC field received over the bus. The LBESR registers lock only on the first occurrence of an ECC error (LBER<CE> or LBER<UCE>). Subsequent ECC errors set LBER<CE2> or LBER<UCE2> until software clears those error bits.

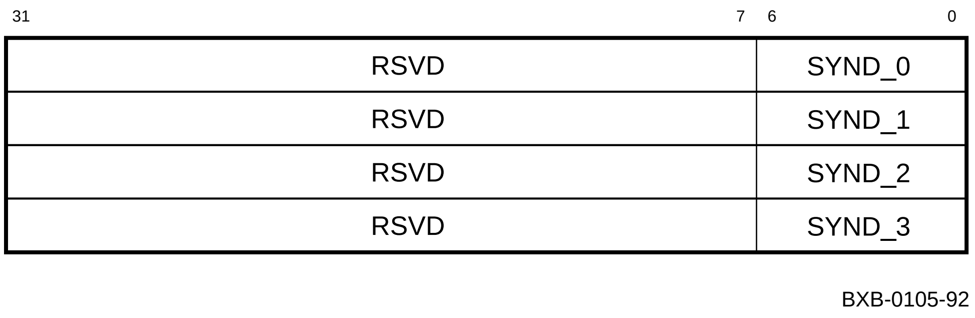


Table 2- 19 LBESR Register Bit Definitions

Name	Bits	Type	Description
RSVD	<31:7>	RO, 0	**Reserved.** Read as zero.
SYND_0	<6:0>	RO, 0	**Syndrome 0.** Syndrome computed from D<31:0> and ECC<6:0> during error cycle.
SYND_1	<6:0>	RO, 0	**Syndrome 1.** Syndrome computed from D<63:32> and ECC<13:7> during error cycle.
SYND_2	<6:0>	RO, 0	**Syndrome 2.** Syndrome computed from D<95:33> and ECC<20:14> during error cycle.
SYND_3	<6:0>	RO, 0	**Syndrome 3.** Syndrome computed from D<127:96> and ECC<27:21> during error cycle.

Syndrome Values

A syndrome of zero indicates no ECC error for the longword. Table 2- 20 gives the syndromes for all single- bit errors. Any non- zero syndrome not listed in Table 2- 20 indicates a double- bit error.

Table 2- 20 Syndromes for Single- Bit Errors

Bit	Syndrome (hex)	Bit	Syndrome (hex)
Data<0>	4F	Data<20>	16
Data<1>	4A	Data<21>	19
Data<2>	52	Data<22>	1A
Data<3>	54	Data<23>	1C
Data<4>	57	Data<24>	62
Data<5>	58	Data<25>	64
Data<6>	5B	Data<26>	67
Data<7>	5D	Data<27>	68
Data<8>	23	Data<28>	6B
Data<9>	25	Data<29>	6D
Data<10>	26	Data<30>	70
Data<11>	29	Data<31>	75
Data<12>	2A	ECC<0>	01
Data<13>	2C	ECC<1>	02
Data<14>	31	ECC<2>	04
Data<15>	34	ECC<3>	08
Data<16>	0E	ECC<4>	10
Data<17>	0B	ECC<5>	20
Data<18>	13	ECC<6>	40
Data<19>	15		

LBECR0, 1 — Bus Error Command Registers

Address BB + 0700, BB + 0740
Access RO

The LBECR registers save the contents of the LSB command and address fields during a command cycle in which an error is detected. See below for a list of errors that lock the LBECR registers.

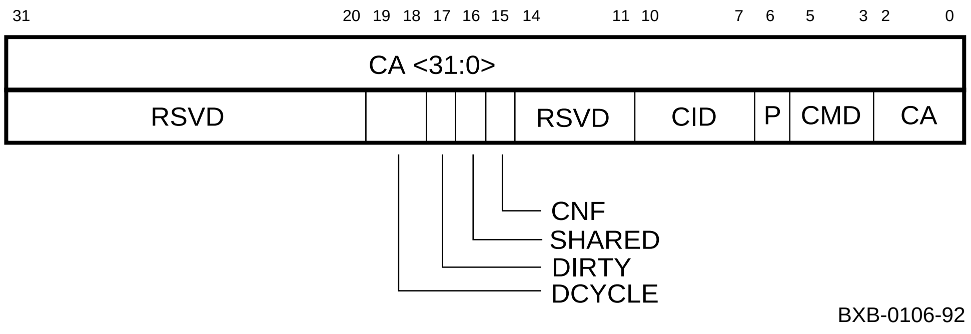


Table 2-21 LBECR Register Bit Definitions

Name	Bits	Type	Description
LBECR0			
CA	<31:0>	RO	**Command/address.** Contents of D<31:0> during the command cycle.
LBECR1			
RSVD	<31:20>	RO	**Reserved.** Read as zero.
DCYCLE	<19:18>	RO	**Data cycle.** Indicates which data cycle had data error.

LBECR<19:18>	Data cycle in error
00	0
01	1
10	2
11	3

Name	Bits	Type	Description
DIRTY	<17>	RO	**Dirty.** Set when DIRTY is asserted for the current command. Not implemented on the IOP module.

Table 2-21 LBECR Register Bit Definitions (Continued)

Name	Bits	Type	Description
SHARED	<16>	RO	**Shared.** Set when SHARED is asserted for the current command. Not implemented on the IOP module.
CNF	<15>	RO	**Confirmation.** Set when CNF is asserted for the current command.
RSVD	<14:11>	RO	**Reserved.** Read as zero.
CID	<10:7>	RO	**Commander ID.** Contents of REQ<3:0> during the command cycle.
P	<6>	RO	**Parity.** Contents of D<38> during command cycle.
CMD	<5:3>	RO	**Command.** Contents of D<37:35> during command cycle. CMD is decoded as follows:

Command	Function
000	Read
001	Write
010	Reserved
011	Write Victim
100	Read CSR
101	Write CSR
110	Reserved
111	Private

CA	<2:0>	RO	**Command/address.** Contents of D<34:32> during command cycle.

Errors That Lock the LBECR Registers

LSB uncorrectable ECC error (LBER<1>)

LSB correctable ECC error (LBER<3>)

LSB command parity error (LBER<5>)

LSB CSR data parity error (LBER<7>)

LSB nonexistent address error (LBER<12>)

LSB arbitration drop error (LMERR<10>)

LEVI P- map parity error (LMERR<3:0>)

LEVI B- cache tag parity error (LMERR<4>)

LEVI B- cache status parity error (LMERR<5>)

LEVI B- map parity error (LMERR<6>)

LILID0–3 — Interrupt Level 0–3 IDENT Registers

Address BB + 0A00 – BB + 0AC0
Access RTC

Each of the four LILID registers is the top- most (oldest) entry in a four- deep queue of interrupts for that IPL. A read to this register sends the oldest interrupt IDENT information to the processor that requests it. The next oldest interrupt IDENT information can then be read at that address. When no active interrupts exist at a specified level, a read of the corresponding LILID register returns zeros.

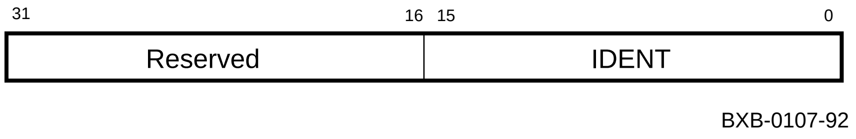


Table 2- 22 LILID Register Bit Definitions

Name	Bits	Type	Description
RSVD	<31:16>	MBZ	**Reserved.** Must be zero.
IDENT	<15:0>	RTC	**IDENT.** This field is loaded with the vector information supplied by the I/O adapter in the INTR/IDENT command packet. If the interrupt was for an IOP error, the vector is loaded from the IPC Vector Register.

LCPUMASK — CPU Interrupt Mask Register

Address	BB + 0B00
Access	R/W

The LCPUMASK register is used to determine which processor is to service a specified level of interrupts. A logical AND of the interrupt level posted from the I/O subsystem and this register form the write data sent from the IOP module to the LIOINTR register.

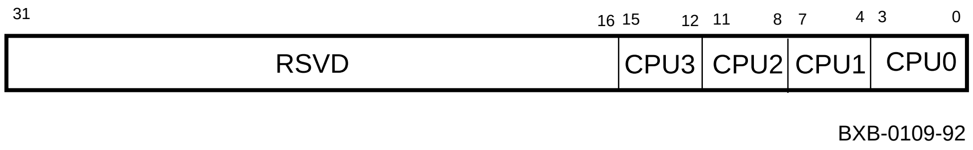


Table 2-23 LCPUMASK Register Bit Definitions

Name	Bits	Type	Description
RSVD	<31:16>	MBZ	**Reserved.** Must be zero.
CPU 3	<15:12>	R/W, 0	**IPL for processor 3.** Each bit in this field represents an IPL level. Bits <12> through <15> correspond to IPL levels 14 through 17. When a bit is set, corresponding interrupts from the I/O subsystem are posted to processor 3.
CPU 2	<11:8>	R/W, 0	**IPL for processor 2.** Each bit in this field represents an IPL level. Bits <8> through <11> correspond to IPL levels 14 through 17. When a bit is set, corresponding interrupts from the I/O subsystem are posted to processor 2.
CPU 1	<7:4>	R/W, 0	**IPL for processor 1.** Each bit in this field represents an IPL level. Bits <4> through <7> correspond to IPL levels 14 through 17. When a bit is set, corresponding interrupts from the I/O subsystem are posted to processor 1.
CPU 0	<3:0>	R/W, 0	**IPL for processor 0.** Each bit in this field represents an IPL level. Bits <0> through <3> correspond to IPL levels 14 through 17. When a bit is set, corresponding interrupts from the I/O subsystem are posted to processor 0.

LMBPR0–3 — Mailbox Pointer Registers

Address BB + 0C00
Access R/W

To access remote I/O registers, software loads the LMBPR with the 64- byte- aligned physical address of the mailbox data structure in main memory. When this register is loaded, the IOP uses the address to fetch request information contained in the mailbox data structure in memory.

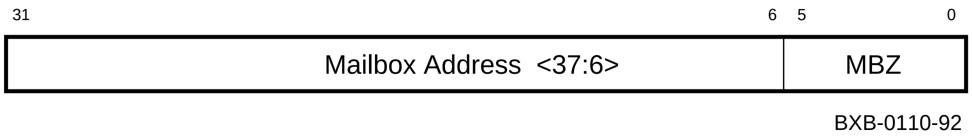


Table 2- 24 LMBPR0–3 Register Bit Definitions

Name	Bits	Type	Description
MBX	<37:6>	R/W, 0	**Mailbox data structure address.** This field contains the 64- byte- aligned physical address of the mailbox data structure in memory where the IOP module locates request information.
MBZ	<5:0>	MBZ	**Reserved.** Must be zero.

LMBPR Addresses

The I/O system architecture requires a single software address for the LMBPR. The IOP, however, implements four LMBPR registers. To prevent lockouts when multiple processors execute mailbox transactions, each processor is assigned a single LMBPR register with each register representing a two- deep queue. Therefore, eight mailbox transactions can be outstanding, two for each processor. The IOP processes only one transaction at a time. A NO ACK is returned on LMBPR register writes to a full queue. Each processor transforms the single software address of the LMBPR (A00 0C00) during the command cycle by adding the processor node ID on D<2:1> so that its designated LMBPR on the IOP is accessed.

Four LMBPR registers implies that only four of a possible six processors can access remote CSRs. The determination of the four processors that are eligible to write to the LMBPR address is specific to the type of processor.

If an LMBPR register is in use when it is written to, the IOP module does not acknowledge the write and CNF is not asserted. Processors use the lack of CNF assertion on writes to the LMBPR to indicate a busy status; the write is tried again later.

This register is write only and is 6 bits wider than a longword. Writes to this register are allowed to use more than 32 bits of the defined CSR write data field since this field is parity protected up to bit 37.

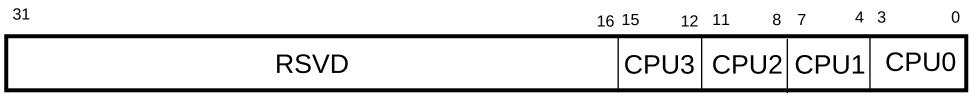
KN7AA processors access LMBPR with STQC instructions. Since this processor has a 34- bit physical address space, only bits <33:0> of LMBPR are loaded.

KA7AA processors have a 32- bit physical address space, so only bits <31:0> of LMBPR are loaded.

LIOINTR — I/O Interrupt Register

Address BSB + 0000
Access R/W

The LIOINTR register is used by the IOP module to signal interrupts from the I/O subsystem to processors.



BXB-0109-92

Table 2-25 LIOINTR Register Bit Definitions

Name	Bits	Type	Description
MBZ	<31:16>	R/W, 0	**Must be zero.** Must always be written as zero.
CPU 3	<15:12>	W1S	**Processor 3 I/O interrupt.** When a bit is set in this field, an interrupt is posted to processor 3.
CPU 2	<11:8>	W1S	**Processor 2 I/O interrupt.** When a bit is set in this field, an interrupt is posted to processor 2.
CPU 1	<7:4>	W1S	**Processor 1 I/O interrupt.** When a bit is set in this field, an interrupt is posted to processor 1.
CPU 0	<3:0>	W1S	**Processor 0 I/O interrupt.** When a bit is set in this field, an interrupt is posted to processor 0.

Interrupt Mapping

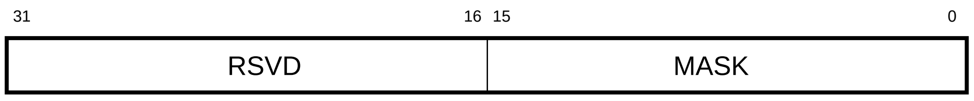
Each interrupt target is assigned four interrupt bits in the LIOINTR register corresponding to the four I/O interrupt levels. A specified processor looks only at the four bits that correspond to its target assignment. This allows interrupts to be targeted to a single processor or to as many as four processors, depending on the data supplied in the bus CSR Write transaction from the IOP module.

This register is in LSB broadcast space. Writes that address this location are accepted without regard to node ID. Thus, all processors accept writes to the register. The register bits are write one to set (W1S). Multiple writes with a value of one to a bit in this register post an equal number of interrupts to the targeted processor. Reads to this location are undefined. Each processor implements only four bits of this register.

LIPINTR — Interprocessor Interrupt Register

Address BSB + 0040
Access R/W

The LIPINTR register is used by processor modules to signal interprocessor interrupts.



BXB-0120-92

Table 2- 26 LIPINTR Register Bit Definitions

Name	Bits	Type	Description
RSVD	<31:16>	RAZ	**Reserved.** Read as zero.
MASK	<15:0>	W1S	**Interprocessor interrupt mask.** When a bit is set, an interprocessor interrupt is posted to the specified processor. Bits are mapped to specific processors within a multiprocessor system as follows:

LIPINTR Bit	Processor Number
<15:6>	Not used
<5>	Processor 5
<4>	Processor 4
<3>	Processor 3
<2>	Processor 2
<1>	Processor 1
<0>	Processor 0

Interprocessor Interrupt

To post an interprocessor interrupt to another processor, a processor writes to the relevant bit in the LIPINTR register. The bits in LIPINTR<15:0> are write one to set (W1S).

This register is in broadcast space. Writes to this location are accepted without regard to node ID. Thus, all processors accept writes to the register. Reads to this location are undefined.

2.9 Console and Initialization

The console initializes the system and bootstraps the operating system. It exists primarily in firmware with some supporting hardware. That hardware is described in this section.

2.9.1 Console Lines

The following sections describe the LSB lines that are dedicated to console support.

2.9.1.1 Console Terminal Lines

Two sets of serial lines are provided for local console terminal and power supply communication. All processor modules provide receiving and transmitting capabilities for these lines. These signals are standard OC TTL levels and are converted to RS232 levels by the cabinet control logic (CCL). Although all LSB signals, including the OC signals, are defined as being asserted true LOW, both transmit lines are inverted on the processor module before assertion on the LSB. This is because the quiescent state of UARTs is HIGH.

After power- up or system initialization, all processor modules arbitrate for the use of the common console lines. The winner is allowed to drive them. The winner asserts the LSB signal CONWIN when it determines that it has won the election. See the appropriate processor technical manual for details of the election.

2.9.1.2 RUN

The primary processor drives the LSB RUN signal. This module asserts the RUN signal when the operating system is running and deasserts the RUN signal when it is running the console or diagnostics or is in an undefined state (due to an error or lack of initialization).

2.9.1.3 CONWIN

This signal is asserted by the primary processor. The CCL connects the external console terminal lines to the power supplies when this signal is deasserted. When this signal is asserted, the processor console lines are connected to the external terminal and the processor power supply lines are connected to the power supply.

2.9.1.4 EXP and SEL

The primary procesor uses these lines to select the power regulator to which the power supply lines connect. Coding is shown in Table 2- 27.

Table 2-27 EXP and SEL Coding

EXP	SEL	Cabinet
0	0	Main cabinet power supplies.
0	1	Right expander cabinet bulk supplies.
1	0	Left expander cabinet bulk supplies.
1	1	Self-test loopback; PSTX data is returned as PSRX data.

2.9.1.5 SECURE

This signal is asserted when the keyswitch on the control panel is in the Secure position and is deasserted when the keyswitch is in any other position.

2.9.1.6 PIU Power Monitoring

The CCL receives three status signals from power converters on the plug-in units (PIUs) and provides these to processor modules for operating system and diagnostic visibility. These signals are described in the following sections.

LDC PWR OK L

This signal is derived from the LDC OK L output of individual local disk converter (LDC) power supplies. When asserted, this signal indicates that the LDCs in disk PIUs are operating correctly.

PIU MOD A OK L

This signal is derived from the power supply in an XMI, Futurebus+, or VAXBI PIU. When asserted, this signal indicates that power regulator A in each I/O PIU is operating correctly.

PIU MOD B OK L

This signal is derived from the power supply in an XMI, Futurebus+, or VAXBI PIU. When asserted, this signal indicates that power regulator B in each I/O PIU is operating correctly.

2.9.2 Control Panel Controls and Indicators

Autorestart and EEPROM update are under the control of the console software. The control panel, shown in Figure 2-16, has a single keyswitch with four positions, described in Table 2-28. It also has LEDs that provide system status, Table 2-29.

Figure 2-16 Control Panel

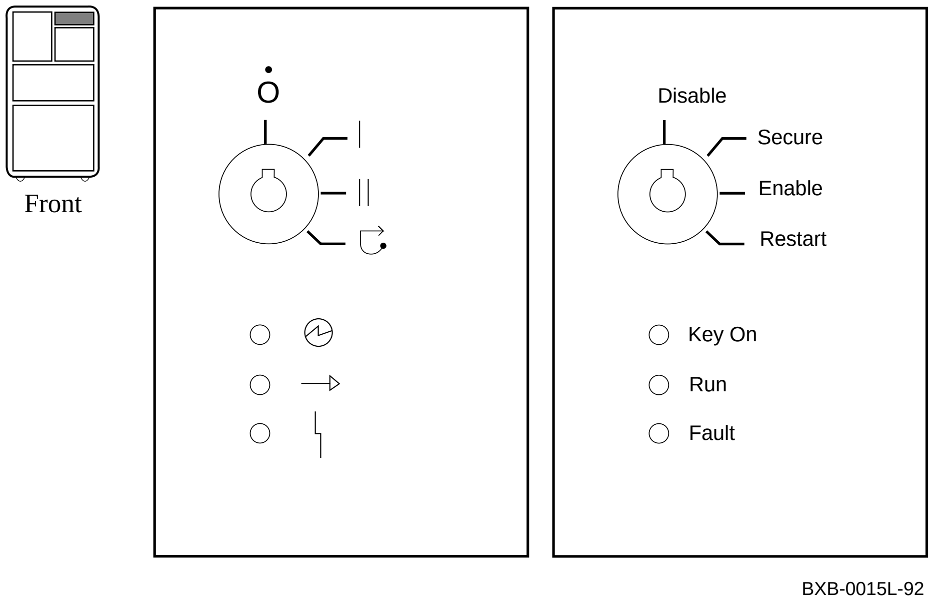


Table 2-28 Control Panel Keyswitch Positions

Position	Power Status
Disable	Removes 48 VDC power from the system. Power is still supplied to the CCL module.
Secure	Prevents entry into console mode; position used while machine executes programs.
Enable	Allows entry into console mode; position used while machine executes programs.
Restart	A momentary switch position, used to reinitialize the system; causes self-test to start running.

Table 2-29 Control Panel Indicator Lights

Light	State	Meaning
Key On (Green)	On	Power supplied to entire system; blower running.
	Off	Power supplied only to CCL module.
Run (Green)	On	Primary processor is running the operating system or user programs.
	Off	Primary processor is in console mode.
Fault (Yellow)	On	Fault on LSB or an I/O bus.
	Slow flash	Power sequencing in progress or airflow error.
	Fast flash	Power system error, airflow error, or detected transition to keyswitch in Disable position.
	Off	No faults were found.

2.9.3 Initialization Mechanisms

The system is reset when a processor node asserts the LSB_RESET signal, when the control panel keyswitch is turned to Restart, or when the IOP module asserts the CCL_RESET signal. When the control panel keyswitch is in the Restart position, CCL_RESET is asserted and LSB_RESET is not asserted. Processor modules use the assertion of CCL_RESET to clear a counter. When CCL_RESET is deasserted, LSB_RESET is asserted for 510 LSB cycles. LSB_RESET is synchronous with the LSB clocks and also finite in duration. The IOP module may also assert and deassert CCL_RESET for system resets initiated by I/O devices.

2.9.3.1 Power-Up

From the perspective of software or firmware, the power system can be in one of two states, either on or off. The source of 48V to an LSB module can be from either the AC line or batteries; the source is irrelevant to software or firmware. LSB modules detect a power-up condition when 48V is present.

2.9.3.2 Power Failure

The power regulators notify the processor and the operating system of an impending power failure (for example, when batteries are low). The operating system can then take appropriate action.

2.9.3.3 System Reset

When LSB_RESET is asserted, all LSB nodes reset all internal logic to a consistent state, from which an orderly system startup can proceed. Sixteen cycles after LSB_RESET is deasserted, nodes may begin requesting access to the bus through assertion of the REQ lines. When the LSB is reset, LSB nodes (with the exception of the IOP) assert BAD to indicate that they have not successfully completed self-test. The BAD signal is not synchronous with the LSB clock.

Each LSB module has a green LED that lights whenever the module is not asserting BAD.

2.9.3.4 Node Reset

When the NRST bit in a node's LCNR register is asserted, all logic on the module is reset, as though the LSB_RESET signal were asserted. To assure that NRST does not cause a processor node to become unsynchronized with the rotating REQ priority system during the NRST, the node asserts ERR. This forces resynchronization of the REQ lines.

While a node is being reset, it may not recognize CSR accesses or memory accesses to it.

2.9.4 Self-Test

All LSB nodes are tested following power-up, system reset, or setting LCNR<NRST>. The state of LCNR<STF> indicates the results of self-test. The IOP module is tested by a processor node, so its self-test status is not reported until after the processor node completes its own self-test.

2.9.5 Module Regulator Failure

DEC 7000 and VAX 7000 systems use a distributed power system. This means that each LSB module has a regulator to create its own +5 and other voltages from a common 48V supply. If one module on the LSB has a defective regulator, the 2V LSB reference voltage is disabled for all nodes, thus preventing any node from using the LSB. Since completion of self-test diagnostics depends on LSB access, the diagnosis of a failing module regulator is different from a logic failure. When a module regulator fails, the green LED must be used in conjunction with the console terminal to diagnose the failure. The following sections explain the symptoms of a module regulator failure.

All Module Regulators Are Operational

If all module regulators are good and all modules pass self-test, the BAD signal is deasserted and the yellow Fault LED on the control panel is off. The green self-test passed LEDs on all modules light.

Regulator Failure on a Processor Module

If one processor module's regulator is bad but all others are good, the yellow Fault LED on the control panel may or may not light. The green self-test passed LED on the processor module with the bad regulator does not light; this LED does light on all other processor modules. (This assumes that only one processor module has a bad regulator.) Additionally, the console software does not arbitrate for the primary processor, and the default console display is printed. The green self-test passed LEDs on all memory modules should light, but the IOP self-test passed LED does not light. This is because the processor needs access to the LSB to provide the self-test for the IOP module.

Regulator Failure on a Memory Module

If one memory module's regulator is bad but all others are good, the yellow Fault LED on the control panel may or may not light. The self-test passed LED on the memory module with the bad regulator does not light; this

LED does light on all other memory modules. (This assumes that only one memory module has a bad regulator.) The green self- test passed LEDs on all processor modules should light, but the IOP self- test passed LED does not light. The default console display is printed.

Regulator Failure on the IOP Module

If the IOP module's regulator is bad but all others are good, the yellow Fault LED on the control panel does not light. The self- test passed LED lights on memory and processor modules, and the default console display is printed. The self- test passed LED on the IOP module does not light. This prohibits self- test passed LEDS from lighting on the interface modules to XMI and Futurebus+ buses.

Table 2- 30 Fault Matrix for Module Regulators

Processor Modules	Memory Modules	IOP Module	Defective Regulator on This Module
One module's STP LED is off; all others are on.	All STP LEDs are on.	STP LED is off.	Processor module with STP LED off
All STP LEDs are on.	One module's STP LED is off; all others are on.	STP LED is off.	Memory module with STP LED off
All STP LEDs are on.	All STP LEDs are on.	STP LED is off.	IOP module

If the FAULT LED on the control panel lights and the console prompt does not print, the LSB is good. The green self- test LEDs do not light on the failing module or modules, and the console self- test display reports the failure. If the system has only one processor, and both the processor and IOP LEDs are off, the processor is bad.

2.10 Errors

Because of the high level of integration and the constrained electrical environment, the LSB is highly reliable. Consequently, the error handling facilities are oriented toward detection of errors, since, in many cases, recovery from errors is not possible. The sole exception to this philosophy is in the handling of memory data, which is covered by error correcting codes (ECC). ECC is used with memory data because the memory chips themselves are expected to suffer occasional soft errors induced by alpha particles. ECC- protected data is passed throughout the LSB environment, permitting correction of individual bit errors in memory data cycles on the bus.

Every node on the LSB monitors the bus and reports any errors, as described in this section. In general, a module continues processing a transaction, even if errors (other than command parity errors) are detected during the transaction. If bad parity is received during a command cycle, the transaction is ignored by all nodes detecting the parity error.

2.10.1 ECC Errors

Each module monitors all memory data cycles for ECC errors. If an uncorrectable error occurs, the module that detects the error asserts ERR (for one cycle) within four cycles after the uncorrectable data appears on the bus. In addition, the module that detects the error captures the following state:

- The ECC syndromes are captured in LBESR0 through LBESR3.
- The command cycle associated with the failed data cycle is captured in LBECR0 and LBECR1.
- The CNF, SHARED, and DIRTY status associated with the command is captured in LBECR1.
- The data cycle (0, 1, 2, or 3) is captured in LBECR1.
- The UCE bit of LBER is set.

The same actions occur for correctable data errors, but only if the CEEN bit in LBCNF is set. In the case of a correctable error, the CE bit of LBER is set, rather than the UCE bit.

Once error syndromes have been captured in the LBESR registers, no additional error data is captured until after the UCE bit and the CE bit are cleared. These bits are cleared by writing a one to them in the LBER. Note that an uncorrectable error that occurs while CE is set causes UCE to be set. Similarly, a correctable error that occurs after UCE is set causes CE to be set (assuming that the CEEN bit in LBCNF is set).

2.10.2 Parity Errors During C/A Cycles

Each module monitors all bus cycles for command parity errors. Any module that detects assertion of CA and even parity on D<38:0> does the following:

- Asserts ERR (for one cycle) within four cycles after the parity error appears on the bus.
- Captures D<38:0> of the command cycle in LBECR0 and LBECR1.

- Sets the CPE bit of LBER.

If a module detects a command cycle parity error, it ignores the remainder of the transaction. Consequently, the module does not check the associated data cycles for parity or ECC errors.

2.10.3 Parity Errors During CSR Data Cycles

Processor modules and the IOP module monitor all CSR data cycles for parity errors. If a parity error occurs, the module that detects the error asserts ERR (for one cycle) within four cycles after the parity error appears on the bus. In addition, the module captures the following state:

- The command cycle associated with the failed data cycle is captured in LBECR0 and LBECR1.
- The CNF, SHARED, and DIRTY status associated with the command is captured in LBECR1.
- The data cycle field in LBECR1 is set to zero.
- The CDPE bit of LBER is set.

2.10.4 Double Errors

For each of the error bits, CE, UCE, CPE, and CDPE, there is a corresponding second bit (CE2, UCE2, CPE2, and CDPE2) that indicates one or more additional errors of this type have occurred while the original error bit was set.

2.10.5 Transmitter During Error

The TDE bit is set by the node that is driving the bus when a parity error or ECC error occurs. Specifically, when a module sets CE, UCE, CPE, or CDPE, it also sets the TDE bit in LBER if it was driving the D<127:0> lines during the cycle in which the error occurred on the bus.

2.10.6 STALL Errors

If a node detects the STALL signal asserted during a cycle in which STALL is not permitted, the node asserts ERR (for one cycle) within four cycles, and sets the STE bit in LBER.

2.10.7 CNF Errors

If a node detects the CNF signal asserted during a cycle in which CNF is not permitted, the node asserts ERR (for one cycle) within four cycles, and sets the CNFE bit in LBER.

2.10.8 Nonexistent Address Errors

If a commander detects the CNF signal deasserted during a cycle in which CNF should be asserted, the node asserts ERR (for one cycle) within four cycles, and sets the NXAE bit in LBER.

The NXAE bit is not set during CSR accesses to LMBPR, even if CNF is not asserted.

NOTE: *Unlike other error bits, NXAE is set only by commanders. This avoids the need for other nodes to check for LMBPR address, and makes it easier to use NXAE during system configuration.*

2.10.9 CA Errors

If a node detects the CA signal asserted during a cycle in which CA is not permitted, the node asserts ERR (for one cycle) within four cycles, and sets the CAE bit in LBER.

2.10.10 SHARED Errors

If a node detects the SHARED signal asserted during a cycle in which SHARED is not permitted, the node asserts ERR (for one cycle) within four cycles, and sets the SHE bit in LBER.

2.10.11 DIRTY Errors

If a node detects the DIRTY signal asserted during a cycle in which DIRTY is not permitted, the node asserts ERR (for one cycle) within four cycles, and sets the DIE bit in LBER.

2.10.12 Data Transmit Check Errors

When a node drives a data cycle or command cycle onto the bus, it checks the received data on the D<127:0> and ECC<27:0> lines at the end of the cycle to verify that the received data matches the data that was sent. If there is a mismatch, the node asserts ERR (for one cycle) within four cycles, and sets the DTCE bit in LBER. All bits of D<127:0> and ECC<27:0> are checked during memory data cycles. During command cycles and the first data cycle of CSR transactions, D<38:0> must be checked, and some or all of D<127:39> and ECC<27:0> may optionally be checked. During the second, third, and fourth data cycles of CSR transactions, all bits of D<127:0> and ECC<27:0> may optionally be checked. Stalled data cycles are checked in the same manner as the first data cycle of the transaction.

2.10.13 Control Transmit Check Errors

When a node drives any control line, it checks the received status on that control line at the end of the cycle to verify that the line was asserted on the bus. If there is a mismatch, the node asserts ERR (for one cycle) within four cycles, and sets the CTCE bit in LBER.

2.10.14 Node-Specific Error Summary

The node-specific error summary (NSES) bit in LBER may be set by a node that has detected an internal error. The node may assert ERR when this bit is set, but it is not required to do so.

2.10.15 Monitoring ERR

When a node detects ERR asserted, it does the following:

- Sets the E bit in LBER.
- Inhibits outgoing arbitration for the next 16 cycles.
- If the node is a processor, resets its arbitration priority according to NID<1:0>. (See Section 2.2.)
- If the node is a processor, reads the LSB error registers and takes appropriate action.

2.10.16 LSB Error Recovery

The error behavior of a node (in response to detection of bits set in the LBER) is module specific. Memory modules take no action. Processors take some appropriate action which may vary depending on the type of processor and operating system.

Effect of Read Errors

If an uncorrectable data error occurs during a read operation, the LSB interface reports the uncorrectable read error to the requester in place of the requested data. The requester can then take appropriate action, such as retrying the read, signaling the read failure to the process, or terminating the transaction.

In addition, the error bits set in the LSB error registers may cause an interrupt to one or more processors, if enabled. Software can determine from the contents of the LSB error registers that the error occurred during a Read operation, and can then ignore the error, log it, or take other appropriate action.

Effect of Write Errors

If an uncorrectable data error occurs during a Write operation, the Write operation in the requester presumably is complete, and no recovery is possible. The error bits set in the LSB error registers may cause an interrupt to one or more processors, if enabled. Software can determine from the contents of the LSB error registers that the error occurred during a Write operation, and can then take appropriate action.

Chapter 3

Power, Cabinet Control, and Cooling Systems

This chapter describes the power, cabinet control, and cooling systems of the system and expander cabinets. Sections include:

- Power System
  - AC Input Box
  - DC Distribution Box
  - Power Regulator
  - Uninterrupted Power Supply
  - Module Regulators
  - 48V DC Bus
- Cabinet Control System
  - CCL Power Source
  - Cabinet Serial Lines
  - Power Sequencing
  - Power Fail Operation
- Cooling System

3.1 Power System

The power system consists of these components:

- AC input box
- DC distribution box
- Power regulators (AC- to- DC)
- Optional uninterrupted power supply
- 48V to 5V DC- to- DC module regulators
- Cabinet control system
- 48V bus
- Signal interconnect cables

The cabinet control system is described in Section 3.2, and cables are listed in Appendix A. Other components of the power system are described below.

3.1.1 AC Input Box

The AC input box provides the interface to the AC line through a three-phase, five- wire power cord. The AC input box includes the main input circuit breaker, AC power distribution to the power regulators, EMI filter, Dranetz port connector, and line transient protection. The main circuit breaker, which is the primary electrical disconnect for the cabinet (including batteries), can be locked in the Off position for the safety of service personnel. The Dranetz port is a nine- pin universal Mate- N- Lok connector. By attaching a monitoring device to this port, the service engineer can read the status of the AC line during system operation.

3.1.2 DC Distribution Box

The DC distribution box is the 48V DC bus distribution point, the interface from the battery packs to the power regulators, and the signal interconnect from the power regulators to the cabinet control system. The DC distribution assembly houses the AC input box and power regulators.

3.1.3 Power Regulator

Each cabinet has a minimum of one and a maximum of three power regulators. The power regulator is a 2.13 Kwatt power supply that operates on a single phase AC power with power factor correction. Each power regulator has three outputs: 48V DC bus at a maximum current of 45A, auxiliary 48V DC at a maximum of 0.25A, and 48V lead acid battery charger.

The number of regulators required, either one or two, depends on the options in the cabinet. An additional power regulator may be used for optional N+1 redundancy. One power regulator can handle a maximum of 85 equivalent power units (EPUs). More than 85 EPUs in a cabinet requires a second power regulator.

The cabinet itself uses 30 EPUs; add to that the EPU values for any equipment in the cabinet from the following tables. Table 3- 1 lists the EPUs for LSB and cabinet options, and Table 3- 2 lists the EPUs for PIU options. Derivation of EPU values is shown in Appendix B.

Table 3- 1 Equivalent Power Units — LSB and Cabinet

Option	EPU Value
KA7AA- AA processor	7
KN7AA- AA processor	7
KN7AA- YA processor	7
MS7AA- AA 64 Mbyte memory	10
MS7AA- BA 128 Mbyte memory	10
MS7AA- CA 256 Mbyte memory	10
MS7AA- DA 512 Mbyte memory	10
MS7BB- AA BBU memory	10
DWLMA- AA XMI PIU	4
SF73- LA disk brick	8
SF74- LA disk brick	8
EF51R- LA SSD	8
EF52R- LA SSD	8
RZ26- VA	1
TLZ06- VA/TZK09- VA	1
RZ73- VA/RZ74- VA	2

Table 3- 2 Equivalent Power Units — Plug- In Units

Module	EPU Value
XMI Options	
CCARD- YA (T2030)	1
CIXCD- AC	4
DEMFA- AA	5
DEMNA- M	3
DWLMA- AA/BA	3
DWMBB- AA	2
DWMBB- LA	3
DWMBB- LB	3
DWMVA- AA	1
KDM70- AA	6
KFMSA- BA	3
KZMSA- AB	3

Table 3- 2 Equivalent Power Units —Plug- In Units (Continued)

Module	EPU Value
VAXBI Options	
DMB32- M	3
DRB32- E	3
DRB32- M	3
DRB32- W	4
DWMBB	2
Futurebus+ Options	
DEFAA	2
DWLAA- AA/BA	2

The total output of a power regulator, including battery charger, is limited to approximately 2400 watts (consistent with line cord ratings of 30A maximum for 180–220V AC or 16A for 380–415V AC).

Each power regulator has a built- in battery charger and discharger for uninterrupted power supply (battery backup) operation.

3.1.4 Uninterrupted Power Supply

The optional uninterrupted power supply is housed in a plug- in unit in the bottom of the cabinet. One battery pack is required for each power regulator in the cabinet; the battery PIU can house one to three battery packs.

Each DEC 7000/VAX 7000 battery pack consists of four 12V, 32Ah batteries (Sonnenschein A- 500 series) connected in series to provide 48V DC for 11 minutes of full system operation when fully charged and new. The amount of time increases to approximately 20 minutes in an N+1 configured system. The exact time depends on the battery charge state, the age of the batteries, the 48V DC load current, and the number of power regulators in the cabinet.

Each DEC 10000/VAX 10000 battery pack consists of eight batteries of the type described above. These batteries provide one hour of full system operation.

The battery option is managed by the power regulators, which provide battery charge power, enable circuitry, status checking, and test capability. The nominal battery charge current is 2–3 amps to a discharged battery in constant current mode until the float cutoff voltage (which is temperature dependent) is reached. At that point the charge becomes a constant voltage source.

Battery end of life occurs when the battery pack is capable of supplying only 8 minutes (DEC 7000/VAX 7000 systems) or 16 minutes (DEC 10000/ VAX 10000 systems) of full system operation. Battery packs are recharged to within 98% of capacity in less than 18 hours.

3.1.5 Module Regulators

Each LSB module has an onboard regulator that creates the required logic and termination voltages. The module regulator operates from a nominal 48V input and supplies a nominal 5V output. The maximum power rating is 150 watts (30 amps).

The module regulator has the means of enabling the output. An input signal is used to enable the output of the regulator when activated (active low). The regulator has overcurrent protection and output overvoltage protection.

The regulator is mounted on the module with solderable pins.

Table 3- 3 is a summary of the input and output characteristics of the module regulator.

Table 3- 3 Input and Output Characteristics of the Module Regulator

Parameter	Minimum	Typical	Maximum
Input voltage range (operating)	40V	48V	60V
Average input current (at low line, full load)	—	—	5A DC
Total output regulation	4.75V	5.00V	5.25V
Output load range	0A	—	30A

3.1.6 48V DC Bus

The 48V DC bus is a prewired cable harness that connects to the LSB, blower, removable media LDC, and the PIU quadrants. The 48V DC bus bars are located on the rear of the DC distribution box. The cable connects to the blower and removable media LDC by Mate- N- Lok connectors, it is hard mounted to the LSB centerplane, and it is connected to the PIU quadrants by blind- mated connectors. (When a plug- in unit is installed in the cabinet, the 48V DC bus and CCL signals are immediately connected.)

3.2 Cabinet Control System

The cabinet control system (CCS) is the physical connection between the control panel, power system, and LSB modules. The CCS consists of the cabinet control logic module (CCL) and cables, and it performs the following functions:

- Power sequencing for regulators on LSB modules
- Power sequencing and reset control of power regulators in I/O plug- in units
- Centralized generation of RESET and ACLO signals during power sequencing
- Conversion of serial line levels
- Monitoring of blower status
- Detection of over- temperature condition in the upper cabinet
- Routing of power status signals from PIUs to the LSB

The CCL is a dedicated module for control functions in the cabinet.

3.2.1 CCL Power Source

The CCL generates its +5V logic voltage and +12 and −12 EIA voltages from the auxiliary 48V DC supply voltage. The auxiliary 48V DC is brought to the CCL module by the DC distribution box- to- CCL cable.

The CCL maintains a common ground reference among the LSB centerplane, the backplanes in I/O PIUs, the control panel, and itself. It does not maintain a common ground reference with the DC distribution box or the power regulators.

The 48V supply voltage to the CCL module is a dedicated power source generated by the power regulators separate from the 48V bus that is distributed throughout the cabinet. The auxiliary 48V supply voltage is present whenever AC line voltage is available and cabinet circuit breakers are closed. The CCL has 250mA of 48V DC available when one power regulator is installed. The power regulators do not implement current sharing for the CCL 48V power source.

The CCL state machine is held in a reset condition when the keyswitch is in the Disable position. The CCL internal logic and state machine may be reset by turning off the control panel keyswitch.

3.2.2 Cabinet Serial Lines

The cabinet implements two transmit and receive pairs of serial data lines. These lines have various sources and destinations depending on the operating mode of the system. The CCL module is the distribution point for these lines, which include a console serial line and a serial line to the power regulators. The logic level of each serial line is converted by the CCL to a format appropriate for its destination.

3.2.3 Power Sequencing

Power sequencing begins when AC power and 48V auxiliary power are present on the CCL and the circuit breaker at the AC input box is closed. When power sequencing begins, the power regulators complete their internal self- tests and wait for the ON COMMAND signal from the control panel. When this signal is received, the 48V DC output is enabled.

The 48V DC auxiliary power source is converted to logic voltages on the CCL. When 48V auxiliary is first applied to the CCL, an onboard power supply monitor chip holds the CCL in a reset condition for 400 ms after internal logic voltages stabilize. When the keyswitch is turned on, and after the blower spins up, logic on the CCL begins power- up sequences for the LSB centerplane and any I/O backplanes and disks in the cabinet. After the release of the reset signals, modules in the LSB centerplane and I/O backplanes perform self- test and system operation begins.

3.2.4 Power Fail Operation

If the optional uninterrupted power supply is installed, it is enabled when the control panel keyswitch is in the Enable position and an AC power failure occurs. The CCL has no indication that the system is operating on battery- backed- up power. Power status is available from the serial line interface from the power regulators to the operating system or the console firmware.

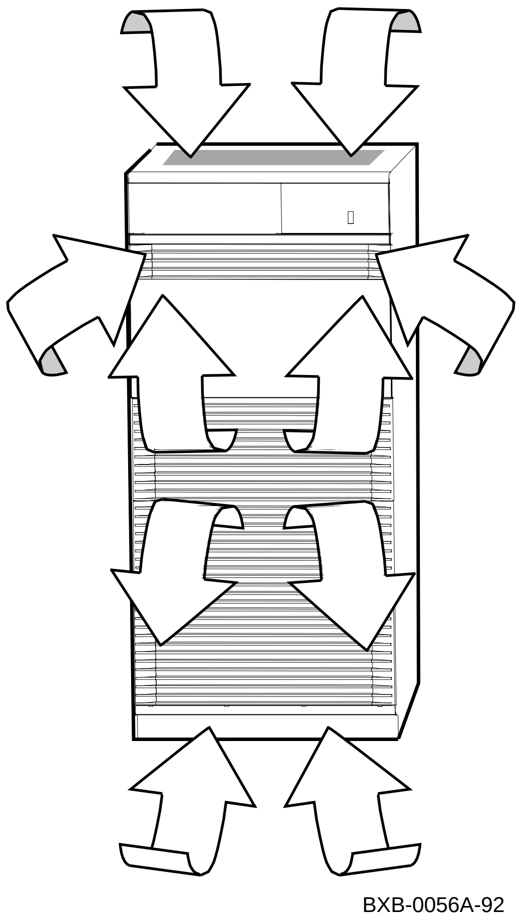
As the batteries discharge during battery- backed- up operation and their output levels drop, the regulators deassert MOD OK, causing the CCL to execute a power- down reset sequence. This reset sequence is followed by the disabling of the module regulators in the cabinet. The CCL state machine then enters an idle loop waiting for AC main power to return (providing the power regulators are maintaining 48V DC auxiliary output). When AC power is gone and the batteries are depleted, power is removed from the CCL.

The CCL powers the system back on without operator intervention when AC main power is restored.

3.3 Cooling System

The cooling system consists of a blower, the CCL module, and connecting cables. The blower is located in the midsection of the cabinet. It draws air downward through the power regulators and LSB centerplane (or optional disk PIUs in the upper part of the expander cabinet) and upward through PIUs in the lower part of the cabinet. Figure 3- 1 shows airflow. The blower supplies airflow whenever the 48V DC bulk power is present.

Figure 3- 1 Airflow in System and Expander Cabinets



The blower generates the signal BLOWER OK, which indicates that blower speed is within specification.

Appendix A

CCL Cables

The CCL module has six connectors for signal cables that provide the physical interface to the cabinet. It also has connectors for expander cabinet power enable and communication and for the blower status signal. The signal cable connectors radiate from the CCL and connect to the LSB, control panel, I/O and storage PIUs, removable media power supply, and DC distribution box. The CCL has a remote input that enables the power regulators in an expander cabinet configuration.

Table A- 1 describes the CCL cables. The remaining tables list the signals in each cable. The signal types used in these tables are defined as follows:

- OPTO —Part of an opto- isolated wire pair.
- OC —Driven as an open collector output and received as a TTL level.
- TTL —Driven and received as a TTL- compatible signal.
- ANALOG —Represents an analog voltage.

Table A- 1 Cable Descriptions

Cable	Part Number	Description
CCL to LSB	17–03121–01	50- pin flat conductor. Path for module regulator signals and some LSB signals.
CCL to control panel	17–03120–01	40- pin flat conductor. Provides the signal path for control panel switches and indicators and for the console. Power for pull- up resistors and LEDs on the control panel is supplied through this cable.
CCL to DC distribution box	17–03124–01	20- conductor cable. The signals listed in Table A- 4 are passed between the power regulators and the CCL through the DC distribution box.
CCL to plug- in units	17–03119–01	6- connector custom harness with a 40- pin header at the CCL. Connects the DC distribution box, CCL, and the four PIUs in the bottom of the cabinet.
CCL to blower	17–03126–01	2- conductor cable.
CCL to removable media	17–03123–01	30- conductor cable. Provides enable and ACOK signals for the removable media device and, in expander cabinets, for disk PIUs located in the upper quadrants.

Table A- 2 CCL to LSB Cable

Signal Name	Type	Description
IOP ONCMD	OPTO	Enable control signal for the regulator on the I/O port module. Return is 48V RTN.
FRONT ONCMD	OPTO	Enable control signal for the regulators on modules plugged into the front of the LSB centerplane. Return is 48V RTN.
BACK ONCMD	OPTO	Enable control signal for the regulators on modules plugged into slots 4 to 8 of the LSB centerplane. Return is 48V RTN.
SECURE	TTL	Driven by the CCL; indicates that the rotary switch on the control panel is in the Secure position.
LSB PSTX	OC	Driven by a processor module; contains power regulator transmit serial data.
LSB PSRX	TTL	Driven by the CCL; contains bulk power supply receive serial data.
LSB CONWIN	OC	Driven by a processor module; indicates a primary processor has been selected. Signal is used by the CCL to connect the CL_PSTX and CL_PSRX serial lines to the PSTX and PSRX serial lines.
LSB RUN	OC	Driven by a processor module; buffered and passed to the control panel to light the Run LED.
PIU MOD B OK	TTL	When asserted, indicates that power regulator B has not failed in any I/O PIU.

Table A-2 CCL to LSB Cable (Continued)

Signal Name	Type	Description
PIU MOD A OK	TTL	When asserted, indicates that power regulator A has not failed in any I/O PIU.
CCL SPARE0	OC	Spare signal.
LDC PWR OK	TTL	When asserted, indicates that no local disk converter modules in the disk PIUs have failed.
CCL SPARE1	OC	Spare signal.
CCL SPARE2	OC	Spare signal.
LSB EXPSEL 0	OC	Driven by a processor with expander cabinet select bit 0.
LSB EXPSEL 1	OC	Driven by a processor with expander cabinet select bit 1.
LSB LOCRX	TTL	Driven by the CCL with local console receive serial data.
LSB LOCTX	OC	Driven by a processor with local console transmit serial data.
CCL RESET	OC	Driven by the CCL or LSB modules for system reset requests.
LSB BAD	OC	Driven by all LSB modules until self- test successfully completes.
GB2CCLSPA	OC	CCL spare signal.

Table A-3 CCL to Control Panel Cable

Signal Name	Type	Description
VCC	POWER	5V power source from the CCL.
KEY ON	TTL	Key on indication to the CCL sequencer.
FAULT LED	OC	Fault LED driver signal, derived from LSB BAD.
RUN LED	OC	Run LED driver signal, derived from RUN.
ON CMD	ANALOG	Isolated switch closure to enable the power regulators.
ON CMD RTN	ANALOG	ON CMD return line.
DEC PB REQ	OPTO	DEC power bus enable signal.
DEC PB RTN	OPTO	DEC power bus enable return signal.
SECURE	OC	Control panel secure mode indication signal.
PANEL RESET	TTL	Momentary reset switch position that results in the assertion of CCL RESET by the cabinet sequencer.
LEFT CL PSTX	OPTO	Left expander cabinet transmit data.
LEFT CL PSTX RTN	OPTO	Left expander cabinet transmit data return.
LEFT CL PSRX	OPTO	Left expander cabinet receive data.
LEFT CL PSRX RTN	OPTO	Left expander cabinet receive data return.
RIGHT CL PSTX	OPTO	Right expander cabinet transmit data.

Table A-3 CCL to Control Panel Cable (Continued)

Signal Name	Type	Description
RIGHT CL PSTX RTN	OPTO	Right expander cabinet transmit data return.
RIGHT CL PSRX	OPTO	Right expander cabinet receive data.
RIGHT CL PSRX RTN	OPTO	Right expander cabinet receive data return.
EIA LOCRX	EIA	Local console EIA receive data.
EIA LOCTX	EIA	Local console EIA transmit data.
GND	POWER	TTL ground reference from the CCL.
OCP PRES	TTL	Control panel present indication.

Table A-4 CCL to DC Distribution Box Cable

Signal Name	Type	Description
ON CMD	ANALOG	Isolated switch closure to enable the power regulators.
ON CMD RTN	ANALOG	ON CMD return line.
LPS OK A	OPTO	48V output OK line from power regulator A.
LPS OK RTN A	OPTO	LPS OK signal return line from the power regulators.
LPS OK B	OPTO	48V output OK line from power regulator B.
LPS OK RTN B	OPTO	LPS OK signal return line from the power regulators.
LPS OK C	OPTO	48V output OK line from power regulator C.
LPS OK RTN C	OPTO	LPS OK signal return line from the power regulators.
CL PSTX	OPTO	Transmit data from the power regulators. This signal is converted to a TTL level and is used to drive the PSRX signal on the CCL module. The null modem function is performed on the CCL.
CL PSTX RTN	OPTO	Return path for CL PSTX.
CL PSRX	OPTO	Receive data to the power regulators. The input to this signal is created from the open collector signal PSTX on the CCL. The null modem function is performed on the CCL.
CL PSRX RTN	OPTO	Return path for CL PSRX.
+48V AUX	POWER	Dedicated 48V CCL power source; two pins provided.
+48V AUX RTN	POWER	48V CCL power source return; two pins provided.
SPARE		The cable has four spare pins.

Table A-5 CCL to Plug-in Unit Cable

Signal Name	Type	Description
PIU 1 Signals		
PIU 1 EN	TTL	PIU power supply enable signal.
PRM RESET A	OC	Received and driven by the CCL during various reset sequences.
PRM ACLO A	OC	Derived from the MOD OK signals from the power regulators.
PRM DCLO 1	OC	Derived from the MOD OK signals from the power regulators plus a delay time.
ACOK 1	OC	Driven by the CCL and is used by DSSI disk units as a control signal; it signals the drives to spin up.
PIU1 MOD A	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator A.
PIU1 MOD B	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator B.
PIU1 LDC OK	OC	Driven by the power supplies in a disk PIU to indicate the status of its internal LDC power supplies.
PIU 2 Signals		
PIU 2 EN	TTL	PIU power supply enable signal.
PRM RESET A	OC	Received and driven by the CCL during various reset sequences.
PRM ACLO A	OC	Derived from the MOD OK signals from the power regulators.
PRM DCLO 2	OC	Derived from the MOD OK signals from the bulk power supplies plus a delay time.
ACOK 2	OC	Driven by the CCL and used by DSSI disks as a control signal; signals the drives to spin up.
PIU2 MOD A	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator A.
PIU2 MOD B	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator B.
PIU2 LDC OK	OC	Driven by the power supplies in a disk PIU to indicate the status of its internal LDC power supplies.
PIU 3 Signals		
PIU 3 EN	TTL	PIU power supply enable signal.
PRM RESET B	OC	Received and driven by the CCL during various reset sequences.
PRM ACLO B	OC	Derived from the MOD OK signals from the power regulators.
PRM DCLO 3	OC	Derived from the MOD OK signals from the power regulators plus a delay time.
ACOK 3	OC	Driven by the CCL and used by DSSI disks as a control signal; signals the drives to spin up.
PIU3 MOD A	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator A.
PIU3 MOD B	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator B.

Table A-5 CCL to Plug-in Unit Cable (Continued)

Signal Name	Type	Description
PIU3 LDC OK	OC	Driven by the power supplies in a disk PIU to indicate the status of its internal LDC power supplies.
PIU 4 Signals		
PIU 4 EN	TTL	PIU power supply enable signal.
PRM RESET B	OC	Received and driven by the CCL during various reset sequences.
PRM ACLO B	OC	Derived from the MOD OK signals from the bulk power supplies.
PRM DCLO 4	OC	Derived from the MOD OK signals from the bulk power supplies plus a delay time.
ACOK 4	OC	Driven by the CCL and is used by DSSI disks as a control signal; signals the drives to spin up.
PIU4 MOD A	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator A.
PIU4 MOD B	OC	Driven by the power supply in I/O PIUs to indicate the status of power regulator B.
PIU4 LDC OK	OC	Driven by the power supplies in a disk PIU to indicate the status of its internal LDC power supplies.

Table A-6 CCL to Blower Cable

Signal Name	Type	Description
BLOWER OK	OPTO	Generated by the cabinet blower to indicate that cabinet airflow and temperature are correct.
BLOWER OK RTN	OPTO	BLOWER OK return line.

Table A-7 CCL to Removable Media Cable

Signal Name	Type	Description
LSB PIU 1 EN	TTL	PIU 5 power supply enable.
ACOK 1	TTL	Driven by the CCL. Signals the drives in PIU 5 to spin up.
LSB PIU 2 EN	TTL	PIU 6 power supply enable.
ACOK 2	TTL	Driven by the CCL. Signals the drives in PIU 6 to spin up.
REM MEDIA ACOK	TTL	Driven by the CCL. Signals the drive in the removable media device to spin up.
IOP ON	TTL	Enables the removable media power supply.

Appendix B

EPU Calculations

Table B- 1 Calculation of Equivalent Power Units —LSB and Cabinet

Option	Watts @ 48V	Milliamps @ 48V	EPU (W @ 48V/20[1])	Rounded EPU Value
KA7AA- AA processor	140.00	2916	7.00	7
KN7AA- AA processor	140.00	2916	7.00	7
KN7AA- YA processor	140.00	2916	7.00	7
MS7AA- AA 64 Mbyte memory	190.00	3958	9.50	10
MS7AA- BA 128 Mbyte memory	190.00	3958	9.50	10
MS7AA- CA 256 Mbyte memory	190.00	3958	9.50	10
MS7AA- DA 512 Mbyte memory	190.00	3958	9.50	10
MS7BB- AA BBU memory	190.00	3958	9.50	10
DWLMA- AA XMI PIU	75.00	1562	3.75	4
SF73- LA disk brick	150.00	3125	7.50	8
SF74- LA disk brick	150.00	3125	7.50	8
EF51R- LA SSD	150.00	3125	7.50	8
EF52R- LA SSD	150.00	3125	7.50	8
RZ26- VA	18.00	375	0.90	1
TLZ06- VA/TZK09- VA	18.00	375	0.90	1
RZ73- VA/RZ74- VA	35.00	729	1.75	2

[1]The number 20 is a constant; it has no units associated with it.

Table B-2 Calculation of Equivalent Power Units —Plug-In Units

Module	PIU Watts	Watts @ 48V (.75 x PIU W)	mA @ 48V	EPU (W @ 48V/20[1])	Rounded EPU Value
XMI PIU					
CCARD-YA (T2030)	5.00	6.66	138	0.33	1
CIXCD-AC	54.78	73.04	1521	3.65	4
DEMFA-AA	68.90	91.86	1913	4.59	5
DEMNA-M	42.99	57.32	1194	2.86	3
DWLMA-AA/BA	51.00	68.00	1416	3.40	3
DWMBB-AA	27.00	36.00	750	1.80	2
DWMBB-LA	45.00	60.00	1250	3.00	3
DWMBB-LB	45.00	60.00	1250	3.00	3
DWMVA-AA	15.00	20.00	416	1.00	1
KDM70-AA	87.00	116.00	2416	5.80	6
KFMSA-BA	46.00	61.33	1277	3.06	3
KZMSA-AB	37.50	50.00	1041	2.50	3
VAXBI PIU					
DMB32-M	42.27	56.36	1174	2.81	3
DRB32-E	49.00	65.33	1361	3.26	3
DRB32-M	40.00	53.33	1111	2.66	3
DRB32-W	59.00	78.66	1638	3.93	4
DWMBB	31.20	41.60	866	2.08	2
Futurebus+ PIU					
DEFAA	31.50	42.00	875	2.10	2
DWLAA-AA/BA	22.50	30.00	625	1.50	2

[1]The number 20 is a constant; it has no units associated with it.

Table B- 3 Voltages of Options in Plug- In Units

Module	+5V	−5.2V	−2.0V	+12V	−12V	+13.5V
XMI PIU						
CCARD- YA (T2030)	1.00	0.00	0.00	0.00	0.00	0.00
CIXCD- AC	5.90	1.90	0.50	0.60	0.60	0.00
DEMFA- AA	12.10	0.00	0.00	0.60	0.10	0.00
DEMNA- M	7.20	0.00	0.00	0.01	0.01	0.50
DWLMA- AA/BA	10.20	0.00	0.00	0.00	0.00	0.00
DWMBB- AA	5.40	0.00	0.00	0.00	0.00	0.00
DWMBB- LA	9.00	0.00	0.00	0.00	0.00	0.00
DWMBB- LB	9.00	0.00	0.00	0.00	0.00	0.00
DWMVA- AA	3.00	0.00	0.00	0.00	0.00	0.00
KDM70- AA	17.40	0.00	0.00	0.00	0.00	0.00
KFMSA- BA	9.20	0.00	0.00	0.00	0.00	0.00
KZMSA- AB	7.50	0.00	0.00	0.00	0.00	0.00
VAXBI PIU						
DMB32- M	6.75	0.00	0.00	0.29	0.42	0.00
DRB32- E	9.80	0.00	0.00	0.00	0.00	0.00
DRB32- M	8.00	0.00	0.00	0.00	0.00	0.00
DRB32- W	11.80	0.00	0.00	0.00	0.00	0.00
DWMBB	6.00	0.00	0.00	0.00	0.10	0.00
Futurebus+ PIU						
DEFAA	6.30	0.00	0.00	0.00	0.00	0.00
DWLAA- AA/BA	4.50	0.00	0.00	0.00	0.00	0.00

Index