EK-KDJ1D-UG-002

KDJ11-D/S CPU Module User's Guide

digital™

EK-KDJ1D-UG-002

KDJ11-D/S CPU Module User's Guide

Prepared by Educational Services
of
Digital Equipment Corporation

First Edition, April 1986

Second Edition, May 1987

Using Digital's networked computer services, this book was produced electronically by the Educational Services Development and Publishing department in Reading, England.

Contents

Preface xi

Chapter 1 ARCHITECTURE

1.1 FEATURES 1-2
1.1.1 Features Not Supported 1-2
1.1.2 Registers 1-3
1.2 SYSTEM CONTROL REGISTERS 1-7
1.2.1 Processor Status Word 1-7
1.2.2 Program Interrupt Request 1-9
1.2.3 CPU Error Register 1-10
1.2.4 Maintenance Register 1-11
1.2.5 Line Time Clock Register 1-12
1.2.6 Native Register 1-13
1.2.7 DLART Registers 1-14
1.2.7.1 Receiver Control/Status Register 1-14
1.2.7.2 Receiver Data Buffer 1-15
1.2.7.3 Transmitter Control/Status Register 1-16
1.2.7.4 Transmitter Data Buffer 1-17
1.3 GENERAL PURPOSE REGISTERS 1-17
1.3.1 Stack Pointer 1-18
1.3.2 Program Counter 1-19
1.4 INTERRUPTS 1-19
1.4.1 Halt Line 1-21
1.4.2 HALT Instruction 1-21
1.5 MEMORY MANAGEMENT 1-22
1.5.1 Physical Address Space 1-22
1.5.1.1 I/O Page 1-23
1.5.1.2 Self-test ROM 1-24
1.5.1.3 Bootstrap 1-25

1.5.2 Virtual Memory Mapping 1–25
1.5.2.1 16-bit Mapping 1–26
1.5.2.2 18-bit Mapping 1–26
1.5.2.3 22-bit Mapping 1–27
1.5.3 Compatibility 1–28
1.5.4 Virtual Addressing 1–29
1.5.5 Interrupts Under Memory Management Control 1–29
1.5.6 Building a Physical Address 1–30
1.5.7 Management Registers (PARs and PDRs) 1–32
1.5.7.1 Page Address Register 1–33
1.5.7.2 Page Descriptor Register 1–35
1.5.8 Fault Recovery Registers 1–36
1.5.8.1 Memory Management Register 0 1–36
1.5.8.2 Memory Management Register 1 1–38
1.5.8.3 Memory Management Register 2 1–38
1.5.8.4 Memory Management Register 3 1–38
1.5.8.5 Instruction Back-up/Restart Recovery 1–39
1.5.8.6 Clearing Status Registers After an Abort 1–40
1.5.8.7 Multiple Faults 1–40
1.5.9 Typical Usage 1–40
1.5.9.1 Typical Memory Page 1–40
1.5.9.2 Non-consecutive Memory Pages 1–42
1.5.9.3 Stack Memory Pages 1–43
1.6 MEMORY SYSTEM REGISTERS 1–44
1.6.1 Memory Error Register 1–44
1.7 FLOATING-POINT 1–45
1.7.1 Floating Point Registers 1–45
1.7.1.1 Floating-point Status Register 1–45
1.7.1.2 Floating-point Exception Registers 1–47
1.7.1.3 Floating-point Accumulators 1–48

Chapter 2 INSTALLATION

2.1 CONFIGURATION 2–1
2.1.1 Power-up Options 2–5
2.1.1.1 Power-up Sequence 2–5
2.1.1.2 Power-down Sequence 2–5
2.1.2 Bootstrap Options 2–5
2.1.3 HALT Option 2–9
2.1.4 Factory Configuration 2–10
2.2 DIAGNOSTIC LEDS 2–11
2.3 MODULE EDGE CONNECTOR PINS 2–13

2.4 HARDWARE OPTIONS 2-16
2.4.1 Restricted LSI-11 Options 2-16
2.5 KDJ11-D/S INSTALLATION 2-19
2.5.1 Pre-installation Considerations 2-19
2.5.2 System Differences 2-20
2.5.3 Installation Procedure 2-20
2.6 SPECIFICATIONS 2-21

Chapter 3 FUNCTIONAL DESCRIPTION

3.1 INTRODUCTION 3-1
3.2 DCJ11 MICROPROCESSOR 3-4
3.2.1 DCJ11 Cycles 3-8
3.2.1.1 NIO 3-9
3.2.1.2 Bus Read 3-9
3.2.1.3 Bus Write 3-10
3.2.1.4 GP Read 3-10
3.2.1.5 GP Write 3-10
3.2.1.6 IAK 3-11
3.3 AIO DECODER 3-11
3.4 GP DECODER 3-13
3.5 MEMORY DECODER 3-14
3.6 I/O DECODER 3-16
3.7 CYCLE DECODER 3-18
3.8 I/O READ/WRITE LOGIC 3-20
3.9 IAK DECODER AND VECTOR GENERATION LOGIC 3-22
3.10 MEMORY STATE MACHINE 3-24
3.11 BUS STATE MACHINE 3-26
3.12 SACKTIMEUP, NXM, BBS7, BTWT 3-26
3.12.1 SACKTIMEUP and NXM 3-26
3.12.2 BBS7 and BWTBT 3-26
3.13 PARITY AND ABORT 3-29
3.14 REGISTERS 3-29
3.14.1 Memory Error Register 3-29
3.14.2 Maintenance Register 3-32
3.14.3 Native Register 3-32
3.14.4 Line Time Clock Register 3-32
3.15 RAM ADDRESSING 3-32
3.15.1 Refresh 3-34
3.16 DLARTS 3-35
3.16.1 DLART Registers 3-35

Chapter 4 BOOT AND DIAGNOSTIC ROM

4.1 INTRODUCTION 4–1
4.1.1 Terminal Requirements 4–2
4.2 BOOT SELECT 4–2
4.2.1 Automatic-boot Mode 4–5
4.2.2 Console Mode 4–5
4.2.2.1 Manual Start 4–6
4.2.3 ROM Part and Version Numbers 4–6
4.2.4 Message Formats 4–6
4.3 CONSOLE MODE 4–7
4.3.1 Entering Console Mode Commands 4–8
4.3.2 HELP Command 4–9
4.3.3 BOOT Command 4–9
4.3.4 LIST Command 4–11
4.3.5 MAP Command 4–12
4.3.6 TEST Command 4–13
4.3.7 WRAP Command 4–14
4.4 TEST RESULTS 4–15
4.4.1 Test 1 Errors 4–15
4.4.2 Test 2 Errors 4–15
4.4.3 Tests 3 through 6 Errors 4–15
4.4.3.1 Test 3 Through 6 Error Messages 4–15
4.4.3.2 Commands to Override Errors 4–17
4.5 AUTOMATIC-BOOT MODE 4–17
4.5.1 Boot Code 4–17
4.5.2 Automatic-boot Sequence (Autoboot) 4–19
4.5.3 Bootstrap Error Messages 4–19
4.5.4 Disk and Tape Autoboot (Except TU58) 4–20
4.5.5 DPV11, DUV11, DLV11-E/F and TU58 Autoboot 4–21
4.5.6 DEQNA Autoboot 4–22
4.5.7 Bootable Media 4–22
4.6 LANGUAGE SELECTION 4–22
4.7 TROUBLESHOOTING 4–23

Index

Examples

4–1 Typical Automatic-boot Message 4–6
4–2 Typical Power-up to Console Mode Message 4–7
4–3 Console Mode Prompt 4–7
4–4 Invalid Entry Message 4–9
4–5 HELP Command 4–9
4–6 BOOT Command Argument Prompt 4–10
4–7 BOOT Command Using DL2 4–11
4–8 LIST Command 4–11
4–9 MAP Command 4–13
4–10 TEST Command 4–14
4–11 WRAP Command Without Switch 4–14
4–12 WRAP Command With Switch 4–14
4–13 On-board RAM Test Error Message 4–16
4–14 Q22-bus RAM Test Error Message 4–16
4–15 J11 Unexpected Trap Error Message 4–16
4–16 Language Inquiry and Error Prompt 4–17
4–17 Unsuccessful Automatic-boot Message 4–19
4–18 Console Mode Boot Error Message (1 of 2) 4–20
4–19 Console Mode Boot Error Message (2 of 2) 4–20
4–20 Language Inquiry 4–23

Figures

1–1 Registers 1–3
1–2 PSW (Processor Status Word) Register 1–8
1–3 PIRQ (Program Interrupt Request) Register 1–9
1–4 CPU Error Register 1–10
1–5 Maintenance Register 1–11
1–6 LTC (Line Time Clock) Register (BEVNT) 1–12
1–7 Native Register 1–13
1–8 Receiver Control/Status Register (RCSR) 1–14
1–9 Receiver Data Buffer (RBUF) 1–15
1–10 Transmitter Control/Status Register (XCSR) 1–16
1–11 Transmitter Data Buffer (XBUF) 1–17
1–12 Physical Address Space 1–23
1–13 Self-test ROM Space 1–24
1–14 Bootstrap Space 1–25
1–15 16-bit Mapping 1–26
1–16 18-bit Mapping 1–27
1–17 22-bit Mapping 1–28
1–18 Virtual to Physical Address Mapping 1–29

1–19 Virtual Address Fields 1–30
1–20 Virtual Address Displacement Field 1–31
1–21 Physical Address Assembly 1–32
1–22 Management Registers (PARs and PDRs) 1–34
1–23 Page Address Register (PAR) 1–35
1–24 Page Descriptor Register (PDR) 1–35
1–25 Memory Management Register 0 (MMR0) 1–37
1–26 Memory Management Register 1 (MMR1) 1–38
1–27 Memory Management Register 3 (MMR3) 1–39
1–28 Typical Memory Page 1–41
1–29 Non-consecutive Memory Pages 1–42
1–30 Typical Stack Memory Page 1–43
1–31 Memory Error Register (MER) 1–44
1–32 Floating-point Status Register (FPS) 1–46
2–1 KDJ11-D/S Module (M7554) 2–2
2–2 KDJ11-D/S Module - Early version (M7554) 2–3
2–3 Power-up Sequence 2–6
2–4 Power-down Sequence 2–7
2–5 Console ODT Exit Sequence 2–10
2–6 Module Edge Connectors 2–17
3–1 KDJ11-DA Block Diagram 3–2
3–2 KDJ11-D/S Block Diagram 3–3
3–3 DCJ11 CPU 3–7
3–4 AIO Decoder 3–12
3–5 GP Decoder 3–14
3–6 Memory Decoder 3–15
3–7 I/O Decoder 3–17
3–8 Cycle Decoder 3–19
3–9 I/O Read/Write logic 3–21
3–10 IAK Decoder and Vector Generation Logic 3–23
3–11 Memory State Machine 3–25
3–12 Bus State Machine 3–27
3–13 SACKTIMEUP, NXM, BBS7, and BWTBT 3–28
3–14 Parity Error and Abort 3–30
3–15 Memory Error Register Access 3–31
3–16 Maintenance Register Access 3–33
3–17 Native Register Access 3–34
3–18 LTC Register Access 3–35
3–19 Memory Address Multiplexer 3–36
3–20 DLARTs 3–37

Tables

1-1 KDJ11 versions 1-1
1-2 Address Map 1-4
1-3 PSW (Processor Status Word) Bits 1-8
1-4 PIRQ (Program Interrupt Request) Bits 1-10
1-5 CPU Error Register Bits 1-11
1-6 Maintenance Register Bits 1-12
1-7 LTC (Line Time Clock) Register Bits 1-13
1-8 Native Register Bits 1-13
1-9 Receiver Control/Status Register (RCSR) Bits 1-14
1-10 Receiver Data Buffer (RBUF) Bits 1-15
1-11 Transmitter Control/Status Register (XCSR) Bits 1-16
1-12 Transmitter Data Buffer (XBUF) Bits 1-17
1-13 General Purpose Registers 1-18
1-14 Stack Pointer Selection 1-18
1-15 Asynchronous Interrupts 1-19
1-16 Synchronous Interrupts 1-20
1-17 Interrupt Priorities 1-21
1-18 KDJ11-D/S Compatibility 1-28
1-19 Virtual Address Description 1-30
1-20 Displacement Field Description 1-31
1-21 Page Descriptor Register (PDR) Bits 1-35
1-22 Memory Management Register 0 (MMR0) Bits 1-37
1-23 Memory Management Register 3 (MMR3) Bits 1-39
1-24 Memory Error Register (MER) Bits 1-44
1-25 Floating-point Status (FPS) Bits 1-46
2-1 Jumpers 2-4
2-2 Boot Select Options 2-8
2-3 Self-test ROM Display Codes 2-11
2-4 J1 Pin Assignments 2-12
2-5 Module Edge Connector Pin Assignments 2-13
2-6 Restricted or Non-compatible LSI-11 Options 2-18
2-7 KDJ11-D/S Specifications 2-21
3-1 DCJ11 Input and Output Signals 3-4
3-2 AIO<3:0> Codes 3-8
3-3 BS<1:0> Codes 3-8
3-4 GP Codes 3-10
3-5 AIO Decoder Outputs 3-13
3-6 GP Decoder Outputs 3-14
3-7 Memory Decoder Outputs 3-15
3-8 I/O Decoder Outputs 3-17

3–9 Cycle Decoder Outputs 3–19
3–10 I/O Read/Write Logic Outputs 3–21
3–11 IAK Decoder Outputs 3–24
3–12 Vector Generation logic Outputs 3–24
4–1 Terminal Requirements 4–2
4–2 Native Register Boot Select Codes 4–3
4–3 ROM Status Codes 4–4
4–4 Manual Restart Addresses 4–6
4–5 Console Mode Commands 4–7
4–6 BOOT Command Interpretation 4–10
4–7 Error Override commands 4–17
4–8 Q22-bus Boot Devices 4–18
4–9 Boot Device Errors 4–20
4–10 Disk and Tape Boot List 4–21
4–11 DPV, DUV, DLV, TU58 and RK05 Autoboot List 4–21

Preface

INTRODUCTION

The KDJ11-D/S CPU module is a low-cost quad-height processor module for use in extended LSI-11 bus (Q22-bus) systems. It is based on the DCJ11 microprocessor, and executes the PDP11/73 instruction set.

PURPOSE

This user's guide provides first-time and sophisticated users of the KDJ11-D/S CPU Module with the information to install, test, operate, and troubleshoot the module.

INTENDED AUDIENCE

This user's guide is written for end users, COEMs, TOEMs, and Field Service.

REVISION NOTES

This manual has been revised to reflect the re-engineered KDJ11-DA. The re-engineered version is refered to in this manual as the KDJ11-D/S. Where differences occur, both the new and the previous version will be described.

GUIDE CONTENT

This user's guide contains four chapters and an index. The following list describes the primary content of each chapter.

1. Architecture

   Describes KDJ11-D/S features, registers, interrupt scheme, and memory management.

2. Installation

   Describes module configuration, options, power-up/down sequences, specifications, back-plane pin-out, and installation.

3. Functional Description

   Briefly describes the functional operation of the major module components.

4. Boot ROMs and Diagnostics

   Describes boot ROM routines and micro-diagnostics.

RELATED INFORMATION

This guide does not replicate generic PDP-11 information often found in other MicroPDP-11 CPU User's Guides. Such related information and its source documents are listed below.

- *PDP-11 Architecture Handbook, EB-23657-18*
  - — Addressing modes
  - — Instruction set.
  - — Floating-point instructions set.
  - — Programming techniques.
  - — Extended LSI-11 bus description.
- *PDP-11 UNIBUS Processor Handbook, EB-26077-41*
  - — Console ODT (On-line Debugging Technique) command descriptions and examples.

ASSOCIATED DOCUMENTS

Order	
EB-26077-41	PDP-11 UNIBUS Processor Handbook
EB-23657-18	PDP-11 Architecture Handbook
EK-DCJ11-UG	DCJ11 Microprocessor User's Guide

CONVENTIONS

The notational conventions used in this guide are described in the following table.

Convention	Meaning
NOTE	Contains general information.
CAUTION	Contains information to prevent damage to equipment.
<mm:nn>	Read as "mm through nn." This use of angle brackets and the colon indicates a bit field, or a set of lines or signals. For example, A<17:00> is the mnemonic for address lines "A17 through A00."
Addresses	Unless otherwise noted, all addresses in this guide are in octal notation.
k, M	Abbreviations for kilo, Mega. When used with bytes and words, k and M represent the actual decimal value of quantities. For example: 8 kbytes = 8192 bytes not 8000 bytes 32 kwords = 32768 words not 32000 words 512 kbytes = 524288 bytes not 512000 bytes 4 Mbytes = 4194304 bytes not 4000000 bytes
<CTRL/n> or ^n	Control sequence. Press the <CTRL> key and the appropriate typing key at the same time.
Abbreviations	Abbreviations are in accordance with *DEC STD 015, 3 February 1983.*

Chapter 1
ARCHITECTURE

This chapter describes features of the KDJ11-D/S module, its registers, interrupt scheme, and memory management.

The range of three modules has been expanded to four modules, as shown in the following table.

Table 1-1: KDJ11 versions

Old version		New version	Comments
KDJ11-DA	Q-bus 0.5 Mb 15 MHz	KDJ11-DA	Re-engineered
		KDJ11-DB	Q-bus 1.5 Mb 15 MHz
KDJ11-SA	BA200 0.5Mb 15 MHz		Replaced by KDJ11-SC
KDJ11-SB	BA200 0.5Mb 18 MHz		Replaced by KDJ11-SD
		KDJ11-SC	BA200 1.5 Mb 15 MHz
		KDJ11-SD	BA200 1.5 Mb 18 MHz

The are three main differences between the previous versions and the new versions of the boards. All the PALs and most of the discrete logic has been replaced by two gate arrays. This has made room on board for an optional increase in on-board memory from 0.5 Mb to 1.5 Mb. As a result of the change in physical layout, the position of the jumpers has been changed, although the previous numbers have been retained. Details of the jumpers will be covered in Chapter 2

1.1 FEATURES

The KDJ11-D/S CPU module (part number M7554) is a low-cost quad-height single board computer for use in extended LSI-11 bus (Q22-bus) systems. It is based on the DCJ11 microprocessor chip, and executes the PDP11/73 instruction set. The module includes the CPU, memory management, local memory, and I/O.

The CPU executes the full PDP11 integer instruction set. Floating point instructions are standard on the KDJ11-D/S; however, the FPA (Floating-point Accelerator) is not an option. Full 22-bit memory management is provided for both instruction and data references in three protection modes: kernel, supervisor, and user. (These are also called operating modes.)

The KDJ11-D/S can address up to 4 Mbytes of memory. The module includes 0.5 Mbytes or 1.5 Mbytes of local memory, as shown below:

- KDJ11-DA — 0.5 Mbytes
- KDJ11-DB — 1.5 Mbytes
- KDJ11-SA — 0.5 Mbytes
- KDJ11-SC — 1.5 Mbytes
- KDJ11-SD — 1.5 Mbytes (BA200 only)

An additional 3.5 Mbytes/2.5 Mbytes of memory can be addressed over the Q22-bus interface. Local (on-board) memory is 0.5 Mbytes/1.5 Mbytes of dynamic RAM with no battery back-up. Its starting address is fixed at 0.

All Q22-bus transaction types are supported. The KDJ11-D/S is the Q22-bus arbiter, and services only level 4 interrupts. Interrupt requests on levels 5, 6, or 7 are serviced at level 4 (BIRQ<5:7> are terminated and not connected to on-board logic).

The CPU supports power-up mode 2, bootstrap on power-up. The bootstrap starting address is fixed at 17773000 in the I/O page. Bootstrap and self-test routines are included in the on-board ROM. The ROMs can be either 32 kbytes (two 16-kbyte ROMs) or 64 kbytes (two 32-kbyte ROMs). Bootstrap options are selectable with on-board jumpers or through a remote switch. Seven indicator signal lines are provided to drive remote indicators for diagnostic purposes.

The HALT/trap option is supported with an on-board jumper. With the jumper removed, the CPU will trap through location 4 when a HALT instruction is executed in kernel mode. If the jumper is installed, the CPU will enter console ODT mode.

The module includes two DLARTs (DL-style UARTs (Universal Asynchronous Receiver/ Transmitter)) for console and printer SLUs (Serial Line Units). DLART baud rate is selectable with on-board jumpers or remote switch. Halt on BREAK is jumper selectable in the console DLART.

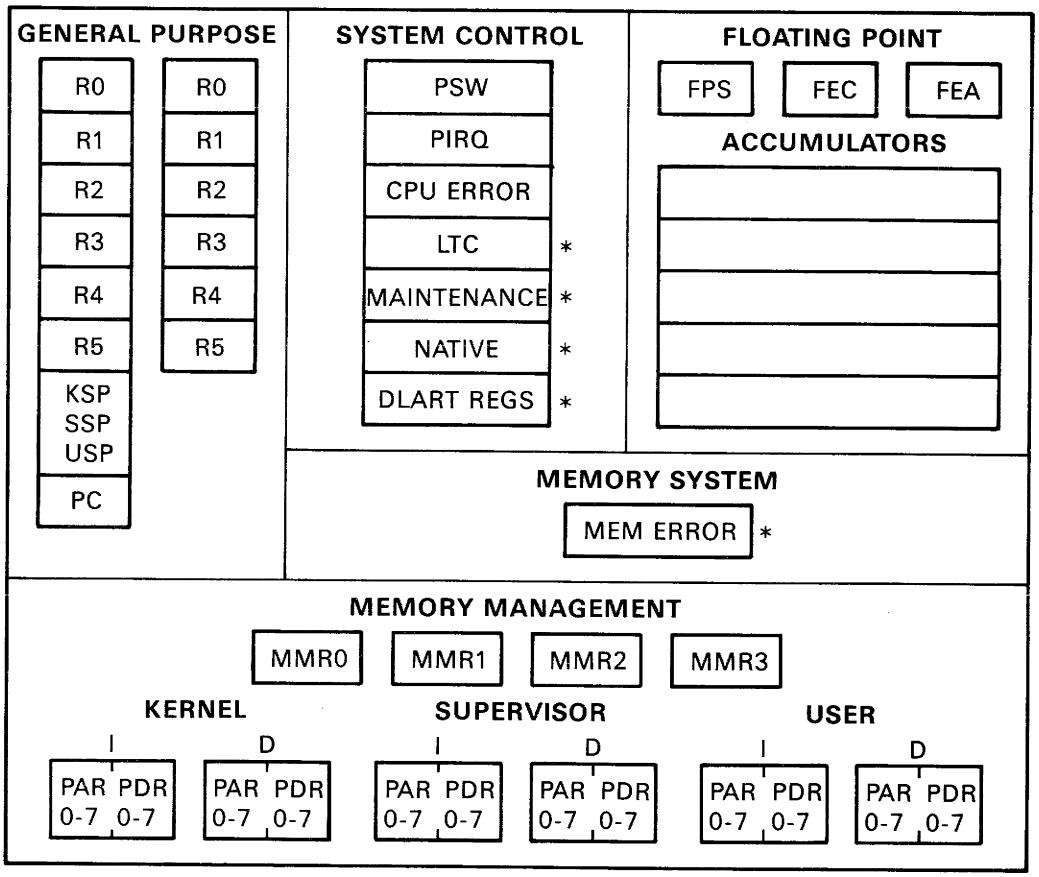
1.1.1 Features Not Supported

The KDJ11-D/S CPU module does not have a cache memory. The FPA (Floating-point Accelerator) is not an option. Power-up modes 0 (power-fail), 1 (ODT), and 3 (user-defined starting address) are not supported.

1.1.2 Registers

User-visible registers are shown in Figure 1–1. They are classified as: System Control, General Purpose, Memory System, Memory Management, and Floating Point registers. The addresses for the memory-addressable registers are listed in Table 1–2. These are user-visible registers which can be accessed with a 22-bit physical address. The other user-visible registers are referenced with the instruction set. Table 1–2 is an address map and lists the addresses and vectors for registers and other system devices.

Figure 1–1: Registers



* EXTERNAL TO THE DCJ11

RE3547

Table 1–2: Address Map

Register/Device		Address	Vector
FPP (Floating Point Processor)		–	244
PSW (Processor Status Word)		17777776	–
PIRQ (Program Interrupt Request)		17777772	240
CPU Error		17777766	–
Maintenance		17777750	–
Memory Error		17772100	–
MMU (Memory Management Unit)			
User Data	PAR 7	17777676	250
	PAR 6	17777674	250
	PAR 5	17777672	250
	PAR 4	17777670	250
	PAR 3	17777666	250
	PAR 2	17777664	250
	PAR 1	17777662	250
	PAR 0	17777660	250
User Instruction	PAR 7	17777656	250
	PAR 6	17777654	250
	PAR 5	17777652	250
	PAR 4	17777650	250
	PAR 3	17777646	250
	PAR 2	17777644	250
	PAR 1	17777642	250
	PAR 0	17777640	250
User Data	PDR 7	17777636	250
	PDR 6	17777634	250
	PDR 5	17777632	250
	PDR 4	17777630	250
	PDR 3	17777626	250
	PDR 2	17777624	250
	PDR 1	17777622	250
	PDR 0	17777620	250
User Instruction	PDR 7	17777616	250
	PDR 6	17777614	250
	PDR 5	17777612	250
	PDR 4	17777610	250
	PDR 3	17777606	250
	PDR 2	17777604	250
	PDR 1	17777602	250
	PDR 0	17777600	250
MMR2 (MMU Status 2)		17777576	250
MMR1 (MMU Status 1)		17777574	–
MMR0 (MMU Status 0)		17777572	250
SLU 0 (Console Serial Line Unit 0)			

Table 1–2 (Cont.): Address Map

Register/Device		Address	Vector
Transmitter Buffer		17777566	64
Transmitter CSR		17777564	64
Receiver Buffer		17777562	60
Receiver CSR		17777560	60
LTC (line time clock)		17777546	100
KDJ11-D/S Native Register		17777520	–
SLU 1 (General Purpose Serial Line Unit 1)			
Transmitter Buffer		17776506	304
Transmitter CSR		17776504	304
Receiver Buffer		17776502	300
Receiver CSR		17776500	300
Bootstrap (512 bytes)		17773776 through 17773000	
MMU (Memory Management Unit)			
MMR3 (MMU Status 3)		17772516	250
Kernel Data	PAR 7	17772376	250
	PAR 6	17772374	250
	PAR 5	17772372	250
	PAR 4	17772370	250
	PAR 3	17772366	250
	PAR 2	17772364	250
	PAR 1	17772362	250
	PAR 0	17772360	250
Kernel Instruction	PAR 7	17772356	250
	PAR 6	17772354	250
	PAR 5	17772352	250
	PAR 4	17772350	250
	PAR 3	17772346	250
	PAR 2	17772344	250
	PAR 1	17772342	250
	PAR 0	17772340	250
Kernel Data	PDR 7	17772336	250
	PDR 6	17772334	250
	PDR 5	17772332	250
	PDR 4	17772330	250
	PDR 3	17772326	250
	PDR 2	17772324	250
	PDR 1	17772322	250
	PDR 0	17772320	250

Table 1–2 (Cont.): Address Map

Register/Device		Address	Vector
Kernel Instruction	PDR 7	17772316	250
	PDR 6	17772314	250
	PDR 5	17772312	250
	PDR 4	17772310	250
	PDR 3	17772306	250
	PDR 2	17772304	250
	PDR 1	17772302	250
	PDR 0	17772300	250
Supervisor Data	PAR 7	17772276	250
	PAR 6	17772274	250
	PAR 5	17772272	250
	PAR 4	17772270	250
	PAR 3	17772266	250
	PAR 2	17772264	250
	PAR 1	17772262	250
	PAR 0	17772260	250
Supervisor Instruction	PAR 7	17772256	250
	PAR 6	17772254	250
	PAR 5	17772252	250
	PAR 4	17772250	250
	PAR 3	17772246	250
	PAR 2	17772244	250
	PAR 1	17772242	250
	PAR 0	17772240	250
Supervisor Data	PDR 7	17772236	250
	PDR 6	17772234	250
	PDR 5	17772232	250
	PDR 4	17772230	250
	PDR 3	17772226	250
	PDR 2	17772224	250
	PDR 1	17772222	250
	PDR 0	17772220	250
Supervisor Instruction	PDR 7	17772216	250
	PDR 6	17772214	250
	PDR 5	17772212	250
	PDR 4	17772210	250
	PDR 3	17772206	250
	PDR 2	17772204	250
	PDR 1	17772202	250
	PDR 0	17772200	250

Table 1–2 (Cont.): Address Map

Register/Device	Address	Vector
Self Test ROM (120 kbytes)	17400000 through 17757777	
User RAM (0.5 Mbytes)	00000000 through 01777777	
User RAM (1.5 Mbytes)	00000000 through 05777777	

1.2 SYSTEM CONTROL REGISTERS

These registers, shown in Figure 1–1, control system functions.

1.2.1 Processor Status Word

Address: 17777776

The PSW (Processor Status Word) Register contains:

- Current and previous operating mode
- General purpose register group in use
- Current priority level
- Condition code status
- Trace/trap bit for debugging

The PSW is initialized at power-up and cleared with a console start. See Figure 1–2 and Table 1–3.

Figure 1-2: PSW (Processor Status Word) Register

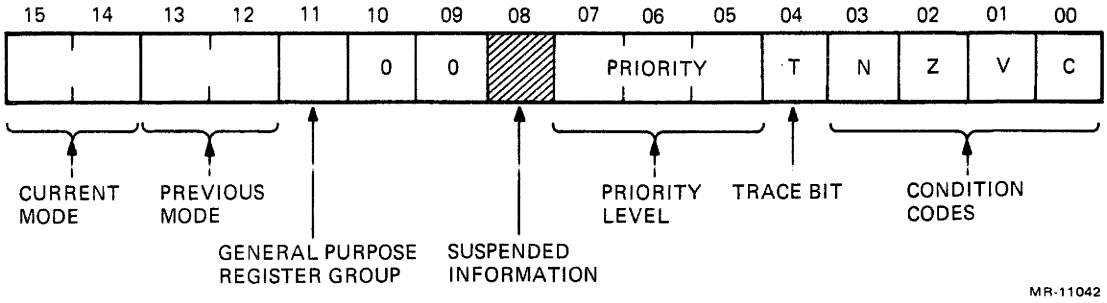


Table 1-3: PSW (Processor Status Word) Bits

Bit	Access	Description
<15:14>	R/W	Current Mode—Indicate the current operating mode as follows:

Bit	Mode
15 14	
0 0	Kernel
0 1	Supervisor
1 0	Illegal
1 1	User

<13:12>	R/W	Previous Mode—Indicate the previous operating mode, and are coded the same as bits <15:14>.
<11>	R/W	Register Set—Selects the current general purpose register set, as follows:

Bit<11>	Register Set
0	R0 through R5
1	R0' through R5'

<10:09>	R	not used
<08>		Suspended Information—reserved
<07:05>	R/W	Priority Level—Indicate the processor's current priority level, as follows:

Table 1-3 (Cont.): PSW (Processor Status Word) Bits

Bit	Access	Description
		Bit 07 06 05 / Level 1 1 1 — 7 1 1 0 — 6 1 0 1 — 5 1 0 0 — 4 0 1 1 — 3 0 1 0 — 2 0 0 1 — 1 0 0 0 — 0
<04>	R/W	Trap—Inactive when cleared. When set, the processor traps to location 14 at the end of the current instruction. Used for debugging and setting breakpoints, this bit cannot be explicitly set by writing the PSW; it can be changed only with the RTI/RTT instructions.
<03>	R/W	Condition code N—Set when the result of the previous operation is negative.
<02>	R/W	Condition code Z—Set when the result of the previous operation is zero.
<01>	R/W	Condition code V—Set when the previous operation resulted in an overflow.
<00>	R/W	Condition code C—Set when the previous operation caused a carry-out.

1.2.2 Program Interrupt Request

Address: 17777772

The PIRQ (Program Interrupt Request) Register implements software interrupts. Setting one of bits <15:09> corresponds to a request on one of the priority levels 7:1. See Figure 1-3 and Table 1-4. Hardware sets bits <07:05> and <03:01> to the encoded value of the highest priority request pending. When the interrupt is acknowledged, the processor vectors to location 240 for the service routine. (The service routine must clear the interrupt request.) The PIRQ Register is cleared by power-up, console start, or the RESET instruction.

Figure 1-3: PIRQ (Program Interrupt Request) Register

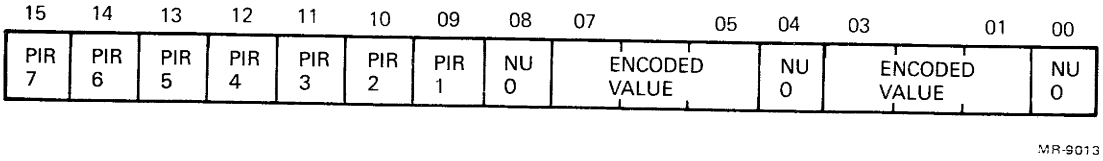


Table 1-4: PIRQ (Program Interrupt Request) Bits

Bit	Access	Description
<15>	R/W	Priority Level 7—When set, requests an interrupt on level 7.
<14>	R/W	Priority Level 6—When set, requests an interrupt on level 6.
<13>	R/W	Priority Level 5—When set, requests an interrupt on level 5.
<12>	R/W	Priority Level 4—When set, requests an interrupt on level 4.
<11>	R/W	Priority Level 4—When set, requests an interrupt on level 3.
<10>	R/W	Priority Level 2—When set, requests an interrupt on level 2.
<09>	R/W	Priority Level 1—When set, requests an interrupt on level 1.
<08>		not used
<07:05>	R	Encoded Value—Indicate the highest priority level set in bits <15:09>.
<04>		not used
<03:01>	R	Encoded Value—Same as bits <07:05>.
<00>		not used

1.2.3 CPU Error Register

Address: 17777766

The CPU Error Register indicates the source of any trap or error condition that caused a trap through location 4. See Figure 1-4 and Table 1-5. The register is cleared by any write reference, power-up, or console start. It is not changed by the RESET instruction.

Figure 1-4: CPU Error Register

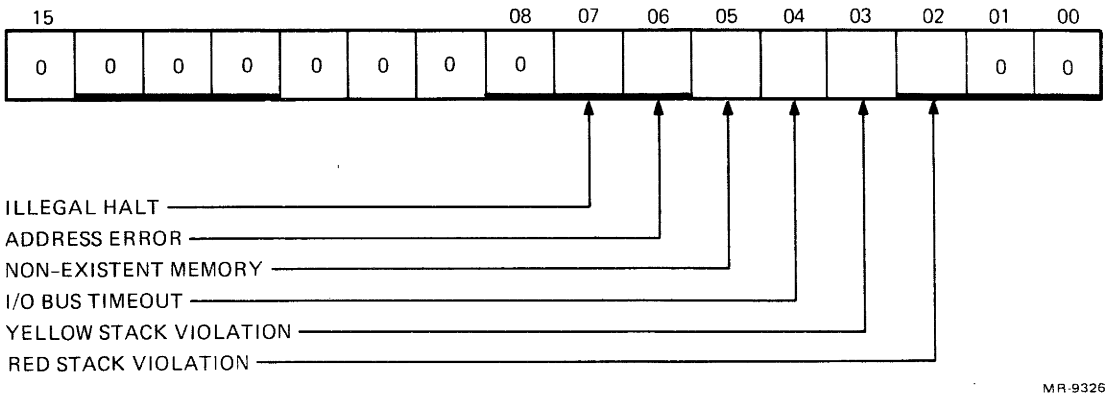


Table 1–5: CPU Error Register Bits

Bit	Access	Description
<15:08>		not used
<07>	R	Illegal HALT—Set when an attempt is made to execute a HALT instruction in user or supervisor mode. (See Maintenance Register<03>, Table 1–6.)
<06>	R	Address Error—Set when an attempt is made to either word-access an odd byte-address or to fetch an instruction from an internal register.
<05>	R	Non-existent Memory—Set when a reference to main memory times-out.
<04>	R	I/O Bus Time-out—Set when a reference to the I /O page times-out.
<03>	R	Yellow Stack Violation—Set on a yellow-zone stack-overflow trap (kernel mode stack reference less than 400_8).
<02>	R	Red Stack Violation—Set on a red stack-trap (a kernel stack-push abort during an interrupt, abort, or stack sequence).
<01:00>		not used

1.2.4 Maintenance Register

Address: 17777750

The Maintenance Register allows software to determine which power-up options are selected and if an FPA (floating point accelerator) is installed. See Figure 1–5 and Table 1–6.

Figure 1–5: Maintenance Register

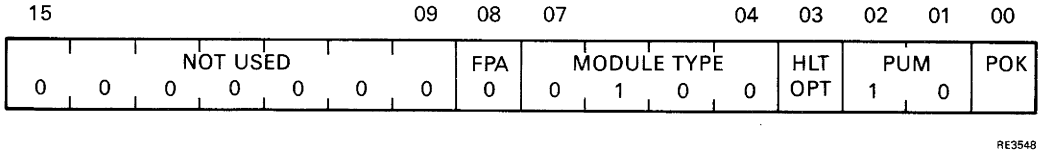


Table 1–6: Maintenance Register Bits

Bit	Access	Description
<15:09>	R	not used—always read zero
<08>	R	FPA Here—Always reads zero. The FPA is not a KDJ11-D/S option.
<07:04>	R	Module Type Number—Encoded value indicating the module type number. The KDJ11-D/S is module type four (0100_2).
<03>	R	Halt Option—Indicates how a HALT instruction will execute in kernel mode. If jumper W1 is not installed the bit is set; when a HALT is executed, the KDJ11-D/S traps through location 4, and sets CPU Error Register<07> (Table 1–5). If jumper W1 is installed, the bit is cleared; when a HALT is executed, the KDJ11-D/S enters console ODT mode.
<02:01>	R	Power-up Mode Option—Indicates the selected power-up mode option. The KDJ11-D/S supports only power-up mode 2 (10_2), bootstrap at starting address 17773000.
<00>	R	Power OK—When set, indicates that Q22-bus signal BPOK is asserted H (system power is within tolerance and stable).

1.2.5 Line Time Clock Register

Address: 17777546

The LTC (Line Time Clock) Register controls recognition of the Q22-bus signal, BEVNT. See Figure 1–6 and Table 1–7.

Figure 1–6: LTC (Line Time Clock) Register (BEVNT)

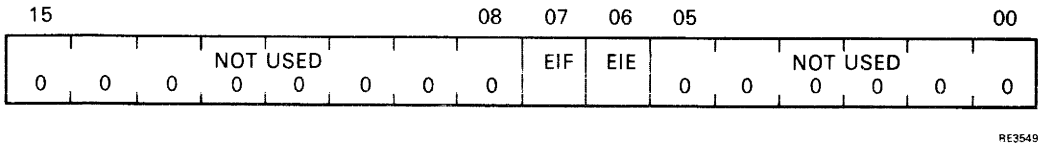


Table 1–7: LTC (Line Time Clock) Register Bits

Bit	Access	Description
<15:08>	R	not used—always read zero
<07>	R	Event Interrupt Flag—Indicates the state of the Q22-bus BEVNT signal when bit <06> is not set. If set when <06> is set, causes an interrupt through vector 100, at priority level 6. Must be deasserted and asserted (BEVNT asserted and deasserted) to cause another interrupt.
<06>	R/W	Event Interrupt Enable—When set, enables level 6 interrupts through vector 100. Cleared by power-up, reboot, or the RESET instruction.
<05:00>	R	not used—always read zero

1.2.6 Native Register

Address: 17777520

The Native Register (Figure 1–7) provides the test and boot functions described in Table 1–8.

Figure 1–7: Native Register

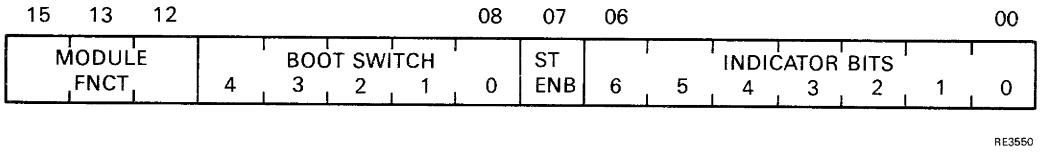


Table 1–8: Native Register Bits

Bit	Access	Description
<15:13>	R	15 = 1 14 = 0 0.5 Mbyte memory (on board) 14 = 1 1.5 Mbyte memory (on board). Bit 14 reflects the setting of jumper W25. The jumper is installed when there is 0.5 Mbytes of memory, and is removed when there is 1.5 Mbytes of memory. 13 = 0 Normal 13 = 1 For special applications. Bit 13 reflects the setting of jumper W4. Bits <15:13> are always zero on early versions.

Table 1–8 (Cont.): Native Register Bits

Bit	Access	Description
<12:08>	R	Boot Switch <4:0>—Binary coded value read by firmware to direct self-test and bootstrap program execution. Up to 32 routines can be selected, either with on-board jumpers W22, W2, W3, W5, and W8 (bits <12:08>, respectively), or with W22 (bit <12>) and a remote 16-position switch (bits <11:08>). See Table 2–2.
<07>	R/W	Self-test Enable—When set, enables the Self-test ROM; that is, physical addresses 17400000 through 17757777 access Self-test ROM space. When cleared, disables the Self-test ROM; that is, physical addresses 17400000 through 17757777 access Q22-bus memory space. Cleared by power-up or reboot; not affected by the RESET instruction.
<06:00>	R/W	Indicator <6:0>—Indicate the state of lines IND<6:0>. Cleared on power-up or reboot; not affected by the RESET instruction. The lines are connected to J1, to drive remote LEDs. The Self-test program reports its progress on IND<3:0>. See Table 2–3.

1.2.7 DLART Registers

Each DLART contains the following registers.

RCSR (Receiver Control/Status Register)
RBUF (Receiver Data Buffer)
XCSR (Transmitter Control/Status Register)
XBUF (Transmitter Data Buffer)

1.2.7.1 Receiver Control/Status Register

Address: 17777560 (SLU 0), 17776500 (SLU 1)

Figure 1–8: Receiver Control/Status Register (RCSR)

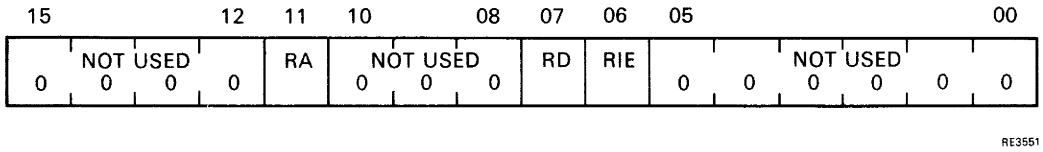


Table 1–9: Receiver Control/Status Register (RCSR) Bits

Bit	Access	Description
<15:12>	R	not used—always read 0
<11>	R	Receiver Active—When set, indicates that a start bit has been detected. Cleared when a stop bit is detected, or on power-up or reboot.

Table 1–9 (Cont.): Receiver Control/Status Register (RCSR) Bits

Bit	Access	Description
<10:08>	R	not used—always read 0
<07>	R	Receiver Done—When set, indicates that the serial interface has received a character; and requests an interrupt if <06> is set. Cleared by reading the receiver data buffer, or on power-up or reboot.
<06>	R/W	Receiver Interrupt Enable—When set, receiver interrupts are enabled (see <07>). When cleared, receiver interrupts are disabled. Cleared by power-up, reboot, or the RESET instruction.
<05:00>	R	not used—always read 0

1.2.7.2 Receiver Data Buffer

Address: 17777562 (SLU 0), 17776502 (SLU 1)

Figure 1–9: Receiver Data Buffer (RBUF)

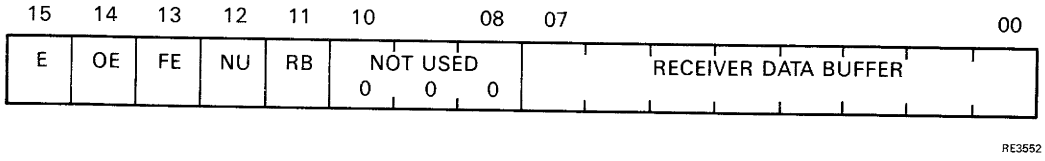


Table 1–10: Receiver Data Buffer (RBUF) Bits

Bit	Access	Description
<15>	R	Error—Set when <14> or <13> is set. Cleared when the error condition is cleared.
<14>	R	Overrun Error—When set, indicates that a new character was received before an old character was read. Will be set if Receiver Done (RCSR<07>) was not cleared when a character was received. Cleared when a character is received and RCSR<07> = 0 (cleared), or on power-up or reboot.
<13>	R	Framing Error—When set, indicates that a received character did not have a valid stop bit. Cleared when a character with a valid stop bit is received, or on power-up or reboot.
<12>	R	not used—always read 0
<11>	R	Received Break—When set, indicates that the received signal has gone from a MARK to a SPACE, and stayed in the SPACE condition for 11 bit-times. Cleared when the received signal returns to a MARK, or on power-up or reboot.
<10:08>	R	not used—always read 0

Table 1–10 (Cont.): Receiver Data Buffer (RBUF) Bits

Bit	Access	Description
<07:00>	R	Receiver Data Buffer—Set to the value of the most recently received character. Cleared by power-up or reboot.

1.2.7.3 Transmitter Control/Status Register

Address: 17777564 (SLU 0), 17776504 (SLU 1)

Figure 1–10: Transmitter Control/Status Register (XCSR)

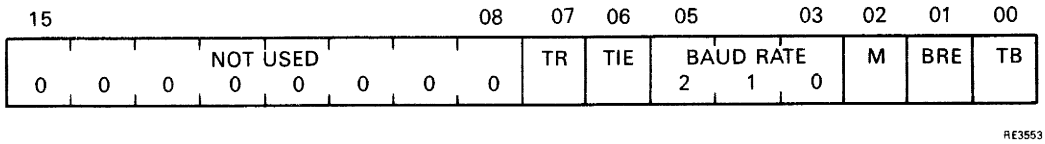


Table 1–11: Transmitter Control/Status Register (XCSR) Bits

Bit	Access	Description
<15:08>	R	not used—always read 0
<07>	R	Transmitter Ready—When set, indicates that the Transmitter Buffer is empty and can accept a new character for transmission; and requests an interrupt if <06> is set. Set on power-up or reboot. Cleared by writing into the Transmitter Buffer.
<06>	R/W	Transmitter Interrupt Enable—Set to enable transmitter interrupts (see <07>). Cleared to disable transmitter interrupts. Cleared by power-up, reboot, or the RESET instruction.
<05:03>	R/W	Baud Rate Select—In the KDJ11-D/S, these bits are ALWAYS CLEARED (see <01>). (In other applications, if <01> is set, these bits determine the transmit/receive baud rate under program control, and are cleared when <01> is cleared, on power-up, or on reboot.)
<02>	R/W	Maintenance—When set, the external receiver input is disconnected and the transmitter output is connected to the receiver input. Cleared by power-up, reboot, or the RESET instruction.

Table 1-11 (Cont.): Transmitter Control/Status Register (XCSR) Bits

Bit	Access	Description
<01>	R/W	Baud Rate Enable—In the KDJ11-D/S, this bit is ALWAYS CLEARED. The baud rate is selected by remote switch or on-board jumpers W4, W6, and W9 for DLART0 and W10, W7, and W12 for DLART1 (baud rate select BRS2, BRS1, and BRS0, respectively). (In other applications, if this bit is set, <05:03> can select the baud rate under program control. When <01> is cleared, the baud rate is selected by remote switch or on-board jumpers. Cleared on power-up or reboot.)
<00>	R/W	Transmit Break—When set, causes the output signal to go to a SPACE condition. A SPACE longer than a character-time causes a framing error and is interpreted as a break. Transmit Ready and Transmit Interrupt continue to operate, allowing software to time the break. Cleared by power-up, reboot, or the RESET instruction.

1.2.7.4 Transmitter Data Buffer

Address: 17777566 (SLU 0), 17776506 (SLU 1)

Figure 1-11: Transmitter Data Buffer (XBUF)

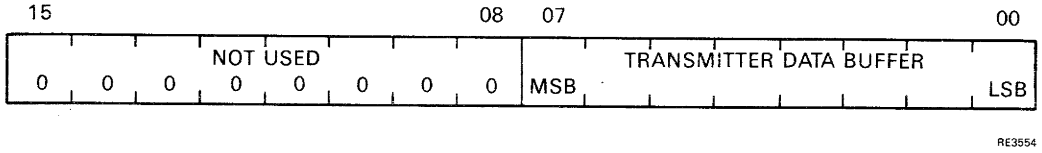


Table 1-12: Transmitter Data Buffer (XBUF) Bits

Bit	Access	Description
<15:08>	R	not used—always read 0
<07:00>	R/W	Transmitter Data Buffer—When a byte is written into this buffer, the Transmitter Ready bit (XCSR<07>) is cleared. This byte is copied into the transmitter serial output register when it is empty and XCSR<07> is cleared. Copying the byte into the serial output register sets XCSR<07>. These bits are cleared on power-up or reboot.

1.3 GENERAL PURPOSE REGISTERS

As shown in Figure 1-1 and listed in Table 1-13, there are 16 general purpose registers. Only eight are visible to the user at any given time. All these registers can be used for accumulators, deferred address, index references, autoincrement, autodecrement, and stack pointers; however, registers 6 and 7 normally have the specific functions described below.

Table 1–13: General Purpose Registers

Register	Mnemonics		
0	R0	R0′	
1	R1	R1′	
2	R2	R2′	
3	R3	R3′	
4	R4	R4′	
5	R5	R5′	
6	KSP	SSP	USP
7	PC		

PSW<15:11> (Table 1–3) determine which eight registers are currently selected. Six of the registers are selected by PSW< BITMAP>(11) as follows:

Bit<11>	Register Set
0	R0 through R5
1	R0′ through R5′

1.3.1 Stack Pointer

Register R6 is the system stack pointer. Three stack pointers are available, one for each protection mode; however, only one is visible to the user at any given time. For most instructions, PSW<15:14> determine the active stack pointer. In MFPI, MFPD, MTPI, and MTPD instructions, when PSW<15:14> select R6 as the destination, PSW<13:12> select the stack pointer. See Table 1–14.

Table 1–14: Stack Pointer Selection

PSW <15:14> or PSW <13:12>	Selected Stack Pointer (R6)
0 0	KSP (Kernel Stack Pointer)
0 1	SSP (Supervisor Stack Pointer)
1 0	illegal—USP selected
1 1	USP (User Stack Pointer)

1.3.2 Program Counter

Register R7 is the PC (Program Counter), and controls the instruction sequence. It contains the 16-bit address of the next word to be processed in the instruction stream. The PC is directly accessible by single and double operand instructions. Although the PC is a general purpose register, it is not normally used as an accumulator.

1.4 INTERRUPTS

The trap, hardware, and software interrupts are listed in Tables 1–15 and 1–16. The priority scheme is shown in Table 1–17.

The Q22-bus provides four interrupt request lines (BIRQ<7:4>) to allow hardware to interrupt the processor; however, the KDJ11-D/S interprets only priority level 4 (BIRQ4) hardware interrupts. BIRQ<5:7> are not used but are terminated on the module.

The PIRQ Register provides seven levels of software interrupt requests. Vectored traps are provided to flag error conditions.

Table 1–15: Asynchronous Interrupts

Interrupt	I/E[1]	Vector Address [2]	Priority Level[3]
Red stack trap CPU Error Register<02>	I	4	NM
Address error CPU Error Register<06>	I	4	NM
Memory management violation MMR0<15:13>	I	250	NM
Timeout/non-existent memory CPU Error Register<05:04>	I	4	NM
Parity error (PARITY, ABORT)	E	114	NM
Trace trap PSW<04>	I	14	NM
Yellow stack trap CPU Error Register<03>	I	4	NM
Power fail (PWRF)	E	24	NM
Floating point exception (FPE)	E	244	NM

[1] I = internal, E = external

[2] NU = not used, UD = user-defined

[3] NM = non-maskable interrupts, unaffected by priority specified in PSW<07:05>.

Table 1–15 (Cont.): Asynchronous Interrupts

Interrupt	I/E[1]	Vector Address [2]	Priority Level[3]
PIR 7 (PIRQ<15>)	I	240	7
IRQ 7 (BIRQ7)	E	NU	4
PIR 6 (PIRQ<14>)	I	240	6
BEVNT	E	100	4
IRQ 6 (BIRQ6)	E	NU	4
PIR 5 (PIRQ<13>)	I	240	5
IRQ 5 (BIRQ5)	E	NU	4
PIR 4 (PIRQ<12>)	I	240	4
IRQ 4 (BIRQ4)	E	UD	4
PIR 3 (PIRQ<11>)	I	240	3
PIR 2 (PIRQ<10>)	I	240	2
PIR 1 (PIRQ<09>)	I	240	1
Halt line (BREAK)[4]	E	none	NM

[1] I = internal, E = external

[2] NU = not used, UD = user-defined

[3] NM = non-maskable interrupts, unaffected by priority specified in PSW<07:05>.

[4] Places system in Console ODT mode if jumper W11 is not installed. (W11 has no affect on Halt line; it only enables/disables BREAK.)

Table 1–16: Synchronous Interrupts

Interrupt	Vector Address
FP Instruction Exception	244
TRAP Instruction	34
EMT Instruction	30
BPT Instruction	14
CSM Instruction	10
HALT Instruction	4
WAIT Instruction[1]	

[1] Does not trap, but frees the bus when waiting for an external interrupt.

Table 1-17: Interrupt Priorities

Priority	Level	Device
Highest	7	Power Fail
	6	BEVNT
	5	none
	4	SLU 0 Receiver
	4	SLU 0 Transmitter
	4	SLU 1 Receiver
	4	SLU 1 Transmitter
Lowest	4	Q22-bus Devices

1.4.1 Halt Line

The Halt line (BHALT) usually has the lowest interrupt priority, except during vector reads when it has the highest priority. This allows the user to break out of potential infinite loops. An infinite loop could occur if a vector has not been properly mapped during memory management operations.

The Halt line is driven by the BREAK signal from the Console SLU (DLART0) if jumper W11 is not installed. If W11 is installed, BREAK is disabled (tied low).

When enabled and asserted, the Halt line forces the KDJ11-D/S into Console ODT mode.

1.4.2 HALT Instruction

HALT instruction execution behavior depends on the current protection mode and the Halt Option jumper, W1.

The HALT instruction is not legal in supervisor and user modes. If executed in either of these modes, the processor traps to location 4, and sets CPU Error Register<07>.

When a HALT instruction is executed in kernel mode:

- If Halt Option jumper W1 is not installed, the processor sets up an emergency stack at location 4, traps through location 4, and sets CPU Error Register<07>.
- If jumper W1 is installed, the processor halts and enters console ODT mode.

1.5 MEMORY MANAGEMENT

The MMU (Memory Management Unit) provides access to all of physical memory with complete memory management and protection in kernel, supervisor, and user modes. It provides relocation and memory protection for multi-user, multi-programming systems. The basic characteristics of the MMU include:

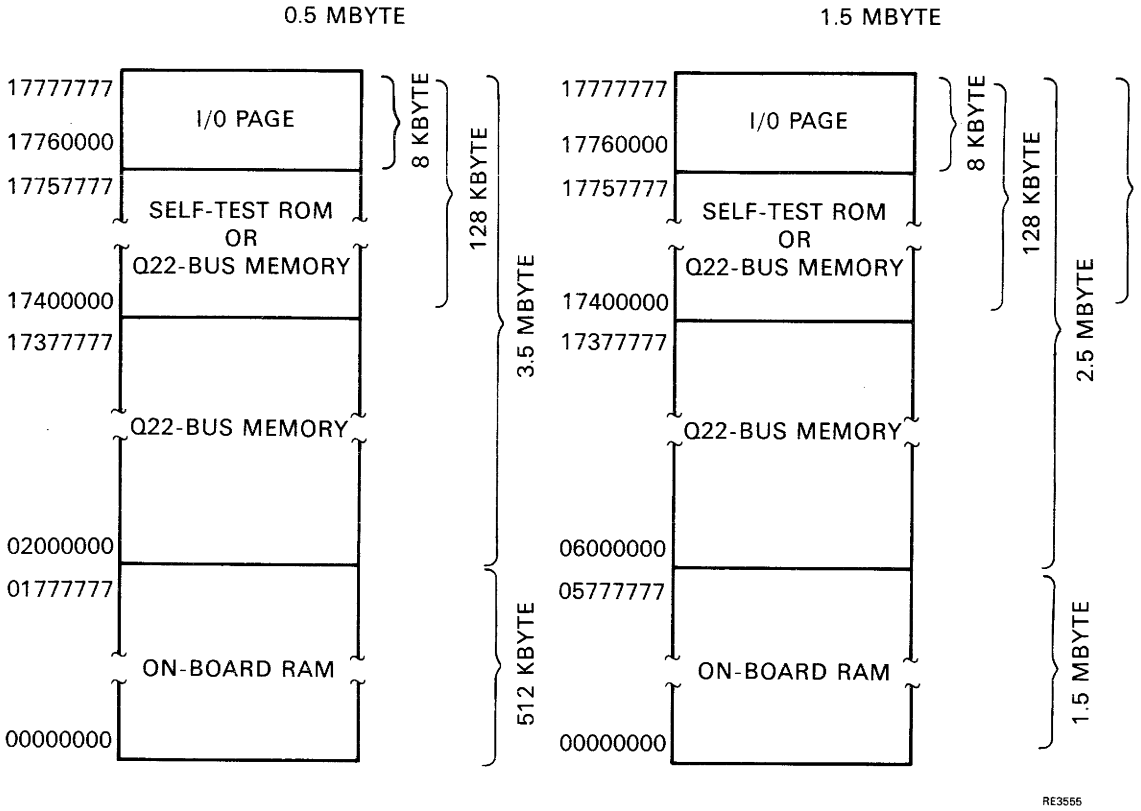
- 16 kernel mode memory pages (8 data, 8 instruction)
- 16 supervisor mode memory pages (8 data, 8 instruction)
- 16 user mode memory pages (8 data, 8 instruction)
- page length from 64 to 8192 bytes
- full protection and relocation for every page
- transparent operation
- 3 memory access control modes
- memory access up to 4 Mbytes

The three protection modes permit layered software protection. Memory management separately manages each mode, allowing each mode to access different sections of memory; and each section can have different access protection. The protection scheme allows a higher mode to enter a lower mode, but a lower mode cannot enter a higher mode. Kernel mode has full privileges and can execute all instructions. The two lower modes, supervisor and user, cannot execute certain instructions.

1.5.1 Physical Address Space

The 4-Mbyte physical address space is allocated to several memory types and/or functions, as shown in Figure 1–12. All memory, except Q22-bus memory, is on-board. Q22-bus memory is optional.

Figure 1–12: Physical Address Space



1.5.1.1 I/O Page

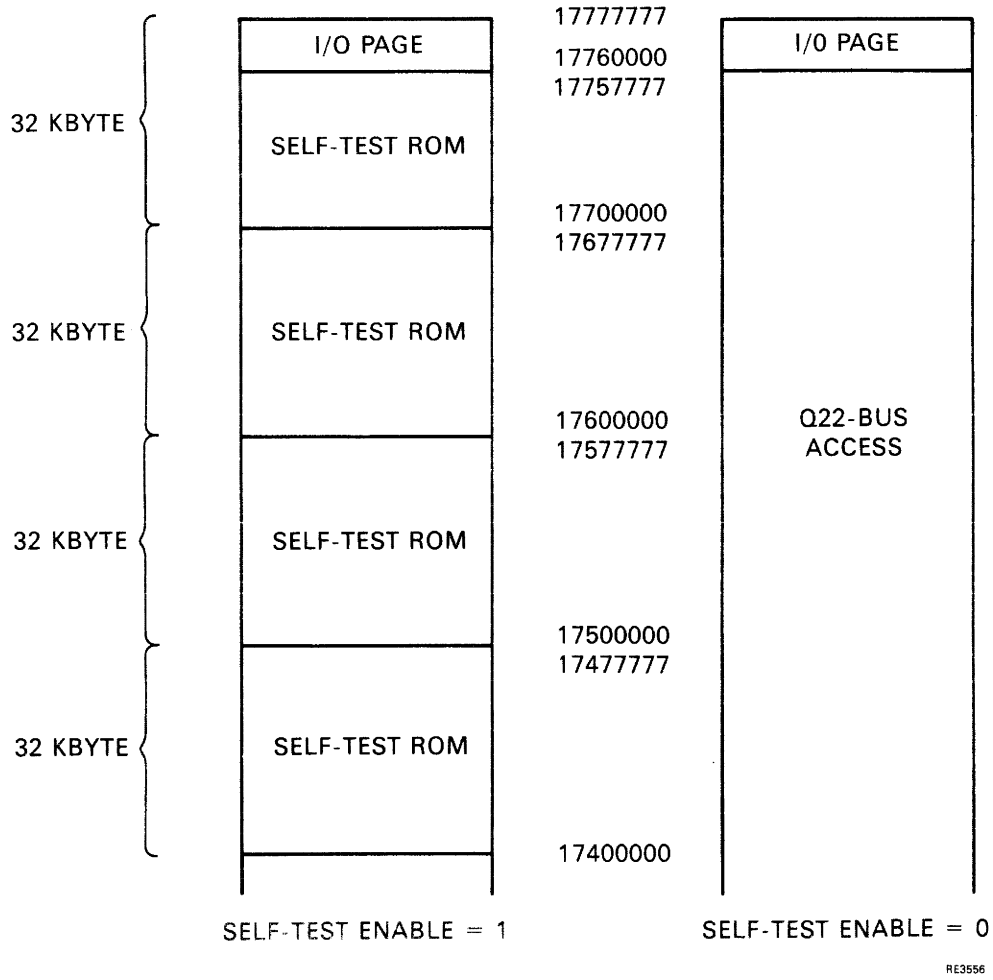
The I/O page is the top 4 kwords (8 kbytes) of memory, and always at physical address locations 17760000 through 17777777 (Figure 1–13). Immediately below the I/O page is Q22-bus memory space and Self-test ROM space. Q22-bus memory is accessed if Self-test Enable (Native Register<07>) is not set, and Self-test ROM space is accessed if Self-test Enable is set.

1.5.1.2 Self-test ROM

Self-test ROM space extends from 17400000 through 17757777. The addresses are "wrapped-around" on 32-kbyte boundaries if 16-kbyte ROMs are installed, and on 64-kbyte boundaries if 32-kbyte ROMs are installed. (During manufacture, either 16-kbyte ROMs and R16 are installed and W13 is removed, or 32-kbyte ROMs and W13 are installed and R16 is removed.)

Because the 8-kbyte I/O page resides at the top of Self-test ROM space, the starting address must be 17400000, 17500000, or 17600000 to access all the 16-kbyte ROM locations; or 17400000 to access all the 32-kbyte ROM locations.

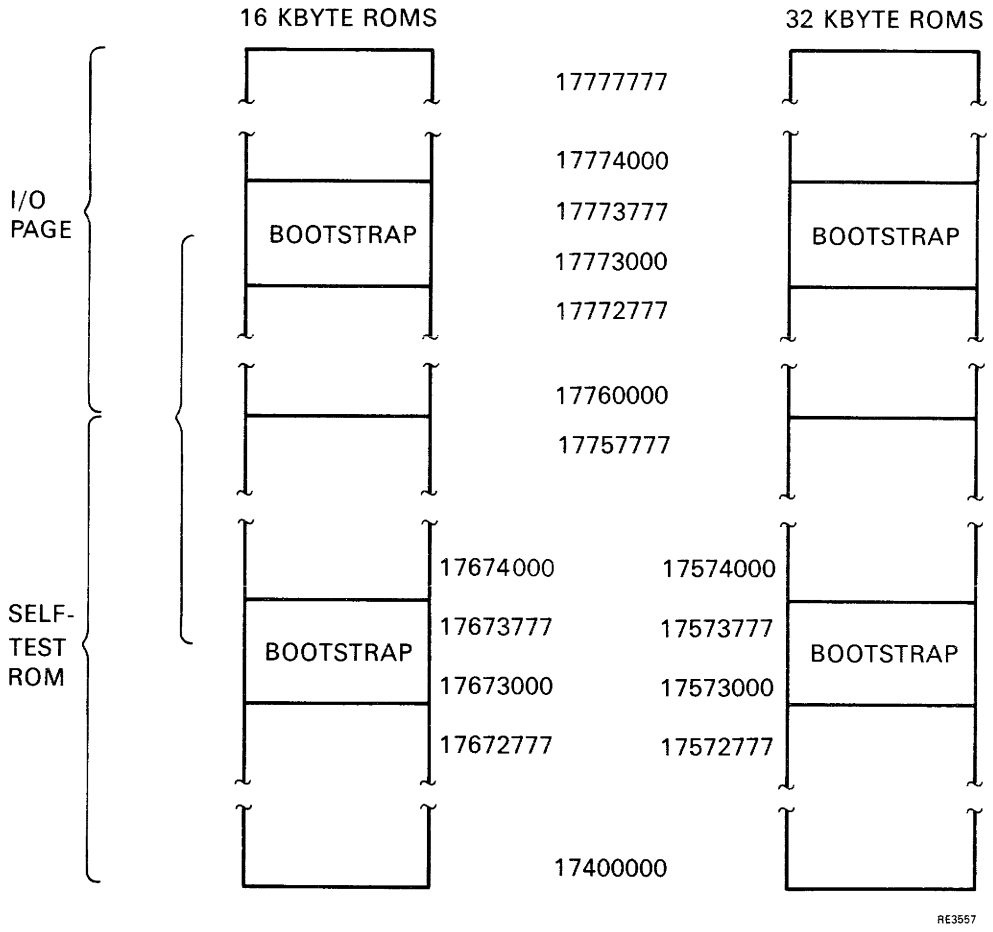
Figure 1–13: Self-test ROM Space



1.5.1.3 Bootstrap

The bootstrap program resides in a 512-byte section of Self-test ROM. Its location is 17673000 to 17673777 when 16-kbyte ROMs are installed, and 17573000 to 17573777 when 32-kbyte ROMs are installed (Figure 1–14). It is also accessible from locations 17773000 to 17773777 in the I/O page. Although the bootstrap program can be read from two locations, it executes only in the I/O page (that is, from starting address 17773000).

Figure 1–14: Bootstrap Space



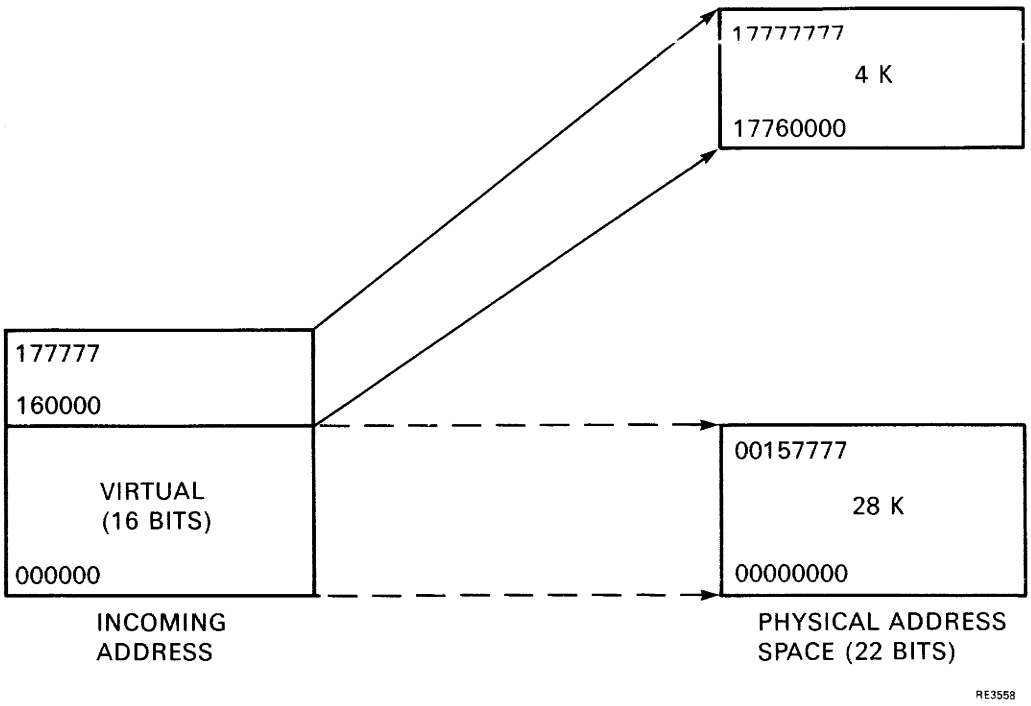
1.5.2 Virtual Memory Mapping

To access memory, the processor can perform 16-, 18-, or 22-bit address mapping.

1.5.2.1 16-bit Mapping

In the lowest 28-kwords, the virtual and physical address are the same (Figure 1–15). The I/O page occupies the upper 4-kword block.

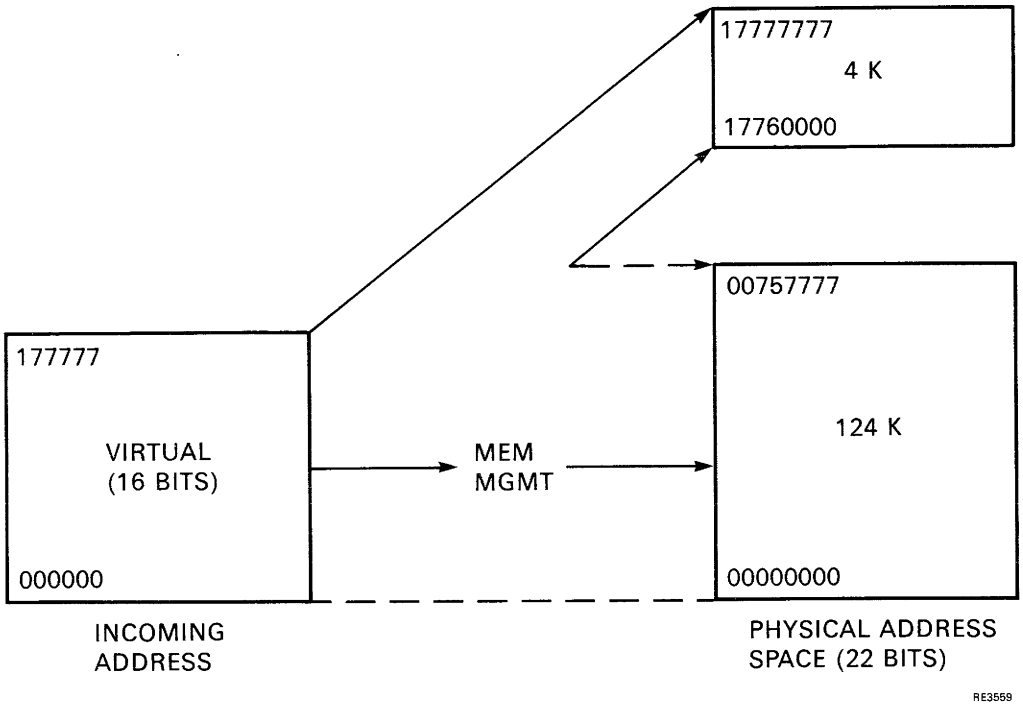
Figure 1–15: 16-bit Mapping



1.5.2.2 18-bit Mapping

Kernel, supervisor, and user modes are each allocated 32-kwords, mapped into physical address space. The 128-kword physical address space, including the 4-kword I/O page, is shown in Figure 1–16.

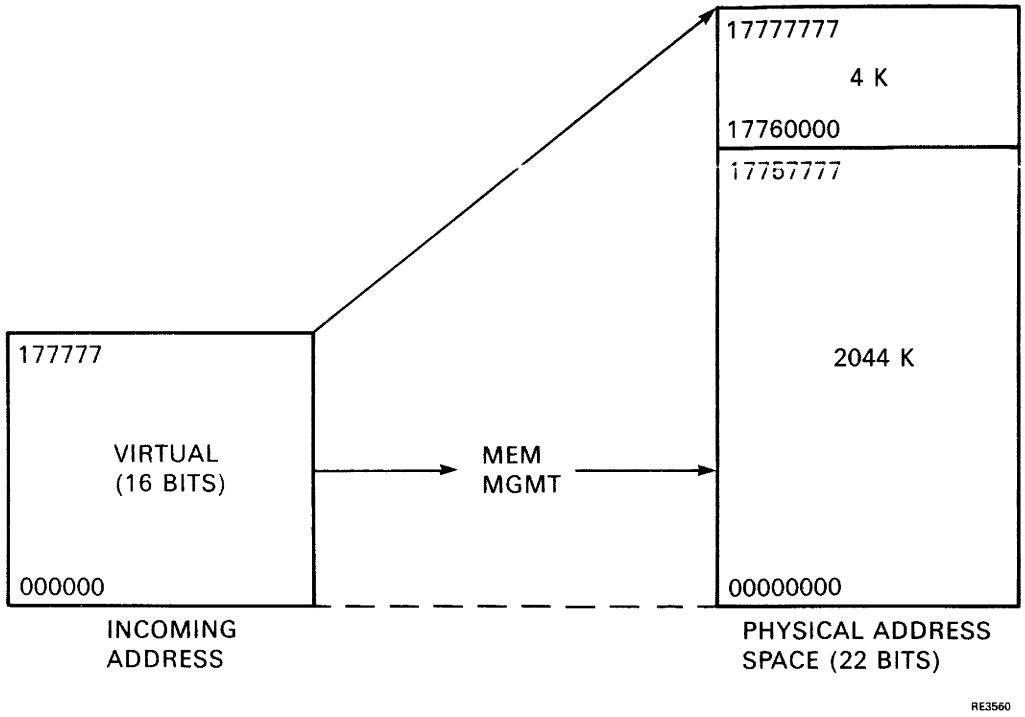
Figure 1–16: 18-bit Mapping



1.5.2.3 22-bit Mapping

The full 22-bit address is used to access all of physical memory. As Figure 1–17 shows, the I/O page remains at the top.

Figure 1–17: 22-bit Mapping



1.5.3 Compatibility

Compatibility with other PDP11 computers is provided by the 16-, 18-, and 22-bit mapping. Software written and developed for other PDP11 systems can be run on the KDJ11-D/S. Table 1–18 lists the compatible systems.

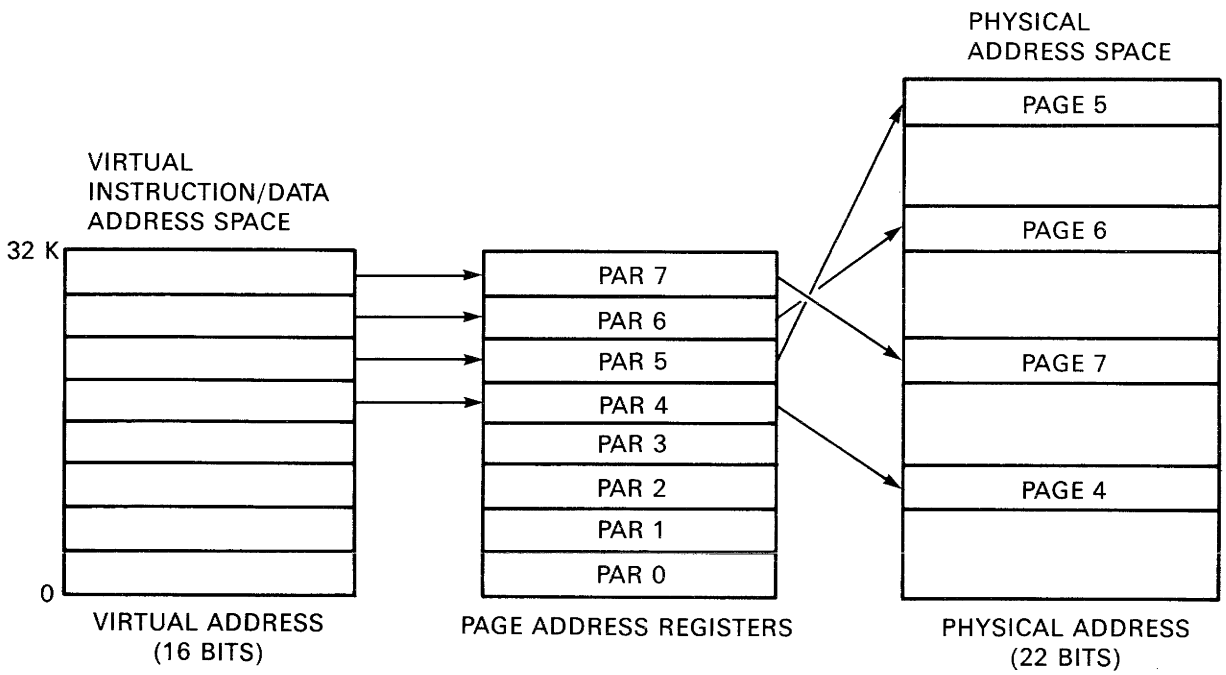
Table 1–18: KDJ11-D/S Compatibility

Mapping	Memory Management	Systems: PDP11/
16-bit	OFF	03, 05, 10, 15, 20
18-bit	ON	23, 35, 40, 45, 50
22-bit	ON	23 PLUS, 24, 44, 70, 73, 83, 84

1.5.4 Virtual Addressing

Using memory management, memory is dynamically allocated in pages, each having 1 to 128 blocks of 64 bytes (that is, from 64 to 8192 bytes per page). The contents of the 16-bit virtual address is combined with relocation information from a PAR (Page Address Register), to provide a 22-bit physical address (Figure 1–18). The starting physical address for each page is an integral multiple of 64 bytes. Pages can be located anywhere in the physical address space.

Figure 1–18: Virtual to Physical Address Mapping



1.5.5 Interrupts Under Memory Management Control

All addresses with memory relocation enabled reference either I space (instruction space) or D space (data space). I space is used for instruction fetches, index words, absolute addresses, and immediate operands. D space is used for all other references. I space and D space each have 16 management registers: eight PARs (Page Address Registers) and eight PDRs (Page Descriptor Registers), in each operating mode (kernel, supervisor, and user). MMR3<02:00> (Table 1–23) can disable D space, mapping all references through I space.

The current mode of operation (kernel, supervisor, or user) and the type of reference (instruction or data) determines which set of 16 page registers (8 PAR and 8 PDR) is used to provide the physical address.

Memory management relocates all addresses. When it is enabled, all traps, aborts, and interrupt vectors are mapped using the kernel-mode D-space mapping registers. Therefore, when a vectored control transfer occurs, the new PC and PSW values are read from two consecutive physical locations at the trap vector mapped with the kernel-mode D-space registers.

The current PC and PSW are pushed onto the stack specified by new PSW<15:14>. These bits also indicate the new mapping register set. The kernel mode program has complete control for servicing all traps, aborts, or interrupts. The kernel program can assign the service of some of these conditions to a supervisor or user mode program; it sets the new PSW mode bits in the vector, to return control to the appropriate mode.

1.5.6 Building a Physical Address

The physical address is obtained from the virtual address (Figure 1–19, and Tables 1–19 and 1–20) and the appropriate PAR set.

Figure 1–19: Virtual Address Fields

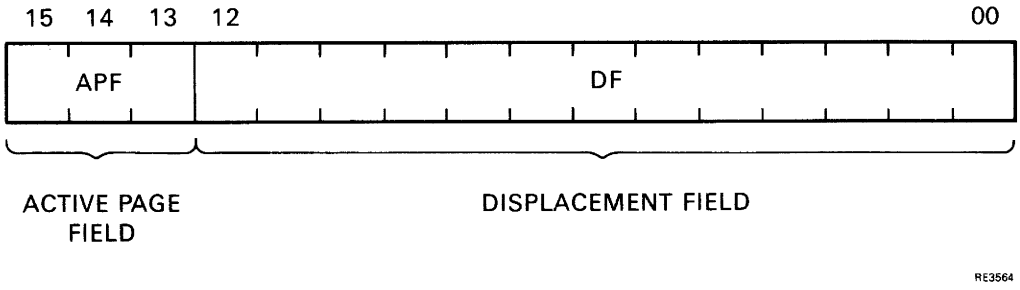


Table 1–19: Virtual Address Description

Bits	Description
<15:13>	Active Page Field—Selects one of the eight PARs to use in forming the physical address.
<12:00>	Displacement Field—Contains an address relative to the start of a page. The maximum number of bytes is 8192, therefore the largest address is 1777. This field is sub-divided into two fields (Figure 1–20 and Table 1–20).

Figure 1–20: Virtual Address Displacement Field

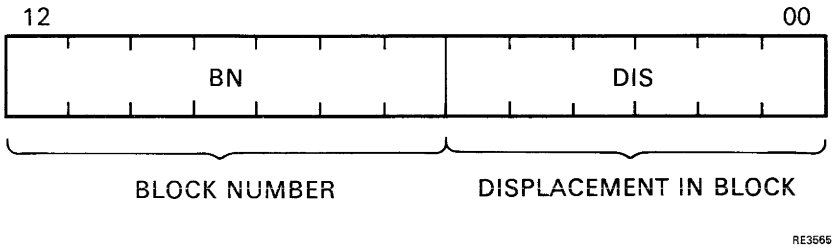


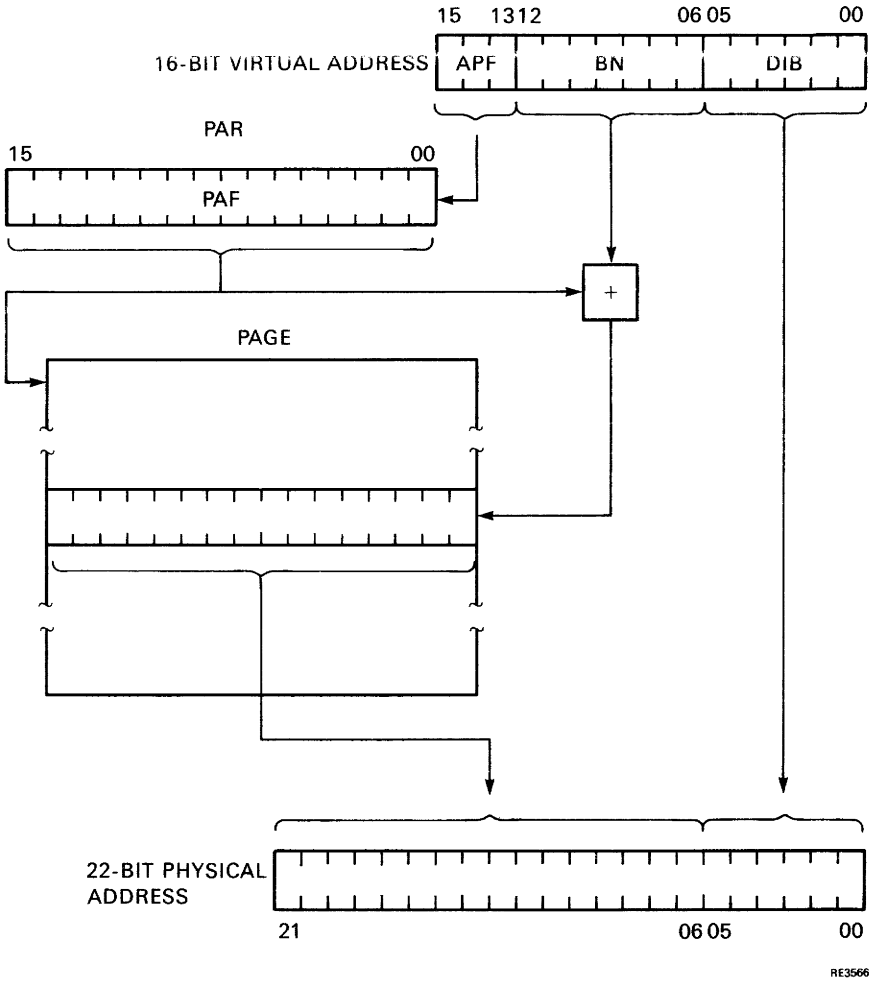
Table 1–20: Displacement Field Description

Bits	Description
<12:06>	Block Number—Selects one of the 128 64-byte blocks in the page.
<05:00>	Displacement in Block—Selects one of the 64 bytes in a block.

Figure 1–21 shows how a physical address is assembled. The sequence for building a physical address is:

1. The PAR set is selected according to the protection mode and type of reference (instruction or data).
2. One of eight PARs is selected by the virtual address APF (Active Page Field).
3. The PAF (Page Address Field) of the selected PAR contains the starting address of the currently active page. This address is a block number (64 bytes/32 words per block).
4. The BN (Block Number) from the virtual address DF (Displacement Field) is added to the PAF; the result is the number of the block in physical memory. This is bits <21:06> of the 22-bit physical address.
5. The DIB (Displacement In Block) field (bits <05:00>) from the virtual address DF is appended to the physical block number (bits <21:06>) to provide the full 22-bit physical address.

Figure 1–21: Physical Address Assembly



1.5.7 Management Registers (PARs and PDRs)

There are three sets of 32 16-bit management registers (Figure 1–22), one set for each operating mode (kernel, supervisor, user). Each set of 32 is divided into 16 I-space registers and 16 D-space registers. Each set of 16 comprises 8 PARs and 8 PDRs. PARs and PDRs are always selected in pairs, by the virtual address APF, bits <15:13>.

The management registers are located in the top 8 kbytes of physical address space; that is, in the I/O page. Table 1–2 lists the addresses assigned to the management registers.

1.5.7.1 Page Address Register

The PAR (Page Address Register) contains the 16-bit PAF (Page Address Field). See Figure 1–23. The PAF specifies the starting address of the page as a block number in physical memory (Section 1.5.6). The contents of the PARs are undefined at power-up, and are not changed by console starts or the RESET instruction.

Figure 1-22: Management Registers (PARs and PDRs)

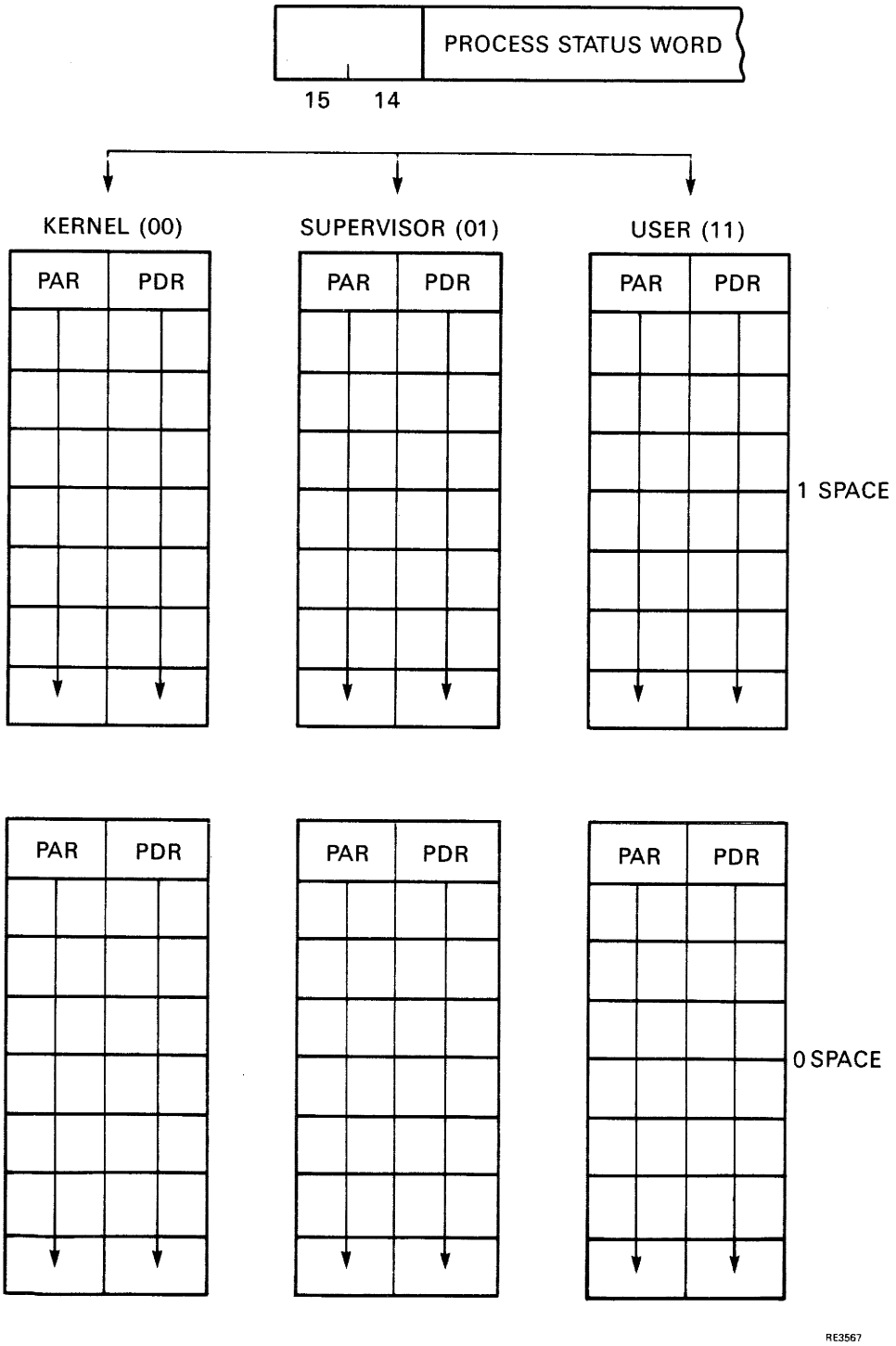
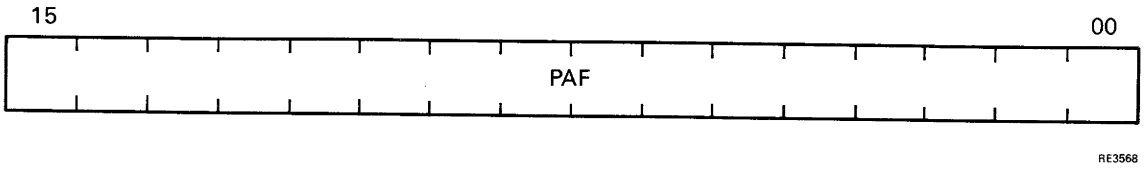


Figure 1-23: Page Address Register (PAR)



1.5.7.2 Page Descriptor Register

The PDR (Page Descriptor Register) contains page expansion, length, and access information. See Figure 1-24 and Table 1-21.

Figure 1-24: Page Descriptor Register (PDR)

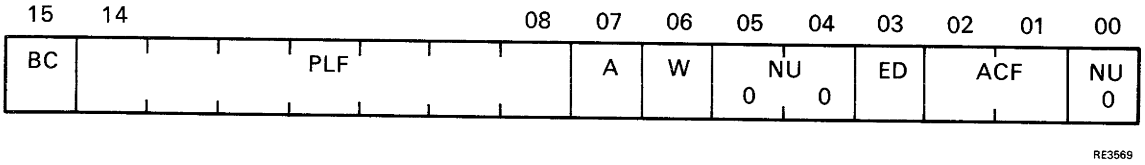


Table 1-21: Page Descriptor Register (PDR) Bits

Bit	Access	Description
<15>	R/W	Bypass Cache—Conditional cache bypass. If set in the PDR accessed during a relocation operation, the reference will go directly to main memory. Read or write hits will result in invalidation of the accessed cache location. (Cache is not used in the KDJ11-D/S.)
<14:08>	R/W	Page Length Field—Specifies the block number defining the page boundary. The virtual address BN field is compared against the PLF to detect length errors. An error occurs when expanding upward if BN>PLF, and when expanding downward if BN<PLF.
<07>	R	Attention—When set, indicates a memory management trap condition. Set by specific codes (trap conditions) in bits <02:01> to collect memory management statistics. Automatically cleared by a write reference to the page's PAR or PDR.
<06>	R	Page Written—When set, indicates the page has been written into (modified) since it was loaded into memory. This bit is automatically cleared by a write reference to the page's PAR or PDR.
<05:04>		not used

Table 1-21 (Cont.): Page Descriptor Register (PDR) Bits

Bit	Access	Description
<03>	R/W	Expansion Direction—When set, the page expands downwards from block number 127, to include blocks with lower addresses. When cleared, the page expands upwards from block number 0, to include blocks with higher addresses.
<02:01>	R/W	Access Control Field—Contains the page access code. The access code specifies how a page can be accessed, and whether a particular access should cause the current operation to abort. The codes are:

Bit 02	Bit 01	Access
0	0	Non-resident—abort all accesses
0	1	Read only—abort on writes
1	0	not used—abort all accesses
1	1	Read/write access

<00>		not used

1.5.8 Fault Recovery Registers

Aborts generated by the memory management hardware are vectored through kernel virtual location 250. MMR0, MMR1, MMR2, and MMR3 (MMU status registers 0:3) indicate why the abort occurred, and provide for program restart.

Note—Aborts

An abort to a location which is itself an invalid address will cause another abort. The kernel program must ensure that kernel virtual address 250 is mapped to a valid address; otherwise a loop will be entered and require console intervention.

1.5.8.1 Memory Management Register 0

Address: 17777572

MMR0 (Memory Management Register 0) provides MMU control and indicates status. See Figure 1-25 and Table 1-22.

Figure 1–25: Memory Management Register 0 (MMR0)

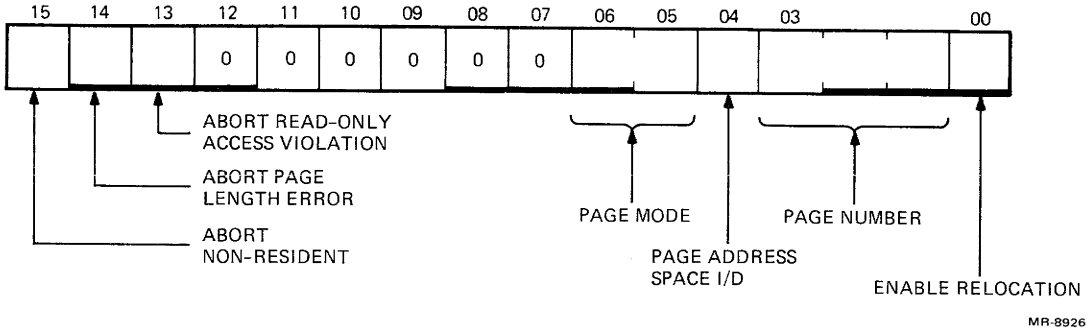


Table 1–22: Memory Management Register 0 (MMR0) Bits

Bit	Access	Description
<15>	R/W	Non-resident Abort—Set by an attempt to access a page with the ACF (PDR<02:01>) = 0 or 2, or an attempt to use relocation when processor mode (PSW<15:14>) = 2. [1]
<14>	R/W	Page Length Abort—Set by an attempt to access a location in a page with a BN (virtual address<12:06>) exceeding the PLF (PDR<14:08>) for the page.[1]
<13>	R/W	Read Only Abort—Set by an attempt to write into a read-only page (PDR<02:01> = 01). [1]
<12:07>		not used
<06:05>	R	Processor Mode—Indicate the processor mode associated with the page causing the abort. If the illegal mode is specified, an abort is generated and bit <15> is set. The modes are:

Bit 06 05	Mode
0 0	Kernel
0 1	Supervisor
1 0	Illegal
1 1	User

<04>	R	Page Space—Indicates the address space associated with the page causing the abort. If set = D space, if cleared = I space.
<03:01>	R	Page Number—Indicate the page number of the page causing the abort.
<00>	R/W	Enable Relocation—If set all addresses are relocated. If cleared, memory management is disabled, and addresses are not relocated.

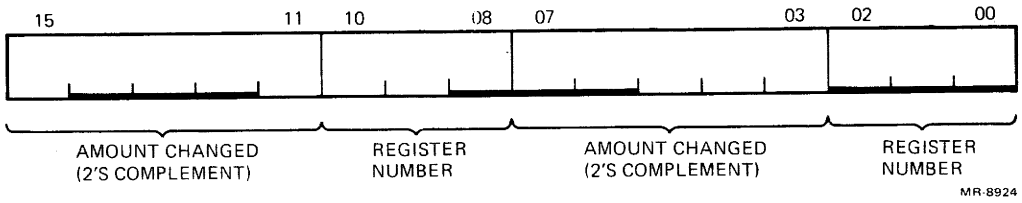
[1]Bits <15:13> can be set with an explicit write, and not cause an abort. If set explicitly or by an abort, setting any of these bits causes memory management to freeze the contents of MMR0<06:01>, MMR1, and MMR2. The registers remain frozen until MMR0<15:13> are cleared with an explicit write.

1.5.8.2 Memory Management Register 1

Address: 17777574

MMR1 records any autoincrement or autodecrement of a general purpose register, including explicit references through the PC. The increment/decrement value is stored in 2's complement (Figure 1–26). The lower byte is used for all source operand instructions; the destination operand can be stored in either byte, depending on the mode and instruction type. MMR1 is normally cleared at the start of each instruction fetch. If its contents are frozen by an abort posted in MMR0, it remains so until MMR0<15:13> are cleared by an explicit write.

Figure 1–26: Memory Management Register 1 (MMR1)



1.5.8.3 Memory Management Register 2

Address: 17777576

MMR2 is loaded with the PC value of the current instruction, and is frozen when any abort condition is posted in MMR0, and remains so until MMR0<15:13> are cleared by an explicit write.

1.5.8.4 Memory Management Register 3

Address: 17772516

MMR3 enables the D space for all three operating modes, enables the request for the supervisor macroinstruction (CSM instruction), and selects either 18- or 22-bit mapping. MMR3 is cleared by power-up, console restart, or a RESET instruction. See Figure 1–27 and Table 1–23.

Figure 1-27: Memory Management Register 3 (MMR3)

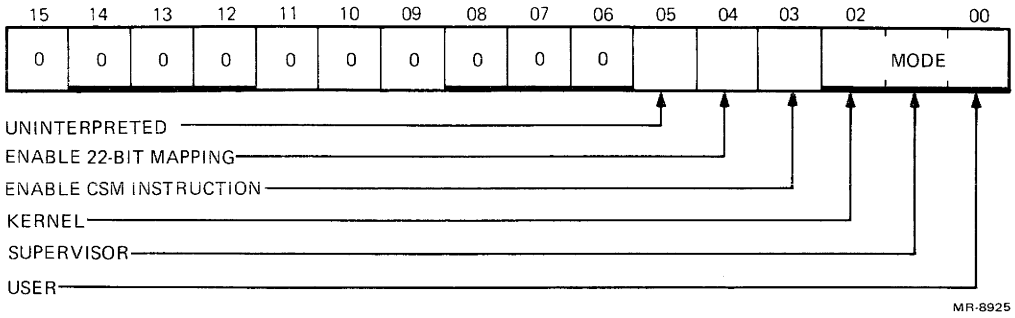


Table 1-23: Memory Management Register 3 (MMR3) Bits

Bit	Access	Description
<15:06>		not used
<05>	R/W	Uninterpreted—Can be set and cleared by software but is not interpreted by the KDJ11-D/S.
<04>	R/W	Enable 22-bit Mapping—When set, 22-bit memory addressing is enabled (the default is 18-bit addressing).
<03>	R/W	Enable CSM Instruction—When set, enables recognition of the CSM instruction.
<02>	R/W	Kernel Data Space—When set, mapping for the kernel data space is enabled.
<01>	R/W	Supervisor Data Space—When set, mapping for the supervisor data space is enabled.
<00>	R/W	User Data Space—When set, mapping for the user data space is enabled.

1.5.8.5 Instruction Back-up/Restart Recovery

Backing-up and restarting a failed instruction includes:

1. Performing the appropriate memory management tasks to clear the cause of the abort; for example, loading a missing page.
2. Restoring the contents of the general purpose registers indicated in MMR1<10:08,02:00> to their value at the start of the instruction (the increment/decrement in MMR1<15:11,07:03> is subtracted/added).
3. Restoring the contents of the PC by loading R7 with the value in MMR2.

This procedure relies on the use of different general purpose register sets for the program segment and the recovery routine.

1.5.8.6 Clearing Status Registers After an Abort

MMR0<15:13> must be cleared (set to zero) at the end of a fault service routine in order for MMR0, MMR1, and MMR2 to resume their monitoring functions.

1.5.8.7 Multiple Faults

The first abort will freeze the contents of MMR0, MMR1, and MMR2. Subsequent errors will not change the contents of these registers, until the first error has been cleared, and MMR0<15:13> have been cleared.

1.5.9 Typical Usage

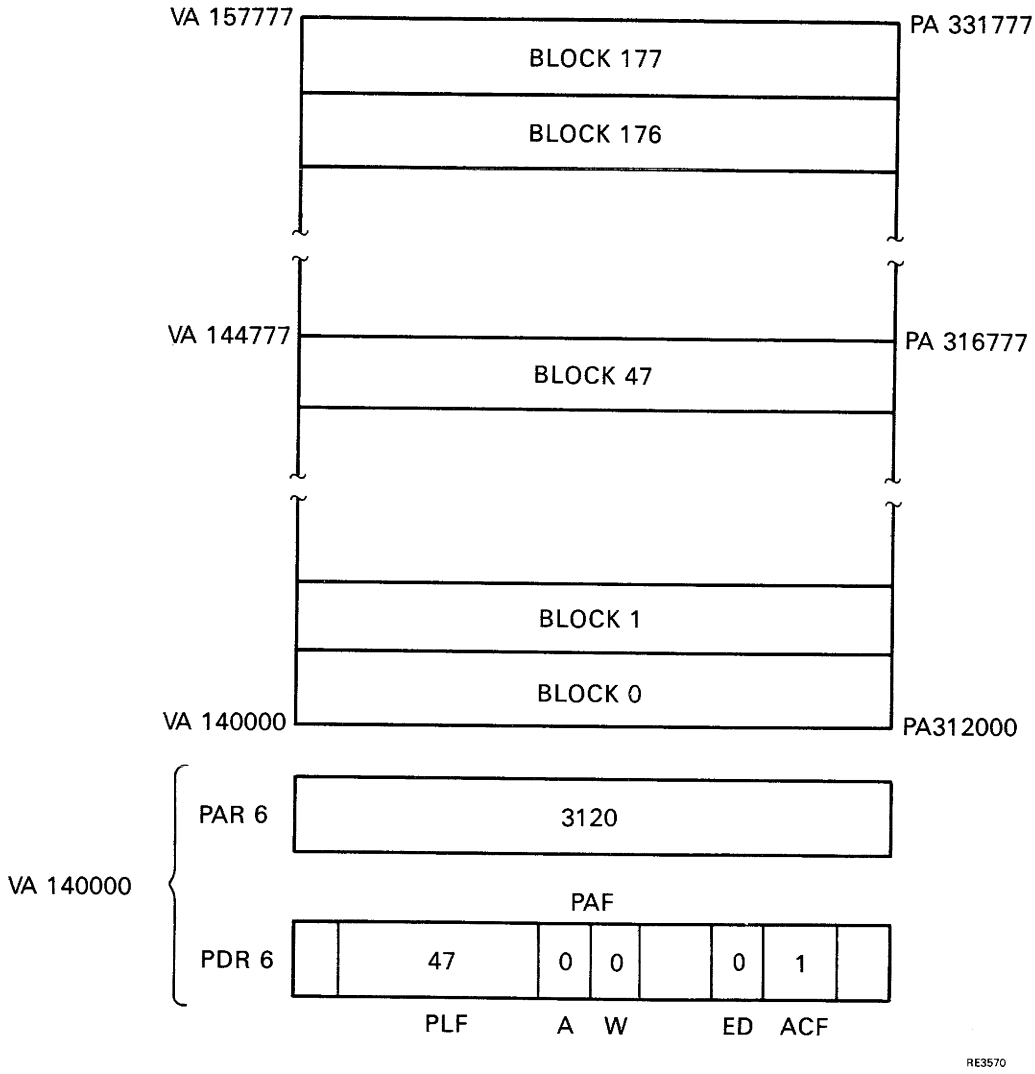
The following sections describe three typical uses of memory management, where a supervisory program operating in kernel mode dynamically assigns various sized memory pages in response to system needs.

1.5.9.1 Typical Memory Page

When the MMU is enabled, the kernel mode program, a supervisor mode program, and a user mode program are each assigned eight active pages for data and eight active pages for instructions. These pages are described by the appropriate PARs and PDRs. Each page contains from 1 to 128 blocks, and is pointed to by the PAF (Page Address Field) in the corresponding PAR. Figure 1–28 shows a typical page in an 128 block segment. The page has the following attributes:

- Page Length = 40 blocks
- Virtual Address range = 140000:144777
- Physical Address range = 312000:316777
- The page has not been modified (not written into)
- Read-only access
- Upward expansion

Figure 1-28: Typical Memory Page



The attributes are determined as follows:

1. PAR 6 and PDR 6 are selected by the VA (virtual address) APF (Active Page Field). That is, VA<15:13> = 6.
2. The PAF (Page Address Field) of PAR 6 determines the page starting address:

   3120_8 X 100_8 bytes/block = 312000_8
3. The PLF (Page Length Field) of PDR 6 determines the page length:

   block number 47_8 + 1 = 50_8 blocks

Any attempt to reference beyond 50_8 blocks will cause a page length error, resulting in an abort vectored through kernel address 250.

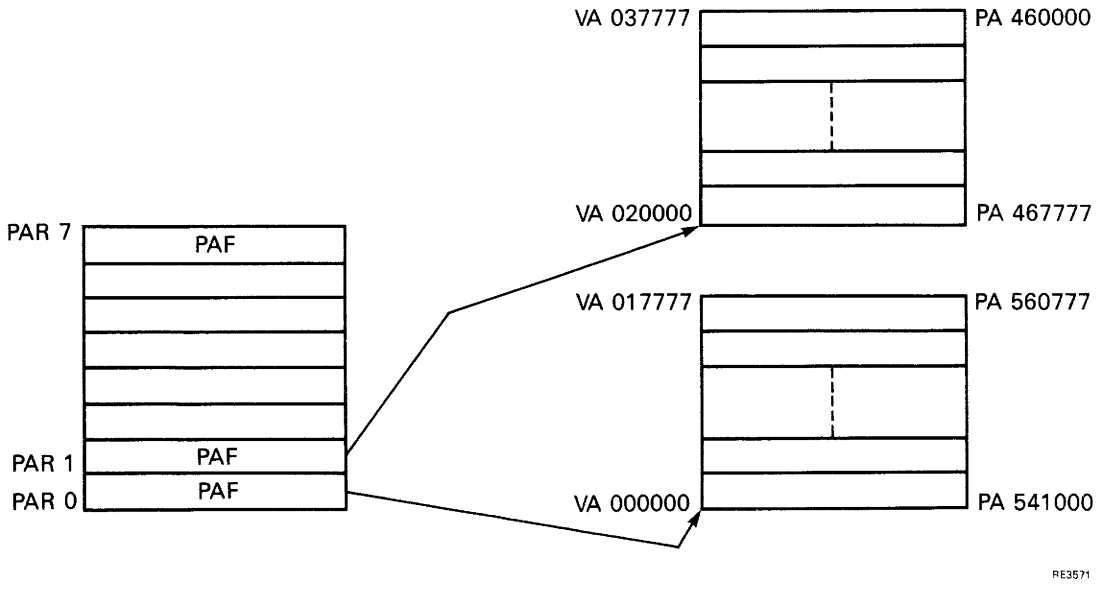
4. The physical addresses are assembled as shown in Figure 1–21.
5. The W bit (PDR6<06>) indicates that this page has not been modified. The state of this bit determines if the page must be written back to mass storage in a swapping or memory overlay scheme.
6. The ACF (Access Control Field) (PDR6<02:01>) indicates this page is protected by read-only access. Any attempt to modify (write) any location in this page will cause an access control violation abort.
7. The ED (expansion direction) bit (PDR6<03>) is zero, indicating upward expansion. If additional blocks are required for this page, blocks with higher addresses will be assigned.

All these attributes can be determined by software. The parameters are loaded into the appropriate PAR and PDR under program control. In normal applications, the page containing the PARs and PDRs is assigned to kernel mode program control.

1.5.9.2 Non-consecutive Memory Pages

Higher virtual addresses do not necessarily map to higher physical addresses. A single memory page must comprise a block of contiguous locations, but consecutive virtual memory pages do not have to reside in contiguous blocks of physical memory. See Figure 1–29. The assignment of memory pages is not restricted to consecutive, non-overlapping physical address locations.

Figure 1–29: Non-consecutive Memory Pages

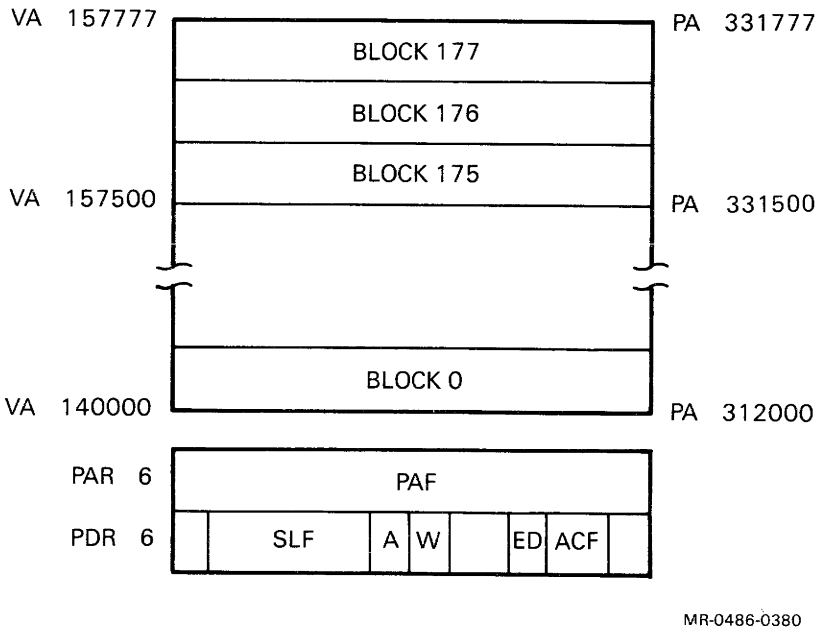


1.5.9.3 Stack Memory Pages

Program variables are often isolated from "pure code" (that is, instructions) by placing the variables in a register indexed stack. The variables are then "pushed" on or "popped" off the stack as needed. Because stacks expand by adding memory locations with lower addresses, a memory page containing stacked variables must expand downward when it needs more room. That is, blocks with lower relative addresses must be added to the page. Therefore, the expansion direction bit (PDR<03>) is set to 1. Figure 1–30 shows a stack memory page with the following parameters.

- PAF = 3120_8
- PLF = 175_8
- ED = 1
- W = 0 or 1 (set by hardware)
- ACF = determined by programmer

Figure 1–30: Typical Stack Memory Page



The stack begins 128 blocks above the relative origin of the memory page and extends downward for three blocks. A page length error abort will occur if a location below the assigned area is referenced; that is, when the BN (Block Number) from the virtual address (VA<12:06>) is less than the PLF (PDR6<14:08>).

1.6 MEMORY SYSTEM REGISTERS

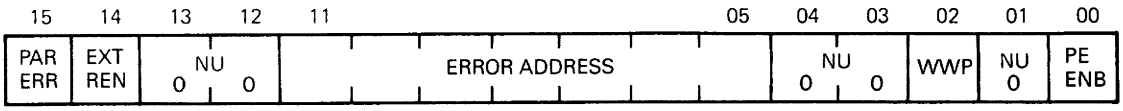
In the KDJ11-D/S, the MER (Memory Error Register) is the only memory system register used (Figure 1–1). Cache is not an option on the KDJ11-D/S, and the CCR (Cache Control Register), and the Hit/Miss Register are not used.

1.6.1 Memory Error Register

Address: 17772100

The MER (Memory Error Register) controls and monitors the memory parity function. Bits in the MER can be used to disable parity error detection, and to force parity errors for diagnostic purposes. See Figure 1–31 and Table 1–24.

Figure 1–31: Memory Error Register (MER)



MR-0486-0381

Table 1–24: Memory Error Register (MER) Bits

Bit	Access	Description
<15>	R/W	Parity Error—When set, indicates a parity error has occurred. This bit is not automatically cleared when the CPU responds to a parity error; it must be explicitly cleared by writing a zero into it. It is also cleared by power-up or bus INIT.
<14>	R/W	Extended MER Read Enable—See <11:05>. Cleared by power-up or bus INIT.
<13:12>	R	not used—always read 0
<11:05>	R/W	Error Address—If a parity error occurs, A<17:11> are stored in these bits, and A<21:18> are latched. When MER<14> = 0, reading the MER transfers MER<11:05> (A<17:11>) to the CPU. Software must then set MER<14> = 1; and read the MER a second time to transfer A<21:18> through MER<08:05>.
<04:03>	R	not used—always read 0
<02>	R/W	Write Wrong Parity—When set, wrong parity data will be written into on-board RAM on a DATO or DATOB bus cycle. Cleared by power-up or bus INIT.
<01>	R	not used—always reads 0
<00>	R/W	Parity Error Enable—When set and a parity error occurs on a Q22-bus DATI or DATIO(B) cycle from a bus master to RAM, BDAL<17:16> are asserted on the bus at the same time data is asserted. When the CPU is reading on-board RAM, a parity error causes a Parity Error trap. This bit is cleared by power-up or bus INIT.

Table 1–24 (Cont.): Memory Error Register (MER) Bits

Bit	Access	Description
		Note—Parity Errors A parity error on Q22-bus data is asserted during the current instruction cycle. A parity error on local memory data is asserted during the next instruction cycle.

1.7 FLOATING-POINT

The processor uses the floating-point instruction set to perform all floating-point arithmetic operations, and converts data between integer and floating-point formats. Similar address modes and the same memory management facilities are used. Floating-point instructions can reference the floating point accumulators, the general registers, or any locations in memory.

See the Associated Documents listed in the Preface for more details on floating-point.

1.7.1 Floating Point Registers

The floating-point registers are shown in Figure 1–1, and are defined as the FPS (Floating-point Status Register), FEA (Floating-point Exception Address Register), FEC (Floating-point Exception Code Register), and six accumulators.

1.7.1.1 Floating-point Status Register

The FPS (Floating-point Status) Register provides mode and interrupt control for the FPP and conditions caused by previous instruction execution. See Figure 1–32 and Table 1–25.

The FPP recognizes six exceptions:

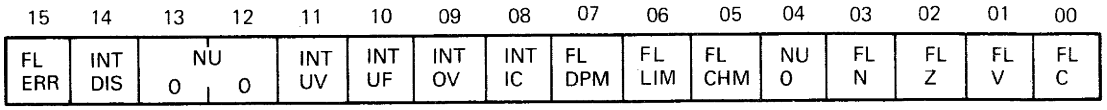
Floating opcode error
Floating divide by zero
Floating or double-to-integer conversion error
Floating overflow
Floating underflow
Floating undefined variable

Illegal opcode and divide by zero interrupts can be disabled only by setting FPS<14>. The other four exceptions can be disabled by clearing the appropriate exception bit in FPS<11:08>.

The four condition codes FPS<03:00> are equivalent to the CPU condition codes (PSW<03:00>).

The FPP sets the error flag and condition codes (FPS<15,03:00>) as part of the output of a floating-point instruction. The bits <15:14,11:05,03:00> can be set and cleared by the LDFPS instruction.

Figure 1–32: Floating-point Status Register (FPS)



MR-9377

Table 1–25: Floating-point Status (FPS) Bits

Bit	Access	Description
<15>	R/W	Floating Error—Set by a floating-point instruction if any of the following occur: • divide by zero • illegal opcode • any of the remaining exceptions with the corresponding interrupt (bits <11:08>) enabled The FPP never resets this bit. If set, it can be cleared only by an LDFPS instruction. Therefore, this bit is up to date only if the most recent floating-point instruction produced an exception. This bit is independent of <14>.
<14>	R/W	Interrupt Disable—If set, all floating-point interrupts are disabled. Primarily, this is a maintenance bit, and should normally be cleared. It must be cleared in order to generate an interrupt if the FPP stores a negative 0.
<13:12>		not used
<11>	R/W	Interrupt on Undefined Variable—An interrupt occurs if this bit is set and a negative 0 is obtained from memory as an operand of an ADD, SUB, MUL, DIV, CMP, MOD, NEG, ABS, TST, or any LOAD instruction. The interrupt occurs before execution. When cleared, negative 0 can be loaded and used in any FPP operation. The interrupt is not generated by negative 0 in any AC operand of an arithmetic instruction; in particular, trap on negative 0 never occurs in addressing mode 0. The FPP will not store the result of negative 0 without a simultaneous interrupt.
<10>	R/W	Interrupt on Underflow—When set, floating underflow will cause an interrupt. The fractional part of the result will be correct. The biased exponent will be too large by 400_8, except for the special case of 0, which is correct (see the LDEXP instruction for an exception). If an underflow occurs when cleared, the interrupt is disabled, and the result is set to exact 0.

Table 1-25 (Cont.): Floating-point Status (FPS) Bits

Bit	Access	Description
<09>	R/W	Interrupt on Overflow—When set, floating overflow will cause an interrupt. The fractional part of the result will be correct. The biased exponent will be too small by 400_8. (See the MOD and LDEXP instructions for special cases of overflow.) If an overflow occurs when cleared, the interrupt is disabled, and the result is set to exact 0.
<08>	R/W	Interrupt on Integer Conversion—An interrupt occurs when this bit is set and a conversion to integer instruction fails. If an interrupt occurs, the destination is set to 0; all other registers are unaffected. If this bit is cleared, the operation is the same as above, but no interrupt occurs. The conversion instruction fails when it generates an integer with more bits than can fit in the long or short integer word specified by <06>.
<07>	R/W	Floating Double-precision Mode—When set, double-precison is used for calculations; when cleared, single-precison is used.
<06>	R/W	Floating Long Integer Mode—Used in conversion between integer and floating-point format. When set, integer format is 2's complement double-precision (32-bit); when cleared, integer format is 2's complement single-precision (16-bit).
<05>	R/W	Floating Chop Mode—When set, the result of any arithmetic operation is "chopped" (truncated). When cleared, the result is rounded.
<04>		not used
<03>	R/W	Floating Negative—Set if the result of the last floating-point operation was negative; otherwise cleared.
<02>	R/W	Floating Zero—Set if the result of the last floating-point operation was zero; otherwise cleared.
<01>	R/W	Floating Overflow—Set if the last floating-point operation resulted in exponent overflow; otherwise cleared.
<00>	R/W	Floating Carry—Set if the last floating-point operation resulted in a carry from the most significant bit. This can only occur in a floating or double-to-integer conversion.

1.7.1.2 Floating-point Exception Registers

One interrupt vector (244) is assigned to process all floating-point exceptions. There are six possible errors, encoded in the 4-bit FEC (Floating-point Exception Code) Register:

02 Floating opcode error
04 Floating divide by zero
06 Floating or double-to-integer conversion error
08 Floating overflow
10 Floating underflow
12 Floating undefined variable

The FEA (Floating-point Exception Address) Register stores the address of the instruction that produced the exception.

One of the following updates the FEC and FEA Registers:

- Illegal opcode.
- Divide by zero.
- Any of the other exceptions with the corresponding interrupt enabled.

If any of the four exceptions, code 06 through 12, occurs with the corresponding interrupt disabled, the FEC and FEA Registers are not updated. Updating is not inhibited if the FID bit (FPS<14>) is set. The FEC and FEA Registers are not updated unless an exception occurs. This means that the STORE STATUS (STST) instruction will return current information only if the most recent floating-point instruction produced an exception. There are no instructions for storing into the FEC and FEA Registers.

1.7.1.3 Floating-point Accumulators

The six 64-bit accumulators, AC0 through AC5, provide for temporary storage and manipulation of 32- and 64-bit floating-point data types.

Chapter 2
INSTALLATION

This chapter describes how the KDJ11-D/S is configured and installed in an LSI-11 system. The module can be installed in systems using either the standard LSI-11 bus backplane or the extended LSI-11 bus (Q22-bus) backplane. Before installing the KDJ11-D/S:

1. Configure the user-selectable features.
2. Be sure that the backplane, mounting box, and options are compatible with the KDJ11-D/S.
3. Consider any system differences between the KDJ11-D/S and the LSI-11 processor it may be replacing. See Section 2.5.2.

2.1 CONFIGURATION

The KDJ11-D/S has 16 jumpers for its user-selectable features (see Figure 2–1 and Table 2–1). Earlier versions of the module have 13 jumpers, and a different module layout (see Figure 2–2 and Table 2–1). There are some additional jumpers which are factory set. Jumpers are installed by pushing an insulated jumper wire (part number 12-18783-00) onto two wirewrap pins on the module.

Figure 2–1: KDJ11-D/S Module (M7554)

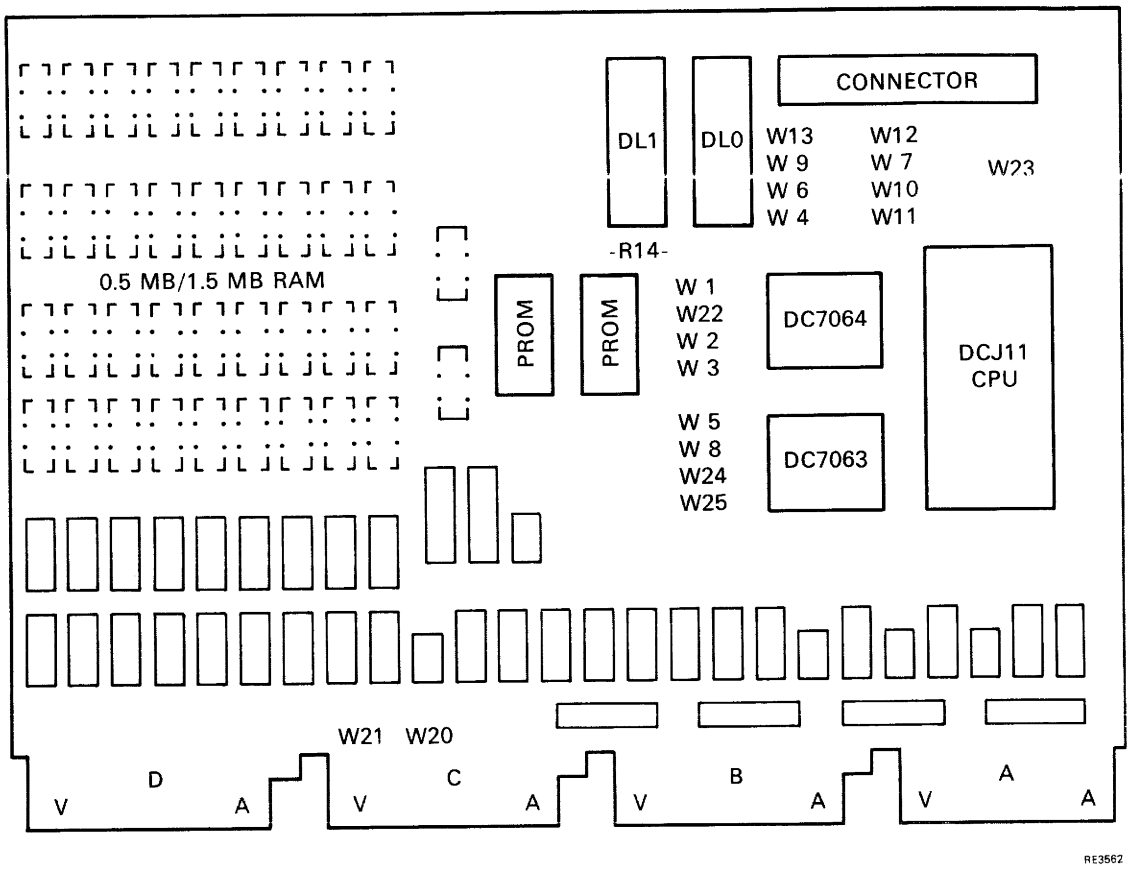


Figure 2–2: KDJ11-D/S Module - Early version (M7554)

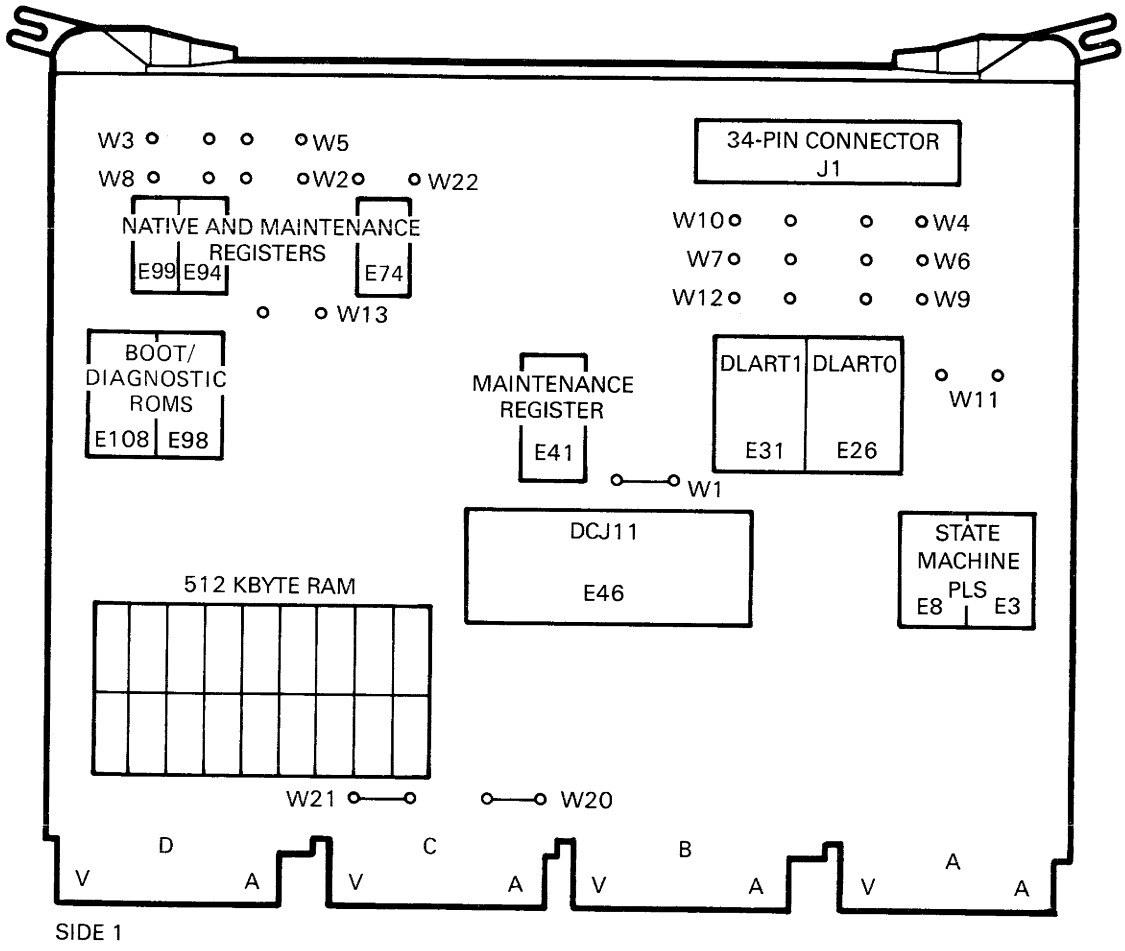


Table 2–1: Jumpers

	Jumper/Position				Comments
HALT	W1				
	In				Trap on HALT disabled[1]
	Out				Trap on HALT enabled
Boot Select	W2				Boot select—see Table 2–2
	W3				Boot select—see Table 2–2
	W5				Boot select—see Table 2–2
	W8				Boot select—see Table 2–2
	W22				Boot select—see Table 2–2
Baud Rate	W4	W6	W9	DLART0	
	W10	W7	W12	DLART1	Baud Rate[2]
	Out	Out	Out		300[1]
	Out	Out	In		600
	Out	In	Out		1200
	Out	In	In		2400
	In	Out	Out		4800
	In	Out	In		9600
	In	In	Out		19200
	In	In	In		38400
BREAK	W11				
	Out				Console BREAK enabled[1]
	In				Console BREAK disabled
ROM Size[3]	W13	R14			
	In	Out			32-kbyte self-test ROMs
	Out	In			16-kbyte self-test ROMs
Backplane[4]	W20	W21			
	In				Backplane pin CM2 to pin CN2[1]
		In			Backplane pin CR2 to pin CS2[1]
–12 V	W23				
	in				–12 V connected to J1
	out				no power to J1
Special Applications[5]	W24				
	in				Special application
	out				Normal
RAM Size[5]	W25				
	in				0.5 Mbyte of on board RAM
	out				1.5 Mbyte of on board RAM

[1]Factory setting or default

[2]To use remote switch, remove all jumpers

[3]W13, R14, and ROMs are factory installed

[4]Not user selectable—soldered in

[5]W24 and W25 are only on the re-engineered versions, and are factory installed

2.1.1 Power-up Options

Unlike other KDJ11 modules, the KDJ11-D/S does not have a power-up option jumper, but supports only power-up mode 2, bootstrap on power-up. (Maintenance Register<01> input is tied to ground and <02> input is tied HIGH.) The bootstrap starting address is fixed at 17773000. The processor sets PC<15:10> to all ones, and PC<09:00> to all zeros (that is, 173000). The processor's priority level is set to seven by loading 340_8 into the PSW (Figure 1–2). The processor then either services pending interrupts, or starts program execution, at the memory location pointed to by the PC. The processor reads the bootstrap option in Native Register<12:08> (Figure 1–7), to direct the bootstrap program.

2.1.1.1 Power-up Sequence

The power-up sequence is shown in Figure 2–3.

2.1.1.2 Power-down Sequence

The power-down sequence is shown in Figure 2–4.

2.1.2 Bootstrap Options

The bootstrap options selected by jumpers W22, W2, W3, W5, and W8 or remote switch are as shown in Table 2–2.

Figure 2–3: Power-up Sequence

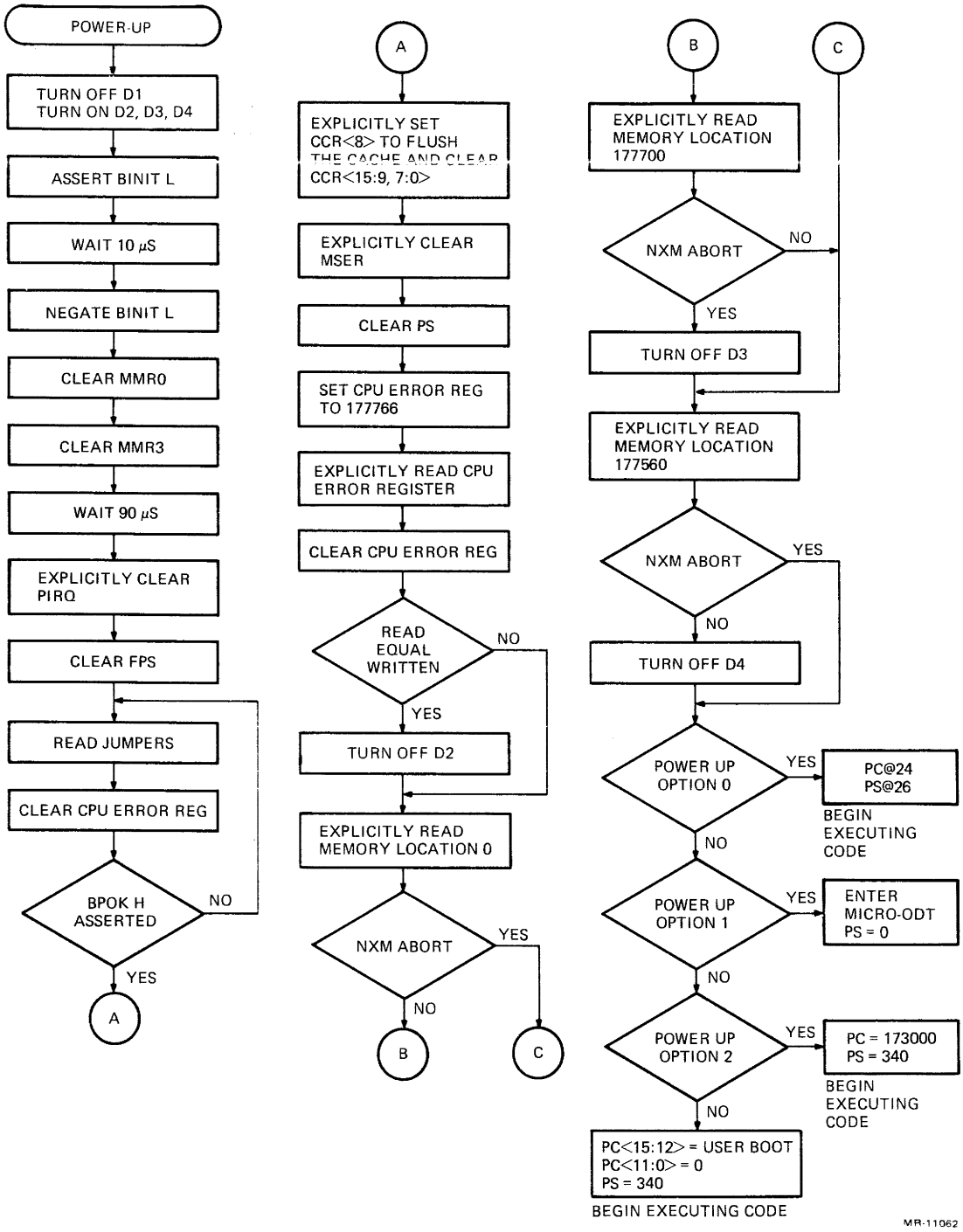


Figure 2–4: Power-down Sequence

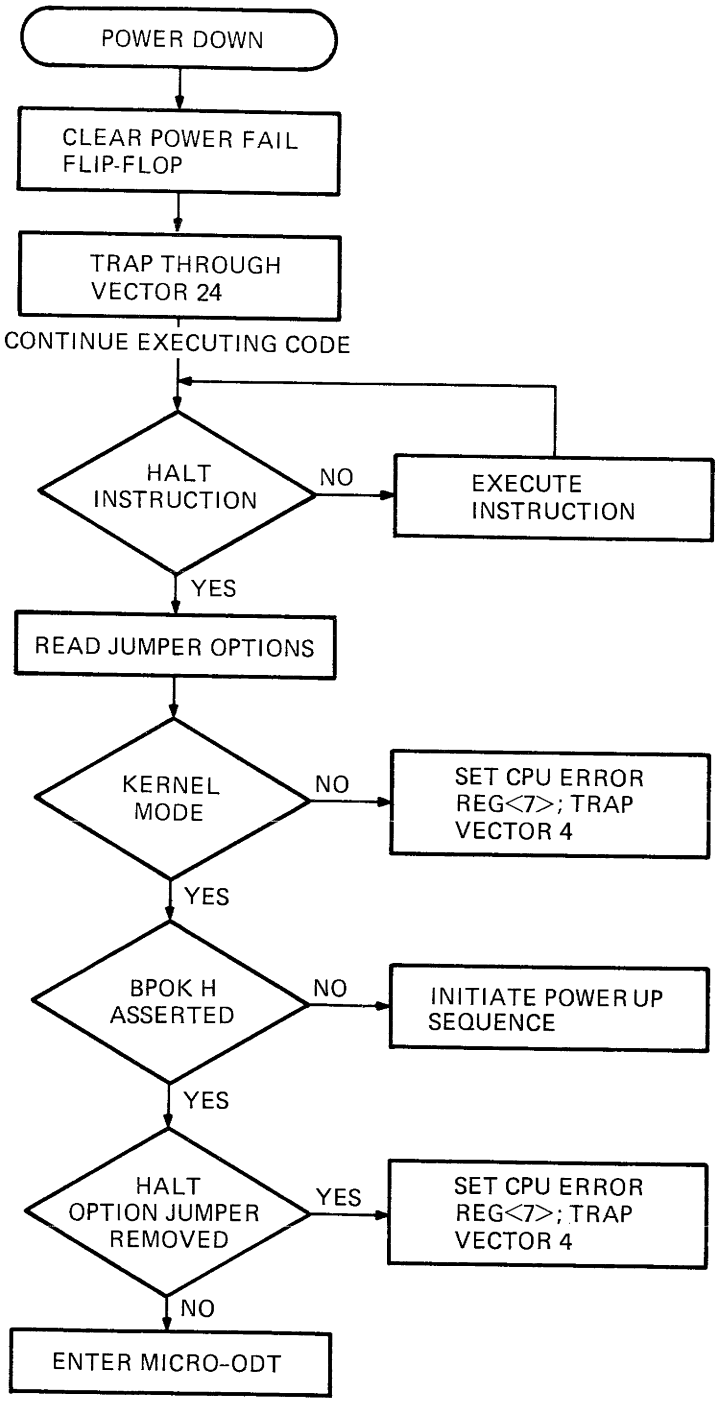


Table 2–2: Boot Select Options

Jumper W: 22	2 3 5 8[1]	Switch Position[2]	Description
W22 Installed			
0	0 0 0 0	0	Test, enter console mode using English text[3,4]
0	0 0 0 1	1	Test, enter console mode using French text
0	0 0 1 0	2	Test, enter console mode using German text
0	0 0 1 1	3	Test, enter console mode using Dutch text
0	0 1 0 0	4	Test, enter console mode using Swedish text
0	0 1 0 1	5	Test, enter console mode using Italian text
0	0 1 1 0	6	Test, enter console mode using Spanish text[3]
0	0 1 1 1	7	Test, enter console mode using Portuguese text
0	1 0 0 0	8	Test, enter console mode (reserved)
0	1 0 0 1	9	Test, enter console mode (reserved)
0	1 0 1 0	10	Test, enter console mode (reserved)
0	1 0 1 1	11	Test, enter console mode (reserved)
0	1 1 0 0	12	Test,[5] autoboot tapes & disks,[6] user selects language
0	1 1 0 1	13	Test, autoboot DPV11, DUV11, DLV11-E/F, TU58 & RK05
0	1 1 1 0	14	Test, autoboot DEQNAs 0 and 1
0	1 1 1 1	15	Manufacturing test loop
W22 Removed			
1	0 0 0 0	0	Test, autoboot tapes & disks[6] using English text[3]
1	0 0 0 1	1	Test, autoboot tapes & disks using French text
1	0 0 1 0	2	Test, autoboot tapes & disks using German text
1	0 0 1 1	3	Test, autoboot tapes & disks using Dutch text

[1]0 = Installed, 1 = Removed

[2]Jumpers W2, W3, W5, and W8 removed to use switch

[3]Only English (positions 00000 and 10000) and Spanish (positions 00110 and 10110) can be selected with version 1.0 ROMs. Eight languages can be selected with version 2.0 ROMs.

[4]Factory or default setting

[5]High speed autoboot, memory address/shorts test bypassed

[6]"tapes & disks" = DU 0-255, DU 0-255 at floating addresses, DL 0-3, DX 0-1, DY 0-1, MU 0, and MS 0. For DU, removable media is booted before fixed media.

Table 2–2 (Cont.): Boot Select Options

Jumper W:		Switch	
22	2 3 5 8[1]	Position[2]	Description
1	0 1 0 0	4	Test, autoboot tapes & disks using Swedish text
1	0 1 0 1	5	Test, autoboot tapes & disks using Italian text
1	0 1 1 0	6	Test, autoboot tapes & disks using Spanish text[3]
1	0 1 1 1	7	Test, autoboot tapes & disks using Portuguese text
1	1 0 0 0	8	Test, autoboot tapes & disks (reserved)
1	1 0 0 1	9	Test, autoboot tapes & disks (reserved)
1	1 0 1 0	10	Test, autoboot tapes & disks (reserved)
1	1 0 1 1	11	Test, autoboot tapes & disks (reserved)
1	1 1 0 0	12	Emulate Power-up mode 24 with no messages
1	1 1 0 1	13	Halt & enter ODT if trap-on-halt disabled, else loop[7]
1	1 1 1 0	14	Test, autoboot DEQNAs 0 and 1
1	1 1 1 1	15	Test, enter console mode, user selects language

[1]0 = Installed, 1 = Removed

[2]Jumpers W2, W3, W5, and W8 removed to use switch

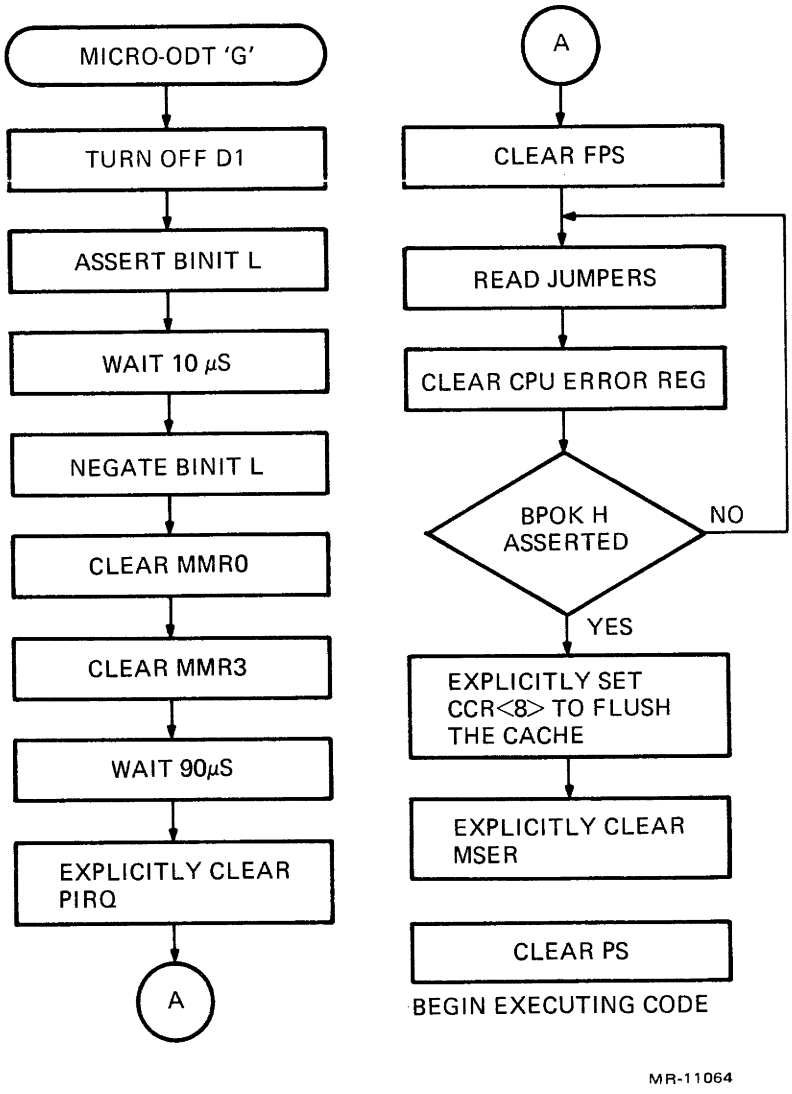
[3]Only English (positions 00000 and 10000) and Spanish (positions 00110 and 10110) can be selected with version 1.0 ROMs. Eight languages can be selected with version 2.0 ROMs.

[7]W1 = Trap on Halt (enabled = removed, disabled = installed)

2.1.3 HALT Option

The HALT option determines whether the action taken after a HALT instruction is executed in the kernel mode. When the HALT instruction is executed, the processor checks the HALT option in Maintenance Register<03>. This bit is set (its input is tied HIGH) when jumper W1 is removed, and the processor traps to location 4 in the kernel data space and sets CPU Error Register<07> (Figure 1–4). If jumper W1 is installed, Maintenance Register<03> is cleared (its input is tied to ground), and the processor enters console ODT mode when a HALT instruction is executed. Console ODT mode is exited with the G command (for more information on ODT, see the Associated Documents listed in the Preface). The exit sequence is shown in Figure 2–5.

Figure 2–5: Console ODT Exit Sequence



2.1.4 Factory Configuration

The factory, or shipped, configuration of the option jumpers is shown in Figure 2–1 and described in Table 2–1.

2.2 DIAGNOSTIC LEDS

The module has seven indicator lines, IND<6:0>, driven by Native Register<06:00> (Figure 1–7). These lines are connected to the on-board connector J1 (Table 2–4), to drive remote LED indicators for monitoring module status. The self-test program reports its status in bits Native Register<03:00>, Table 2–3. An error is indicated when the display remains set to a value for more time than is normally required to run the associated test.

Table 2–3: Self-test ROM Display Codes

Bits <03:00>	Value[1]	Description
0000	0	HALT switch on, CPU fault, power supply fault, or control has passed from ROM code to secondary boot
0001	1	Preliminary CPU testing—limited error messages
0010	2	Console SLU testing
0011	3	CPU testing
0100	4	On-board memory testing
0101	5	External memory testing
0110	6	Floating-point, LTC interrupt, SLU0 interrupt, and SLU1 interrupt testing
0111	7	not used
1000	8	not used
1001	9	not used
1010	A	not used
1011	B	not used
1100	C	not used
1101	D	Wrap mode in progress
1110	E	Boot in progress
1111	F	Console mode in progress

[1]Hexadecimal value

Table 2–4: J1 Pin Assignments

Signal Name	Pin	Function	To/From
	1	Signal Ground	
	2	Signal Ground	
	4	Signal Ground	
	33	Signal Ground	
	34	Signal Ground	
+5F H	17	+5 Vdc Fused	
+5F H	18	+5 Vdc Fused	
+12F H	27	+12 Vdc Fused	
-12 L[1]	8	-12 Vdc	
BHALT L	12	Q22-bus Processor Halt L	
BREAK H	23	Console Break Signal	DLART0
CBR0 H	26	Console Baud Rate 0	DLART0
CBR1 H	25	Console Baud Rate 1	DLART0
CBR2 H	24	Console Baud rate 2	DLART0
IND0 H	15	Indicator Bit 0	Native Reg
IND1 H	16	Indicator Bit 1	Native Reg
IND2 H	19	Indicator Bit 2	Native Reg
IND3 H	20	Indicator Bit 3	Native Reg
IND4 H	7	Indicator Bit 4	Native Reg
IND5 H	28	Indicator Bit 5	Native Reg
IND6 H	31	Indicator Bit 6	Native Reg
PBR0 H	9	Printer Baud Rate 0	DLART1
PBR1 H	10	Printer Baud Rate 1	DLART1
PBR2 H	11	Printer Baud Rate 2	DLART1
RECVDL0 H	29	Console Receive Data +	DLART0[2]
RECVDL0 L	30	Console Receive Data -	DLART0[2]
RECVDL1 H	6	Printer Receive Data +	DLART1[2]
RECVDL1 L	5	Printer Receive Data -	DLART1[2]

[1] Normally not connected.
[2] Indirectly

Table 2–4 (Cont.): J1 Pin Assignments

Signal Name	Pin	Function	To/From
STB0 H	13	Boot Switch Bit 0	Native Reg
STB1 H	14	Boot Switch Bit 1	Native Reg
STB2 H	21	Boot Switch Bit 2	Native Reg
STB3 H	22	Boot Switch Bit 3	Native Reg
TXMTDL0 L	32	Console Transmit Data +	DLART0[2]
TXMTDL1 L	3	Printer Transmit Data +	DLART1[2]

[2]Indirectly

2.3 MODULE EDGE CONNECTOR PINS

The KDJ11-D/S is a quad-height module, meaning it has four backplane edge-connectors: A, B, C, and D, corresponding to the backplane connector rows (Figure 2–6). The Q22-bus signals are carried in connector rows A and B. In certain configurations, backplane rows C and D form the *CD interconnect* for certain slots. The component side of the module is side 1 and the solder side is side 2. Each connector has 18 pins on each side of the module, labeled A through V (36 pins per connector). For example, pin BE2 is:

B—connector (row) B
E—pin E (fifth from top)
2—side 2 (solder side)

Table 2–5 lists the edge connector pin assignments for the KDJ11-D/S.

Table 2–5: Module Edge Connector Pin Assignments

Component Side		Solder Side	
Pin	Signal Name	Pin	Signal Name
AA1	BIRQ5 L[1,2]	AA2	+5V
AB1	BIRQ6 L[1,2]	AB2	-12V[1]
AC1	BDAL16 L	AC2	GND
AD1	BDAL17 L	AD2	+12V
AE1	SSPARE 1[1]	AE2	BDOUT L
AF1	SRUN H	AF2	BRPLY L
AH1		AH2	BDIN L
AJ1	GND	AJ2	BSYNC L
AK1	MSPARE A[1]	AK2	BWTBT L

[1]Not connected on the module
[2]Terminated but not used

Table 2–5 (Cont.): Module Edge Connector Pin Assignments

Component Side		Solder Side	
Pin	Signal Name	Pin	Signal Name
AL1	MSPARE A[1]	AL2	BIRQ4 L
AM1	GND	AM2	BIAKI L
AN1	BDMR L	AN2	BIAKO L
AP1	BHALT L	AP2	BBS7 L
AR1	BREF L	AR2	BDMGI L
AS1	+12B[1]	AS2	BDMGO L
AT1	GND	AT2	BINIT L
AU1	PSPARE 1	AU2	BDAL0 L
AV1	+5B[1]	AV2	BDAL1 L
BA1	BDCOK H	BA2	+5V
BB1	BPOK H	BB2	-12V[1]
BC1	BDAL18 L	BC2	GND
BD1	BDAL19 L	BD2	+12V[1]
BE1	BDAL20 L	BE2	BDAL2 L
BF1	BDAL21 L	BF2	BDAL3 L
BH1	SSPARE 8[1]	BH2	BDAL4 L
BJ1	GND	BJ2	BDAL5 L
BK1	MSPARE B[1]	BK2	BDAL6 L
BL1	MSPARE B[1]	BL2	BDAL7 L
BM1	GND	BM2	BDAL8 L
BN1	BSACK L	BN2	BDAL9 L
BP1	BIRQ7 L[1,2]	BP2	BDAL10 L
BR1	BEVNT L	BR2	BDAL11 L
BS1	PSPARE 4[1]	BS2	BDAL12 L
BT1	GND	BT2	BDAL13 L
BU1	PSPARE 2[1]	BU2	BDAL14 L
BV1	+5V	BV2	BDAL15 L
CA1		CA2	+5V

[1]Not connected on the module
[2]Terminated but not used

Table 2–5 (Cont.): Module Edge Connector Pin Assignments

Component Side		Solder Side	
Pin	Signal Name	Pin	Signal Name
CB1		CB2	
CC1		CC2	GND
CD1		CD2	
CE1		CE2	
CF1		CF2	
CH1		CH2	
CJ1		CJ2	
CK1		CK2	
CL1		CL2	
CM1		CM2	IAK L[3]
CN1		CN2	IAK L[3]
CP1		CP2	
CR1		CR2	DMG L[4]
CS1		CS2	DMG L[4]
CT1	GND	CT2	
CU1		CU2	
CV1		CV2	
DA1		DA2	+5V
DB1		DB2	
DC1		DC2	GND
DD1		DD2	
DE1		DE2	
DF1		DF2	
DH1		DH2	
DJ1		DJ2	
DK1		DK2	
DL1		DL2	
DM1		DM2	

[3]Soldered-in jumper W20 connects these pins

[4]Soldered-in jumper W21 connects these pins

Table 2–5 (Cont.): Module Edge Connector Pin Assignments

Component Side		Solder Side	
Pin	Signal Name	Pin	Signal Name
DN1		DN2	
DP1		DP2	
DR1		DR2	
DS1		DS2	
DT1	GND	DT2	
DU1		DU2	
DV1		DV2	

2.4 HARDWARE OPTIONS

The KDJ11-D/S module can be configured into an operating system using a variety of backplanes, power supplies, enclosures, and LSI-11 type modules.

2.4.1 Restricted LSI-11 Options

LSI-11 options not compatible or restricted for use with the KDJ11-D/S are listed in Table 2–6. Backplanes, memories, or I/O devices which do not use 22-bit addresses may generate or decode erroneous addresses if installed in systems that use 22-bit addresses.

Memory-addressing devices which use only 16- or 18-bit addresses can be installed in a 22-bit backplane; however, system memory size must be restricted to the address range of such devices (that is, 64 kbytes with 16-bit devices, and 256 kbytes with 18-bit devices).

Caution—16-bit and 18-bit Memories

Neither 16-bit nor 18-bit memories can be used with the KDJ11-D/S. Such memories would be "invisible" to the CPU. However, the address space for such memories would overlap on-board memory address space, causing DMA problems.

Devices using backplane pins BC1, BD1, BE1, BF1 or DC1, DD1, DE1, DF1 for signals other than BDAL<21:18> are not compatible with the Q22-bus. Such devices cannot be used without modification.

Note—18-bit DMA Devices

Eighteen-bit DMA devices have the potential to work in Q22-bus systems if I/O is buffered in the 18-bit address space.

Figure 2–6: Module Edge Connectors

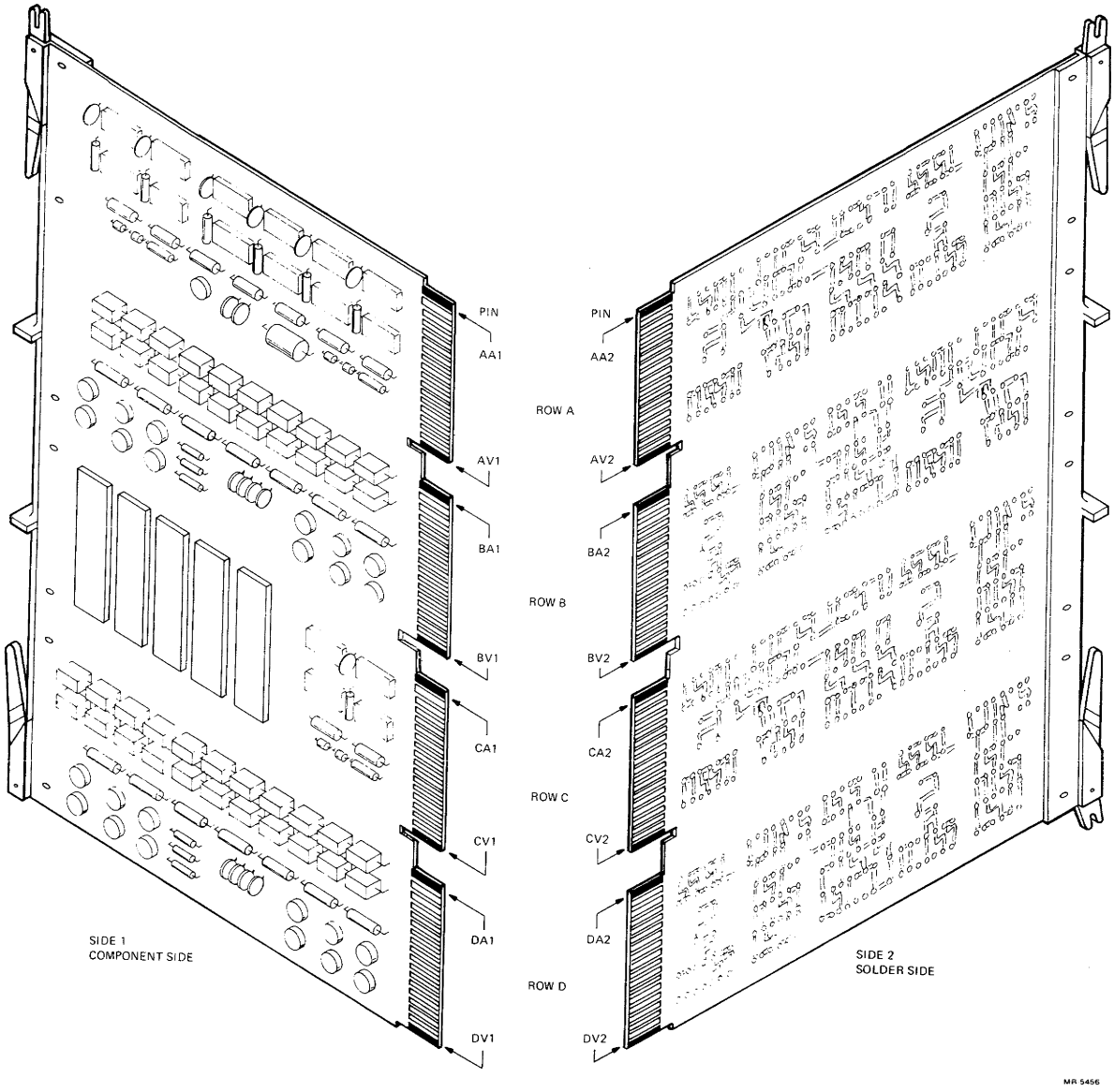


Table 2–6: Restricted or Non-compatible LSI-11 Options

Option	Module	Description
Backplanes (18-bit addressing only)		
DDV11-B		6 * 9 Backplane
H9270		4 X * Backplane
H9273-A		4 * 9 Backplane
VT103 BP		4 * 4 Backplane (54-14008)
Memory (16-bit addressing only)		
MMV11-A	G653	Core memory (Q22-bus required on C/D backplane connectors)
MRV11-AA	M7942	ROM
MRV11-BA	M8021	UV PROM-RAM
MSV11-B	M7944	MOS
Memory (18-bit addressing only)		
MRV11-C	M8048	PROM/ROM
MSV11-C	M7955	MOS
MSV11-D/E	M8044/M80	MOS
MXV11-A	M8047	Multifunction module (18-bit addressing only on memory; the memory can be disabled)
Options		
AAV11	A6001	D/A converter (BC1 not used for BDAL18)
ADV11	A012	A/D converter (BC1 not used for BDAL18)
BDV11	M8012	Bootstrap/terminator (CS Rev D or later for use with KDF11-A, or KDF11-B, EDD M8012-ML0002. CS Rev E or later for use in 22-bit systems, ECO M8012-ML005)
DLV11-J	M8043	Serial line interface (CS Rev E or later for use with KDF11-A, or KDF11-B, ECO M8043-M8002)
DRV11-B	M7950	DMA interface (18 -bit DMA only)
KPV11-B,-C	M8016-YB,-YC	Power-fail/LTC terminator (18-bit termination only)
KUV11	M8018	WCS (For use only with KD11-B and KD11-BA processors)
KWV11-A	M7952	Programmable real-time clock (BC1 not used for BDAL18)

Table 2–6 (Cont.): Restricted or Non-compatible LSI-11 Options

Option	Module	Description
REV11	M9400	Terminator, DMA refresh, bootstrap (Bootstrap for use only with KDF11-B and KD11-HA processors. Termination for 18 bits only. DMA refresh may be used in any system.)
RKV11-D	M72609	RK05 controller interface (16-bit DMA only)
RLV11	M8013 & M8014	RL01, 2 controller (18-bit DMA only. BC1 and BL1 not used for BDAL18 and BDAL19. Requires CD-interconnect on C/D connectors.)
RXV21	M8029	RX02 interface (18-bit DMA only)
TEV11	M9400-YB	Terminator (18 bits only)
VSV11	M7064	Graphics display (18-bit DMA only)
Bus Cable Cards		
M9400-YD		Cable connector (18-bit bus only)
M9400-YE		Connector with 240 Ohm terminators (18-bit bus only)
M9401		Cable connector (18-bit bus only)
Boot ROMs		
MXV11-A2		Boot ROMs
MXV11-B2		Boot ROMs

2.5 KDJ11-D/S INSTALLATION

2.5.1 Pre-installation Considerations

Before installing a KDJ11-D/S in an existing or new system, the following must be considered:

- The boot mechanism
- 18- or 22-bit addressing
- Bus loading
- Power requirements
- Single or multiple box system
- System differences (Section 2.5.2).

Note—Bus and Power Supply Loading

The ac and dc loading for the final configuration should be conform to the Q22-bus loading rules. Power supply ratings should not be exceeded.

2.5.2 System Differences

Note—Parity Errors

A parity error on Q22-bus data is asserted during the current instruction cycle. A parity error on local memory data is asserted during the next instruction cycle.

In addition to parity error assertion noted above, the following is a list of major KDJ11-D/S features, for comparison with other processors using the DCJ11 chip.

- On-board memory (0.5 Mbytes/1.5 Mbytes). (Optional, additional 3.5 Mbytes/2.5 Mbytes accessible over Q22-bus.)
- No cache memory.
- On-board bootstrap and self-test ROM (32-kbyte or 64-kbyte).
- Jumper or remote switch selectable bootstrap routine.
- Two on-board DL-type serial line interfaces (console and general purpose).
- Jumper or remote switch selectable serial line baud rate.
- Jumper selectable halt-on-BREAK option.
- Jumper selectable trap-on-HALT option.
- Mechanization of LTC<07> differs from other DCJ11-based processors.
- Supports only Power-up mode 2 (bootstrap on power-up).
- Q22-bus arbiter. Accepts only level 4 (BIRQ4) external interrupt requests.
- No on-board diagnostic LEDs (seven lines are provided to drive remote indicators).
- The FPA is not an option.

2.5.3 Installation Procedure

Caution—Memory Overlap

KDJ11-D/S on-board memory addresses are in the range 00000000 through 1777776. Other system memory devices *should not* be configured to overlap this range. Bootstrap does not check for such memory overlap; if it exists it may result in system disk corruption.

The following guidelines should be followed when installing or replacing a KDJ11-D/S.

1. Verify that the correct dc power is available before installing the module.
2. Be sure that dc power is not applied to the backplane when removing or inserting the module.
3. Verify the jumper configuration.
4. Install the KDJ11-D/S in backplane slot 1.

2.6 SPECIFICATIONS

Table 2–7 lists the physical, electrical, and environmental specifications for the KDJ11-D/S.

Table 2–7: KDJ11-D/S Specifications

Physical Specifications	
Module Number:	M7554
Module Type:	Q22-bus, quad-height
Dimensions:	Height: 26.56 cm (10.45 in) Length: 30.25 cm (11.91 in) Width: 1.27 cm (0.50 in)
Weight:	0.91 kg (32.0 oz)
Electrical Specifications	
Power (early version):	+5.0 Vdc / 3.15 A (typical) 3.47 A (maximum) +12.0 Vdc / 0.17 A (typical) 0.19 A (maximum)
Power (KDJ11-DA):	+5.0 Vdc / 2.50 A (typical) 2.80 A (maximum) +12.0 Vdc / 0.18 A (typical) 0.20 A (maximum)
Power (KDJ11-DB):	+5.0 Vdc / 2.90 A (typical) 3.20 A (maximum) +12.0 Vdc / 0.17 A (typical) 0.19 A (maximum)
Power (KDJ11-SD):	+5.0 Vdc / 3.20 A (typical) 3.50 A (maximum) +12.0 Vdc / 0.16 A (typical) 0.18 A (maximum)
Bus Loading:	ac: 3 unit loads dc: 1 unit load
Environmental Specifications	
Temperature	
Storage Range:	-40° to 66° C (-40° to 150.8° F)[1]
Operating Range:	5° to 60° C (41° to 140.0° F)[2]
Humidity	
Storage Range:	10% to 95%
Operating Range:	10% to 95% non-condensing
Altitude	
Storage:	The module is neither mechanically nor electrically damaged up to 4.9 km (16100 ft).
Operating:	The module can be operated up to 2.4 km (7900 ft).[2]
Airflow:	Provide adequate airflow to limit the inlet-to-outlet temperature rise across the module to 10° C (50° F).

[1]Before operating a module that is not at an operating range temperature, stabilize its temperature by placing the module in an operating range environment for at least five minutes.

[2]De-rate maximum operating temperature 1.82° C/km (1.0° F/3300 ft) above sea level.

Chapter 3
FUNCTIONAL DESCRIPTION

This chapter describes the functional areas of the KDJ11-D/S at the block diagram level. All the block diagrams apply to both the early versions and the re-engineered versions of the modules. The dashed lines show which functional units are contained within a gate array for the re-engineered versions.

3.1 INTRODUCTION

The KDJ11-D/S CPU module is a quad-height, single-board computer for use in Q22-bus systems. Based on the DCJ11 microprocessor chip, it executes the PDP11/73 instruction set and includes the CPU, memory management, local memory, and I/O. The module does not support cache nor an FPA (Floating-point Accelerator). Figures 3–1 and 3–2 are block diagrams of the module showing the major functional areas and interconnections, for the early and re-engineered versions. Note that most of the functional areas are connected to one or more of five major sets of signals:

- BUS STATE—signals to/from the Bus State-machine which controls Q22-bus cycle timing.
- BUS LOGIC—signals to/from Q22-bus combinational (not including BDAL<21:00> logic).
- DAL<21:0>—22 data and address lines between the major functional areas.
- LA<21:0>—22 latched address lines (latched from DAL<21:0>).
- CPU—signals to/from the DCJ11 and its associated logic.

For earlier versions of the KDJ11-D/S (see Figure 3–1), most of the module's functions are implemented with large-scale integration, including PALs (Programmable Array Logic), PLAs (Programmable Logic Array), and PLSs (Programmable Logic Sequencer). The re-engineered version of the KDJ11-D/S (see Figure 3–1) uses very large-scale integration to implement the module's functions, in the form of two gate arrays. These gate arrays are the DC7063 for control functions, and the DC7064 for the data paths.

Figure 3-1: KDJ11-DA Block Diagram

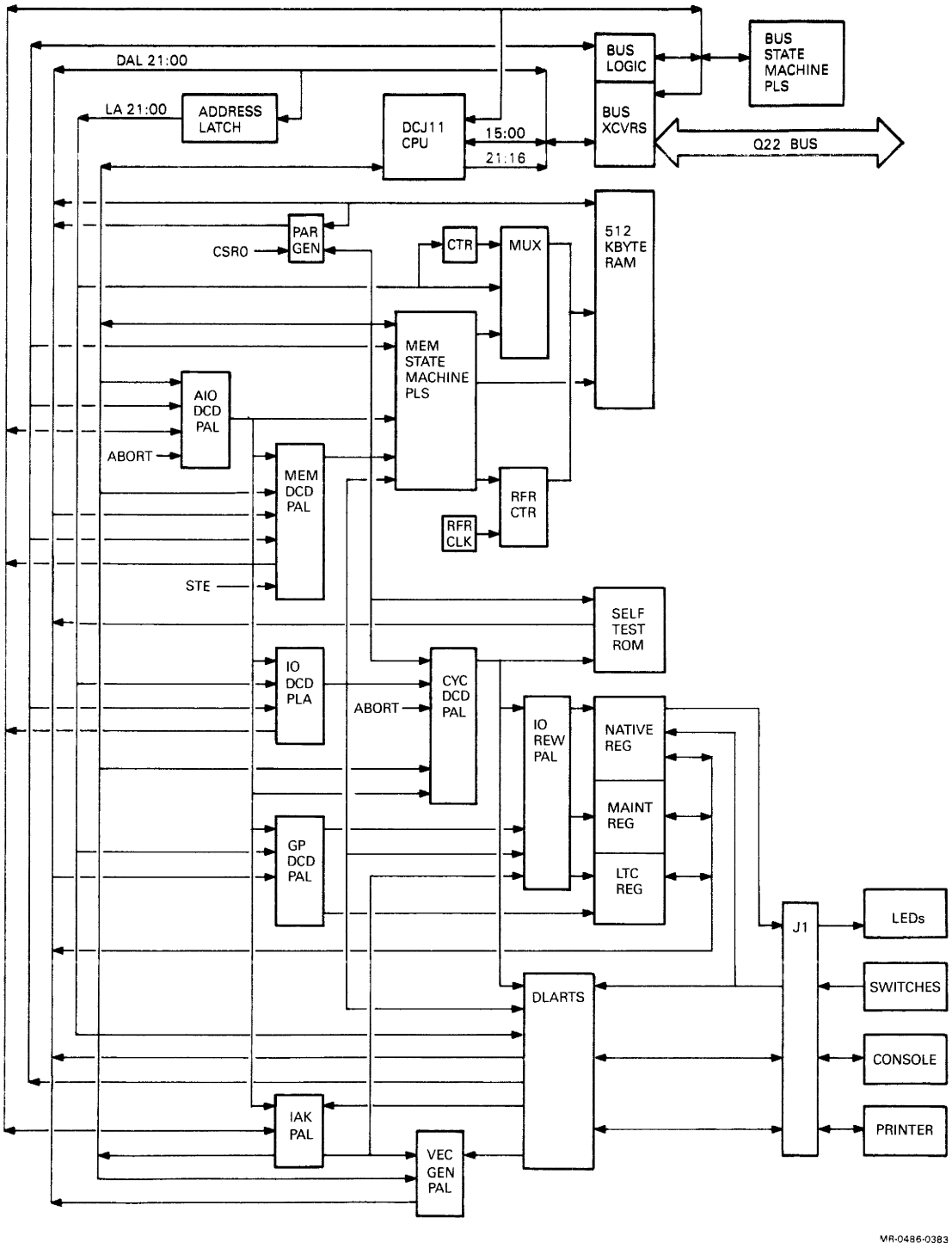
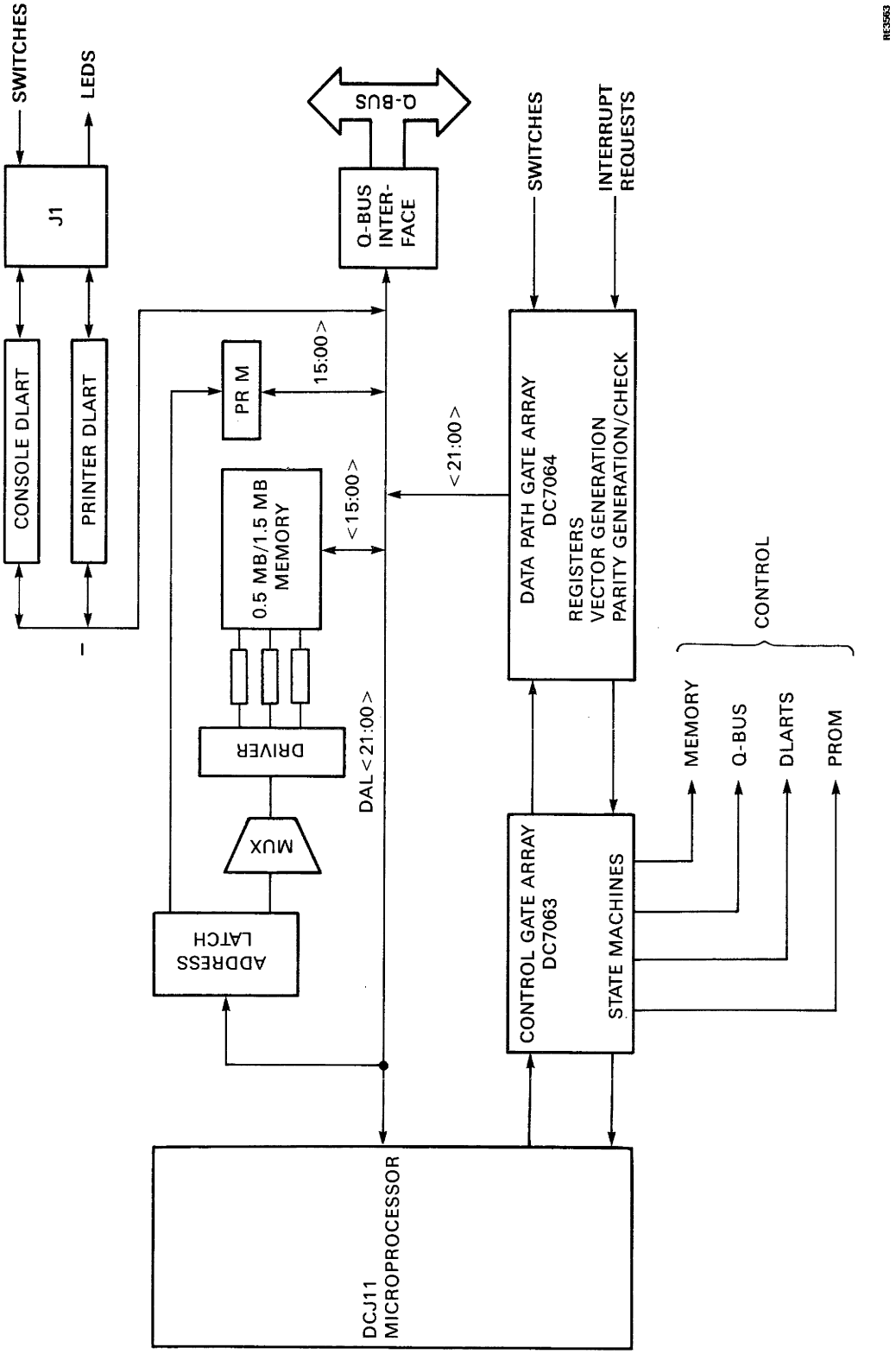


Figure 3-2: KDJ11-D/S Block Diagram



3.2 DCJ11 MICROPROCESSOR

Note—DCJ11 Information

For DCJ11 features not implemented in the KDJ11-D/S, and more detail on features described below, see the *DCJ11 MICROPROCESSOR USER'S GUIDE, EK-DCJ11-UG*. The following subsections are specific to the KDJ11-D/S.

The DCJ11 executes the PDP11/73 instruction set (see the Associated Documents listed in the Preface), and controls the rest of the module. It initiates all KDJ11-D/S data transfers and operations. The DCJ11 is a 60-pin VLSI chip. The input/output signals are shown in Figure 3–3 and described in Table 3–1.

Table 3–1: DCJ11 Input and Output Signals

Signal	Description
Input Signals	
RHALT	Receive Halt—Asserted by either the console BREAK line (if jumper W11 is not installed) or Q22-bus BHALT. HALT is the lowest priority non-maskable interrupt except during vector read cycles when it is the highest priority.
LTC	Line Time Clock—(Event) Asserted by Q22-bus BEVNT (via the LTC Register). When enabled and asserted, this priority level 6 maskable-interrupt causes the DCJ11 to trap through vector address 100.
PE	Parity Error—Asserted when a parity error is detected in local memory or Q22-bus data. PE also asserts ABORT (described below). When both PE and ABORT are asserted, a parity error abort is generated and the DCJ11 traps through vector address 114, without completing the current instruction. PE is sampled only during the stretched portion of a cycle. PE is asserted as follows: Q22-bus data error—during the CURRENT instruction cycle. Local memory data error—during the NEXT instruction cycle.
PWRF	Power Fail—Asserted when Q22-bus BPOK is deasserted, indicating an ac power failure. This non-maskable high priority interrupt causes a trap through vector address 24.
IRQ0	Interrupt Request 0—Asserted by Q22-bus BIRQ4 or a transmit or receive interrupt request from either DLART. All such interrupts are level 4 priority interrupts.
DV	Data Valid—Generated by the Bus State Machine and the Memory State Machine. Latches DAL data into the DCJ11.
DCOK	DC power OK (INIT)—Asserted by Q22-bus BDCOK to initialize the DCJ11. DCOK forces the DCJ11 to execute the power-up sequence.
RDMR	Receive DMA Request—Asserted by Q22-bus BDMR (and BPOK). The DCJ11 samples RDMR at the start of all cycles. If the cycle does not include a write operation, the DCJ11: Stretches the cycle. Places DAL<15:0> in the high-impedance state.

Table 3–1 (Cont.): DCJ11 Input and Output Signals

Signal	Description
	Asserts MAP during the second part of the cycle, to acknowledge the request.
	If the cycle includes a write operation, the DCJ11 stretches the cycle, but does not place DAL<15:0> in the high-impedance state, and does not assert MAP.
CONT	Continue—Generated by the Bus State Machine and the Memory State Machine. CONT is asserted by either of the state machines when it has finished using the DALs. CONT terminates a stretched cycle (see SCTL, below), enabling the DCJ11 to continue on to its next cycle.
XTAL<1:0>	Crystal—Input connections for the external 15.206 MHz (or 18 MHz) crystal oscillator.
Input/Output Signals	
ABORT	Asserted either internally by the DCJ11, during the first part of a DCJ11 I/O cycle; or externally, during the stretched portion of a DCJ11 non-I/O cycle (see SCTL, below). When ABORT is asserted: Internally because a memory management error occurred, the DCJ11 traps through vector address 250. Internally because an address error occurred, the DCJ11 traps through vector address 4. Externally by PE (above), the DCJ11 traps through vector address 114 (parity error abort). Externally by NXM (Non-existent Memory), the DCJ11 traps through vector address 4.
DAL<15:0>	Data/Address Lines—These time-multiplexed I/O lines are the 16-bit DCJ11 data and address bus. During the first part of a DCJ11 I/O transfer cycle, the type of cycle determines what is on DAL<15:0>: Bus Read or Bus Write—16 bits of physical address <15:0>. GP Read or Write—8-bit GP code <7:0>. Interrupt Acknowledge—priority level <3:0>. During the second part of an I/O transfer cycle, DAL<15:0> carry 8 or 16 bits of data. On read cycles, external logic asserts the lines. To read a byte, the DCJ11 reads the full word, but ignores the unwanted byte. On write cycles, 16 lines are asserted to write a word, and 8 lines are asserted to write a byte.

Table 3-1 (Cont.): DCJ11 Input and Output Signals

Signal	Description
Output Signals	
JDAL<21:16>	Data/Address Lines—These six time-multiplexed output lines are the six most significant bits of a 22-bit physical address. The address is valid at the start of every DCJ11 bus cycle. During the second part of the cycle, internal status is asserted on JDAL<21:16> (for manufacturing test).
ALE	Address Latch Enable—Asserted to indicate that MAP, DAL<21:0>, AIO<3:0>, and BS<1:0> are all valid. Used by external logic as a latch enable.
STRB	Strobe—Asserted one clock period after ALE is asserted. The deassertion of STRB is the end of one microcycle and the start of another.
MAP	Map Enable—Time-multiplexed. Asserted during the first part of a cycle to set MMR3<5> and enable the external I/O map (not used in the KDJ11-D/S). Asserted during the second part of a cycle to acknowledge assertion of input RDMR.
SRUN	(Predecode) Asserted to indicate that the contents of the DCJ11 PB (Prefetch Buffer) are valid and being decoded. In the KDJ11-D/S it is wired to backplane pin AF1, to drive a remote RUN-indicator.
AIO<3:0>	Address Input/Output—The code asserted on these lines identifies the type of the current DCJ11 cycle. See Table 3–2.
BS<1:0>	Bank Select—Time-multiplexed. During the first part of a DCJ11 Bus Read or Bus Write cycle, BS<1:0> identify the type of device being physically addressed on DAL<21:0>. See Table 3–3. During the second part of a DCJ11 I/O cycle, BS<1:0> carry cache access information (not used by the KDJ11-D/S).
CLK2	Clock 2—The system clock for external logic. CLK2 has the same frequency as the oscillator connected to XTAL<1:0> inputs, and the same state of the DCJ11 internal clock.
SCTL	Stretch Control—Asserted during the stretched portion of a DCJ11 cycle. Used to: Latch data with the leading or trailing edge of SCTL during write cycles. Enable externally generated ABORTs. SCTL is deasserted when input CONT (above) is asserted.
BFCTL	Buffer Control—Asserted when the DCJ11 is not driving the DAL lines. This occurs during the portion of a read cycle when data is on the DAL lines and during the stretched portion of any non-write cycle. Not asserted when the DCJ11 is driving the DAL lines.

Figure 3-3: DCJ11 CPU

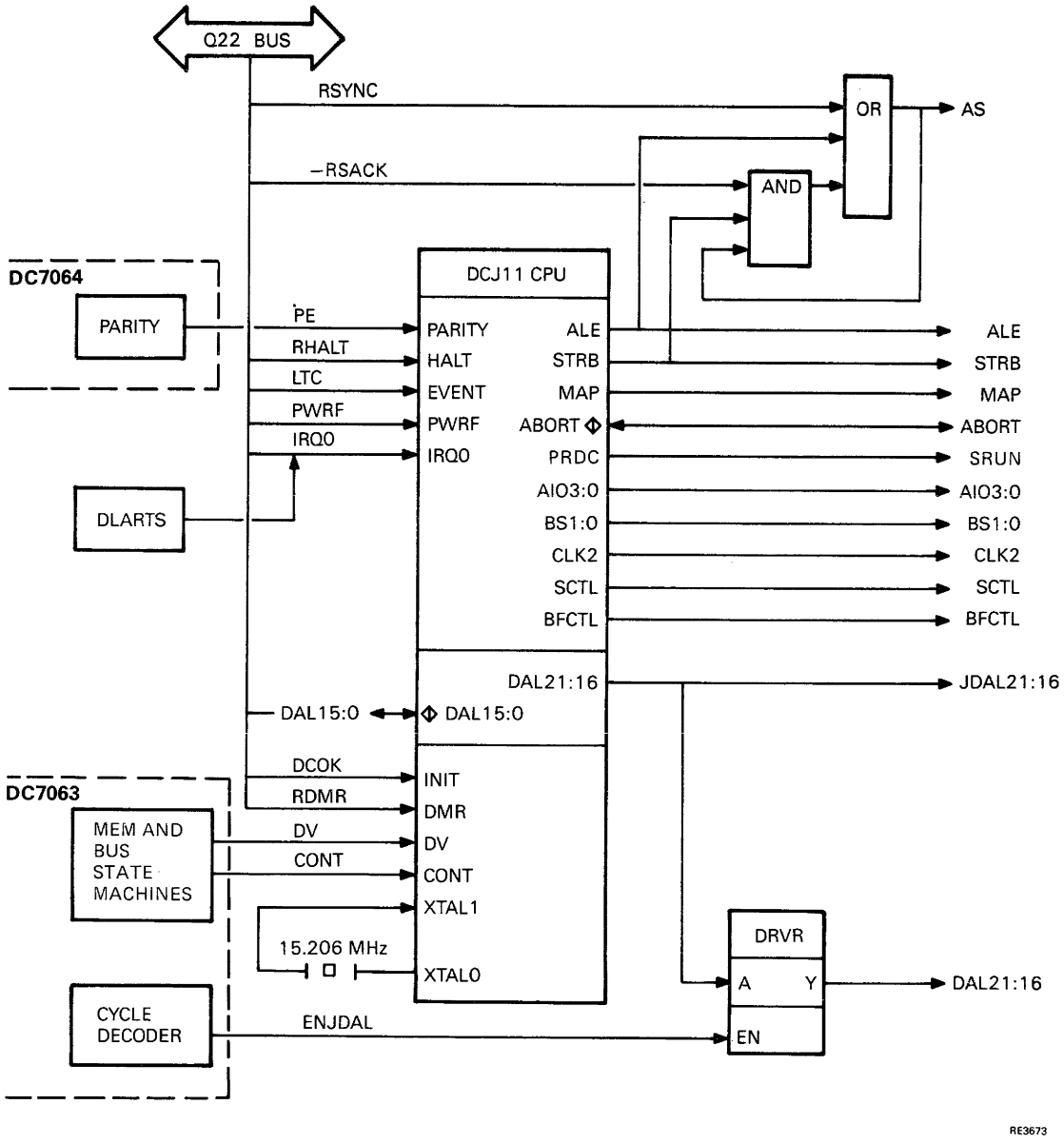


Table 3–2: AIO<3:0> Codes

AI0	Cycle	Cycle
1 1 1 1	NIO	Non-I/O, DCJ11 internal operation
1 1 1 0	GP Read	GP Read
1 1 0 1	IAK	Interrupt acknowledge, vector-read
1 1 0 0	Bus Read	Instruction stream request-read
1 0 1 1	Bus Read	RMW (read/modify/write), no bus lock
1 0 1 0	Bus Read	RMW, bus lock
1 0 0 1	Bus Read	Data stream read
1 0 0 0	Bus Read	Instruction stream demand-read
0 1 0 1	GP Write	GP word-write
0 0 1 1	Bus Write	Bus byte-write
0 0 0 1	Bus Write	Bus word-write

Table 3–3: BS<1:0> Codes

BS	
1 0	Device
1 1	Internal Register—DCJ11 internal, memory-addressable registers, including the PSW, PIRQ, CPU Error, MMU, and Hit/Miss[1] registers.
1 0	External I/O Device—Any device or register external to the DCJ11 and resident in 17760000 through 17777777 (the I/O Page), but excluding internal and system registers (BS<0> = 1).[2]
0 1	System Registers—Memory-addressable registers in the range 17777740 through 17777750.[2]
0 0	Memory—Any location in the physical address space in the range 00000000 through 17757777 (below the I/O Page).

[1]The Hit/Miss register is not used in the KDJ11-D/S.

[2]The Maintenance Register (17777750) is accessible as both an External I/O Device and a System Register.

3.2.1 DCJ11 Cycles

The DCJ11 executes six basic types of cycles (see Table 3–2). These cycles (also called "microcycles" and "bus cycles") are classed according to the type of activity on DAL<16:0> (also called the "I/O Bus"). Each microcycle is associated with the execution of one DCJ11 microinstruction. The execution of one DCJ11 macroinstruction (that is, PDP11 instruction) can require several microcycles. The basic DCJ11 cycles are:

- NIO (non-I/O)
- Bus Read
- Bus Write
- GP Read
- GP Write
- Interrupt Acknowledge

A cycle starts (and ends) when STRB is deasserted. All cycles require at least four clock periods; however, cycles can be extended, or stretched, by an internal event or external logic. When stretched, a cycle is extended for at least four clock periods; beyond that, a cycle can be stretched indefinitely in increments of two clock periods. SCTL is asserted at the start of the stretched portion of the cycle. A stretched cycle is ended when CONT is asserted.

A cycle will *not* be stretched if it is an NIO cycle, and RDMR is not asserted during the cycle.

The "first" or "early" part of a cycle is defined as the first two clock periods of the cycle. The "second" or "late" part of a cycle is the remaining clock periods—two in a normal cycle, six or more in a stretched cycle.

The DCJ11 asserts a code for the current cycle on AIO<3:0> (see Table 3–2). The AIO Decoder (Section 3.3) decodes AIO<3:0>, and generates signals to control other module functions. The DCJ11 cycles are described in the following sections.

3.2.1.1 NIO

During an NIO (non-I/O) cycle the DCJ11 is performing an internal operation, and is not using DAL<21:0>. The cycle is stretched if RDMR is asserted.

3.2.1.2 Bus Read

These cycles include:

- Instruction stream request-read or demand-read.

  The DCJ11 executes a *request-read* to prefetch information. All other types of reads are *demand-reads*. If external logic generates an abort during a request-read, macroinstruction flow is not affected (the abort is ignored). Aborts generated during a demand-read or by on-board logic during a request-read are recognized and serviced through the vectors listed in Table 1–15.

- Read/modify/write (read portion).

  An AIO read code is asserted during the first part of the cycle. The second part of the cycle is a Bus Write cycle with a different AIO code asserted.

- Data stream read.

The DCJ11 asserts BS<1:0> to identify the type of device being read as either an I/O device, memory, or a memory-addressable register. The DCJ11 always reads a full word, and ignores the unwanted byte if necessary. The cycle will be stretched for any of the following conditions.

- Anything other than a memory reference (BS<1:0> not = 00).
- DMA grant (MAP is asserted during the second part of the cycle).

- ABORT is asserted during an instruction stream demand-read, data stream read, or read-modify-write cycle.

The DCJ11 latches the read data at the start of the third clock period or when DV (Data Valid) is asserted during the stretched part of the cycle.

3.2.1.3 Bus Write

These are either word-write or byte-write cycles. The DCJ11 asserts BS<1:0> to identify the type of device being written as either an I/O device, memory, or a memory-addressable register. The DCJ11 drives all 16 bits of DAL<15:0> for byte-write cycles. The address specifies which byte has valid data; data in the other byte is undefined. All Bus Write cycles are stretched. The write data is valid when SCTL is asserted.

3.2.1.4 GP Read

The DCJ11 executes this cycle to read the low byte of the Maintenance Register; that is, power-up mode, HALT/trap option, POK (Power OK), and module type number. The DCJ11 asserts a GP code on DAL<7:0> (see Table 3–4). This code is decoded by the GP DECODER (Section 3.4). The DCJ11 reads a full word; however, the high byte is driven to zero by external logic, and ignored by the DCJ11. The GP Read cycle is always stretched. The DCJ11 latches the data at the start of the third clock period or when DV (Data Valid) is asserted during the stretched part of the cycle.

Table 3–4: GP Codes

Code[1]	Type	Function
000	READ	Power-up mode, HALT/trap option, POK
002	READ	Power-up mode, HALT/trap option, POK
214	WRITE	Negate INIT
014	WRITE	Assert INIT
100	WRITE	Acknowledge BEVNT
140	WRITE	Acknowledge PWRF

[1]Unlisted codes are not implemented in the KDJ11-D/S

3.2.1.5 GP Write

The DCJ11 executes this cycle to:

- Clear the Event interrupt enable flip-flop.
- Assert or negate Q22-bus BINIT.
- Clear the Event interrupt request flip-flop.
- Clear the power-fail flip-flop.

The GP code asserted on DAL<7:0> (Table 3–4) during the first part of the cycle is decoded by the GP Decoder (Section 3.4). The cycle is always stretched. The GP Write data can be either a word or byte, and is asserted in the stretched part of the cycle. The write data is valid when SCTL is asserted.

3.2.1.6 IAK

This cycle is also called a "vector read cycle," and is executed to service an interrupt request on IRQ0. The DCJ11 asserts the bit-encoded acknowledged level on DAL<3:0> in the first part of the cycle (for IRQ0, DAL<3:0> = 0001). The cycle is always stretched. The interrupting device asserts the interrupt vector on DAL<15:0>, during the stretched part of the cycle. The DCJ11 latches the vector at the start of the third clock period or when DV (Data Valid) is asserted during the stretched part of the cycle. External logic can abort the cycle during the stretched part of the cycle. If ABORT is asserted, the DCJ11 ignores the interrupt request and continues processing.

3.3 AIO DECODER

Figure 3–4 shows the AIO Decoder. Its inputs are:

1. AIO<3:0> from the DCJ11, to determine the current DCJ11 cycle (see Table 3–2).
2. RSACK. A Q22-bus DMA device asserts BSACK to acknowledge bus mastership when it has received a DMA grant (BDMGO) from the KDJ11-D/S.
3. RBS7. A bus master asserts Bank 7 Select (BBS7) along with DAL<12:0> to reference a location in the I/O Page.

Figure 3–4: AIO Decoder

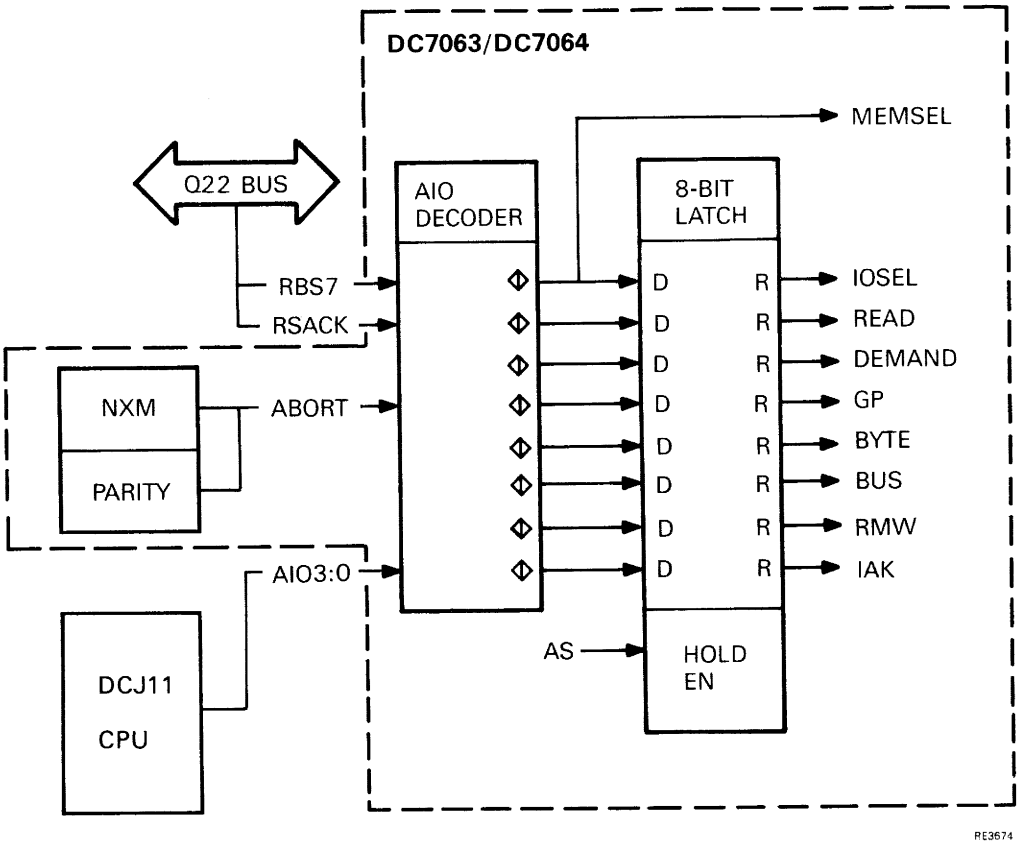


Table 3–5 describes the conditions for asserting the AIO Decoder outputs. The AIO Decoder outputs are latched by AS (Address Strobe—see Figure 3–3).

NOTE

The AIO decode logic is implemented in both gate arrays.

Table 3–5: AIO Decoder Outputs

Output	Conditions
MEMSEL	Memory Select—Asserted for all Bus Read and Bus Write cycles. MEMSEL is a direct input to the Memory Decoder, and a latched input (IOSEL) to the I/O Decoder. MEMSEL can only be asserted if RSACK is not asserted. MEMSEL is not asserted: For NIO, IAK, GP Read, or GP Write cycles. If ABORT is asserted. If both RSACK and RBS7 are asserted (that is, a Q22-bus device is attempting a DMA reference to the I/O Page). **Note—Q22-bus References to On-board I/O** All Q22-bus device references to on-board I/O devices are illegal.
READ	Asserted for all Bus, IAK (vector), and GP read cycles. Note that the complement of READ (that is, -READ) can be interpreted as a write cycle signal (see Table 3–9).
DEMAND	Asserted for all cycles except NIO, GP Read, and Instruction Stream Request-read.
GP	General Purpose—Asserted for GP Read and GP Write cycles.
BYTE	Asserted only for a Bus Byte Write cycle.
BUS	Asserted for all bus read and write cycles.
RMW	Read/Modify/Write—Asserted for both RMW cycles.
IAK	Interrupt Acknowledge (vector read)—Asserted only for an IAK cycle.

3.4 GP DECODER

Figure 3–5 shows the General Purpose Decoder. It decodes LA<7:0>; that is, the latched DAL<7:0> value, asserted in the first part of a GP cycle (Table 3–4). The GP Decoder outputs are described in Table 3–6. Note that signals INIT H and INIT L are both outputs and inputs (feedback), and are complements of each other.

Figure 3-5: GP Decoder

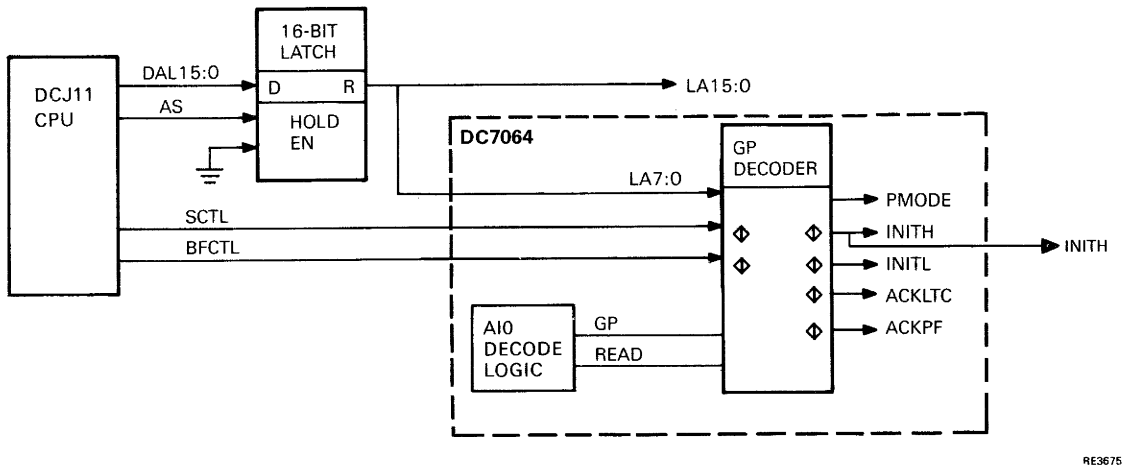


Table 3-6: GP Decoder Outputs

Output	Conditions
PMODE	(GP code 000 or 002) Power-up Mode—Asserted during the stretched part of a GP Read cycle. It is one of the inputs which setsup the I/O Read/Write logic (Section 3.8) to read Maintenance Register<07:00> (Module Type, Halt/Trap Option, Power-up Mode, and POK—see Table 1-6).
INIT	(GP code 214)—Asserted during the stretched part of a GP Write cycle. It asserts Q22-bus BINIT, clears certain DLART register bits (see Section 1.2.7), and is part of the clear input to the DMG (DMA Grant) flip-flop. Cleared by GP code 014.
INIT	(GP code 014)—Asserted during the stretched part of a GP Write cycle. It clears Memory Error Register<15:14> and <02:00> (Table 1-24) and the Event Interrupt Enable flip-flop. Cleared by GP code 214.
ACKLTC	(GP code 100) Acknowledge LTC—Asserted during the stretched part of a GP Write cycle. It clears the Event Interrupt Request flip-flop. Also asserted by INIT L.
ACKPF	(GP code 140) Acknowledge Power-fail—Asserted during the stretched part of a GP Write cycle. It clears the PWRF (Power-fail) flip-flop. Also asserted by INIT L.

3.5 MEMORY DECODER

Figure 3-6 shows the Memory Decoder. It decodes BS<1:0> (Table 3-3) and DAL<21:9> to generate three outputs which are part of the logic to select either on-board ROM, on-board RAM, or Q22-bus memory (Table 3-7). All the Memory Decoder outputs are latched by AS (Address Strobe—see Figure 3-3).

Figure 3–6: Memory Decoder

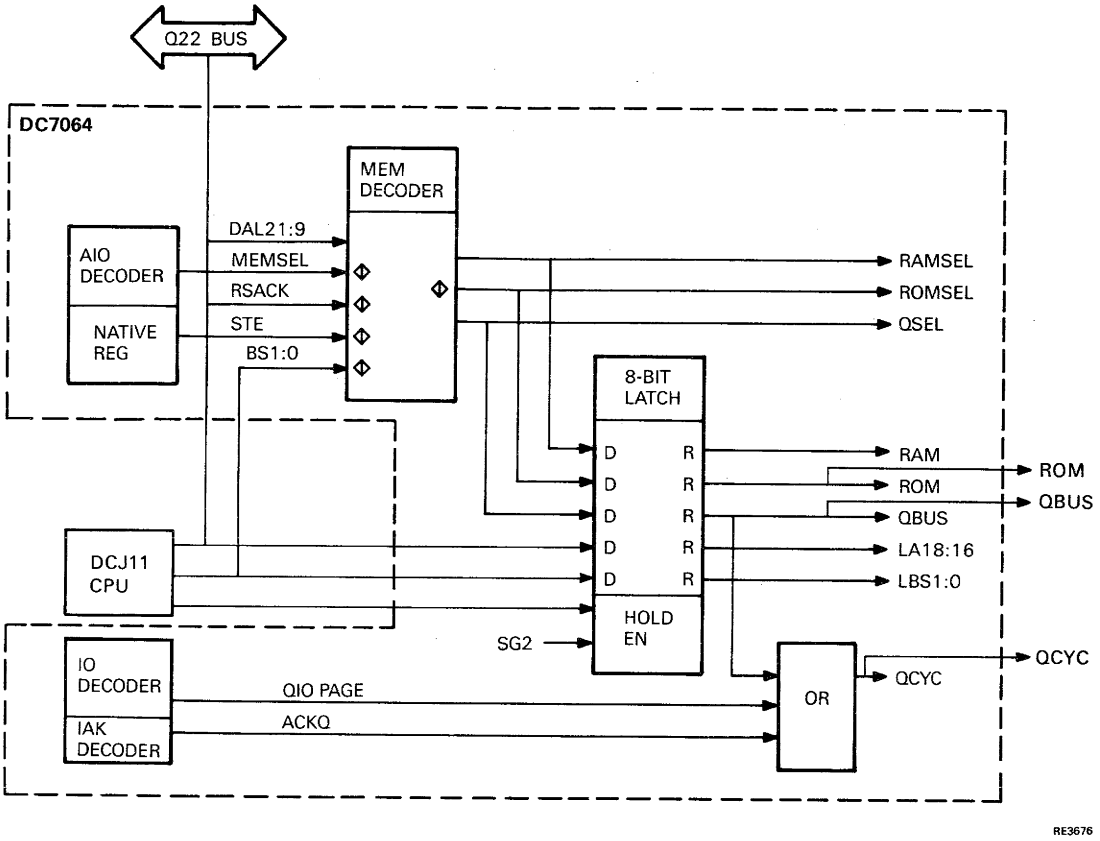


Table 3–7: Memory Decoder Outputs

Output	Conditions
ROMSEL	ROM Select—Asserted when:
	The bootstrap is referenced:
	BS<1:0> = 10 and DAL<12:9> = 13 and MEMSEL is asserted, and RSACK is not asserted.
	Self-test ROM is referenced:
	DAL<21:17> = the range 17400000 through 17777777 and BS<1:0> = 00 and Native Register<07> = 1 (STE) and MEMSEL is asserted, and RSACK is not asserted.

Table 3–7 (Cont.): Memory Decoder Outputs

Output	Conditions
RAMSEL	RAM Select—Asserted when On-board RAM is referenced: BS<1:0> = 00 and DAL<21:19> = the range 00000000 through 01777777 (00000000 through 5777777 if 1.5 Mb of RAM) and MEMSEL is asserted, and RSACK is not asserted. or DAL<21:19> = the range 00000000 through 01777777 (00000000 through 5777777 if 1.5 Mb of RAM) and both MEMSEL and RSACK are asserted.
QSEL	Q22-bus Select—Asserted when: MEMSEL is asserted, and RSACK is not asserted and BS<1:0> = 00 and Q22-bus memory is referenced: DAL<21:19> = the range 02000000 through 17377777 (0.5 Mbyte modules) DAL<21:19> = the range 06000000 through 17377777 (1.5 Mbyte modules) or DAL<21:17> = the range 17400000 through 17777777 and Native Register<07> = 0 (not STE). QSEL is not asserted for the same input conditions that assert ROMSEL and RAMSEL. In addition, QSEL is not asserted anytime that: RSACK is asserted or MEMSEL is not asserted or BS<1:0> = 01 or 10 or 11.

3.6 I/O DECODER

The I/O Decoder, Figure 3–7, is an 82S100 PLA. It performs the initial selection to access a specific External I/O Register, or System Register (Maintenance Register). It also prevents Q22-bus devices from accessing these registers, the bootstrap, and internal registers by inhibiting QIOPAGE (see Table 3–8).

Figure 3–7: I/O Decoder

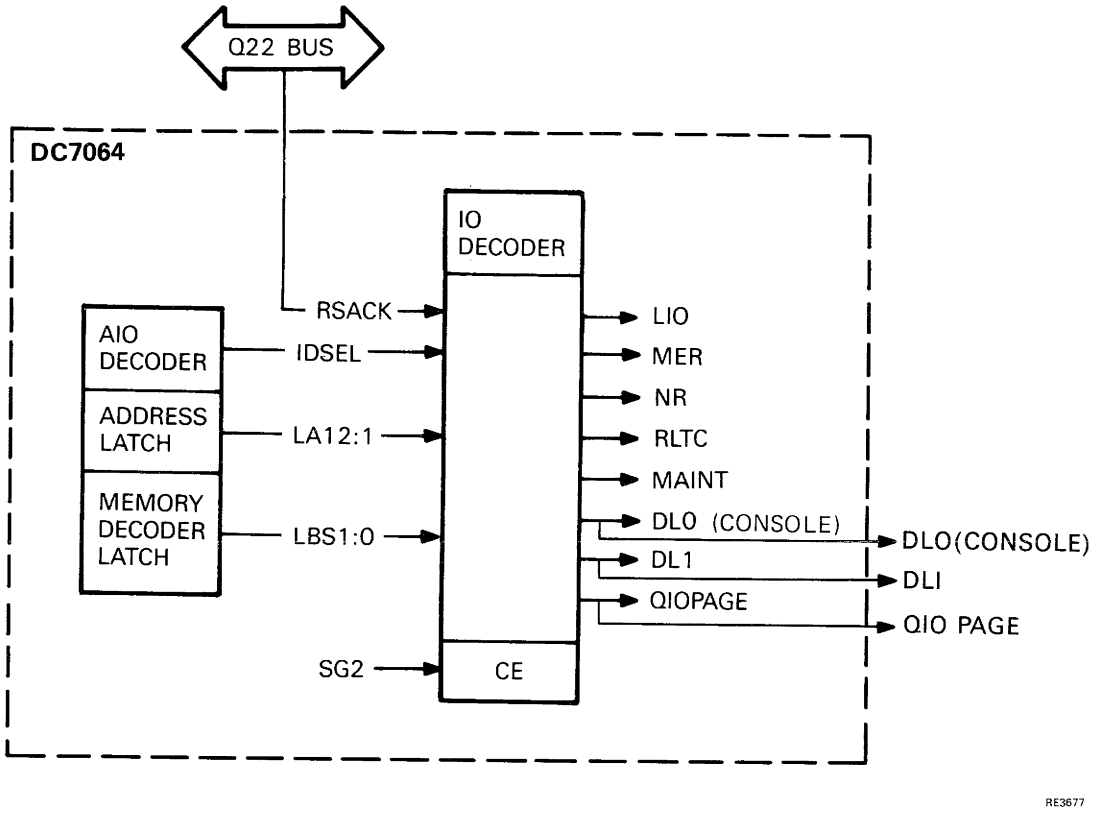


Table 3–8: I/O Decoder Outputs

Output	Conditions
LIO	Local I/O—Asserted when: IOSEL is asserted and RSACK is deasserted, and either: BS<1:0> = 11 or Any of the following outputs are asserted: MER, NR, RLTC, MAINT, DL1, DL0
MER	Memory Error Register select—Asserted when IOSEL is asserted and RSACK is not asserted, and: BS<1:0> = 10. DAL<12:1> = 12100.
NR	Native Register select—Asserted when IOSEL is asserted and RSACK is not asserted, and:

Table 3–8 (Cont.): I/O Decoder Outputs

Output	Conditions
	BS<1:0> = 10. DAL<12:1> = 17520.
RLTC	Read LTC select—Asserted when IOSEL is asserted and RSACK is not asserted, and: BS<1:0> = 10. DAL<12:1> = 17546. Note that this signal is used as enable for both reads and writes (see the I/O Read/Write Decoder, Table 3–10).
MAINT	Maintenance Register select—Asserted when IOSEL is asserted and RSACK is not asserted, and: BS<1:0> = 01. DAL<12:1> = 17750.
DL1	DLART 1 select—Asserted when IOSEL is asserted and RSACK is not asserted, and: BS<1:0> = 10. DAL<12:3> = 1650. DAL<2:1> select the specific DLART register.
DL0	DLART 0 select—Asserted when IOSEL is asserted and RSACK is not asserted, and: BS<1:0> = 10. DAL<12:3> = 1756. DAL<2:1> select the specific DLART register.
QIOPAGE	Q22-bus I/O Page reference—Deasserted when any of the above are asserted. That is, a Q22-bus device cannot reference any of the registers resident in the I/O Page. QIOPAGE is also not asserted when any of the following conditions exist: RSACK is asserted. IOSEL is deasserted. BS<1:0> = 11 or 00. BS<1:0> = 10 and DAL<12:9> = 13. QIOPAGE is asserted when IOSEL is asserted and RSACK is not asserted, and: BS<1:0> = 10. DAL<12:9> = none of the above addresses.

3.7 CYCLE DECODER

Figure 3–8 shows the Cycle Decoder. Three of its outputs define the type of operation performed on the devices selected by the I/O Decoder. In addition, the CYCLE output is used in the Memory State Machine as part of the logic to exit its Idle loop and enter one of its major sequences. The Cycle Decoder also provides the output enable for DAL<21:16> (latched JDAL<21:16> from the DCJ11).

Figure 3–8: Cycle Decoder

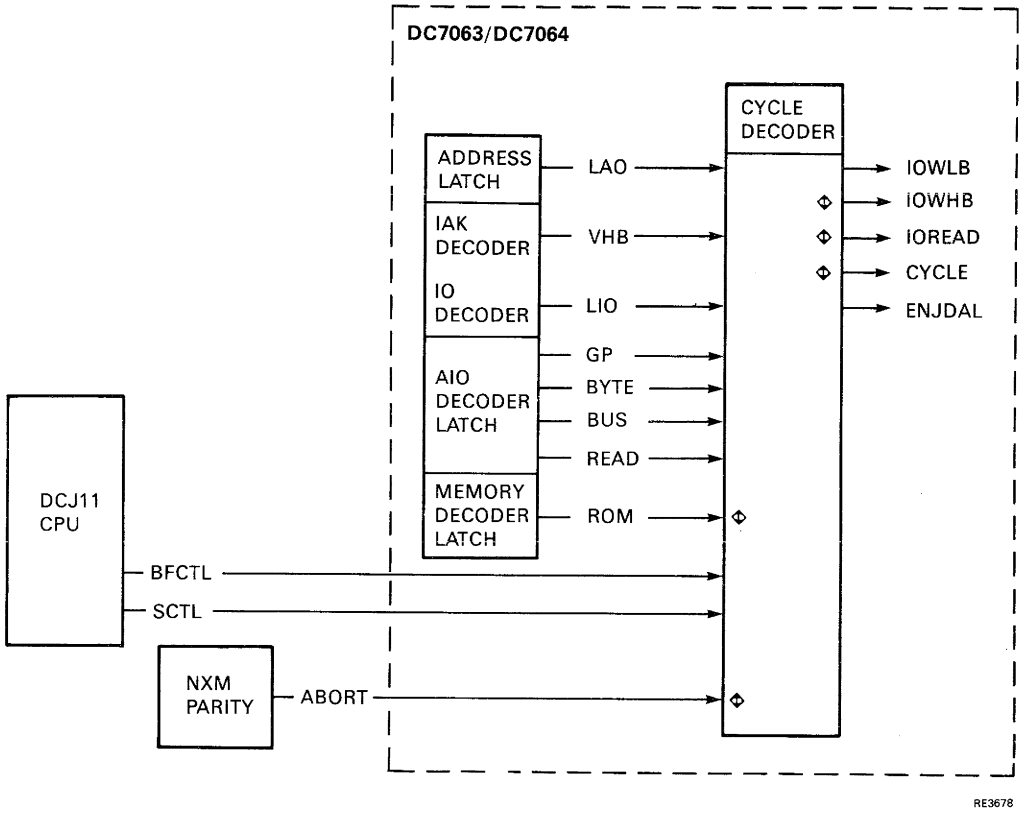


Table 3–9: Cycle Decoder Outputs

Output	Conditions
IOWLB	I/O Write Low Byte—Asserted when SCTL and LIO are asserted, and either: BYTE is asserted (byte only), and READ and LA0 are not asserted (write low byte). or READ and BYTE are not asserted (write word).
IOWHB	I/O Write High Byte—Asserted when SCTL and LIO are asserted, and either: BYTE and LA0 and are asserted (high byte only), and READ is not asserted (write). or BYTE and READ are not asserted (write word).

Table 3–9 (Cont.): Cycle Decoder Outputs

Output	Conditions
IOREAD	Asserted to read a byte or word when SCTL and BFCTL are asserted, and either: LIO is asserted (register read). or ROM is asserted (ROM read).
CYCLE	Asserted for all the above conditions. That is, any time that LIO or ROM is asserted to access External I/O Registers, Internal Registers, System Registers, or ROM. Also asserted when: VHB is asserted (see IAK Decoder, Table 3–11). GP is asserted (GP read or write—see GP Decoder, Table 3–6). ABORT is asserted.
ENJADD	Asserted when both SCTL and BFCTL are not asserted (includes the IOWLB, IOWHB, and IOREAD conditions above).

3.8 I/O READ/WRITE LOGIC

Figure 3–9 shows the I/O Read/Write logic. All but one of its outputs are enables to read or write either the Maintenance, LTC, Memory Error, or Native register. Its ZHB output grounds the inputs to DAL<15:8> when the Maintenance or LTC registers or an interrupt vector is being read.

Figure 3–9: I/O Read/Write logic

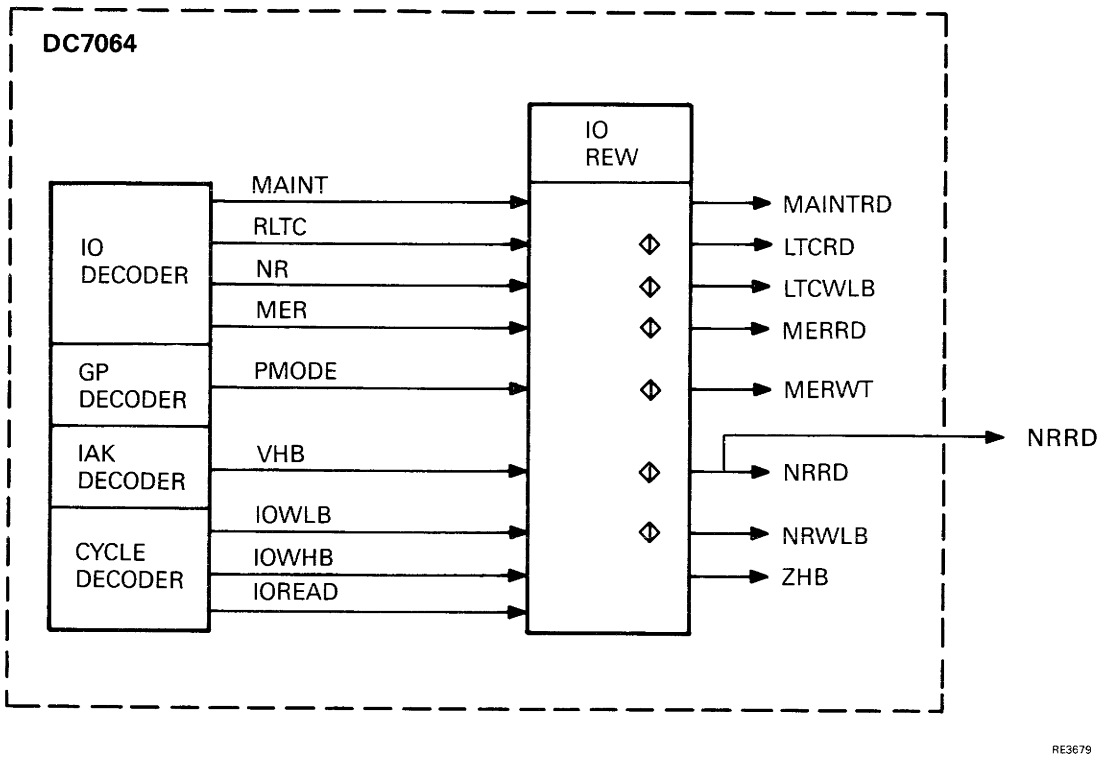


Table 3–10: I/O Read/Write Logic Outputs

Output	Conditions
MAINTRD	Maintenance Register Read—Asserted when either: IOREAD and MAINT are asserted, or PMODE is asserted (see GP Decoder, Table 3–6).
LTCRD	LTC Register Read—Asserted when IOREAD and RLTC are asserted.
LTCWLB	LTC Register Write Low Byte—Asserted when IOWLB and RLTC are asserted.
MERRD	Memory Error Register Read—Asserted when IOREAD and MER are asserted.
MERWT	Memory Error Register Write—Asserted when IOWLB and MER are asserted.
NRRD	Native Register Read—Asserted when IOREAD and NR are asserted.
NRWLB	Native Register Read—Asserted when IOWLB and NR are asserted.

Table 3–10 (Cont.): I/O Read/Write Logic Outputs

Output	Conditions
ZHB	Zero High Byte—Asserted when the Maintenance or LTC registers are read, and when VHB is asserted for a vector read.

3.9 IAK DECODER AND VECTOR GENERATION LOGIC

Figure 3–10 shows the Input Acknowledge Decoder and Vector Generation Logic. The basic sequence of operation is:

1. When either (or both) of the DLARTs asserts (high) an interrupt request (DL0RX, DL0TX, DL1RX, DL1TX), IRQ0 is asserted to the DCJ11, requesting an interrupt.

For the rest of this sequence, assume DL0RX is asserted.

1. Eventually, the DCJ11 will execute an IAK—Vector Read cycle, causing the AIO Decoder to assert IAK.
2. IAK does two things:
   a. it latches DL0RX
   b. it's ANDed with BFCTL from the DCJ11 to assert VEC.
3. In the IAK Decoder:
   a. Latched-DL0RX asserts VHB (Vector High Byte) high.
   b. VEC,VHB (feedback), and latched-DL0RX deassert the corresponding output (that is, -DL0RX is asserted low). This pulls the DL0RX line (from the DLART, not from the latch) low.
   c. Assuming there are no other requests pending, IRQ0 is deasserted.
4. In the Vector Generator:
   a. Both VEC and VHB must be asserted to enable any of the outputs.
   b. Latched-DL0RX asserts (high) DAL<5:4>; that is vector address 60.

Note that the IAK Decoder is mechanized so that the four DL outputs are either not asserted and in the high-impedance state, or asserted low, negating the DL input from the DLARTs. In addition, the mechanization is such that the requests are prioritized:

DL0RX highest
DL0TX
DL1RX
DL1TX
QBIRQ lowest

If the interrupt request is a Q22-bus level 4 request, QBIRQ is asserted, asserting IRQ0. In this case, the request is not latched, but IAK and BFCTL assert VHB as in 3.b. above. Assuming neither DLART is requesting an interrupt, only IAK Decoder output ACKQ is asserted and input to the Q22-bus State Machine.

Figure 3–10: IAK Decoder and Vector Generation Logic

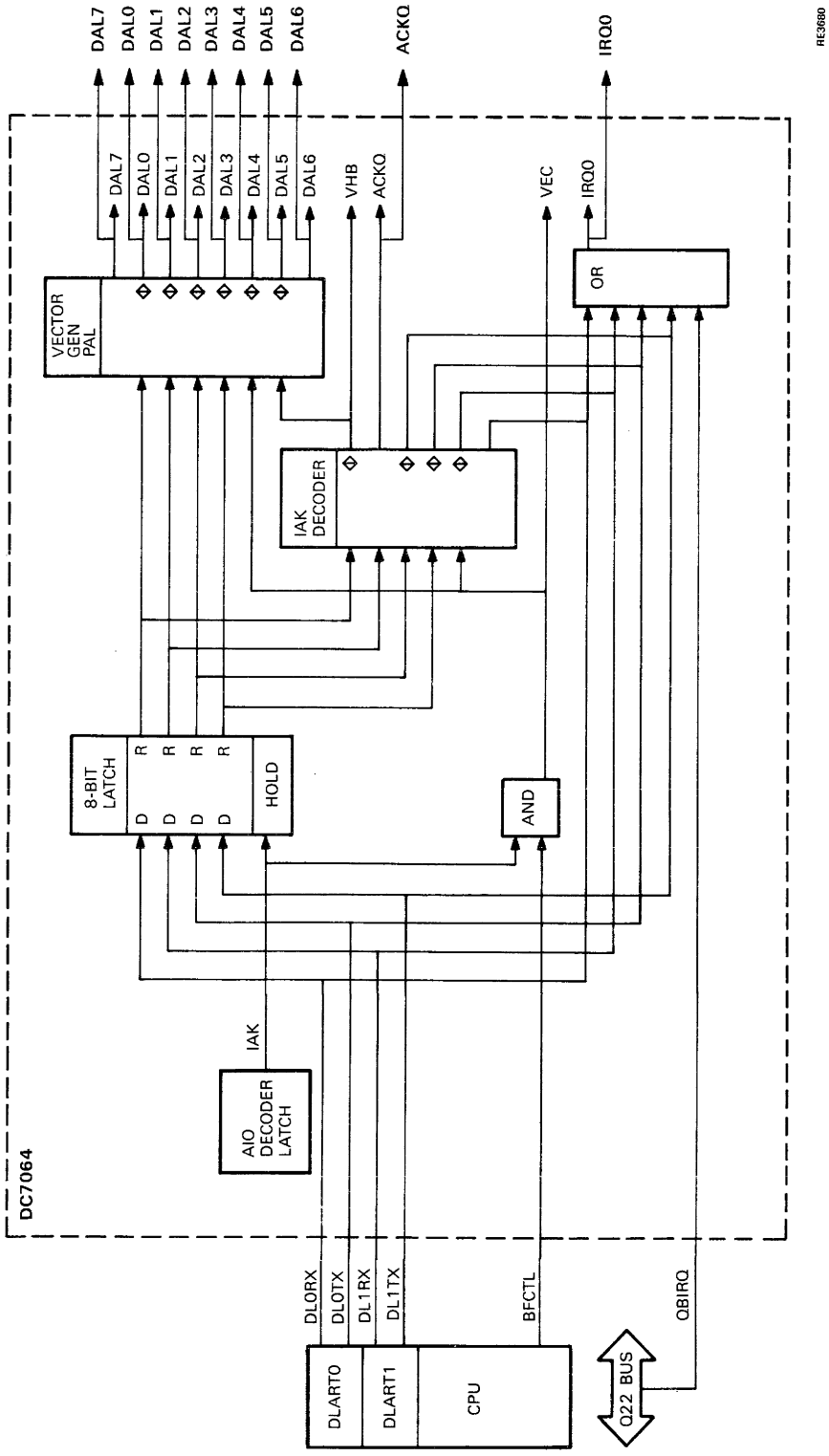


Table 3–11: IAK Decoder Outputs

Output	Conditions
VHB	Vector High Byte—Asserted when any DLART interrupt request is asserted.
DL0RX	DLART 0 Receiver Interrupt Request—Deasserted low (-DL0RX is asserted) when VEC, VHB, and latched-DL0RX are asserted. When not low, this line is in the high impedance state.
DL0TX	DLART 0 Transmitter Interrupt Request—Deasserted low (-DL0TX is asserted) when VEC, VHB, and latched-DL0TX are asserted, and DL0RX is in the high impedance state. When not low, this line is in the high impedance state.
DL1RX	DLART 1 Receiver Interrupt Request—Deasserted low (-DL1RX is asserted) when VEC, VHB, and latched-DL1RX are asserted, and DL0RX and DL0TX are in the high impedance state. When not low, this line is in the high impedance state.
DL1TX	DLART 1 Transmitter Interrupt Request—Deasserted low (-DL1TX is asserted) when VEC, VHB, and latched-DL1TX are asserted, and DL0RX, DL0TX, and DL1RX are in the high impedance state. When not low, this line is in the high impedance state.
ACKQ	Acknowledge BIRQ—Asserted when VEC is asserted and latched-DL0RX, latched-DL0TX, latched-DL1RX, and latched-DL1TX are all deasserted.

Table 3–12: Vector Generation logic Outputs

Vector	Conditions
60	DAL<5:4> = H. Asserted when VEC, VHB, and latched DL0RX are asserted.
64	DAL<5:4> and <2> = H. Asserted when VEC, VHB, and latched DL0TX are asserted.
300	DAL<7:6> = H. Asserted when VEC, VHB, and latched DL1RX are asserted.
360	DAL<7:6> and <5:4> = H. Asserted when VEC, VHB, and latched DL1TX are asserted.

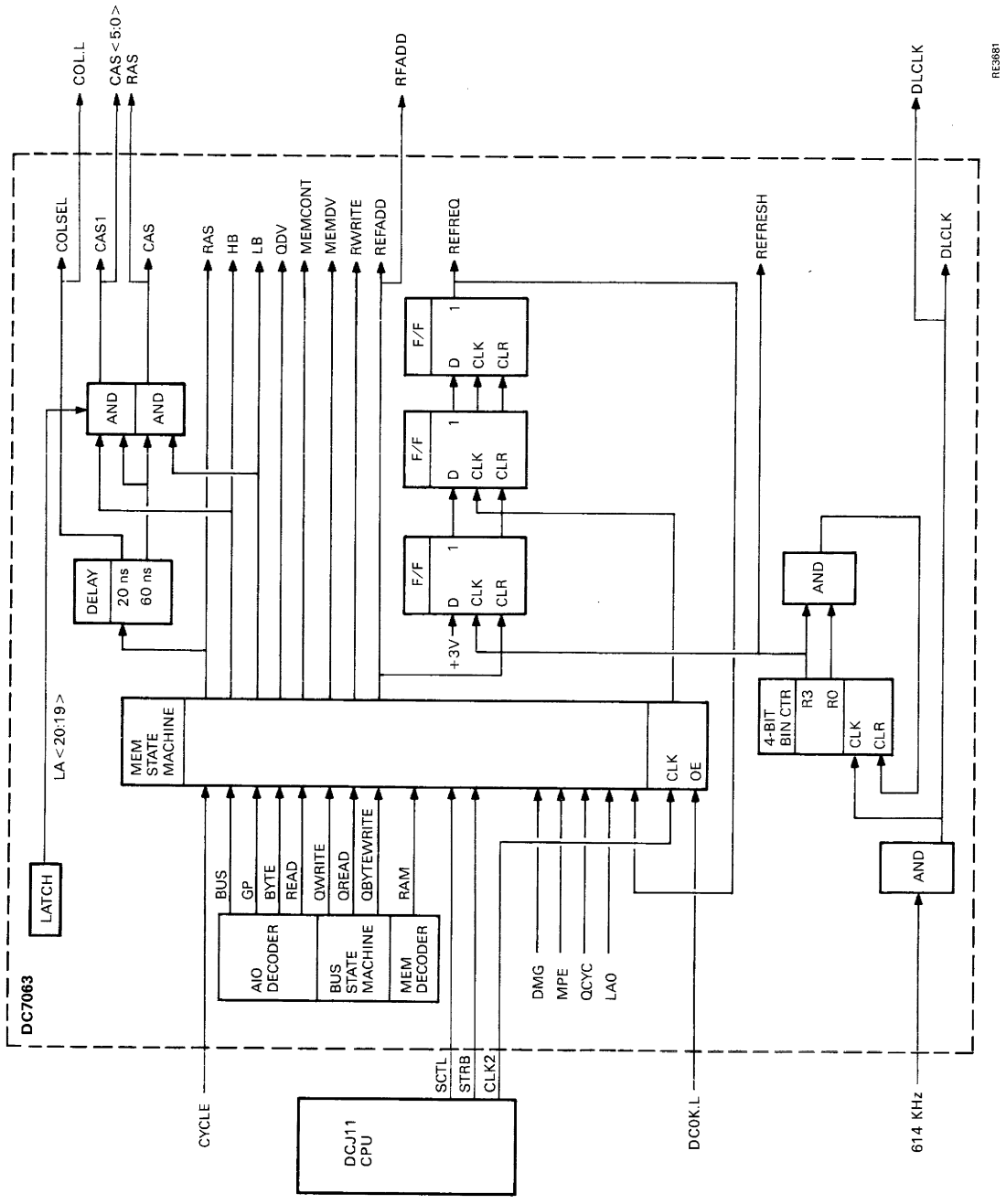
3.10 MEMORY STATE MACHINE

Figure 3–11 shows the Memory State Machine. All the inputs and outputs are used.

The logic states define four major sequences:

- Refresh
- Local I/O
- Write
- Read

Figure 3–11: Memory State Machine



3.11 BUS STATE MACHINE

The Bus State Machine, Figure 3–12, consists of two parts, a Master and a Slave.

There are three Master/Slave interconnections. Slave output QCONT is input to the Master, and Master outputs TDIN and TDOUT are input to the Slave. Slave output QREAD is fed back to its input.

The bus state machine implements the functions of a Q22-bus arbiter and generates the signals required of a Q22-bus master and slave device, including:

TSYNC—Transmit SYNC
TDIN—Transmit Data In
TDOUT—Transmit Data Out
TIAK—Transmit Interrupt Acknowledge
TDMG—Transmit DMA Grant
TRPLY—Transmit RPLY

Note—Q22-bus or Local Memory

The KDJ11-D/S setsup the address portion of a Q22-bus transaction for every memory reference. In the Master state machine, TSYNC is asserted only for a Q22-bus transaction. If the memory address is an on-board reference, TSYNC (and BSYNC) will not be asserted, effectively aborting the bus transaction.

3.12 SACKTIMEUP, NXM, BBS7, BTWT

This section describes the SACK time-out timer, and the mechanization for BBS7 and BWTBT (Figure 3–13).

3.12.1 SACKTIMEUP and NXM

The SACKTIMEUP counter is a 16-state binary counter. It starts counting as soon as the clear input is removed; that is as soon as the Bus State Machine Master asserts either TDIN, TDOUT, or TDMG. The counter is clocked by the 614.4 kHz oscillator. As soon as the counter reaches state 8, SACKTIMEUP is asserted. In other words, SACKTIMEUP is asserted approximately 13 μs after the counter is started.

NXM (Non-existent Memory) is asserted if RRPLY is not asserted within 13 μs after TDIN or TDOUT is asserted. A NXM error asserts ABORT (see Figure 3–14).

3.12.2 BBS7 and BWTBT

At the start of a bus transaction, the bus master asserts the slave device address, BBS7 to reference the I/O Page, and either asserts or deasserts BWTBT to indicate the type of transaction. During the address portion of a bus transaction, the Bus State Machine Master asserts JDATALTCH which selects the "1" inputs to the BBS7/BWTBT input multiplexer.

BBS7 is asserted if the I/O Decoder has asserted QIOPAGE (Table 3–8). That is, the device address is in the I/O Page (but not a KDJ11-D/S register address).

BWTBT is asserted if the AIO decoder has not asserted READ (Table 3–5). That is, the transaction will be a data-out (or write) transaction. If READ is asserted, BWTBT is deasserted.

When JDATALTCH is not asserted, BBS7 is deasserted, and BWTBT is either asserted or deasserted by BYTE from the AIO Decoder, to indicate a byte or word transfer.

Figure 3–12: Bus State Machine

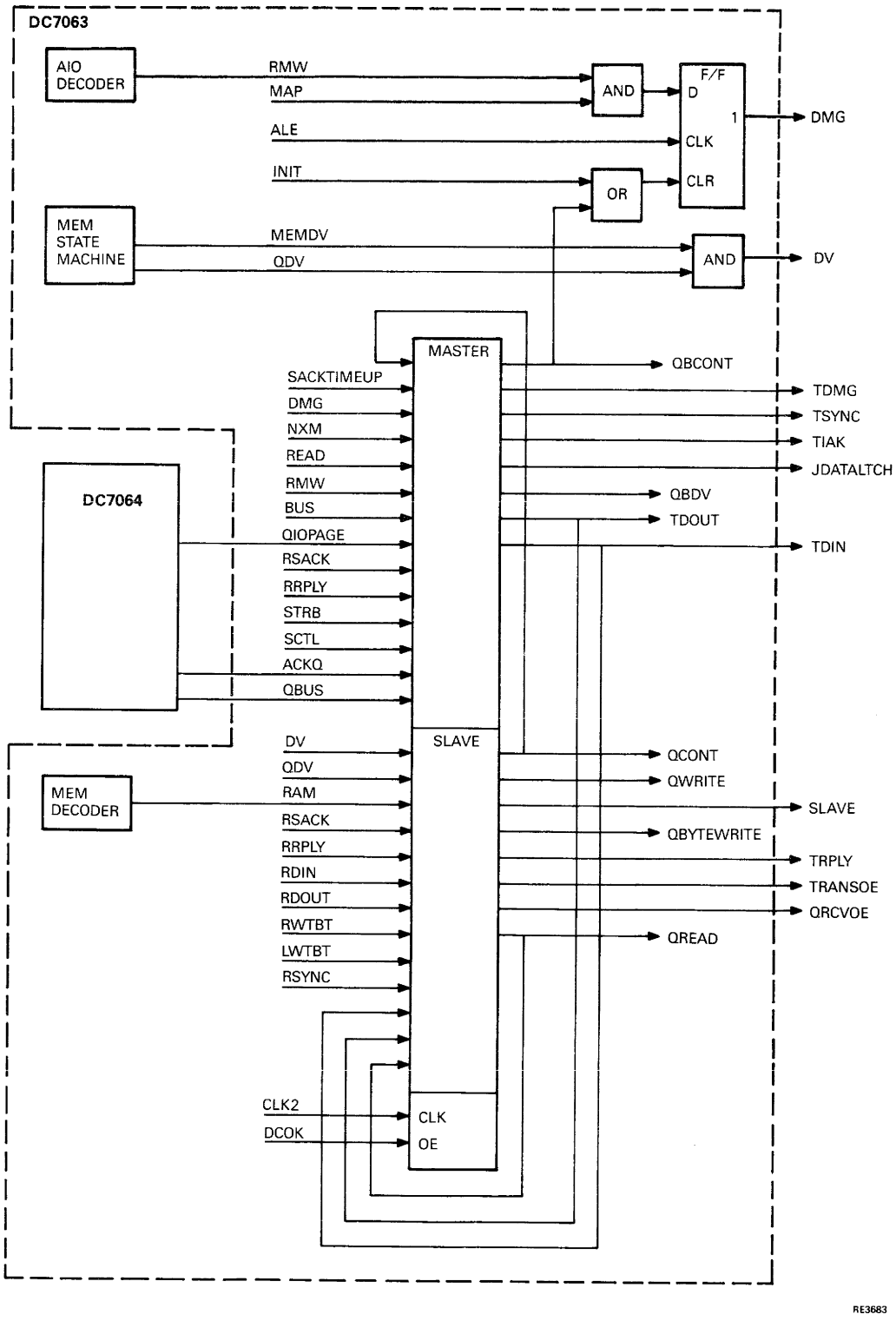
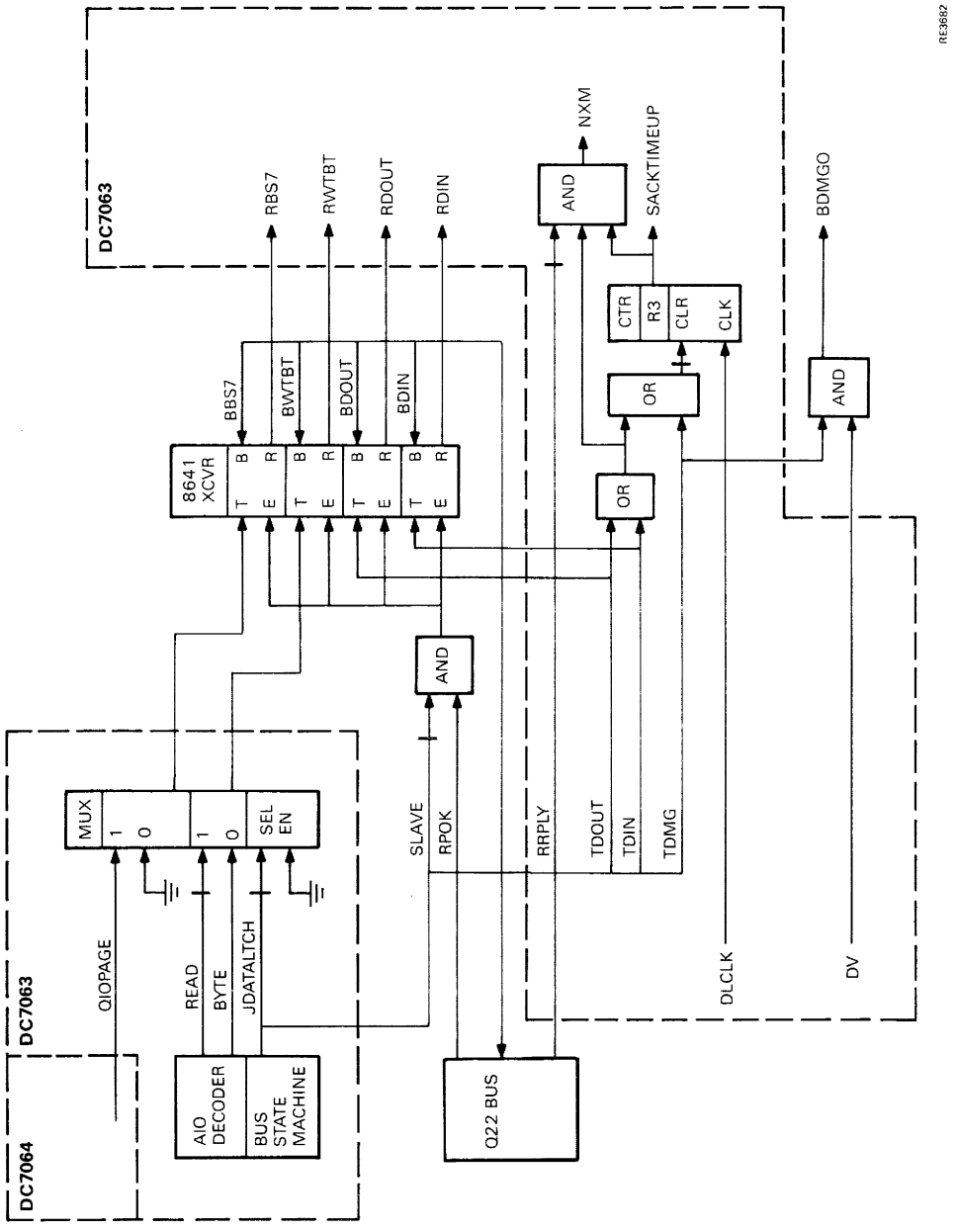


Figure 3–13: SACKTIMEUP, NXM, BBS7, and BWTBT



3.13 PARITY AND ABORT

Parity is generated on all data written into RAM, and parity bits 16 and 17 are set accordingly. See Figure 3–14. If the low byte, DAL<7:0>, has an even number of bits, PDI16 (Parity Data In) is asserted. If the high byte, DAL<15:8>, has an odd number of bits, PDI17 is asserted. When RAM is read, PDO16 and DAL<7:0> should contain an odd number of bits. If not, ERROR is asserted. Similarly, PDO17 and DAL<15:8> should contain an even number of bits; an odd number of bits will assert ERROR.

A parity error asserts ABORT and PE to the DCJ11 (Table 3–1). For DCJ11 reads, a RAM parity error causes an unmaskable interrupt through vector 114. For Q22-bus controlled RAM reads, parity errors are reported by asserting DAL<17,16>.

Note—Parity Errors

A parity error on Q22-bus data is asserted during the current instruction cycle. A parity error on local memory data is asserted during the next instruction cycle.

Parity detection is enabled by MER<00> (CSR<0>). CSR<2> is a diagnostic bit and forces a parity error. (See Section 3.14.1).

3.14 REGISTERS

This section describes how the following registers are accessed:

MER—Memory Error Register
MAINT—Maintenance Register
NR—Native Register
LTC—Line Time Clock Register

3.14.1 Memory Error Register

The MER bits are described in Chapter 1, Figure 1–31, and Table 1–24. Figure 3–15 shows the MER access paths.

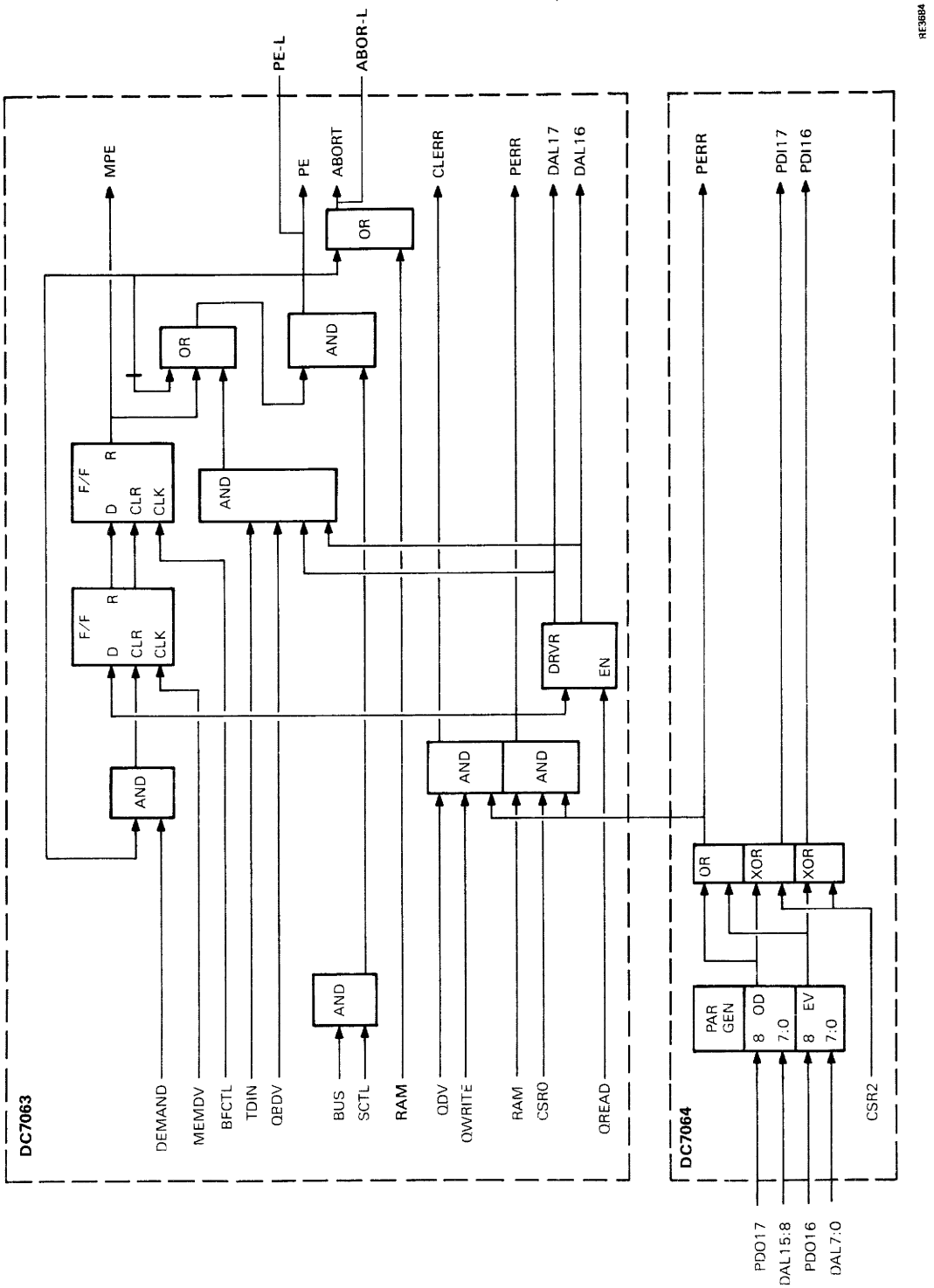
MER<15,14,02,00> comprise four flip-flops set by TDAL<15,14,2,0>. (TDAL<15:0> is DAL<15:0> fed through a pair of tristate octal drivers.) These four bits assert respectively CSR<15,14,2,0> (Control/Status Register). MER<11:05> comprise two sets of 8D flip-flops.

CLERR (Clock Error—Figure 3–14) is the preset input to MER<15>. During a RAM read, MER is not asserted, and inputs LA<17:11> are selected through the multiplexer. The multiplexer feeds the first set of MER<11:05> flip-flops. When enabled, this set stores address bits <17:11>. LA<18> is directly input to MER<05> in the second set of MER<11:05> flip-flops. Ground is input to MER<11:06> in this set.

When a parity error occurs, CLERR is asserted, setting CSR<15>. At the same time, CLERR enables both sets of MER<11:05> flip-flops, storing address bits <21:5>. (Although ground is input to bits <11:06> in the second set of flip-flops, these bits represent address bits <21:19> and must be zero to address RAM.)

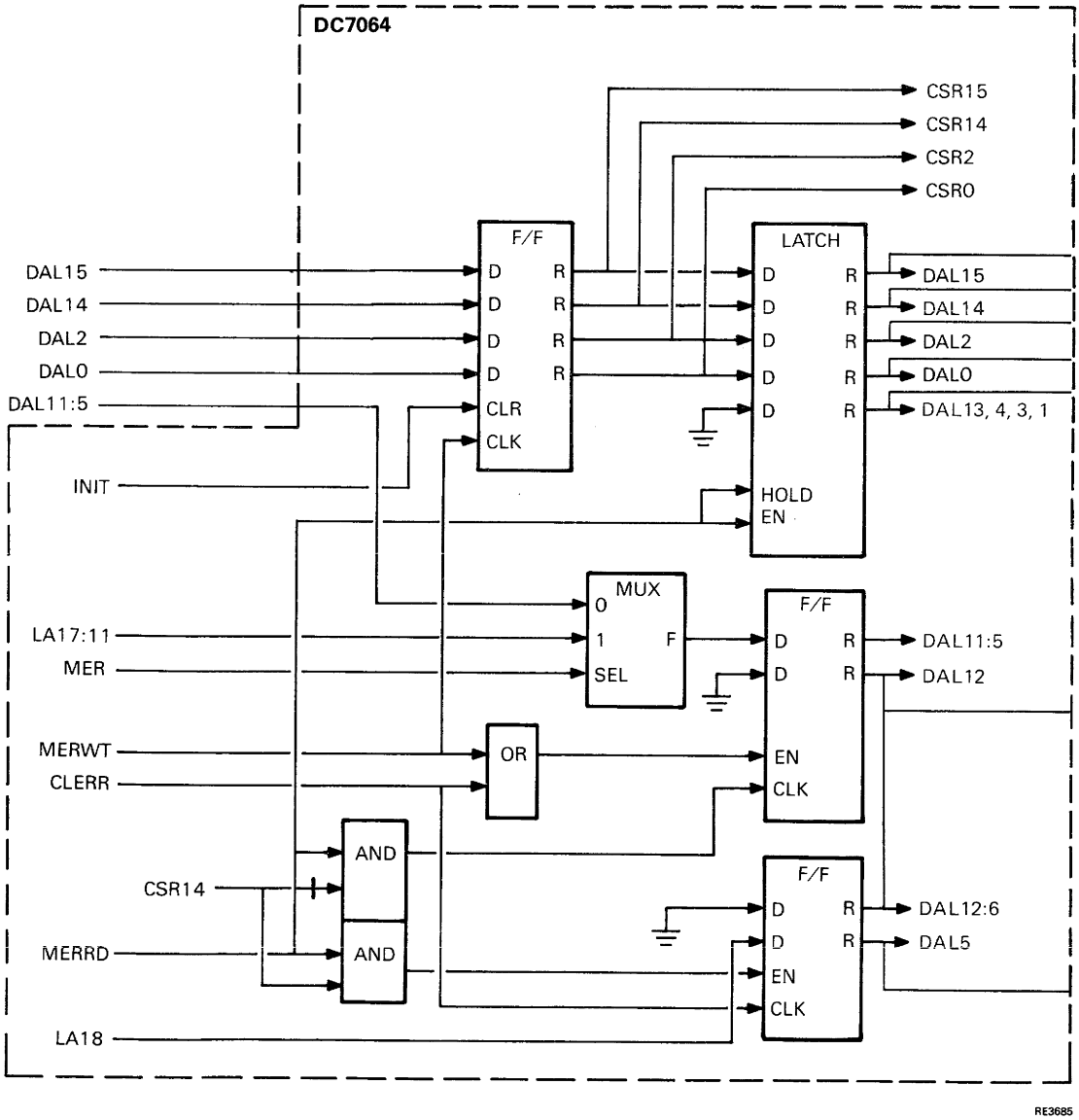
To read the MER, MERRD is asserted through the I/O Read/Write logic (Table 3–10) enabling the output of the latch for MER<15,14,02,00> to DAL<15,14,2,0>. CSR<14> is normally cleared, and MERRD enables the output of the first set of MER<11:05> flip-flops to DAL<11:5>. In other words, bits <17:11> of the error address are asserted on DAL<12:5>. To read the second set of MER<11:05> flip-flops, CSR<14> is set to 1, and the MER is read a second time, putting address bits <21:18> on DAL<8:5>.

Figure 3-14: Parity Error and Abort



To write the MER, MERWT is asserted from the I/O Read/Write logic, clocking the inputs to MER<15,14,11:05,02,00>. At the same time MER is asserted from the I/O Decoder (Table 3–8), selecting TDAL<11:5> through the multiplexer to the first set of MER<11:05> flip-flops. MER<11:05> are written only for diagnostic purposes.

Figure 3–15: Memory Error Register Access



3.14.2 Maintenance Register

The Maintenance Register bits are described in Chapter 1, Figure 1–5 and Table 1–6. Figure 3–16 shows Maintenance Register access. The Maintenance Register is two quad drivers with inputs strapped to ground, +3 V and jumper W1. It is read when MAINTRD is asserted by the I/O Read/Write logic (Table 3–10).

When HALT option jumper W1 is removed, Maintenance Register<03> is set. When a HALT instruction is executed, the processor traps to location 4 in the kernel data space and sets CPU Error Register<07> (Chapter 1, Figure 1–4). When jumper W1 is installed, bit<03> is cleared and the processor enters console ODT mode when a HALT instruction is executed.

3.14.3 Native Register

The NR (Native Register) bits are described in Chapter 1, Figure 1–7 and Table 1–8. The NR comprises 16 drivers and eight flip-flops (Figure 3–17).

The eight flip-flops are NR<07:00> and are set from TDAL<7:0> by the trailing edge of NRWLB when it is asserted by the I/O Read/Write logic (Table 3–10). NR<07> is the Self-test Enable bit, and NR<06:00> are the remote indicator bits. IND<6:0> go to the 34-pin connector (Table 2–4). The flip-flops are cleared if DCOK is not asserted.

The "contents" of the NR are placed on DAL<15:0> when NRRD is asserted by the I/O Read/Write logic. NR<15:13> indicate the KDJ11-D/S revision level, and NR<12:08> indicate the bootstrap option. Jumpers W22, W8, W5, W3, and W2, or jumper W22 and STB<3:0> select the bootstrap option. STB<3:0> come from the 34-pin connector.

3.14.4 Line Time Clock Register

The LTC Register bits are described in Chapter 1, Figure 1–6 and Table 1–7. The LTC Register comprises two flip-flops and eight drivers (Figure 3–18).

LTC<05:00> are not used. LTC<06> is the event interrupt enable. The first flip-flop is the enable flip-flop, and is set from TDAL<6> by the trailing-edge of LTCWLB from the I/O Read/Write logic (Table 3–10). The output of the enable flip-flop is input to the LTC<06> driver and the request flip-flop. If LTC<06> is set, LTC will be asserted (low) to the DCJ11 EVENT input (Table 3–1) when REVNT (Receive BEVNT) is asserted. Note that the state of REVNT is also indicated by the NR<07> driver. The enable flip-flop is cleared by INIT and the request flip-flop is cleared by ACKLTC, both from the GP Decoder (Table 3–6).

When LTCRD is asserted by the I/O Read/Write Decoder, the driver inputs are gated onto DAL<7:0>.

3.15 RAM ADDRESSING

The 512-kbyte on-board RAM is addressed by LA<18:1> through the Memory Address Mux (multiplexer), Figure 3–19. The address comprises a 9-bit row address and a 9-bit column address, MUX<7:0>. The row address, LA<18,16:9>, is selected through the mux when COLSEL is not asserted (Figure 3–11). When COLSEL is asserted, the column address is selected. The column address comprises LA<17,8:5> and the output of an up/down counter. The counter supplies the four least significant bits of column address. The counter allows a 16-word block to be written into RAM from the Q22-bus by latching a single row address in RAM, and incrementing the column address with the trailing edge of TRPLY.

Figure 3–16: Maintenance Register Access

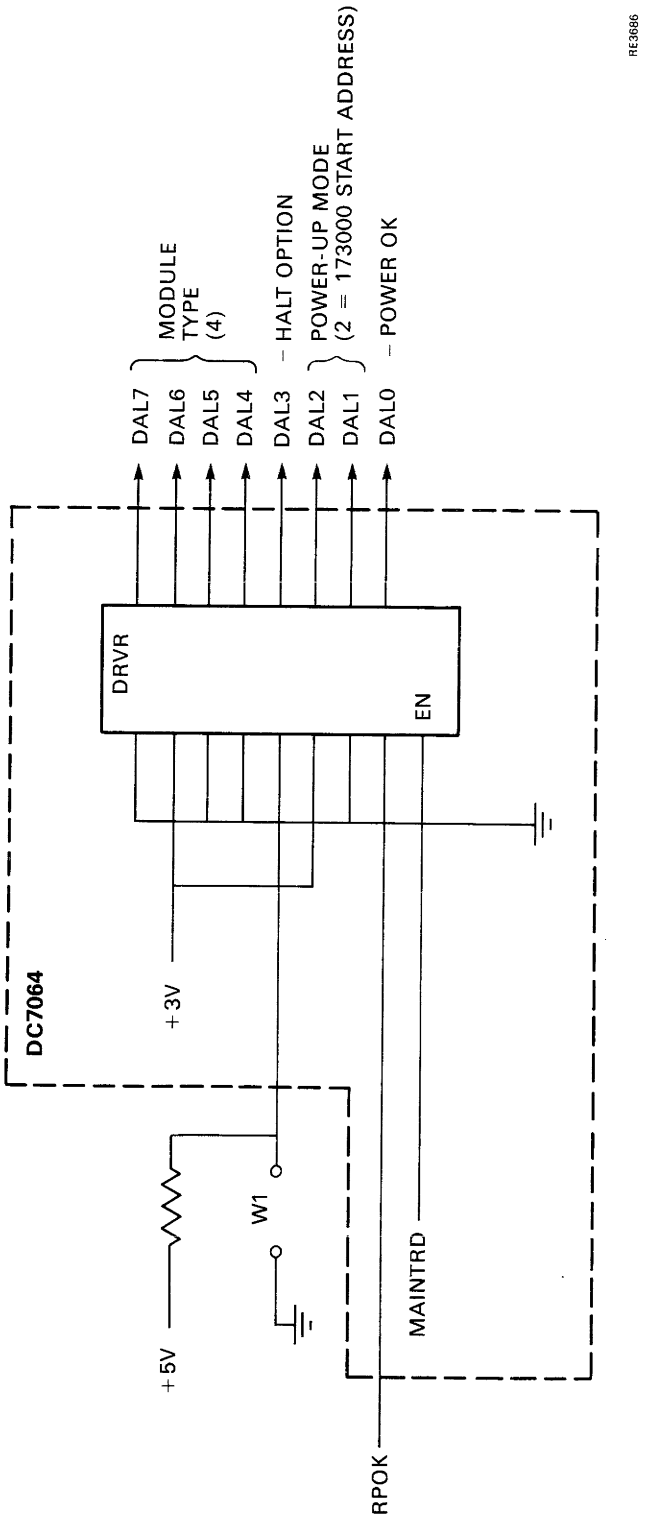
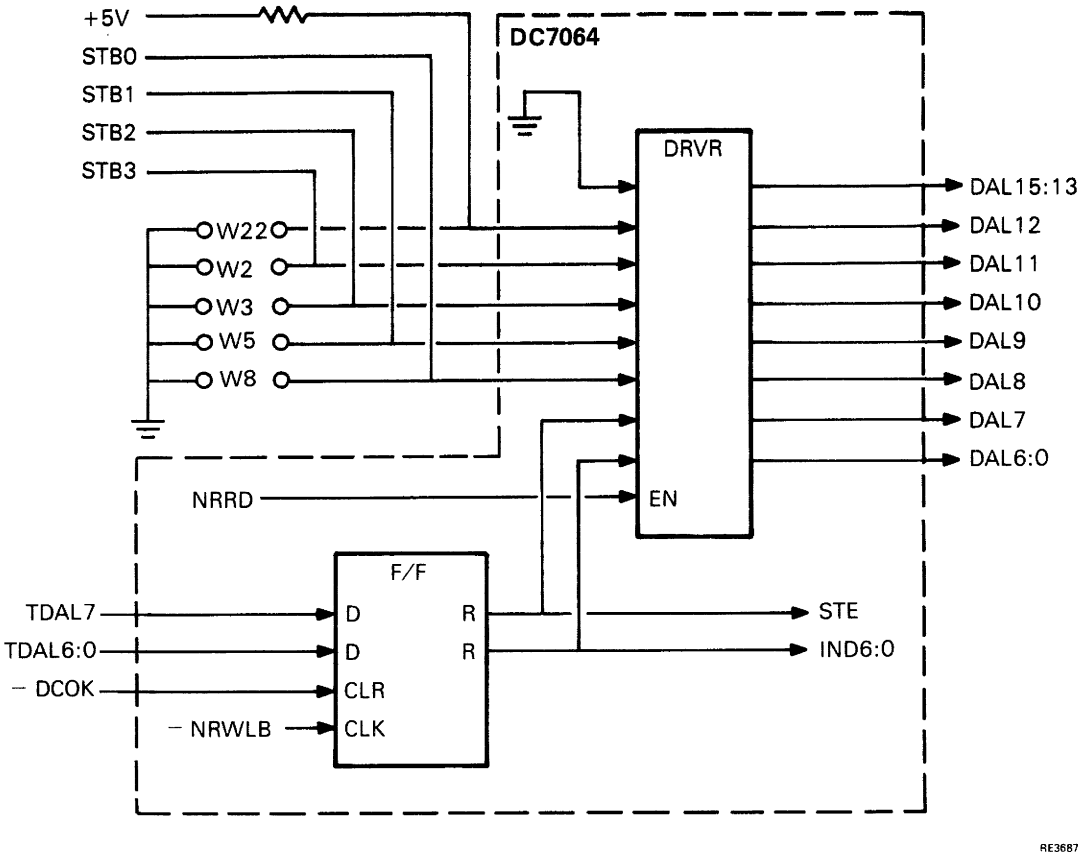


Figure 3–17: Native Register Access



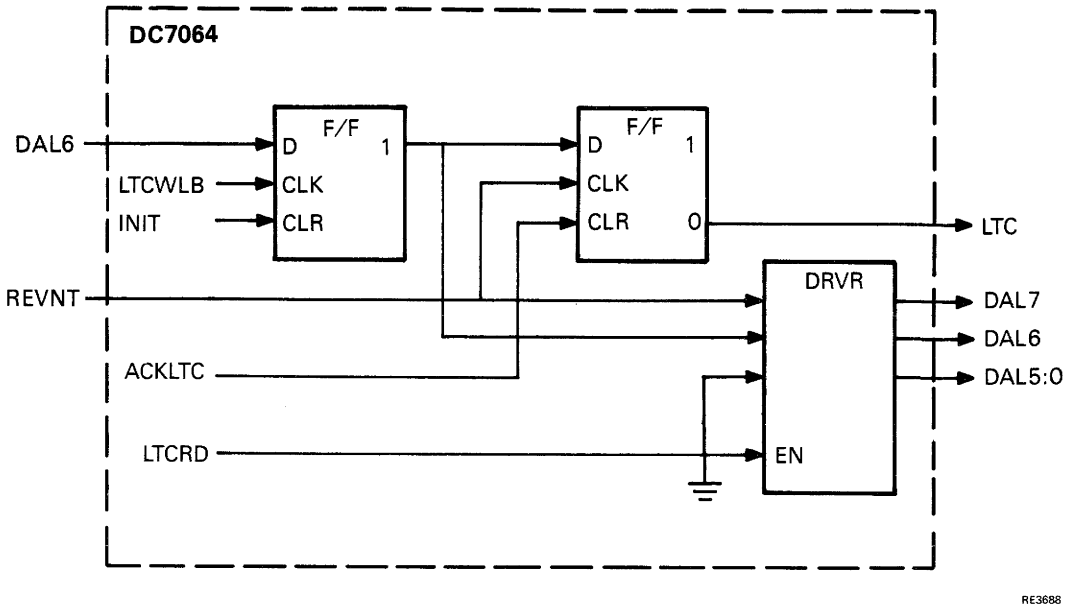
3.15.1 Refresh

The RAM must be refreshed at least once every 4 ms. The refresh is done by asserting RAS, and strobing each of 256 column addresses (MUX<7:0>).

The refresh cycle is controlled by the Memory State Machine, Figure 3–11. When the Memory State Machine asserts REFADD, the Memory Address Mux is disabled, and its outputs go to the high-impedance state. REFADD also enables the Refresh Address Counter, and its outputs are now gated to MUX<7:0>, as the RAM column address (Figure 3–19).

The eight-bit Refresh Address Counter is clocked by REFRESH, which is the output of the Refresh Counter, Figure 3–11. REFRESH has a period of approximately 15 μs.

Figure 3–18: LTC Register Access



3.16 DLARTS

The two DL-style UARTs, Figure 3–20, are identical except for the Halt-on-Break option implemented with jumper W11 in DLART0. The DLARTs are enabled respectively by DL0 and DL1 from the I/O Decoder (Table 3–8). Serial data in and out is fed from and to the 34-pin connector through receivers and drivers. Parallel data is carried on DAL<15:0>.

The baud rate is selected either externally through the 34-pin connector, or with on-board jumpers W12, W10, W9, W7, W6, and W4. The DLARTs are clocked by the 614.4 kHz oscillator (Figure 3–11).

3.16.1 DLART Registers

The DLART register bits are described in Chapter 1, Figures 1–8 through 1–11 and Tables 1–9 through 1–12. Each DLART (Figure 3–20) contains four registers:

RCSR—Receiver Control/Status Register
RBUF—Receiver Data Buffer
XCSR—Transmitter Control/Status Register
XBUF—Transmitter Data Buffer

To access the registers, the DLART is enabled by either DL0 or DL1. The Cycle Decoder asserts either IOREAD or IOWLB (Table 3–9), and the specific register is addressed by LA<2:1>. The register read/write data is on DAL<15:0>.

Figure 3-19: Memory Address Multiplexer

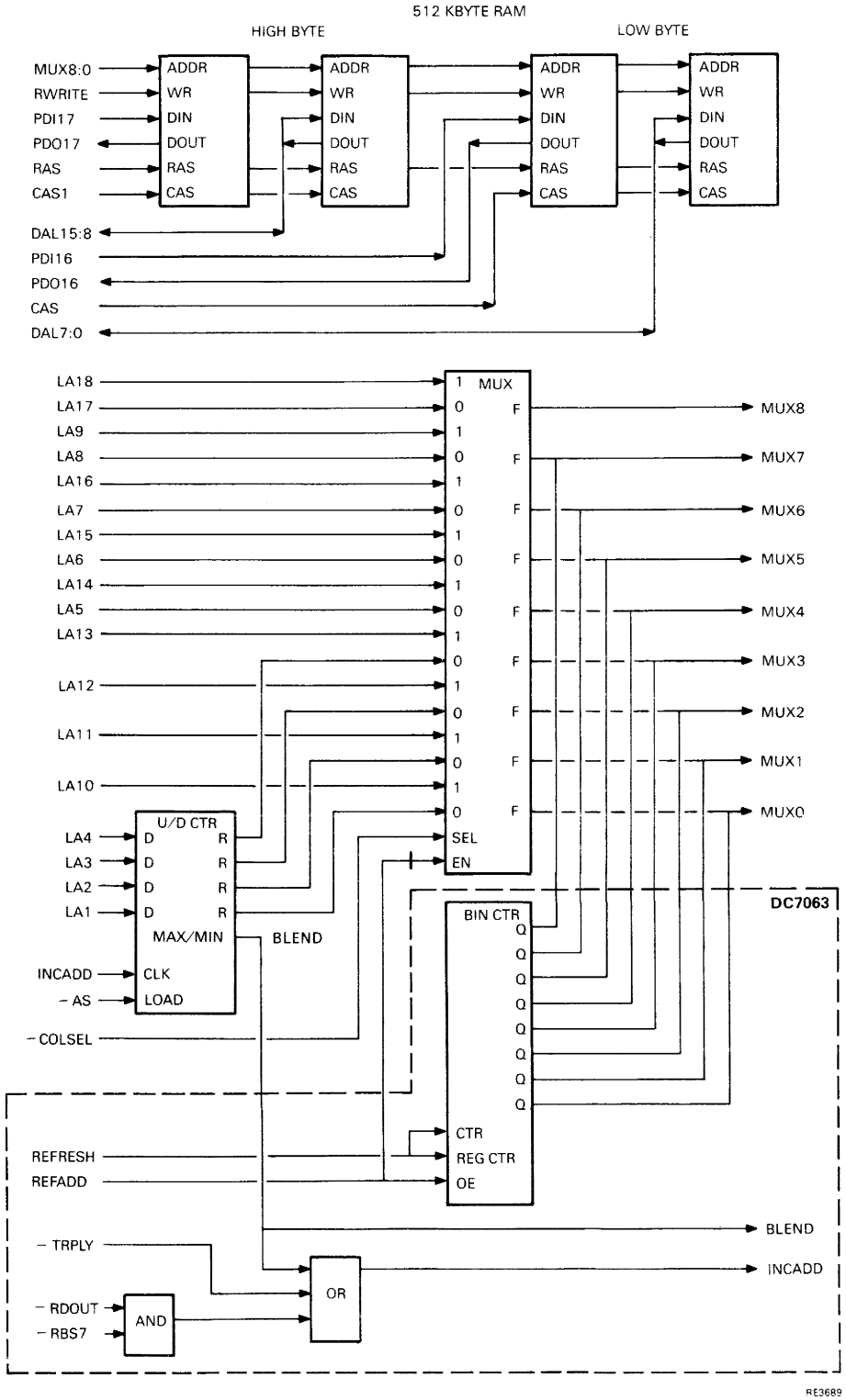
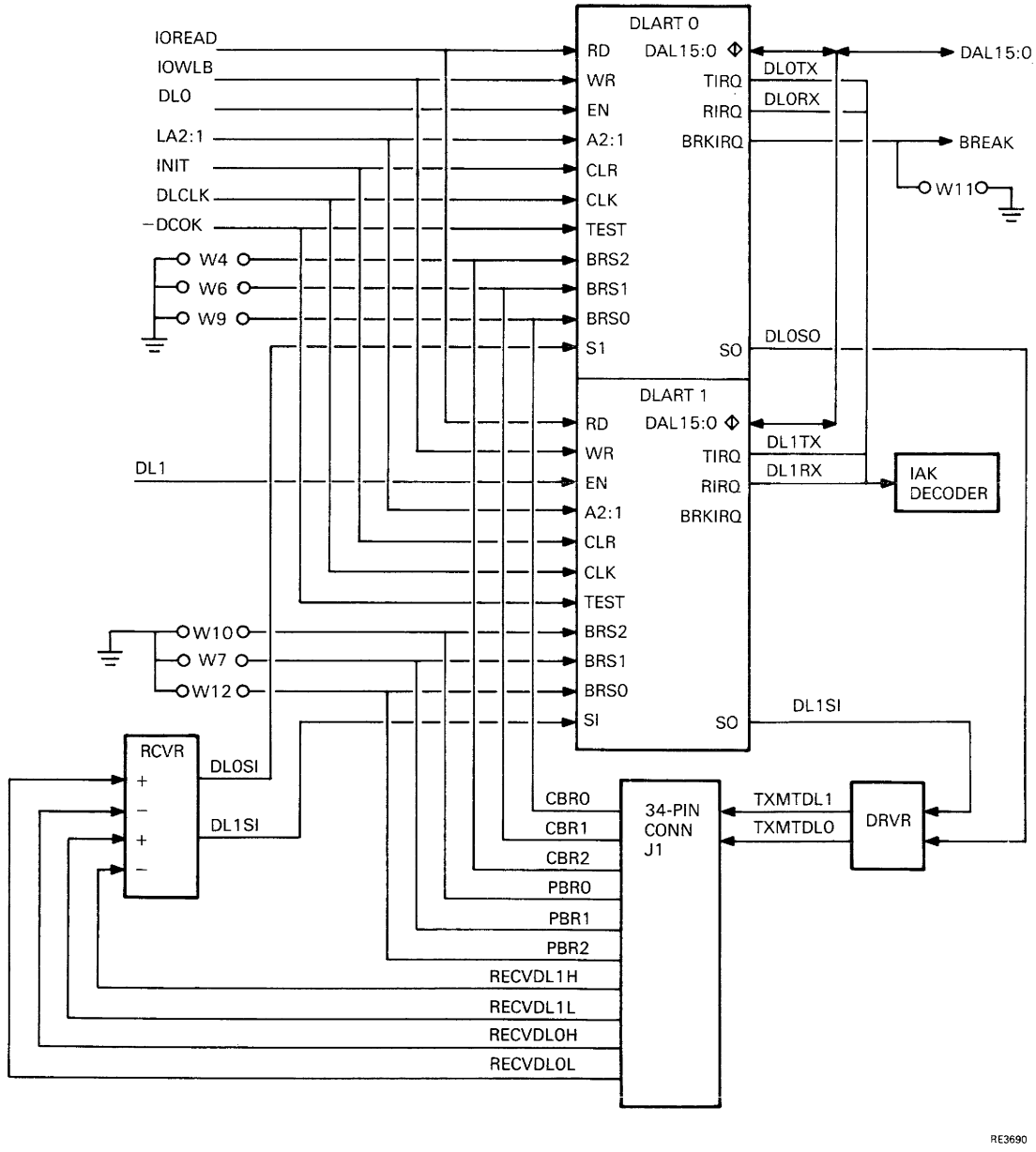


Figure 3-20: DLARTs



Chapter 4
BOOT AND DIAGNOSTIC ROM

This chapter describes the commands and displays for the diagnostic and bootstrap routines resident in ROM on the KDJ11-D/S module.

4.1 INTRODUCTION

Bootstrap and diagnostic programs are resident in two ROMs (read-only memories) on the KDJ11-D/S. The programs (ROM code) test the module and memory at power-up or restart, and boot user's software from various devices.

The ROM code has three general areas:

- The first area includes the diagnostics which are run when the ROM code is started. The diagnostics verify that the KDJ11-D/S and additional Q22-bus memories (if any) are working correctly. Note that test run time is longer if additional memories are installed.
- The second area includes bootstrap routines for most DIGITAL tape, disk and network products.
- The third area includes all of the support routines and user commands.

The ROM code tests only the KDJ11-D/S CPU module and additional Q22-bus RAM modules; it does not test any other modules in the system. The ROM code does not test Q22-bus logic directly. If additional memory is installed, Q22-bus logic test coverage is increased. Generally, many Q22-bus logic problems in the KDJ11-D/S CPU will appear as boot-routine failures. The ROM code does not provide fault isolation to the chip level.

4.1.1 Terminal Requirements

In order to correctly display messages in various languages the console terminal must have the capabilities described in this section. Language selection depends on the self-test ROM version (see Section 4.2.3).

For version 1.0 ROM code, the terminal needs to display only standard ASCII for both English and Spanish (bit 7 of all input is ignored).

For version 2.0 ROM code, certain languages require the terminal to have MCS (multinational character set) capability in addition to 7-bit ASCII. A terminal with MCS capability is required to correctly display all the language selections (see Example 4–20). The terminal characteristics should be such that characters 0 through 127 are ASCII and characters 128 through 255 are MCS. As listed in Table 4–1, certain languages also use 8-bit input.

Table 4–1: Terminal Requirements

Language	Output			Input
English	ASCII		7 bit	7 bit
French	ASCII	MCS	8 bit	7 bit
German	ASCII	MCS	8 bit	8 bit[1]
Dutch	ASCII		7 bit	7 bit
Swedish	ASCII	MCS	8 bit	8 bit[1]
Italian	ASCII		7 bit	7 bit
Spanish	ASCII	MCS	8 bit	7 bit
Portuguese	ASCII	MCS	8 bit	7 bit

[1]7-bit input can be used if bit 7 is set to 0, and the user enters only the minimum characters required to uniquely identify each command

If a VT220 is used, it must be set to VT220 mode to display MCS characters.

4.2 BOOT SELECT

The CPU automatically executes the ROM code each time the KDJ11-D/S is powered-up or restarted (using a remote RESTART switch). The bootstrap sequence and subsequent ROM code execution mode is determined by the value of NR<12:08> (Table 4–2). These boot-select bits are set by jumpers or a remote switch.

During ROM code execution, NR<03:00> contain a status code (Table 4–3). This code is available for remote indication on lines IND<03:00>.

Table 4–2: Native Register Boot Select Codes

NR <12:08>	Switch Position	Description
W22 Installed[1]		
00000	0	Test, enter console mode using English text[2]
00001	1	Test, enter console mode using French text
00010	2	Test, enter console mode using German text
00011	3	Test, enter console mode using Dutch text
00100	4	Test, enter console mode using Swedish text
00101	5	Test, enter console mode using Italian text
00110	6	Test, enter console mode using Spanish text[2]
00111	7	Test, enter console mode using Portuguese text
01000	8	Test, enter console mode (reserved)
01001	9	Test, enter console mode (reserved)
01010	10	Test, enter console mode (reserved)
01011	11	Test, enter console mode (reserved)
01100	12	Test,[3] autoboot tapes & disks,[4] user selects language
01101	13	Test, autoboot DPV11, DUV11, DLV11-E/F, TU58, & RK05
01110	14	Test, autoboot DEQNAs 0 and 1
01111	15	Manufacturing test loop
W22 Removed		
10000	0	Test, autoboot tapes & disks[4] using English text[2]
10001	1	Test, autoboot tapes & disks using French text
10010	2	Test, autoboot tapes & disks using German text
10011	3	Test, autoboot tapes & disks using Dutch text
10100	4	Test, autoboot tapes & disks using Swedish text

[1]NR <12> = W22 (1 = removed, 0 = installed). NR<11:08> = W2, W3, W5, W8 (1 = Removed, 0 = Installed), or remote switch position with W2, W3, W5, and W8 removed.

[2]Only English (codes 00000 and 10000) and Spanish (codes 00110 and 10110) can be selected with version 1.0 ROMs. Eight languages can be selected with version 2.0 ROMs.

[3]High-speed autoboot, memory address/shorts test is bypassed.

[4]"tapes & disks" = DU 0-255, DU 0-255 at floating addresses, DL 0-3, DX 0-1, DY 0-1, MU 0, and MS 0. For DU, removable media is booted before fixed media.

Table 4-2 (Cont.): Native Register Boot Select Codes

NR <12:08>	Switch Position	Description
10101	5	Test, autoboot tapes & disks using Italian text
10110	6	Test, autoboot tapes & disks using Spanish text[2]
10111	7	Test, autoboot tapes & disks using Portuguese text
11000	8	Test, autoboot tapes & disks (reserved)
11001	9	Test, autoboot tapes & disks (reserved)
11010	10	Test, autoboot tapes & disks (reserved)
11011	11	Test, autoboot tapes & disks (reserved)
11100	12	Emulate power up mode 24 with no messages
11101	13	Halt & enter ODT if trap-on-halt disabled, else loop[5]
11110	14	Test, autoboot DEQNAs 0 and 1
11111	15	Test, enter console mode, user selects language

[2]Only English (codes 00000 and 10000) and Spanish (codes 00110 and 10110) can be selected with version 1.0 ROMs. Eight languages can be selected with version 2.0 ROMs.

[5]W1 = Trap-on-Halt (disabled = installed, enabled = removed)

Table 4-3: ROM Status Codes

NR <03:00>	Value[1]	Description
0000	0	HALT switch on, CPU fault, power supply fault, or control has passed from ROM code to secondary boot
0001	1	Preliminary CPU testing—limited error messages
0010	2	Console SLU testing
0011	3	CPU testing
0100	4	On-board memory testing
0101	5	External memory testing
0110	6	Floating-point, LTC interrupt, SLU0 interrupt, and SLU1 interrupt testing
0111	7	not used

[1]Hexadecimal value

Table 4–3 (Cont.): ROM Status Codes

NR <03:00>	Value[1]	Description
1000	8	not used
1001	9	not used
1010	A	not used
1011	B	not used
1100	C	ODT in progress
1101	D	Wrap mode in progress
1110	E	Boot in progress
1111	F	Console mode in progress

[1]Hexadecimal value

4.2.1 Automatic-boot Mode

In this mode, the ROM code automatically loads and starts a program from the user's disk or tape. After user software is started, ROM code is not entered again until the KDJ11-D/S is powered-up or restarted. Automatic-boot mode is described in Section 4.5.

4.2.2 Console Mode

Console mode can be entered in two ways:

- Depending on the contents of NR<12:08>, console mode is entered after testing is completed. In console mode, the ROM code allows the user to determine the execution sequence by entering keyboard commands through the console terminal.
- Console mode can also be entered if the user types <CTRL/C> during testing or during the boot sequence; in this case the Native register bits are ignored.

Console mode is described in Section 4.3.

Note—User Input Ignored

User input from the console keyboard is ignored until the first digit of the memory test count is displayed (Example 4–1), indicating that ROM code is monitoring the keyboard.

4.2.2.1 Manual Start

This code allows the user to start the ROM code and enter console mode without running any tests. The user can then attempt to bootstrap test media if a fatal error occurs. (The media should be write protected.) To manually start, select the boot ODT option (Table 4–2, position NR<12:08> = 11101) and give the following command to micro-ODT:

```
173xxxG
```

where 173xxx is defined in Table 4–4. (For more information on ODT, see the Associated Documents listed in the Preface.)

Table 4–4: Manual Restart Addresses

Address	Description
173000	Normal entry point for power-up or restart. Subsequent action is determined by boot select jumpers or switch.
173002	Relocate ROM code to RAM. Start the code and run from RAM to allow changes to be made. The memory test for the first 40 kbytes of memory is bypassed.
173004	Enter console mode with no testing. For maintenance purposes, this overrides testing errors to allow booting external diagnostics.

4.2.3 ROM Part and Version Numbers

ROMs in sockets E26 and E34 on the KDJ11-D/S module (M7554) have the following part numbers:

	Part Number		Version Number
	23-204E5-00	23-205E5-00	1.0
	23-261E5-00	23-262E5-00	2.0
Socket:	E26	E34	
Byte:	High	Low	

4.2.4 Message Formats

The ROM code displays various messages on the console terminal during a normal power-up sequence. Example 4–1 shows the messages for a typical system bootstrap in automatic-boot mode. The user's software is RT11 and is booted from device DU unit 0.

Example 4–1: Typical Automatic-boot Message

```
9 8 7 6 5 4 3 2 1
DUO

RT-11FB (S) V05.01
```

The descending number sequence (9 8 7 . . .) is displayed to indicate that tests are executing. Messages following the device name and unit number (DU0) are generated by the booted software, not the ROM code. At that point the ROM code is not executing and all commands and messages are determined by the user's software.

Example 4–2 shows the messages displayed when a typical system is powered-up, runs the internal diagnostics, then enters console mode. The ROM code will wait for the user to select the next action.

Example 4–2: Typical Power-up to Console Mode Message

```
9 8 7 6 5 4 3 2 1

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

4.3 CONSOLE MODE

Console mode allows the user to select a boot device, list available boot programs, run ROM-resident tests, obtain a map of all memory and I/O page locations, and "wrap" the console SLU to the second SLU.

When console mode is entered, the ROM code displays the message shown in Example 4–3 and waits for the user to enter a command.

Example 4–3: Console Mode Prompt

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

Console mode provides the user with a choice of six commands, listed in the prompt message. For a brief description of the commands, the user can type either:

? <RETURN> or H <RETURN>

Table 4–5 lists the console mode commands and control characters. The six commands and control sequences are described in Sections 4.3.2 through 4.3.7.

Table 4–5: Console Mode Commands

Command	Description
HELP	List console mode commands
BOOT	Boot from selected device
LIST	List ROM boot programs
MAP	Size memory and map I/O page
TEST	Run tests 3 through 6
WRAP	Wrap SLU0 to SLU1
?	Alternate form of HELP command
/A	BOOT command switch: non-standard CSR address

Table 4-5 (Cont.): Console Mode Commands

Command	Description
/A	WRAP command switch: wrap SLU0 to specified SLU
/O	BOOT command switch: override boot block definition
<DELETE>	Delete previous command character
<RETURN>	Command delimiter
<CTRL/C>	Aborts operation. Enters/restarts console mode
<CTRL/D>	Aborts WRAP and reenters console mode
<CTRL/H>	Selects hardcopy terminal mode[1]
<CTRL/L>	Display language inquiry message
<CTRL/R>	Redisplay command line
<CTRL/U>	Delete command line
<CTRL/V>	Selects video terminal mode[1,2]

[1]Affects result of deleting characters
[2]Default on ROM restart

4.3.1 Entering Console Mode Commands

All of the commands can be executed by typing any or all of the command characters in the correct sequence, starting with the first, and followed by <RETURN>. For example, the WRAP command can be executed by typing any of the following:

W <RETURN>
WR <RETURN>
WRA <RETURN>
WRAP <RETURN>

<CTRL/H> selects hardcopy console terminal mode causing deleted characters to be identified with / (slash) characters when <DELETE> is used.

<CTRL/R> redisplays the command line. <CTRL/R> is normally used on hardcopy terminals to reprint command lines that have been obscured by / when using <DELETE>.

<CTRL/V> selects video console terminal mode causing deleted characters to be erased from the screen when <DELETE> is used. This is the default setting when the ROM code is restarted.

Input is limited to 26 characters (including spaces). None of the commands needs more than 26 characters. If more than 26 characters are entered, the ROM code deletes all of the input characters, redisplays the prompt, and waits for input.

On input, all lower case letters are converted to upper case. Leading spaces and tabs are ignored. Two or more tabs or spaces in sequence are treated a single space. All tabs are converted to and echoed as spaces.

Example 4–4: Invalid Entry Message

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: MP

Invalid input

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

If an invalid command is entered (such as MP—see Example 4–4), an invalid message is displayed and the prompt is re-displayed to request additional input.

4.3.2 HELP Command

The HELP command, Example 4–5, displays a brief description of all console mode commands. It can be executed by typing either:

? <RETURN> or H <RETURN>

Console mode is restarted at the end of this command.

Example 4–5: HELP Command

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: H

Command     Description

Boot        Load and start a program from a device
List        List boot programs
Map         Map memory and I/O page
Test        Run continuous self test - Type CTRL C to exit
Wrap        Wrap Console to SLU1, type CTRL D to exit

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

4.3.3 BOOT Command

The BOOT command allows the user to select a boot device. The command uses arguments and optional switches.

Arguments to the command specify the device name and unit number. The device name is a two letter mnemonic which describes the device. An optional third letter specifies the controller. If the device name is omitted, the program will prompt for it and the unit number (Example 4–6). If the unit number is omitted, the program assumes unit zero. The unit number range is 0 through 255, depending on the device and the boot program.

The BOOT command can be entered in two ways:

1. B <RETURN>—The system will prompt for the device name and unit number, as shown in Example 4–6. Type the device name and unit number and <RETURN>.
2. B <SPACE> device-name unit-number <RETURN>—see Example 4–7.

The optional switches that can be used with the BOOT command are:

- /A—Request that the user type in a non-standard CSR address for the controller. The ROM code prompts for the non-standard address.

- /O—Override the standard boot block definition.

The switch is typed immediately after the BOOT command and before the device name and unit number, for example:

B/A

When the BOOT command is entered without an argument, the ROM code prompts for the additional information, as shown in Example 4–6.

Example 4–6: BOOT Command Argument Prompt

```
Enter device name and unit number then press the RETURN key:
```

The device name and unit number are then entered. If a ? is typed at this point, the ROM code will list the boot programs available, redisplay the argument prompt, and wait for a selection.

Table 4–6 lists examples of BOOT commands and the corresponding ROM code action.

Table 4–6: BOOT Command Interpretation

Command Entered	ROM Code Action
B DU	Boot DU0 using standard controller address
B DUA	Boot DU0 using standard controller address
B DUB	Boot DU0 using first floating controller address
B DU1	Boot DU1
B DUA1	Boot DU1
B/A DU10 Address = 17760400	Boot DU10 with non-standard CSR address = 17760400
B/A/O DU10 Address = 17760400	Boot DU10 with non-standard CSR address = 17760400 without boot block validity check
B/O DU8	Boot DU8 and start the program without boot block validity check
B D U 0	Invalid format (illegal space in device name DU)
BDU0	Invalid format (space required between BOOT command and device name)
B DU 10:	Boot DU10 (colon after the unit is ignored)

Example 4–7 shows a boot from a DL2.

Example 4–7: BOOT Command Using DL2

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: B DL2

DL2

RT-11FB (S) V05.01

.SET TT QUIET

.R DATIME
Date? [dd-mmm-yy]?
```

Example 4–8: LIST Command

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: L

Device          Units           Description

 DU             0-255           RDnn, RXnn, RC25, RAnn
 DL             0-3             RL01, RL02
 DX             0-1             RX01
 DY             0-1             RX02
 DD             0-1             TU58
 DK             0-7             RK05
 MU             0-255           TK50
 MS             0-3             TK25, TS05
 XH             0-1             DECnet ETHERNET
 NP             0-1             DECnet DPV11
 NU             0-15            DECnet DUV11
 NE             0-15            DECnet DLV11-E
 NF             0-15            DECnet DLV11-F

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

4.3.4 LIST Command

This command, shown in Example 4–8, displays a list of all available boot programs found in the ROM. The list includes the device name, unit number range, and a short device description.

The device name is usually a two letter mnemonic. The valid letter range is A through Z. The ROM code converts lower case letters to upper case at input.

The unit number range is the valid range for a particular boot program. The range varies from 0 to 255, depending on the device.

The description, or device type, is intended to be the name on the outside of the physical device. For example, device name DL is described as an RL02.

Console mode is restarted at the end of the LIST command.

4.3.5 MAP Command

The MAP command, shown in Example 4–9:

- displays the current ROM code version number
- determines and displays the size of (''sizes'') consecutive memory
- identifies all memory in the system
- maps all locations in the I/O page
- identifies the ETHERNET controllers and displays the station addresses
- identifies disk (DU) and tape (MU) MSCP devices at their standard addresses of 17772150/2 and 17774500/2.
- indicates whether the BEVNT signal is present in the LTC register.

Memory is mapped from location 0 to the I/O page, in 1-kbyte increments. Every location is not mapped because it takes too much time. The map routine will try to identify the size of each system memory and each memory's CSR address (if applicable). (Note that if two memories share some common addresses or have CSRs with the same address, the MAP command will not work correctly.) If two or more non-contiguous memories are present, ROM code will display their descriptions separated by a blank line. If a Q22-bus memory is installed and shares addresses with on-board memory, the MAP command will not see the Q22-bus memory addresses which overlap on-board memory addresses.

Caution—Memory Overlap

KDJ11-D/S on-board memory addresses are in the range 00000000 through 17777776 (00000000 through 57777776 for 1.5 Mbyte modules). Other system memory devices *should not* be configured to overlap this range. Bootstrap does not check for such memory overlap; if it exists it may result in system disk corruption.

After all memory is mapped, the ROM code waits for the user to press <RETURN>. The command will then continue the map, displaying all responding I/O page addresses. The I/O page map addresses are 17760000 to 17777776. In addition, all responding CPU addresses are listed with a short description.

With the exception of memory CSRs, and devices DU, MU, and XH, responding Q22-bus addresses are not described. Disk and tape MSCP devices DU and MU are identified only at their standard addresses of 17772150 and 17772512 (DU) and 17774500 and 17774502 (MU). ETHERNET devices XH are identified at addresses 17774440, 17774456, 17774460, and 17774476.

When an ETHERNET device is identified during the I/O page part of the MAP command, the hexadecimal station address is read (and displayed) from bits <07:04> and <03:00> of the six consecutive bytes starting at address 17774440 or 17774460.

When the LTC register is read during the I/O page part of the MAP command, BEVENT = 0 or BEVENT = 1 will be displayed following the LTC display. This indicates the presence (1) or absence (0) of the Q22-bus BEVNT signal. The LTC test will not fail if BEVNT is not present.

To prevent displayed data from being scrolled off the screen, the ROM code waits for the user to press <RETURN>. The ROM code assumes the terminal display is at least 24-lines * 80-columns.

Console mode is restarted at completion of the MAP command.

Example 4–9 shows the MAP command in English language format. The descriptions are mnemonic and not translated. For other than English languages, only the console mode prompt is translated.

Example 4–9: MAP Command

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: M

KDJ11-D/S ROM V1.0
512 K Bytes

00000000 - 01777776     512 KB     CSR = 17772100

Press the RETURN key when ready to continue

17772100                MCSR
17772150 - 17772152     DU
17772200 - 17772216     SIPDR0-7
17772220 - 17772236     SDPDR0-7
17772240 - 17772256     SIPAR0-7
17772260 - 17772260     SDPAR0-7
17772300 - 17772316     KIPDR0-7
17772320 - 17772336     KDPDR0-7
17772340 - 17772356     KIPAR0-7
17772360 - 17772376     KDPAR0-7
17772516                MMR3
17773000 - 17773776     CPU ROM
17776500 - 17776506     SLU1
17777520                NR
17777546                LTC CSR
17777560 - 17777566     SLU0
17777572 - 17777576     MMR0,1,2
17777600 - 17777616     UIPDR0-7
17777620 - 17777636     UDPDR0-7
17777640 - 17777656     UIPAR0-7
17777660 - 17777660     UDPAR0-7
17777750                MREG
17777766                CPUER
17777772                PIRQ
17777776                PSW

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

4.3.6 TEST Command

This command causes the ROM code to run most of the power-up tests in a continuous loop. The ROM code starts at test 3, runs all applicable tests and sub-tests, then restarts the loop after test 6 is complete. Testing can be aborted and console mode restarted by typing <CTRL/C> at any time. If an error occurs, the Tests 3 through 6 error routine is entered (see Section 4.4.3). Two actions are possible at this point:

- Console mode can be restarted by typing <CTRL/C>
- Loop through all of the tests, ignoring errors, by typing L <RETURN> (see Table 4–7).

On exit from the test loop, the ROM code displays the total number of loops (passes) and the total number of errors (if any) in the following format:

```
nnn/xxx
```

where *nnn* is the number of errors and *xxx* is the number of times the tests were attempted.

In Example 4–10 the TEST command is entered to run all loopable tests. After four passes, the testing sequence is aborted with no errors by entering <CTRL/C>.

Example 4–10: TEST Command

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: T

Continuous self test - Type CTRL C to exit

0/4

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

4.3.7 WRAP Command

This command takes all input from the console terminal (DLART0) and transmits it to the second SLU (DLART1), or a selected SLU. All input from DLART1 or the selected SLU is sent to the console terminal. This allows the user at the KDJ11-D/S console to communicate with another system through KDJ11-D/S DLART1 or another SLU selected by the user. The command has one optional switch, /A.

Entering the WRAP command without the switch, as in Example 4–11, wraps the console to DLART1 (address 17776500). Entering the WRAP command with the switch, as in Example 4–12, causes the ROM code to request an alternative SLU address (instead of DLART1). The valid alternate address range is 17776500 to 17776676.

All characters are wrapped, with the exception of <CTRL/D>. Entered from the console, <CTRL/D> aborts the WRAP command and restarts console mode.

Example 4–11 shows the WRAP command being entered without the switch. The console will be wrapped to the second SLU at address 17776500.

Example 4–11: WRAP Command Without Switch

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: W

Wrap Console to SLU1, type CTRL D to exit
```

Example 4–12 shows the WRAP command being entered with an alternate SLU address.

Example 4–12: WRAP Command With Switch

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: W/A

Address = 17776520

Wrap Console to SLU1, type CTRL D to exit
```

4.4 TEST RESULTS

4.4.1 Test 1 Errors

When started, the ROM code runs a series of tests which verify the basic MMU operation and the ROM code. At this point in the testing sequence, the comprehensive error message display routines are disabled. If an error occurs during Test 1, the ROM code displays the following error message:

```
KDJ11-D/S  1.00
```

This message indicates that a fatal error condition occurred. The ROM code ignores any keyboard input, except to redisplay the error message each time input is received. The message is the same in any user-selected language.

4.4.2 Test 2 Errors

This test checks the console SLU. When the SLU0 test is running, the ROM code assumes that error messages cannot be displayed. Therefore, if an error occurs, the ROM code will loop on the error.

4.4.3 Tests 3 through 6 Errors

These are the main CPU and memory tests. These are also the tests which continuously loop when the TEST command is entered. If an error is detected during these tests, the ROM code displays a brief error message. All errors are treated as fatal errors; the user is expected to fix the problem before continuing. The error messages are described in Section 4.4.3.1. Section 4.4.3.2 describes how errors can be bypassed for troubleshooting.

4.4.3.1 Test 3 Through 6 Error Messages

Three examples of the error message format are shown in Examples 4–13 through 4–15. In each of the examples, the three lines of the message are interpreted as follows (the fourth line is the KDJ11-D/S prompt—see Example 4–16 and Section 4.4.3.2).

- Line 1

  KDJ11-D/S is the *CPU identifier*, and is the same in any error message. 3.15 is the *test.subtest* number and is test-dependent. The test number is also contained in NR<03:00> (Table 4–3).

- Line 2

  This line is "standard" in any error message.

- Line 3

  This line has four parts:

  1. A short description of the failed area:

     J11 = J11 test error

     J11 FP = floating point test error

     J11 MMU = memory management test error

J11 nnn = unexpected trap to virtual address nnn

LTC CSR = line time clock test error

SLU0 = console SLU test error

SLU1 = second SLU test error

ROM = ROM checksum test error

RAM = onboard memory test error

RAM CSR = onboard memory parity test error

Qbus RAM = Qbus memory test error

Qbus CSR = Qbus memory parity test error

All areas except Qbus RAM and Qbus CSR are on the KDJ11-D/S module.

2. The virtual PC of the failure. Generally, this information is useful only with a program listing.
3. Physical address of the failure. Generally, this information is useful only with a program listing.
4. Only displayed with RAM errors, this part displays:

```
address/found data <> expected data
```

that is, the failing location, the bad data, and the expected data.

Example 4–13: On-board RAM Test Error Message

```
KDJ11-D/S  3.15
Error, see troubleshooting section in Owner's manual for assistance
RAM   VPC=024722    PA=17604722  01000000/125200 <> 125252

KDJ11-D/S>
```

Example 4–14: Q22-bus RAM Test Error Message

```
KDJ11-D/S  3.15
Error, see troubleshooting section in Owner's manual for assistance
Qbus RAM CSR  VPC=nnnnnn

KDJ11-D/S>
```

Example 4–15: J11 Unexpected Trap Error Message

```
KDJ11-D/S  3.15
Error, see troubleshooting section in Owner's manual for assistance
J11 004  VPC=024722

KDJ11-D/S>
```

If an error occurs and a user language has not been selected, the ROM code will prompt for a language, then display the error message, as in Example 4–16, which shows English selected. Note that each line of the language inquiry is displayed in the associated language.

Example 4–16: Language Inquiry and Error Prompt

```
English         Type 1 and press the <RETURN> key
Francais        Tapez 2 et appuyez sur <RETOUR>
Deutsch         Geben Sie 3 ein und drucken Sie <WR>.
Nederlands      Typ 4 en druk op <RETURN>
Svenska         Skriv 5 och tryck sedan pa <RET>
Italiano        Introdurre 6 e premere <RITORNO>
Espanol         Presione el 7 y luego la tecla <RETORNO>
Portuguese      Escreva 8 seguido de <RETURN>

KDJ11-D/S>      1

KDJ11-D/S  3.015
Error, see troubleshooting section in Owner's manual for assistance
ROM    VPC=024722

KDJ11-D/S>
```

4.4.3.2 Commands to Override Errors

When the error message is displayed, the ROM code displays the KDJ11-D/S prompt and waits for input. The available commands are listed in Table 4–7.

Table 4–7: Error Override commands

Command	Result
<CTRL/O>4	Override error and enter console mode.[1]
L	Restart tests at Test 2. Loop through tests ignoring errors. Type <CTRL/C> to exit loop.[2]

[1]The 4 is included with the <CTRL/O> override command to avoid accidental entry of the command.

[2]Do not confuse the L (loop) error override command with the L(IST) console mode command

Caution—Bypassing Errors

System media should be either removed or write-protected before bypassing an error.

4.5 AUTOMATIC-BOOT MODE

This section describes the automatic-boot sequence and boot routines.

4.5.1 Boot Code

This part of the boot and diagnostic ROM code provides the primary bootstrap for the devices listed in Table 4–8.

Table 4–8: Q22-bus Boot Devices

Mnemonic	Device	
DU	RA60/8n, RX50, RD5n, RC25	Disk MSCP[1]
DL	RL01, RL02	
DX	RX01	
DY	RX02	
DD	TU58	
DK	RK05	
MU	TK50	Tape MSCP[2]
MS	TS05, TK25	
XH	ETHERNET	DECnet
NP	DPV11	DECnet
NU	DUV11	DECnet
NE	DLV11-E	DECnet
NF	DLV11-F	DECnet

[1]DU is a general purpose boot device mnemonic for disk MSCP (Mass Storage Control Protocol) devices.

[2]MU is a general purpose boot device mnemonic for tape MSCP devices.

The primary boot program normally reads a 256-word secondary boot program, from the device into memory (starting at location 0). Following successful load of the secondary bootstrap, the bootstrap is started with the contents of general purpose registers:

R0 = the unit number booted
R1 = the controller address

The controller address contained in R1 at the start of the bootstrap is not always the first address of the controller. Before it is started, the secondary bootstrap program is checked to see if it is in the expected format for bootable media. (Bootable media is defined in Section 4.5.7). The secondary bootstrap is started at location 0 for all bootstraps except: DPV11, DUV11, DLV11-E and DLV11-F, which are started at location 6.

4.5.2 Automatic-boot Sequence (Autoboot)

In the autoboot (automatic-boot sequence), the ROM code continuously tries to load the secondary bootstrap from an ordered list of devices, as described in Sections 4.5.4 and 4.5.5. The specific boot device list is determined by the boot select jumpers/switch (Table 4–2). The load is attempted until a bootable device is found or until the user aborts the autoboot by typing <CTRL/C>.

Note—DEQNA

The DEQNA is not included in either boot device list because autoboot will not terminate if there is no response over the Ethernet. It will continuously retry. The DEQNA is selected separately and runs by itself. See Section 4.5.6.

If a secondary bootstrap is successfully loaded, NR<03:00> is set to 0. The ROM code displays the boot device's mnemonic and unit number before transferring control to the secondary boot.

4.5.3 Bootstrap Error Messages

There are two types of bootstrap error messages. One is associated with autoboot at power-up or restart, and the other with the console mode BOOT command.

If the autoboot is not successful after two passes, the ROM code displays the message in Example 4–17, indicating that the autoboot was not successful but will make continuous passes (until successful or aborted). The example shows the power-up or restart count-down sequence and the unsuccessful autoboot message. Note that if a language had not been selected and the autoboot failed, only the first line of the message:

```
KDJ11-D/S  E.01
```

would be displayed.

Example 4–17: Unsuccessful Automatic-boot Message

```
9 8 7 6 5 4 3 2 1

KDJ11-D/S  E.01
No bootable devices found.
Boot in progress, type CTRL C to exit.
```

When an error occurs in a boot program called with the console mode BOOT command, the ROM code displays a specific error message for the failing device and unit number. Table 4–9 lists the possible error messages. Note that all the errors listed may not apply to all boot programs.

Table 4–9: Boot Device Errors

Drive not ready
Media not bootable
Non existent controller, address = 177nnnnnn
Non existent drive
Invalid unit number
Invalid device
Controller error
Drive error

Examples 4–18 and 4–19 show console mode BOOT command error messages.

Example 4–18: Console Mode Boot Error Message (1 of 2)

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: B DL3

KDJ11-D/S E.05
Non existent drive

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

Example 4–19: Console Mode Boot Error Message (2 of 2)

```
Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key: B DU1

KDJ11-D/S E.04
Non existent controller, address = 17772150

Commands are Help, Boot, List, Map, Test and Wrap.
Type a command then press the RETURN key:
```

4.5.4 Disk and Tape Autoboot (Except TU58)

When disk and tape autoboot is selected, the ROM code attempts to boot software from the first available device that is present and ready with bootable software. Table 4–10 lists the devices in the order in which they are checked. The table also lists the unit numbers checked for each bootstrap.

Table 4–10: Disk and Tape Boot List

Order	Name	Unit Range	Address	Description
1	DU	0-255	172150	Disk MSCP[1] (removable media)
2	DU	0-255	172150	Disk MSCP (fixed media)
3	DU	0-255	FFA[2]	Disk MSCP (removable media)
4	DU	0-255	FFA	Disk MSCP (fixed media)
5	DL	0-3	174400	RL01/02
6	DX	0-1	177170	RX01
7	DY	0-1	177170	RX02
8	MU	0	174500	Tape MSCP
9	MS	0	172520	TSV05/TK25

[1]MSCP devices are listed in Table 4–8
[2]First floating address

If the autoboot comes to the end of the list without booting a device, it will return to the top of the list and continuously loop through the list, until successful or aborted by entering <CTRL/C>.

Note that the boot select jumpers/switch also select the language for displayed messages.

4.5.5 DPV11, DUV11, DLV11-E/F and TU58 Autoboot

When disk and tape autoboot is selected, the ROM code attempts to boot software from the first available device that is present and ready with bootable software. Table 4–11 lists the devices in the order in which they are checked. The table also lists the unit numbers checked for each bootstrap.

Table 4–11: DPV, DUV, DLV, TU58 and RK05 Autoboot List

Order	Name	Unit Range	Address	Description	
1	NP	0	Floating (rank 22)	DPV11	DECnet
2	NU	0	Floating (rank 4)	DUV11	DECnet
3	NE	0	175610	DLV11-E	DECnet
4	NF	0	175610	DLV11-F	DECnet
5	DD	0-1	176500	TU58	
6	DK	0-7	177404	RK05	

These devices are in a separate selection because they all tend to slow-down the autoboot. The DECnet boots are included because of the relatively long device-response times between attempted boots. For a similar reason, the TU58 is included in this list. The TU58 controller has the same address as SLU1 (17776500), and the autoboot does not know whether it is present until the TU58 fails to respond to a request. Because this requires a relatively long time, the TU58 is included in this list.

This selection does not include a language for messages. If the boot is successful, a message is not required. For error or status messages, the ROM code will prompt the user to select a language (see Section 4.6).

4.5.6 DEQNA Autoboot

The DEQNA is not included in either boot device list because autoboot will not terminate if there is no response over the Ethernet. It will continuously retry. If the first DEQNA is not present, or fails the citizenship test, the autoboot will try the second DEQNA. The DEQNA is selected separately and runs by itself.

This selection does not include a language for messages. If the boot is successful, a message is not required. For error or status messages, the ROM code will prompt the user to select a language (see Section 4.6).

4.5.7 Bootable Media

After the first block is read from a disk or the second record is read from a tape, the ROM code checks the first two words in memory. If the data is not correct, the ROM code generates a non-bootable media error.

The boot block is bootable if the first word (memory location 0) contains a value in the range 240_8 to 277_8, and the second word (memory location 2) contains a value in the range of 400_8 to 777_8. This definition applies to all disk and tape boots in the ROM code, but does not apply to the DECnet boots for the DPV11, DUV11, DLV11-E and the DLV11-F. These four DECnet boots are considered bootable if the value of memory location 0 is 0.

If the /O switch is used with the console mode BOOT command, the ROM code verifies only that location 0 is not 0 (HALT) before starting a secondary boot. (This does not apply to the DECnet boots for DPV11, DUV11, DLV11-E and DLV11-F.) The /O switch allows software that does not meet the bootable media standard to be booted. The switch can only be selected with the console mode BOOT command; there is no similar switch or function for the automatic-boot sequence.

4.6 LANGUAGE SELECTION

If a local language has not been selected, the ROM code will prompt the user to select a language as required for messages and commands. If version 1.0 ROMs are installed, only English and Spanish are available. If version 2.0 ROMs are installed, eight languages are available.

Certain boot selections do not include a language selection because these modes are normally automatic; messages requiring translation are not normally displayed. When <CTRL/C> is typed to abort one of these modes, the ROM code enters console mode and prompts the user to select a language, as in Example 4–20. The example shows the selected language is German.

At any time in console mode, the user can select another language by typing <CTRL/L> to display the language inquiry message and prompt.

Example 4–20: Language Inquiry

```
English         Type 1 and press the <RETURN> key
Francais        Tapez 2 et appuyez sur <RETOUR>
Deutsch         Geben Sie 3 ein und drucken Sie <WR>
Nederlands      Typ 4 en druk op <RETURN>
Svenska         Skriv 5 och tryck sedan pa <RET\TEXT>
Italiano        Introdurre 6 e premere <RITORNO>
Espanol         Presione el 7 y luego la tecla <RETORNO>
Portuguese      Escreva 8 seguido de <RETURN>

KDJ11-D/S>      3
```

4.7 TROUBLESHOOTING

The ROM code is limited to detecting errors on the KDJ11-D/S module or external Q22-bus memory. If the KDJ11-D/S is failing, the module would normally be replaced. If the problem is in external RAM, system-level diagnostics may provide further isolation. The documentation for systems with the KDJ11-D/S installed should define the system maintenance philosophy and describe system-level diagnostic programs. The message:

```
Error, see troubleshooting section in Owner's manual for assistance
```

in Examples 4–13 through 4–15, refers to such system-level documentation.

Index

A

ABORT, 3–5
Aborts note, 1–36
Access control field, 1–36, 1–42
ACF, 1–36, 1–42
ACKLTC, 3–14
ACKPF, 3–14
ACKQ, 3–24
Active page field, 1–31
Address
  Bootstrap starting address, 1–2
  Input/Output, 3–6
  Latch Enable, 3–6
  Local memory starting address, 1–2
  Modes, xii
AIO
  3:0, 3–6
  Decoder output signals
    BUS, 3–13
    BYTE, 3–13
    DEMAND, 3–13
    GP, 3–13
    IAK, 3–13
    MEMSEL, 3–13
    READ, 3–13
    RMW, 3–13
ALE, 3–6
APF, 1–31
/A switch, 4–9, 4–14
Autoboot, 4–19

B

Bank Select, 3–6
BFCTL, 3–6
BHALT, 1–21
16-bit and 18-bit memories caution, 2–16
18-bit DMA devices note, 2–16
Block number, 1–31
BN, 1–31
Bootstrap starting address, 1–2
BS 1:0, 3–6
Buffer Control, 3–6
BUS, 3–13
Bus
  Cycle, 3–8
  Loading note, 2–19
  State Machine
    Output signals
      TDIN, 3–26
      TDMG, 3–26
      TDOUT, 3–26
      TIAK, 3–26
      TRPLY, 3–26
      TSYNC, 3–26
Bypassing errors caution, 4–17
BYTE, 3–13

C

Cache, 1–2
  Control register, 1–44
  Hit/Miss register, 1–44
CCR, 1–44
CD interconnect, 2–13
CLERR, 3–29
CLK2, 3–6
Clock 2, 3–6
Clock Error, 3–29
Commands
  Console mode
    /A switch, 4–9, 4–14
    BOOT, 4–9
    CTRL/C, 4–5, 4–13, 4–14
    CTRL/D, 4–14
    CTRL/H, 4–8

Commands
Console mode (cont'd.)
CTRL/L, 4–22
CTRL/R, 4–8
CTRL/V, 4–8
HELP, 4–9
LIST, 4–11
MAP, 4–12
/O switch, 4–10
TEST, 4–13
WRAP, 4–14
Error override
CTRL/O 4, 4–17
L, 4–13, 4–17
CONT, 3–5
Continue, 3–5
Control and Status Register, 3–29
Crystal, 3–5
CSR, 3–29
CTRL/C command, 4–5, 4–13, 4–14
CTRL/D command, 4–14
CTRL/H command, 4–8
CTRL/L command, 4–22
CTRL/O 4 command, 4–17
CTRL/R command, 4–8
CTRL/V command, 4–8
CYCLE, 3–20
Cycle Decoder
Output signals
CYCLE, 3–20
ENJADD, 3–20
IOREAD, 3–20
IOWHB, 3–19
IOWLB, 3–19

D

DAL 15:0, 3–5
Data/Address Line, 3–5, 3–6
Data Valid, 3–4
DC7063, 3–1
DC7064, 3–1
DCJ11
Bus cycle, 3–8
Cycle, 3–8
Stretched, 3–9
Vector read, 3–11
I/O Bus, 3–8
Information note, 3–4
Input/Outputs
ABORT, 3–5
DAL 15:0, 3–5
Inputs
DCJ11
Inputs (cont'd.)
CONT, 3–5
DCOK, 3–4
DMR, 3–4
DV, 3–4
EVENT, 3–4
HALT, 3–4
INIT, 3–4
IRQ0, 3–4
LTC, 3–4
PARITY, 3–4
PE, 3–4
PWRF, 3–4
RDMR, 3–4
RHALT, 3–4
XTAL 1:0, 3–5
Macroinstruction, 3–8
Microcycle, 3–8
Microinstruction, 3–8
Outputs
AIO 3:0, 3–6
ALE, 3–6
BFCTL, 3–6
BS 1:0, 3–6
CLK2, 3–6
DAL 21:16, 3–6
JDAL 21:16, 3–6
MAP, 3–6
PRDC, 3–6
SCTL, 3–6
SRUN, 3–6
STRB, 3–6
DCOK, 3–4
DC power OK, 3–4
DEMAND, 3–13
Demand-read, 3–9
DEQNA note, 4–19
DF, 1–31
DIB, 1–31
Displacement
Field, 1–31
In block, 1–31
DL0, 3–18
DL0RX, 3–24
DL0TX, 3–24
DL1, 3–18
DL1RX, 3–24
DL1TX, 3–24
DLART, 1–2
DL-style UART, 1–2
DMA Request, 3–4

D space, 1–29
DV, 3–4

E

ENJADD, 3–20
ERROR, 3–29
Errors
  Bypassing caution, 4–17
Extended LSI-11 bus, xi, xii

F

FEA, 1–47
FEC, 1–47
Floating-point
  Accelerator, 1–2
  Exception address register, 1–47
  Exception code register, 1–47
  Instruction set, xii
  Status register, 1–45
FPA, 1–2
FPS, 1–45

G

Gate arrays, 3–1
General Purpose, 3–5
GP, 3–5
  Codes
    000, 3–14
    002, 3–14
    014, 3–14
    100, 3–14
    140, 3–14
    214, 3–14
  Decoder output signals
    ACKLTC, 3–14
    ACKPF, 3–14
    INIT H, 3–14
    INIT L, 3–14
    PMODE, 3–14

H

HALT/trap, 1–2

I

I/O
  Bus, 3–8
  Decoder
    Output signals
      DL0, 3–18
      DL1, 3–18
I/O
  Decoder
    Output signals (cont'd.)
      LIO, 3–17
      MAINT, 3–18
      MER, 3–17
      NR, 3–17
      QIOPAGE, 3–18
      RLTC, 3–18
  Read/Write logic
    Output signals
      LTCRD, 3–21
      LTCWLB, 3–21
      MAINTRD, 3–21
      MERRD, 3–21
      MERWT, 3–21
      NRRD, 3–21
      NRWLB, 3–21
      ZHB, 3–22
IAK, 3–13
  Decoder
    Output signals
      ACKQ, 3–24
      DL0RX, 3–24
      DL0TX, 3–24
      DL1RX, 3–24
      DL1TX, 3–24
      VHB, 3–24
INIT
  H, 3–14
  L, 3–14
Initialize, 3–4
Instruction set, xii
Interrupt Request 0, 3–4
IOREAD, 3–20
IOWHB, 3–19
IOWLB, 3–19
IRQ0, 3–4
I space, 1–29

J

JDAL 21:16, 3–6
Jumper part number, 2–1

K

Kernel
  Mode, 1–2
  Stack pointer, 1–18
KSP, 1–18

L

L command, 4–13, 4–17
Line time clock, 1–12, 3–4
LIO, 3–17
Local memory starting address, 1–2
LTC, 1–12, 3–4, 3–32
  Interrupt request, 3–32
LTCRD, 3–21
LTCWLB, 3–21

M

M7554, 1–2
Macroinstruction, 3–8
MAINT, 3–18
MAINTRD, 3–21
Management registers, 1–29
MAP, 3–6
Map Enable, 3–6
MCS, 4–2
Memory
  16-bit and 18-bit memories caution, 2–16
  Cache, 1–2
  Decoder output signals
    QSEL, 3–16
    RAMSEL, 3–16
    ROMSEL, 3–15
  Local, 1–2
  On-board, 1–2
  Q22-bus or Local Memory note, 3–26
Memory error register, 1–44
MEMSEL, 3–13
MER, 1–44, 3–17
MERRD, 3–21
MERWT, 3–21
Microcycle, 3–8
Microinstruction, 3–8
Mode
  Kernel, 1–2
  Power-up
    0,1,3, 1–2
    2, 1–2
  Protection modes, 1–2
  Supervisor, 1–2
  User, 1–2
Module part number, 1–2
MSCP, 4–18
Multinational character set, 4–2
MUX 7:0, 3–32

N

NIO, 3–8
Non-I/O, 3–8
NR, 3–17
NRRD, 3–21
NRWLB, 3–21

O

/O command switch, 4–10
ODT, xii, 1–2
On-line Debugging Technique, 1–2
Operating mode, 1–2

P

PAF, 1–31, 1–33
Page
  Address field, 1–31, 1–33
  Address register, 1–29
  Descriptor register, 1–29, 1–35
  Length field, 1–35
PAL, 3–1
PAR, 1–29
Parity
  Data In bit 16, 3–29
  Data In bit 17, 3–29
  Data Out bit 16, 3–29
  Data Out bit 17, 3–29
  Error, 3–4
  Error note, 3–29
Part number
  Jumper, 2–1
  Module, 1–2
  ROM, 4–6
PB, 3–6
PC, 1–19
PDI16, 3–29
PDI17, 3–29
PDO16, 3–29
PDO17, 3–29
PDR, 1–29, 1–35
PE, 3–4
PIRQ, 1–9
PLA, 3–1
PLF, 1–35
PLS, 3–1
PMODE, 3–14
Power Fail, 3–4
Power supply loading note, 2–19
Power-up mode
  0,1,3, 1–2
  2, 1–2
Predecode, 3–6

Prefetch Buffer, 3–6
Processor status word, 1–7
Program
  Counter, 1–19
  Interrupt request, 1–9
Programmable
  Array Logic, 3–1
  Logic Array, 3–1
  Logic Sequencer, 3–1
Programming techniques, xii
Protection modes, 1–2
PSW, 1–7
PWRF, 3–4

Q

Q22-bus, xi, xii
QIOPAGE, 3–18
QSEL, 3–16

R

RAMSEL, 3–16
RBUF, 1–14
RCSR, 1–14
RDMR, 3–4
READ, 3–13
Read/Modify/Write, 3–8
Receive
  DMA Request, 3–4
  Halt, 3–4
Receiver
  Control/status register, 1–14
  Data buffer, 1–14
Registers
  AC0:AC5, 1–48
  CCR, 1–44
  CPU error, 1–10
  DLART registers, 1–14
  Fault recovery, 1–36
  FEA, 1–47
  FEC, 1–47
  Floating point, 1–3
  FPS, 1–45
  General purpose, 1–3, 1–17
  Hit/Miss, 1–44
  KSP, 1–18
  LTC, 1–12
  Maintenance, 1–11
  Management, 1–29
  Memory
    Addressable, 1–3
    Management, 1–3, 1–32
Registers
  Memory (cont'd.)
    System, 1–3
  MER, 1–44
  MMR0, 1–36
  MMR1, 1–38
  MMR2, 1–38
  MMR3, 1–29, 1–38
  MMU status, 1–36
  Native, 1–13
  PAR, 1–33
  PC, 1–19
  PDR, 1–35
  PIRQ, 1–9
  PSW, 1–7
  R6, 1–18
  R7, 1–19
  RBUF, 1–14
  RCSR, 1–14
  Receiver
    Control/status, 1–14
    Data buffer, 1–15
  SSP, 1–18
  Stack pointer, 1–18
  System control, 1–3, 1–7
  Transmitter
    Control/status, 1–16
    Data buffer, 1–17
  User-visible, 1–3
  USP, 1–18
  XBUF, 1–14
  XCSR, 1–14
Request-read, 3–9
RHALT, 3–4
RLTC, 3–18
RMW, 3–8, 3–13
ROM, 4–1
  Code, 4–1
  Part number, 4–6
ROMSEL, 3–15
RUN indicator, 3–6

S

SCTL, 3–6
Serial Line Unit, 1–2
SLU, 1–2
SRUN, 3–6
SSP, 1–18
Stack pointer
  KSP, 1–18
  SSP, 1–18
  System, 1–18

Stack pointer (cont'd.)
USP, 1–18
STRB, 3–6
Stretch Control, 3–6
Stretched cycle, 3–9
Strobe, 3–6
Supervisor
Mode, 1–2
Stack pointer, 1–18
System stack pointer, 1–18

T

TDAL 15:0, 3–29
TDIN, 3–26
TDMG, 3–26
TDOUT, 3–26
TIAK, 3–26
Transmitter
Control/status register, 1–14
Data buffer, 1–14
TRPLY, 3–26
TSYNC, 3–26

U

UART, 1–2
Universal asynchronous receiver/transmitter, 1–2
User
Mode, 1–2
Stack pointer, 1–18
USP, 1–18

V

VHB, 3–20, 3–24

X

XBUF, 1–14
XCSR, 1–14
XTAL 1:0, 3–5

Z

ZHB, 3–22

Digital Equipment Corporation • Bedford, MA 01730