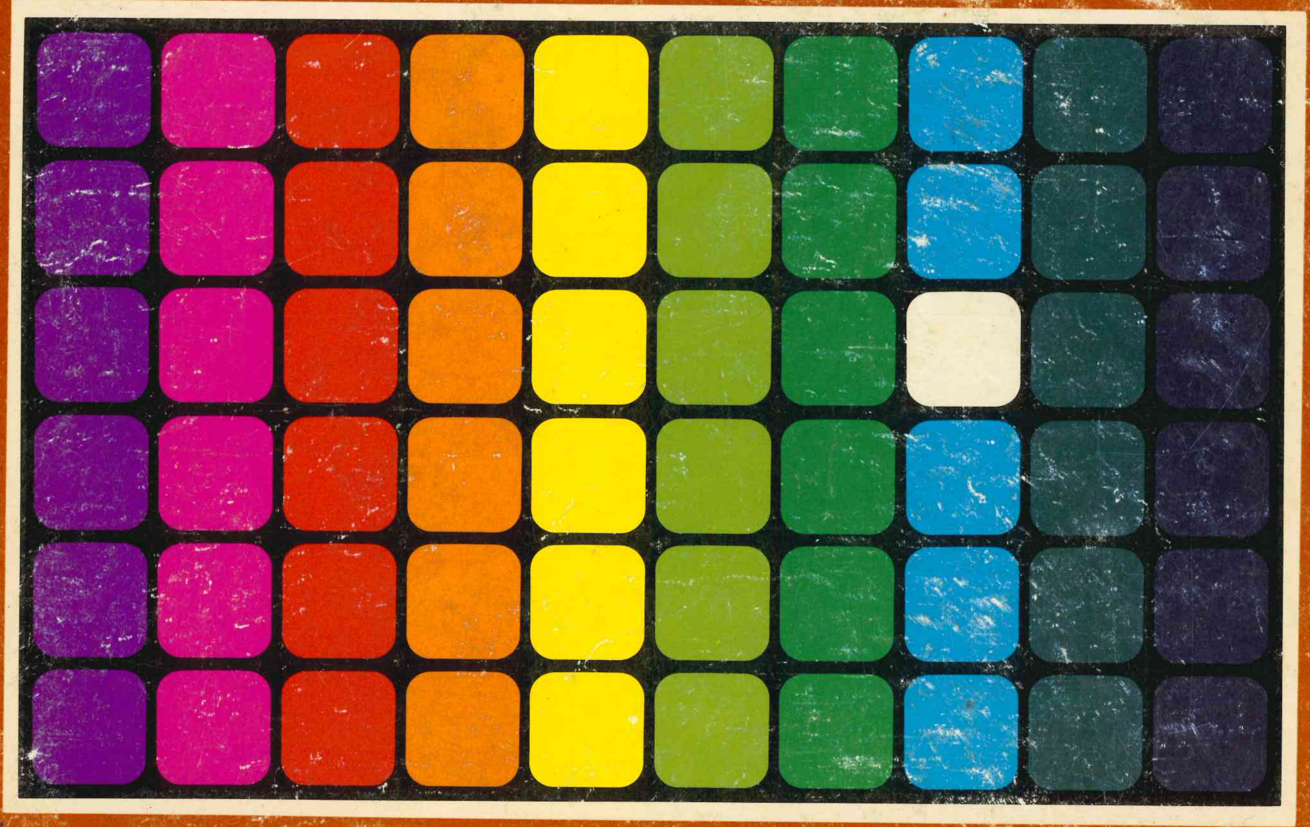
DICK BAILEY

digital

COMPANY CONFIDENTIAL

REMOTE DIAGNOSIS



KC750 MICRODIAGNOSTICS AND TECHNICAL MANUAL

EK-KC75Ø-TM-ØØ1

KC75Ø MICRODIAGNOSTICS
AND TECHNICAL MANUAL

COMPANY CONFIDENTIAL

1st Edition, October 198Ø

Printed in U.S.A.

2/81-15

CONTENTS

FIGURES

TABLES

PREFACE

This manual explains the application, installation, and function of the KC750 option. This option allows remote diagnosis of the VAX-11/750 computer system and provides it with a self-microdiagnostic capability.

A remote diagnostic module (RDM) is the primary component of the KC750 option. The RDM is a stored-program microcomputer residing on a circuit board that installs in the VAX-11/750 CPU backplane. This manual shows DIGITAL Field Service engineers how to install the KC750 option and use it to troubleshoot a VAX-11/750 system. The text includes an overview of how the RDM works, designed to aid Field Service engineers in using the KC750 option.

Chapter 1 provides an overview of the KC750 option and how it fits into DIGITAL's general diagnostic service.

Chapter 2 explains how to use the KC750 option. This includes customer responses to system problems, VAX console panel indicators, RDM operating states, and diagnostic commands.

Chapter 3 covers installation of the KC750 option. This includes connecting the RDM and modem, and verifying the installation.

Chapter 4 provides a functional description of the RDM. This helps Field Service engineers working with the RDM.

Chapters 5 through 8 describe the RDM software used to run microdiagnostics on the VAX-11/750.

CONVENTIONS

The following syntax and dialogue conventions apply to this manual.

Command Syntax Conventions

```
COM <argument-1> <argument-2> [optional] [LITERAL]
{<argument-3>/<argument-4>}
```

COM — The only characters of a command name you may type to enter the command.

<argument> — Angle brackets indicate information you must enter with the command (such as an address). Each argument is given a generic name spelled in lowercase letters, and surrounded by angle brackets. Do not type the brackets; they only designate arguments. An argument must be typed unless it is surrounded by square brackets, in which case it is optional.

{<a-3>/<a-4>} — Braces indicate that you must select one argument from a list of two or more arguments. The braces enclose the list and a slash separates the choices. Do not type the braces or slashes; they clarify the choices.

[optional] — Square brackets indicate an optional argument in the command line. Do not type the brackets; they only designate the options.

[LITERAL] — Literal arrguments appear in all uppercase letters; if entered, the argument must be typed exactly as shown.

Dialogue Conventions

This manual uses sample dialogues to demonstrate commands being described. In these dialogues, the following symbols represent terminal keys that perform special functions.

Symbol	Key(s)	Description
CTRL/C	CTRL and C	Cancels current program execution.
CTRL/U	CTRL and U	Deletes all characters typed on current line. System prints carriage return/line feed sequence; operator may retype the entire line.
CTRL/O	CTRL and O	Suppresses output to terminal.
CTRL/S	CTRL and S	Suspends output to terminal until operator types CTRL/Q.
CTRL/Q	CTRL and Q	Resumes output interrupted by CTRL/S.
<DEL>	DELETE	Deletes last character typed.
<RET>	RETURN	Serves as normal delimiter for all input to system.
CTRL/P	CTRL and P	Changes system state from program I/O to VAX console.
CTRL/D	CTRL and D	Changes system state from VAX console to RDM console.
CTRL/R	CTRL and R	Displays current command string.

NOTE

Type control characters by pressing the CTRL key in combination with another character. The combined code has special meaning to the system.

Boldface
In sample dialogues, text in **boldface** indicates data the operator types. Text not in **boldface** indicates data the computer types.

1.1 SCOPE

This chapter provides an overview of the KC750 option. It refers to applicable documents, defines the different option versions, and lists the required specifications for modems used with the option.

1.2 OPTION OVERVIEW

A fully configured KC750 option consists of a remote diagnostic module (RDM) that fits into the VAX-11/750 CPU backplane, a modem that allows the RDM to communicate with a remote operator, and cabling that connects the RDM and modem. Figure 1-1 shows how the KC750 option components relate to the VAX-11/750.

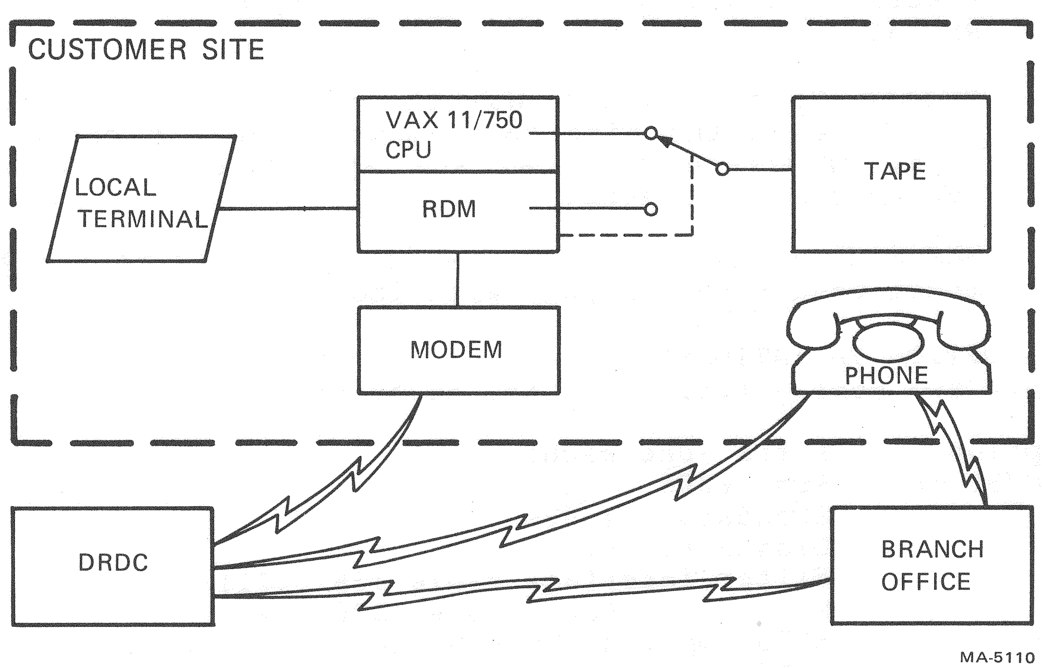


Figure 1-1 VAX-11/750 Remote Diagnosis Block Diagram

The option provides the VAX-11/750 with two diagnostic tools: remote diagnosis and microdiagnosis.

Remote diagnosis lets you connect the VAX-11/750 directly to DIGITAL's Remote Diagnostic Center (DRDC) by turning a keyswitch on the VAX console panel. This DRDC connection allows a remote engineer to log on the system and control the console the same as a local user. DRDC can then monitor system execution of user programs and run diagnostic programs on the CPU and peripherals.

A major benefit of the KC750 option is the RDM's ability to test the VAX CPU at the microlevel. This includes the ability to control execution of microdiagnostic programs on the CPU, even if the only part of the CPU that works is the clock.

Microdiagnostic programs are stored on cassette tape at the customer site. To test the CPU, the RDM reads these programs from the tape and stores them in its own memory. Under RDM control, the microdiagnostics then drive the CPU. During diagnosis, the RDM relays responses from the CPU to the console terminal, the DRDC, or both.

1.3 APPLICABLE DOCUMENTS

The following documents apply to the operation and installation of the KC750 option.

- Electronic Industries Association (EIA) -- Document No. RS-232.
- System Configuration Worksheets -- This package is available in the Site Management Guide. It contains bus maps, interface layouts, and appliques for laying out system configurations.
- Modem instruction manuals -- These are supplied by the modem vendor and shipped with individual units.

1.4 OPTION VARIATIONS

There are two configurations of the KC750 option.

Option	Parts (one each)
KC750-CA	RDM L0006 Standalone modem Diagnostic kit Filter interface cable (7016921) Modem interface cable (BC05D-25) KC750 Options User Guide (EK-KC750-UG) KC750 Options Installation Guide (EK-KC750-IN)
KC750-DA	RDM L0006 Diagnostic kit Filter interface cable (7016921) KC750 Options User Guide (EK-KC750-UG) KC750 Options Installation Guide (EK-KC750-IN)

KC75Ø-CA is the usual configuration for VAX-11/75Øs. KC75Ø-DA applies to VAX-11/75Ø installations outside the US (where local governments provide the modem).

1.5 MODEM SPECIFICATIONS

Modems used with KC75Ø options must conform to certain specifications and function states. They must be built in accordance with CCITT Standard V24 and EIA Standard RS-232C. The modem type is low speed asynchronous with a transfer rate of 3ØØ baud. Appendix E gives detailed modem specifications.

1.6 SELECTABLE OPTION STATES REQUIRED

Certain modems, such as the Racal/Vadic and General DataCom (GDC), already conform to the following list of required states when shipped. (See Note.) Any other modem used with this option must be configured to the function states listed. Refer to the instruction manual of the modem vendor to configure its functions. (Outside the US, refer to the local registration number established for modems dedicated to the RDM. The customer uses this number to order the modem.)

Modem Option	Required State
Originate/Answer	ON
Automatic/Answer	ON
Data Set Ready	ON
Loss of Carrier Disconnect	OFF
Abort Timer Disconnect 17 s (min) to 3Ø s (max)	ON
Receive Space Disconnect	OFF
Send Space Disconnect	OFF
Transmit Reversals in Manual Analog Loop	OFF
Answer Mode Indication	OFF
Early Data Set Ready	ON
Make Busy (CN Circuit)	OFF
Fail Safe State of CN	OFF
Common Ringer	OFF
RTS Control	DTE mode
DTR Control	DTE mode
Analog Loop	DTE mode
Disconnect (Unattended)	DTE mode
CB-CF (COM/SEP)	Separate
Grounding AA/AM COM/SEP	Common
Remote Telephone Operator (REM OPR)	Remote

NOTE

DIGITAL DFØ2 modems are also shipped with some KC75Ø options; however, these modems may not have the proper jumper configuration. If you receive a DFØ2 modem, read Appendix B to verify the modem has the right jumper settings.

CHAPTER 2
OPERATING PROCEDURES

2.1 SCOPE

This chapter explains how to use the KC750 option to diagnose the VAX-11/750. The chapter contains two main sections corresponding to the two diagnostic approaches used by the RDM. Paragraph 2.3 explains how to use the KC750 option to perform remote diagnosis. Paragraph 2.4 explains how to use the RDM command set to perform microdiagnosis.

Remote diagnosis is usually the first procedure used to test the system, since KC750 option customers make their initial request for service through the DIGITAL Remote Diagnosis Center (DRDC). If the CPU is working, the DRDC logs on the customer system and tests the CPU and peripherals with diagnostics. If the CPU is not working, the DRDC may use the RDM command set to test the CPU. Should remote diagnosis fail, an on-site technician may need to use the RDM (along with other tools) to test the system.

2.2 KEYSWITCH AND FRONT PANEL INDICATORS

Using the RDM requires a knowledge of the VAX-11/750 keyswitch and front panel indicators. The keyswitch determines various operating states; the front panel indicators show what the RDM may do at a given moment. Figure 2-1 shows the front panel console with keyswitch and indicators. Table 2-1 describes the various keyswitch positions. Table 2-2 describes the front panel indicators that apply to the KC750 option.

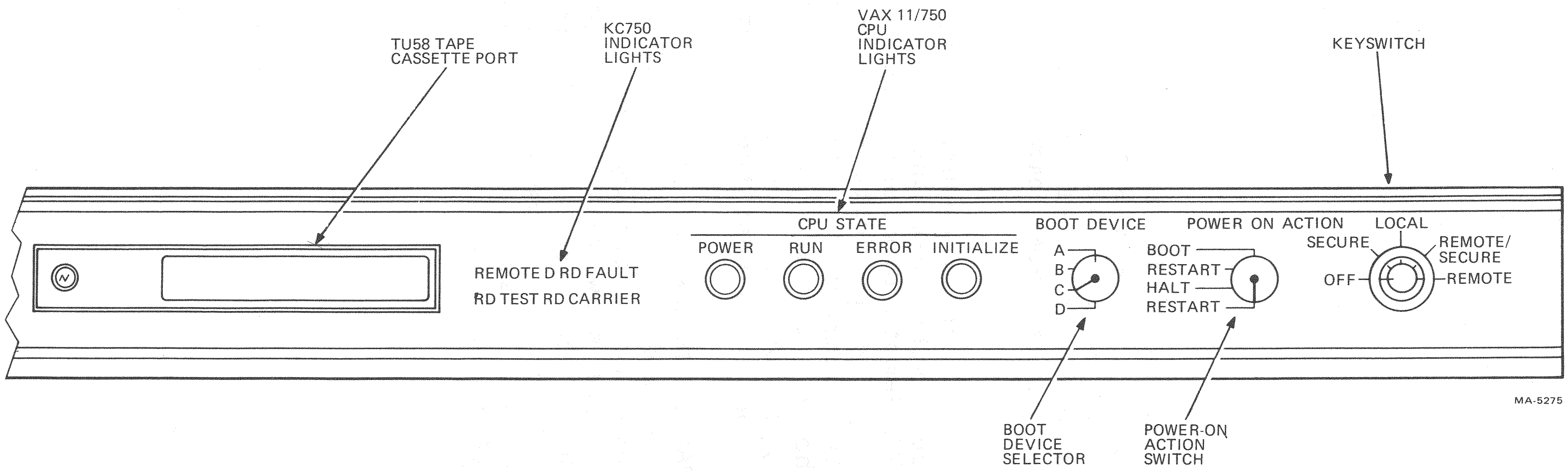


Figure 2-1 VAX-11/75Ø Front Panel

Table 2-1 VAX-11/750 Keyswitch Positions

Position	Description
LOCAL SECURE	System only responds to local terminal. Program I/O state enforced. This is SECURE position on keyswitch.
LOCAL	System only responds to local terminal. System responds to CTRL/D and CTRL/P to change states.
REMOTE SECURE	System only responds to remote terminal. Program I/O state enforced, except that remote terminal can make system enter talk state.
REMOTE	System only responds to remote terminal. System responds to CTRL/D and CTRL/P to change states.

Table 2-2 KC750 Option Front Panel Indicators

Indicator	Description
REMOTE D	Indicates keyswitch is in REMOTE SECURE position.
REMOTE	Indicates keyswitch is in REMOTE position.
RD FAULT	Indicates RDM logic failure. Fault indicator should come on for about ten seconds during console power-up as part of logic self-test.
RD TEST	Indicates DRDC host computer is performing diagnostic tests.
RD CARRIER	Indicates carrier signal detected from DRDC.

2.3 REMOTE DIAGNOSIS

To perform remote diagnosis on a VAX-11/750 equipped with the KC750 option, phone the DIGITAL Remote Diagnosis Center (DRDC) at one of the following numbers.

800-525-6570	Outside Colorado (US only)
800-332-7189	Inside Colorado

Outside the US, ask the local DIGITAL branch office for the procedure to contact the DRDC.

When calling, be prepared to give the following information.

- Customer name (company)
- Caller's name and phone number
- Address and location of system
- System type and serial number
- Description of problem
- Device and device unit number on which diagnostic media will be mounted

If the DRDC decides to run a remote diagnostic session, it requests that you remove all disks (except the system/diagnostic disk) and tapes from the system. Then you must follow this procedure.

1. Mount the diagnostic media and a scratch disk.

2. Turn the POWER ON ACTION switch to HALT.

3. Turn the keyswitch to REMOTE; this transfers system control to the DRDC.

2.4 MICRODIAGNOSIS

There are two ways to use the RDM to diagnose the VAX-11/750 at the microlevel. First, you may use RDM commands to perform particular actions, such as examining CPU memory locations or stepping through a particular microinstruction. Second, you may use RDM commands to load microdiagnostic programs into the RDM, and have the RDM drive the VAX CPU control lines.

Many factors determine which method to use. Usually Field Service technicians run a microdiagnostic first. If that fails to run or produces unsatisfactory information, they resort to RDM commands. The following paragraphs explain how to access and use RDM commands, including those that load and run microdiagnostics. Chapters 5 through 8 describe in detail how to use microdiagnostics to test the VAX-11/750.

2.4.1 Entering Diagnostic Commands

The VAX-11/750 must be powered up and in the proper operating state to enter RDM diagnostic commands. Power is applied to the system when you turn the keyswitch from OFF to another position. The VAX-11/750 powers up in either the VAX console state or the program I/O state, depending on the position of the front panel switches. To test the VAX-11/750 from the local terminal, turn the keyswitch to LOCAL. On power-up, the RDM runs an initialization routine that tests RDM logic. Table 2-3 describes RDM power-up self-test activities.

Initialization causes the fault indicator to turn on and then off. A fault exists in the RDM if the fault indicator does not turn on, or turns on but not off. The RDM may continue to function despite a fault; however, you should not use the RDM when a fault is suspected.

Table 2-3 RDM Power-Up Activities

Activity	Description
ROM Test	Performs a checksum calculation on the set of four RDM ROM chips. Compares results to a hard-coded character located in the last position of each memory chip. Any mismatch between the calculated and hard-coded character sends an error message (ERR - ROM) to the active terminal, and the fault indicator remains on.
RAM Test	Writes, reads back, and verifies a set of bit patterns for each memory location. Any mismatch sends an error message (ERR - RAM) to the active terminal and the fault indicator remains on.

The current system operating state determines which VAX and RDM commands the system recognizes. Table 2-4 describes the operating states available. Figure 2-2 shows how to get from one state to another. RDM commands that test the VAX-11/750 are recognized only if entered while the system is in the command mode of the RDM console state.

Table 2-4 System Operating States

State	Description
Program I/O	This state is normal operating state of the VAX-11/750. CPU is under macro program control.
VAX Console	This state places CPU under the control of its own console microcode. It supports CPU console commands. Active terminal displays console prompt (>>>).
RDM Console Control Mode	This state is the same as program I/O state if CPU is running, except that CTRL/D changes system to RDM console state command mode; running program supplies terminal prompt. If CPU is halted, this state is the same as VAX console state where VAX console prompt (>>>) is disabled.
RDM Console Command Mode	Under this state, system recognizes RDM commands only. (See Paragraph 2.3.3 for a list of available commands.) RDM prompt (RDM>) displayed. Character output from the CPU to the terminal is disabled.

Table 2-4 System Operating States (Cont)

State	Description
Talk	This state allows direct communication between an operator at the local terminal and an operator at the DRDC. See Paragraph 2.3.3.26 for further discussion of the talk state.
Micro-diagnostic	This state places RDM under control of the microdiagnostic monitor (MICMON). Microdiagnostic prompt (MIC>) displayed.

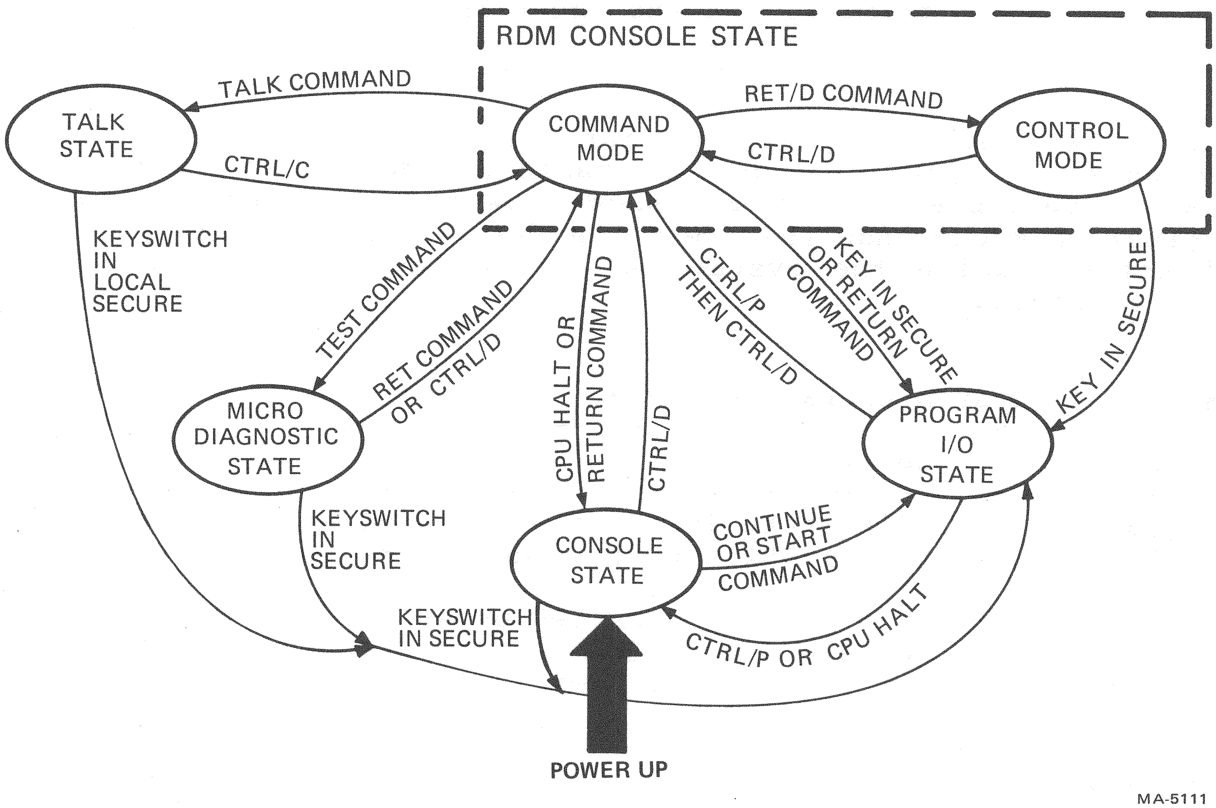


Figure 2-2 RDM Operating State Transitions

To enter diagnostic commands at the local terminal, turn the keyswitch to LOCAL. If the CPU is running (under program control), type CTRL/P followed immediately by CTRL/D . This takes the system from the program I/O state to the command mode of the RDM console state. If the CPU is halted (program execution), the system is in the VAX console state; type a CTRL/D to invoke the command mode of the RDM console state. Next, type the desired diagnostic command.

NOTE

If you use an LA120 terminal with the RDM, set the terminal's auto-disconnect feature to OFF. Otherwise, CTRL/D will not invoke the command mode of the RDM console state.

Example

The keyswitch is in LOCAL position, and CPU status is halted (program not running, but CPU clock working). The operator wants to execute a trace of CPU control store addresses. The system leaves console state and enters RDM console state; then the operator gives diagnostic commands to stop the CPU clock and do a trace.

```
>>>  CTRL/D            (>>> is console state prompt.)

RDM> STO<RET>          (RDM> is RDM console state prompt.)

RDM> TR<RET>
```

2.4.2 RDM Commands

Table 2-5 lists the RDM commands. The rest of this chapter describes these commands in alphabetical order.

Table 2-5 RDM Command Set

Command	Syntax	Function
Clear	CL	Clear stop-on-micromatch.
Copy	COP	Copy to local terminal.
Copy-Disable	COP/D	Disable local copy function.
Deposit	D [/modifier][address]<data>	Deposit to VAX memory location.
Examine	E [/modifier][address]	Examine VAX memory location.
Examine-Console	E/C [address]	Examine RDM status registers.
Initialize	INI	Initialize.
Link	LIN	Enter link control file.
Load	LO <file name.ext>[address]	Load TU58 file to VAX memory.
Local	CTL	Enable local copy control.
Local-Disable	CTL/D	Disable local copy.
Microaddress	UA <address>	Load CS address bus.
Microaddress/C	UA/C <address>	Load CS address bus until next M clock only.
Parity	PAR <address>	Run control store parity check.
Perform	PER	Perform link control files.
Repeat-Last-Command	REP	Repeat console command.
Repeat-Next-Command	R <command>	Repeat following command.
Return	RET	Return to program I/O state.
Return/D	RET/D	Return to RDM control mode.
Set	SE [address]	Set stop-on-micromatch.
Show	SH	Show CPU state.
Show-Version	SH/V	Show current version of RDM firmware.
Step	STE	Step through single microinstruction.
Step-Tick	STE/T	Step through single clock tick.
Stop	STO	Stop clock.
Talk	TA	Enter talk state.
Test	TE	Load and run microdiagnostics.
Test-Com	TE/C	Load microdiagnostic and await command.
Test-File	TE <file name.ext>	Load and run user RDM program.
Trace	TR	Display trace of CS address.

2.4.2.1 Clear

Purpose: Clears stop-on-micromatch function.

Syntax: CL

This command disables the stop-on-micromatch function (enabled by Set command), but does not change the contents of the match register. This allows you to sync test equipment on the match without stopping the micromachine. Refer to Appendix A for backplane test points.

Sample Dialogue:

```
RDM> CL<RET>

RDM>
```

2.4.2.2 Copy

Purpose: Enables local copy function.

Syntax: COP

This command allows the local terminal to receive a copy of the dialogue between the RDM and the DRDC. Only the DRDC can enter COP.

Sample Dialogue:

```
RDM> COP<RET>

RDM>
```

2.4.2.3 Copy Disable

Purpose: Disables local copy function.

Syntax: COP/D

This command stops the local terminal from printing DRDC dialogue.

Sample Dialogue:

```
RDM> COP/D<RET>

RDM>
```

2.4.2.4 Deposit

Purpose: Deposits data to a VAX memory location.

Syntax: D [/modifier] <address> <data>

This command allows you to write hexadecimal data to a hexadecimal VAX memory address specified in the command string. Modifiers (/W for word, /L for long word, and /B for byte) define the data type to be deposited. When you use a particular modifier, it becomes the default data type. If you omit the modifier, then the current default data type is assumed.

If you use a /N modifier, the address field does not appear in the command; the last location where data was deposited is incremented by the default data type and used as the default address.

You may replace the address field with an asterisk (*) or a plus sign (+). An asterisk causes the RDM to use the address referenced in the last Examine or Deposit command. A plus sign (+) increments the address by one (as in the /N modifier).

You may deposit the same data to blocks of successive addresses by using the Repeat command syntax (for example, R D + <data>).

Sample Dialogue:

Example 1

RDM> **D/L Ø 11223344<RET>**	Long word 11223344 deposited to address zero.
RDM>	

Example 2

RDM> **D/W 2 6677<RET>**	Word 6677 deposited to address 2.
RDM> **E/L Ø <RET>**	Long word examined at location Ø.
P ØØØØØØ 66773344	P indicates physical address.
RDM>	

Example 3

RDM> **D/B 3 88<RET>**	Byte 88 deposited to address 3.
RDM> **E/L Ø<RET>**	Long word examined at location Ø.
P ØØØØØØ 88773344	
RDM>	

Example 4

RDM> D Ø 1Ø1Ø1Ø1Ø<RET>	Long word 1Ø1Ø1Ø1Ø deposited to address Ø.
RDM> E *<RET>	Long word examined at address Ø.
P ØØØØØØ 1Ø1Ø1Ø1Ø	
RDM> D/B * 11<RET>	Byte 11 deposited to address Ø.
RDM> D + 22<RET>	Byte 22 deposited to address 1.
RDM> D + 33<RET>	Byte 33 deposited to address 2.
RDM> D + 44<RET>	Byte 44 deposited to address 3.
RDM> E/L Ø<RET>	Long word examined at address Ø.
P ØØØØØØ 44332211	
RDM>	

2.4.2.5 Examine

Purpose: Examines data at a VAX memory address.

Syntax: E [/modifier] [address]

This command allows you to read data located at a hexadecimal VAX memory address specified in the command string. Modifiers (/W for word, /L for long word, and /B for byte) define the data type to be examined. When you use a particular modifier, it becomes the default data type. If you omit the modifier, then the current default data type is assumed.

If you omit the address field in the command, the last memory location examined is incremented by the specified or default data type and used as the default address. You may replace the address field with an asterisk (*) to use the address referenced in the last Examine or Deposit command.

You may examine blocks of successive addresses by using the Repeat command syntax (for example, R E).

Sample Dialogue:

Example 1

RDM> **E/L Ø<RET>**	Long word examined at memory location Ø.
P ØØØØØØ 12345678	P indicates physical address; ØØØØØØ is address examined, and 12345678 is data examined.
RDM>	

Example 2

RDM> **E<RET>**	Address incremented by default data type (long word) and examined.
P ØØØØØ4 9ABCDEFØ	
RDM>	

Example 3

RDM> **E/W Ø<RET>**	Word examined at address Ø.
P ØØØØØØ 5678	
RDM>	

Example 4

RDM> **E<RET>**	Address incremented by default data type (word) and examined.
P ØØØØØ2 1234	

Example 5

RDM> **E/B 1<RET>**	Byte examined at address 1.
P ØØØØ1 56	
RDM> **E *<RET>**	Byte examined at address 1.
P ØØØØØ1 56	
RDM>	

2.4.2.6 Examine Console

Purpose: Inspects data at RDM random access memory (RAM) or control register address.

Syntax: E/C [address]

This command displays the contents of the RDM status registers and RAM addresses. If you do not specify an address, the current default address is incremented and examined. The default address is the one used in the previous Examine Console command. When using this command, you may only inspect data at addresses within the hexadecimal range 8000 to 8FF8.

Sample Dialogue:

RDM> **E/C F820<RET>**	RDM status register F820 examined.
R F820 DF	R indicates RDM RAM or status register;
RDM>	F820 is address examined, and DF is contents of examined register in hexadecimal.

2.4.2.7 Initialize

Purpose: Initializes CPU.

Syntax: INI

This command allows the local user or DRDC to simulate a power-fail sequence in order to recover the CPU from a hung condition. The command asserts the CPU's ac low signal, followed by dc low five milliseconds later. This sequence initializes the CPU to a newly powered-up state; however, main memory is not swept clean, to ensure the presence of a good error correction code (ECC).

You may use Initialize with Repeat Next Command (example 2) to generate a scope loop of power-fail activity. This might help in dealing with CPU power-fail and recovery problems.

Sample Dialogue:

Example 1

```
RDM> INI<RET>

RDM>
```

Example 2

```
RDM> R INI<RET>

RDM>
```

2.4.2.8 Link

Purpose: Builds executable control files in RDM RAM.

Syntax: LIN

This command allows you to build a list of instructions in RDM RAM that you can execute with the Perform command. After each instruction typed into the link list, the RDM prompts the next instruction with LNK>. When you have no further instructions to enter in the link list, type a CTRL/C in response to the LNK> prompt. If the last link list instruction is Perform, the link program will loop continuously when run until you type CTRL/C.

NOTE

A link list is destroyed by power-down or by any of the following commands: Return, Return/D, Test, Test Command, or Test File.

Sample Dialogue:

Refer to the sample dialogue provided in Paragraph 2.4.2.15 (Perform).

2.4.2.9 Load

Purpose: Loads files from TU58 tape to VAX main memory.

Syntax: LO <file name.extension> [address]

This command causes the RDM to locate a specific file on the TU58, read it from tape, and deposit it at sequential addresses in VAX main memory. The specified address is the starting address in VAX memory of the deposited file. If you do not specify an address, the default address is zero. The file name must have exactly six characters. The extension must have exactly three characters.

The Load command allows you to load files from the TU58 when the CPU cannot bootstrap them. This happens when the CPU is not functional enough to bootstrap, or the desired file is not properly hooked to an RT-11 boot block. In either case, the RDM needs only the CPU base clock to achieve the load.

You may use this command to load diagnostics such as EVKAA, in order to get useful error information about a failing CPU.

The command also enables a remote diagnostic script to load the diagnostic supervisor and a level 4 diagnostic from the same tape cartridge. This would otherwise be impossible, since only one of those files can be hooked to the boot block and therefore booted.

Sample Dialogue:

Example 1

RDM> **LO ECKAL.EXE<RET>**	Diagnostic ECKAL loaded into VAX main memory starting at address 0.
RDM>	

Example 2

RDM> **LO EVKAA.EXE 10000<RET>**	Diagnostic EVKAA loaded into VAX main memory starting at address 10000.
RDM>	

Example 3

RDM> **LO ECSAA.EXE FE00<RET>**	Diagnostic ECSAA loaded into VAX main memory starting at address FE00.
RDM>	

2.4.2.1Ø Local

Purpose: Provides dual local/remote control of RDM.

Syntax: CTL

This command allows the local user or DRDC to communicate with and control the RDM. You may enter this command only from the DRDC.

CAUTION

When both local and remote terminals communicate with the RDM, do not enter data from both terminals at the same time. Otherwise command strings could become mixed.

Sample Dialogue:

```
RDM> CTL<RET>

RDM>
```

2.4.2.11 Local Disable

Purpose: Disables Local command function.

Syntax: CTL/D

This command cancels the RDM's ability to accept commands from or send copy to the local terminal during remote diagnosis. You may enter this command only from the DRDC.

Sample Dialogue:

```
RDM> CTL/D<RET>

RDM>
```

2.4.2.12 Microaddress

Purpose: Latches contents of specific microaddress into CPU control store latches.

Syntax: UA <address>

This command halts the CPU and allows you to load a selected address on the control store (CS) address bus. When you restart the CPU, the new address takes control and the CPU checks the parity of the addressed data.

> **NOTE**
> **When you use this command, you disable any stop-on-micromatch function previously set.**

You may use this command to perform the following functions.

1. Apply the contents of an address with bad parity to the CPU logic checkers. Then find the bad bits statically with a scope, logic probe, or other test equipment.

2. Perform a scope loop of a given microaddress at RDM execution speed. Do this by specifying Repeat execution (Paragraph 2.4.2.17) of the command and typing CTRL/O to suppress terminal output.

3. Start the micromachine at a desired address.

> **NOTE**
> **To restart the CPU at the new address, use the Continue command.**

Sample Dialogue:

```
RDM> UA 17FD<RET>

     CLK STOPPED CSAD 17FD    NEXT 0020

RDM>
```

2.4.2.13 Microaddress/C

Purpose: Temporarily loads a specific microaddress on the control store (CS) address bus.

Syntax: UA/C <address>

This command places a selected address on the CS address lines until the next master (M) clock tick. Data from the specified address stays on the CS bus until the next CPU clock tick. At that point, the CPU latches data on the CS bus prior to the command. This command, therefore, does not change the program flow in the CPU.

NOTE

When you use this command, you disable any stop-on-micromatch function previously set.

You may use this command to isolate a failure in the CPU CS latching mechanism to one of the following areas.

- CS bus
- Control store
- CS latch
- CS control signal

Sample Dialogue:

```
RDM> UA/C 17FD<RET>

RDM>
```

2.4.2.14 Parity

Purpose: Checks VAX control store (CS) parity.

Syntax: PAR <address>

This command runs a parity check of the CPU control store. The parity check starts at the selected address and ends at the first discovered parity error. If no parity error is found, the parity check stops at preset error address 17FD. The command displays the location of any parity error found. You must stop the CPU clock before you use this command.

Sample Dialogue:

RDM> **PAR Ø<RET>**	CPU stopped and parity check run from address Ø.
PARITY ERROR CSAD ØAD1	Parity error found at address ØAD1.
RDM>	

NOTE

You may include the writable control store (WCS) in the parity check by specifying hexadecimal addresses 2ØØØ through 24ØØ. You must load the WCS after power-up to avoid apparent errors that would result from power-up. The WCS is an option on the VAX-11/75Ø.

2.4.2.15 Perform

Purpose: Executes a link program.

Syntax: PER

This command executes the program in the link file until the program stops or you type a CTRL/C.

Sample Dialogue:

```
RDM> LIN<RET>             User builds a link
                          program that deposits
LNK> D/B Ø 11<RET>        data to successive
                          bytes in the
LNK> D + 22<RET>          long word
                          at address Ø,
LNK> D + 33<RET>          and then
                          examines the long
LNK> D + 44<RET>          word at address
                          Ø.  The last instruction
LNK> E/L Ø<RET>           is Perform, to make
                          the program loop
LNK> PER<RET>             until a CTRL/C is
                          typed.
LNK> CTRL/C               CTRL/C terminates building
                          of link list.

RDM> PER<RET>             Link list performed
                          continuously until
     D/B Ø 11             CTRL/C.

     D + 22

     D + 33

     D + 44

     E/L Ø

     P ØØØØØØ 44332211    P indicates physical address.

     PER

     D/B Ø 11

     D + 22

     D + 33

     D + 44

     E/L Ø
```

```
     CTRL/C

     P ØØØØØØ 44332211

     ^C

RDM>
```

2.4.2.16 Repeat Last Command

Purpose: Repeats execution of the last RDM command.

Syntax: REP

This command continuously executes the preceding command until you type CTRL/C. It helps generate scope signals off the continuous execution of selected commands. When using Repeat, type CTRL/O to stop terminal output. Repeat also helps examine or deposit to blocks of memory area. (See Paragraphs 2.4.2.4 and 2.4.2.5.)

Sample Dialogue:

RDM> **E Ø<RET>**	Location Ø examined.
P ØØØØØØ 12345678	P indicates physical address.
RDM> **REP<RET>**	Location Ø examined continuously until CTRL/C typed.
P ØØØØØØ 12345678 P ØØØØØØ 12345678	
CTRL/C ^C	
RDM>	

2.4.2.17 Repeat Next Command

Purpose: Repeats execution of selected RDM command.

Syntax: R <command>

This command continuously executes a given command until you type CTRL/C. It helps generate scope signals off the continuous execution of selected commands. When using Repeat, type CTRL/O to stop terminal output. Repeat also helps examine or deposit to blocks of memory area. (See Paragraphs 2.4.2.4 and 2.4.2.5.)

Sample Dialogue:

RDM> **R E Ø<RET>**	Location Ø continuously examined until CTRL/C typed.
P ØØØØØØ 12345678 P ØØØØØØ 12345678 P ØØØØØØ 12345678	P indicates physical address.
CTRL/C ^C	
RDM>	

2.4.2.18 Return

Purpose: Returns system to program I/O state.

Syntax: RET

This command switches the system from RDM console state to program I/O state.

NOTE

When you use this command, you disable any stop-on-micromatch function previously set.

Sample Dialogue:

```
RDM> RET<RET>
$
```

2.4.2.19 Return/D

Purpose: Returns system to the control mode of RDM console state.

Syntax: RET/D

This command allows you to switch the system from the command mode to the control mode of RDM console state. In control mode, the system is at the prompt level of the program currently running in the CPU. However, unlike program I/O state, a CTRL/D returns the system to the RDM console command mode whether the CPU is running or not. Any previously set stop-on-micromatch function remains enabled after you type the RET/D command.

Sample Dialogue:

```
RDM> RET/D<RET>
$
```

2.4.2.20 Set

Purpose: Enables stop-on-micromatch function.

Syntax: SE [address]

This command stops the CPU clock when the contents of the control store (CS) address bus equal the contents of the RDM match register. You may specify the address to match by entering an address in the command syntax. If you omit an address in the command syntax, the CPU stops at the address which matches the one already contained in the match register. When the CPU stops, the RDM displays the addresses of both the last instruction executed and the next instruction to execute.

Use Set to stop the CPU at a particular address from which to do a trace of the VAX control store. This allows you to see what path the CPU took to reach that point in the control store.

Sample Dialogue:

```
RDM> SE 3FE<RET>

     CPU STOPPED CSAD Ø3FE    NEXT Ø3FF

RDM>
```

2.4.2.21 Show

Purpose: Displays current CPU state.

Syntax: SH

This command displays the current operating state of the CPU: running, halted, or stopped. Running means the CPU is executing code. Halted means the CPU is not executing code, but its clock is running. Stopped means the CPU clock is stopped. The command also displays the addresses of both the last instruction executed and the next instruction to execute.

Sample Dialogue:

```
RDM> SH<RET>

     CPU STOPPED CSAD 3402    NEXT 3403

RDM>
```

2.4.2.22 Show Version

Purpose: Displays version and date of RDM firmware.

Syntax: SH/V

This command displays the current revision level and date of the firmware running in the RDM. If your procedure depends on the revision of the RDM you have, ask the RDM for the version number.

Sample Dialogue:

```
RDM> SH/V<RET>

13-MAR-8Ø RDM LV=1Ø

RDM>
```

2.4.2.23 Step

Purpose: Steps CPU through single microinstruction.

Syntax: STE

This command causes the CPU to execute a single microinstruction and stop. It then displays the address (CSAD) of the current microinstruction latched in the CPU and the address of the next microinstruction to latch.

Sample Dialogue:

```
RDM> STE<RET>

     CSAD 1FØ2    NEXT 1FØ3

RDM>
```

2.4.2.24 Step Tick

Purpose: Advances CPU in base (B) clock increments.

Syntax: STE/T

This command advances the CPU by one B clock tick, instead of one master (M) clock tick as in the Step command. You may use Step Tick by itself, or with Repeat Next Command after you stop the CPU clock.

Step Tick displays the address (CSAD) whose contents are currently latched in the CPU; these contents currently control the CPU. It also displays the address of the data the CPU will latch on the next M clock tick.

This command allows inspection of CPU activity within a single M clock (or CPU latching) cycle.

The relationship between the B clock and M clock of the CPU is as follows. One M tick nominally occurs for every three B ticks. A different action takes place on each of the three B ticks to prepare for the next M tick.

On the first B tick, the eight HI NEXT bits in the control store address <13:Ø6> are terminated to read back as ones. CSAD bit 13 is a special case; it is jumpered to ground on the backplane and read back as zero unless the grounding jumper has been removed (for example, after WCS installation).

On the second B tick, the microinstruction located at the NEXT address moves to the CS bus.

On the third B tick (the next M tick), the CPU latches the contents of the CS bus and the cycle begins again.

Note in example 2 that it takes three B ticks for a new address (IFE9) to move on and off the CS address bus.

Use Step Tick to diagnose problems in the VAX-11/75Ø system that are synchronized with the system base clock. In combination with other RDM commands, you may establish a position within the microexecution of a function and then advance through the execution of that function in B clock increments.

Sample Dialogue:

Example 1

```
ROM> STE/T<RET>

     CSAD 17DB          NEXT IFE9

RDM>
```

Example 2

```
RDM> R STE/T<RET>

     CSAD 17DB          NEXT IFE9
     CSAD 17DB          NEXT IFE9
     CSAD IFE9          NEXT 17DF
     CSAD IFE9          NEXT 17DF
     CSAD IFE9          NEXT 17DF
     CSAD 17DF          NEXT IFEA

CTRL/C  ^C

RDM>
```

2.4.2.25 Stop

Purpose: Stops CPU clock.

Syntax: STO

This command stops the CPU clock, then displays the addresses of both the last instruction executed and the next instruction to execute.

Sample Dialogue:

```
RDM> STO<RET>

CPU  STOPPED CSAD 2F13        NEXT 2F14

RDM>
```

2.4.2.26 Talk

Purpose: Changes system from RDM console state to talk state.

Syntax: TA

This command allows communication between local and remote terminals. You enter it from the terminal currently controlling the VAX-11/750 system. The Talk command displays characters typed at either terminal on both terminals.

If you enter the talk state with the keyswitch turned to LOCAL, the system asserts DTR to the modem and awaits a telephone link to DRDC. Completion of the DRDC link establishes a talk state.

If you enter the talk state with the keyswitch turned to REMOTE or REMOTE SECURE, the command immediately places the system in the talk state.

Type a CTRL/C to return from talk state to RDM console state. To return from talk state to program I/O state, turn the keyswitch to LOCAL SECURE.

Sample Dialogue:

RDM> **TA<RET>**

ENTER TALK MODE

RDM>

2.4.2.27 Test

Purpose: Loads and runs microdiagnostics.

Syntax: TE

This command loads the microdiagnostic monitor into RDM RAM, and the monitor runs a series of microdiagnostics on the VAX-11/750 system. The monitor (MICMON) and microdiagnostics reside on TU58 tape cassettes. You must insert the right cassette before using Test.

MICMON cancels the tests if it detects an error or you type CTRL/C while the tests are running. The monitor then prints an error message and gives the MIC> prompt on a new line. The MIC> prompt means you are communicating with MICMON in the microdiagnostic state.

If you type CTRL/C while the RDM is loading MICMON into RAM, the RDM stops loading MICMON and returns to RDM console command mode.

If the microdiagnostics run to completion, the RDM automatically returns to RDM console command mode. If tests have not run to completion, perform one of the following procedures to exit the microdiagnostic state.

- Turn the keyswitch to LOCAL SECURE or REMOTE SECURE to return to the program I/O state.
- Type CTRL/D to return to RDM console command mode.

Refer to Chapter 5 for information and sample dialogues on how to apply the Test command.

2.4.2.28 Test Command

Purpose: Loads microdiagnostic monitor.

Syntax: TE/C

This command loads the microdiagnostic monitor (MICMON) into RDM RAM, and gives the monitor control of the RDM. With Test Command (unlike Test), MICMON waits for terminal commands before running any microdiagnostics. With MICMON in RAM, the system enters the microdiagnostic state and the active terminal displays the MIC> prompt.

MICMON and the microdiagnostics reside on TU58 tape cassettes. You must insert the right cassette before using Test Command.

If you type a CTRL/C while the ROM is loading MICMON into RAM, the RDM stops loading MICMON and returns to RDM console command mode.

To exit the microdiagnostic state, perform one of the following procedures.

- Turn the keyswitch to LOCAL SECURE or REMOTE SECURE to return to the program I/O state.

- Type CTRL/D to return to RDM console command mode.

Refer to Chapter 6 for information and sample dialogues on how to apply Test Command.

2.4.2.29 Test File

Purpose: Loads user program into RDM RAM and runs it.

Syntax: TE <file name.extension>

This command allows you to load programs other than the microdiagnostic monitor into RDM RAM and execute them. The file name in the command argument must consist of exactly six characters. The extension must consist of exactly three characters.

Sample Dialogue:

```
RDM> TE MICMON.TST<RET>

RDM>
```

2.4.2.30 Trace

Purpose: Displays the 65 most current control store (CS) addresses.

Syntax: TR

This command displays in reverse order the CS addresses stored in the RDM diagnostic control store (DCS). Except in the microdiagnostic state, the RDM stores the addresses of the 64 most recently executed VAX CS instructions in its DCS. The address of the next instruction to execute is stored in a DCS trap register and also displayed.

Type a CTRL/C or CTRL/D to stop the display of the 65 stored addresses at any point. Either command returns you to the RDM> prompt at once.

The CPU clock must be stopped before you use the Trace command. You may use Trace to see what path the CPU took just before its clock stopped.

Sample Dialogue:

```
     RDM> TR<RET>

                CSAD 2391      NEXT IF2F
                CSAD 4234
                CSAD 3178
     CTRL/C     CSAD 13^C

     RDM>
```

2.4.3 Error Messages

The terminal prints appropriate error messages for a variety of conditions. Error messages print on the line following the command string that caused the error condition.

With some exceptions, error messages consist of two-character codes; where the message ERR- prints, followed by the code. Table 2-6 lists these error codes.

Example

RDM>TES<RET>	Command to load and run microdiagnostics mistyped.
ERR - SYNTAX ERROR	
RDM>	

Table 2-6 Error Message Codes

Error Code	Definition
TAP:14	Tape -- read length error, not all records fit.
TAP:13	Tape -- flag received, not command or data.
TAP:12	Tape -- directory error.
RDM:11	Invalid operation code in macro.
RDM:10	Operation already in progress.
TRM:0E	Terminal -- remote line CRC error.
TRM:0D	Terminal -- length of input longer than buffer.
TRM:0B	Terminal -- command input buffer overloaded.
TAP:09	Tape -- file not found.
TAP:08	Tape -- invalid packet received.
TAP:07	Tape -- no end packet, invalid operation code received.
TAP:06	Tape -- tape count byte received exceeds maximum.
TAP:05	Tape -- tape check sum error received.

NOTE
UARTs are RDM resident.

Error Code	Definition
TAP:04	Tape UART -- overflow received.
TAP:03	Tape UART -- data set ready dropped.
TAP:02	Tape UART -- error received from UART.
TAP:01	Tape UART -- device timed out.
CPU:04	CPU UART -- overflow received.
CPU:03	CPU UART -- data set ready dropped.
CPU:02	CPU UART -- error received from UART.
CPU:01	CPU UART -- device timed out.
TRM:04	Terminal UART -- overflow received.
TRM:03	Terminal UART -- data set ready dropped.
TRM:02	Terminal UART -- error received from UART.
TRM:01	Terminal UART -- device timed out.

Table 2-6 Error Message Codes (Cont)

Error Code	Definition
REM:Ø4	Remote UART -- overflow received.
REM:Ø3	Remote UART -- data set ready dropped.
REM:Ø2	Remote UART -- error received from UART.
REM:Ø1	Remote UART -- device timed out.
TAP:FF	Tape -- diagnostic failure.
TAP:EE	Tape -- partial operation (end of medium).
TAP:F8	Tape -- bad unit number.
TAP:F7	Tape -- no cartridge.
TAP:F5	Tape -- write protocol.
TAP:EF	Tape -- data check error.
TAP:EO	Tape -- see error (block not found).
TAP:DF	Tape -- motor stopped.
TAP:DO	Tape -- bad operation code.
TAP:C9	Tape -- bad record number.
SYNTAX ERROR	Error in entering console commands.
INVALID COMMAND	RDM does not recognize command.
CMI:nn	Error in VAX main memory. (Two digits, nn, are error code.) Results from EXAMINE if area addressed has error.
CMI:ØØ	Nonexistent memory.
CMI:Ø1	Corrected read data.
CMI:Ø2	Read data substitute.
ROM	ROM failed RDM power-up self-test.
RAM	RAM failed RDM power-up self-test.

3.1 SCOPE

This chapter describes how to unpack, install, and inspect the KC750 remote diagnostic module and modem.

3.2 CUSTOMER REQUIREMENTS

Paragraphs 3.2.1 through 3.2.4 describe the steps you and the customer must take to prepare for a KC750 option installation.

3.2.1 System Configuration Worksheet

Make sure the prospective customer site is evaluated and System Configuration Worksheets supplied to DIGITAL's Remote Diagnostic Center (DRDC) before scheduling any installations of a KC750 option. The DRDC requires complete and accurate worksheets to build a configuration file.

3.2.2 Telephone Company Notification (US Only)

The customer must furnish the following information to the telephone company before the installation of a KC750-CA option.

1. Modem manufacturer's name
2. Modem model number
3. Modem FCC registration number (supplied by DIGITAL district office)
4. Modem ringer equivalence number (supplied by DIGITAL district office)
5. Modem speed
6. Type of modem voice jack direct-connect receptacle
7. Telephone number of line/RJ11C, if installed

Table 3-1 lists the data for modems supplied with the KC750-CA option.

Table 3-1 KC750-CA Option Modem Data

Modem Mfg.	Model No.	FCC Registration No.	Ringer Equiv. No.
GDC	103A3	AG697J-62418-DM-E	0.6B
Vadic	VA355P	AJ4964-70263-DM-N	1.0B
DIGITAL	DF02	A09994Q-67693-DM-R	Q.3A

All three modems use an RJ11C voice jack and operate at 300 bits/s.

For the KC750-DA option, the customer must order both the phone line and modem. The modem must be a Bell Model 103J or an equivalent.

3.2.2.1 Discontinuing Telephone Service (US Only) -- Normally the modem should not disturb the telephone network. However, if the telephone network malfunctions due to a defective modem, the local telephone company will discontinue service. If practical, the telephone company notifies customers before discontinuing service. However, this is not always possible.

If service is discontinued, the local telephone company should perform the following steps.

1. Notify customers promptly that service has been discontinued.

2. Give customers the opportunity to correct the situation which caused the temporary break in service.

3. Inform customers of their right to bring a complaint to the FCC. Complaint procedures can be obtained from the local telephone company or the modem manufacturer.

3.2.2.2 Notifying Customers of Telephone Service Changes (US Only) -- Any local telephone company changes of equipment or procedures must be consistent with FCC regulations. Customers should receive adequate written notice if the changes might create any incompatibility between customer terminal equipment and new equipment or procedures. This allows them the opportunity to maintain uninterrupted service.

3.2.3 Customer Acknowledgement

The customer must sign an Installation Acknowledgement form to ensure DIGITAL's full and free access to equipment. (Appendix C includes a sample form.)

3.2.4 Customer Responsibilities

The customer is responsible for preparing the system for remote diagnosis after installation of the KC750 option. Preparation includes mounting the system diagnostic pack and scratch media.

3.3 MODEM REPAIR AND REPLACEMENT (US Only)

Return modems shipped with the KC750-CA option to Stockroom 126 in Woburn for repair and replacement. If one vendor's modem is replaced with a different vendor's modem (or even a different model type), the customer must notify the telephone company of the new FCC registration number, ringer equivalence, and other items listed in Paragraph 3.2.2.

The on-site Field Service engineer is responsible for notifying the customer of changes and providing technical information which the customer must report to the local telephone company.

3.4 UNPACKING AND INSPECTION

Open the shipping container carefully. Check the contents against the inventory list in Paragraph 1.4 and inspect all parts for damage. Immediately report any damage to the responsible carrier and branch office supervisor. Do not start installation if any item is missing or damaged; wait until the item is replaced or repaired.

3.5 INSTALLATION PROCEDURES

Perform the following procedure to install the KC750 option. Refer to Figure 3-1 for the cabling layout.

1. Power down the system by turning the keyswitch to OFF.

2. Select RDM remote port baud rate (Table 3-2). Baud rate is usually 300.

3. Install the L0006 module in slot 6 of the processor cabinet. Refer to Figure 3-2 for module placement in front of cabinet.

   no, only P3 & P4

4. Move all cables on the processor backplane (except the TU58 power cable) from the left side of slot 6 to the right side, keeping the same vertical placement. The TU58 power cable remains in place. Refer to Figure 3-3 for cable placement in cabinet.

5. Plug the RDM filtered cable assembly (7016921) into the bottom 22 pins of set C on the right side of slot 6. Attach the assembly to the backplane so that pin 1 is on top and pin 22 is on the bottom. Refer to Appendix A for the backplane pin assignments of the various signals.

If not already fitted

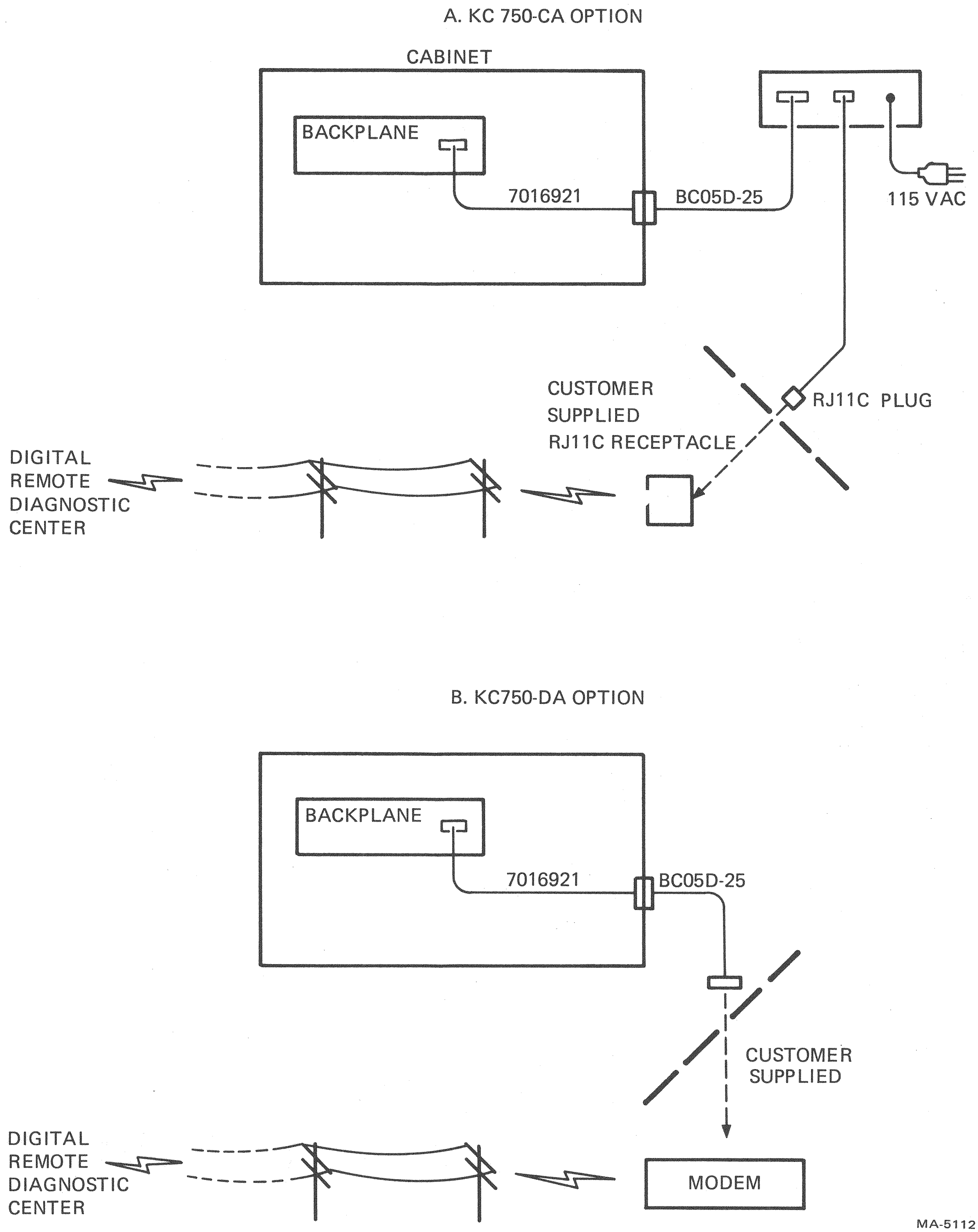


Figure 3-1 KC75Ø Option Cabling

Table 3-2 Switch Settings for RDM Remote Port Baud Rates

Baud Rate	E190 -1	E190 -2	E190 -3	E190 -4
300	ON	ON	OFF	ON
400	OFF	ON	OFF	ON
600	ON	OFF	OFF	ON
1,200	OFF	OFF	OFF	ON
2,400	ON	ON	ON	OFF
3,600	OFF	ON	ON	OFF
4,000	ON	OFF	ON	OFF
4,800	OFF	OFF	ON	OFF
7,200	ON	ON	OFF	OFF
9,600	OFF	ON	OFF	OFF
19,200	ON	OFF	OFF	OFF

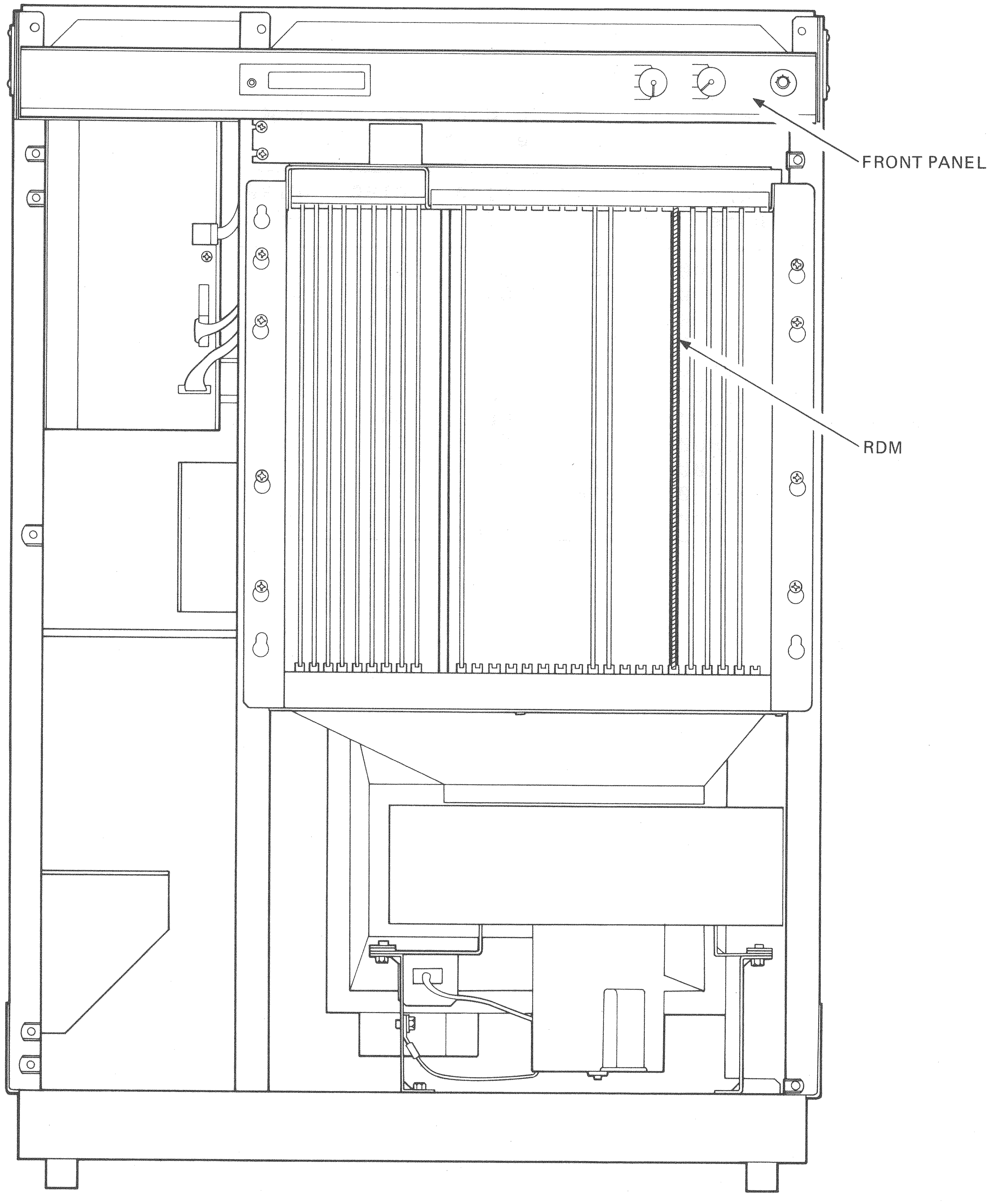


Figure 3-2 RDM in VAX-11/750 Cabinet (Front View)

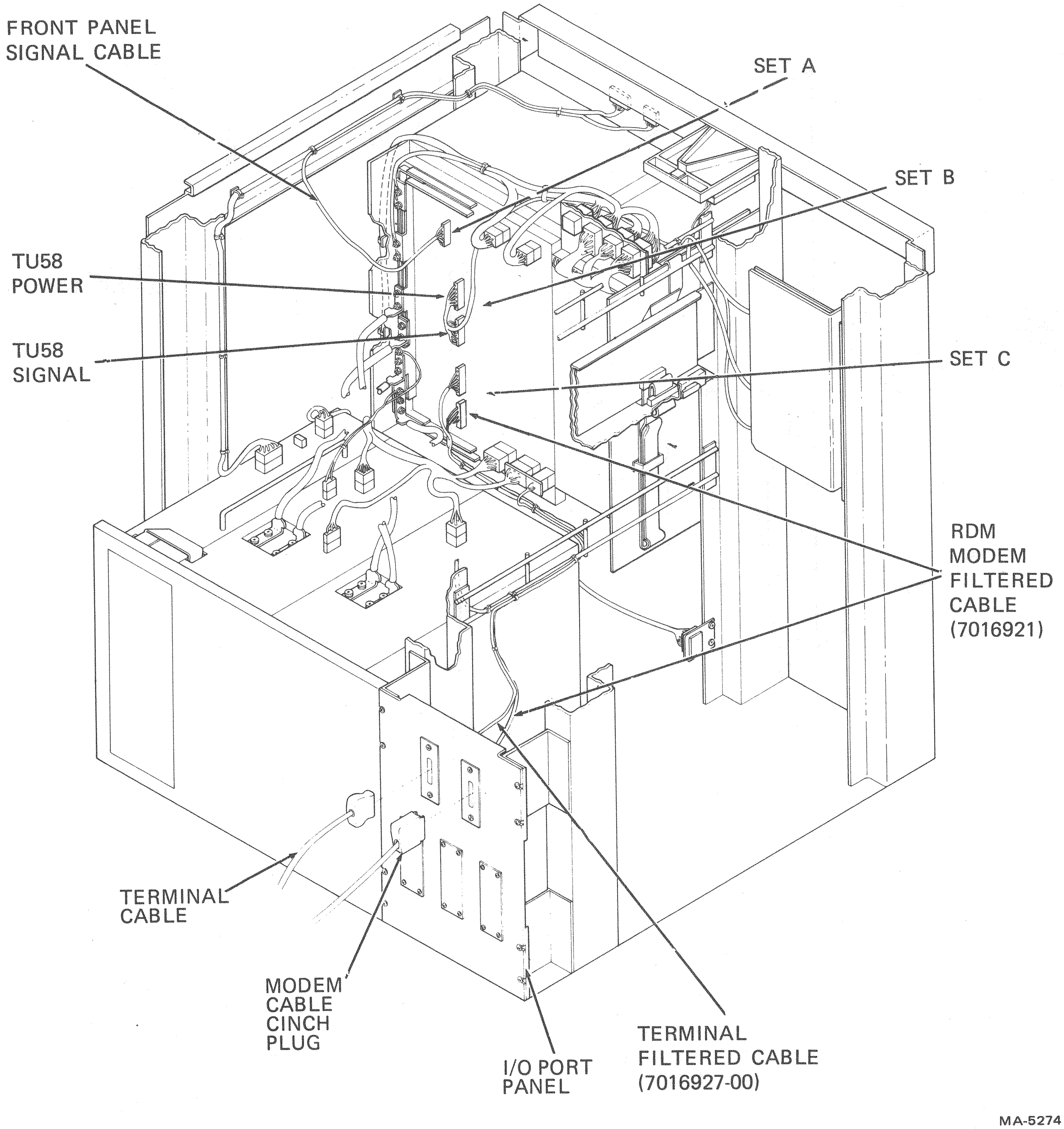


Figure 3-3 KC750 Option Cabling in VAX-11/750 Cabinet (Rear View)

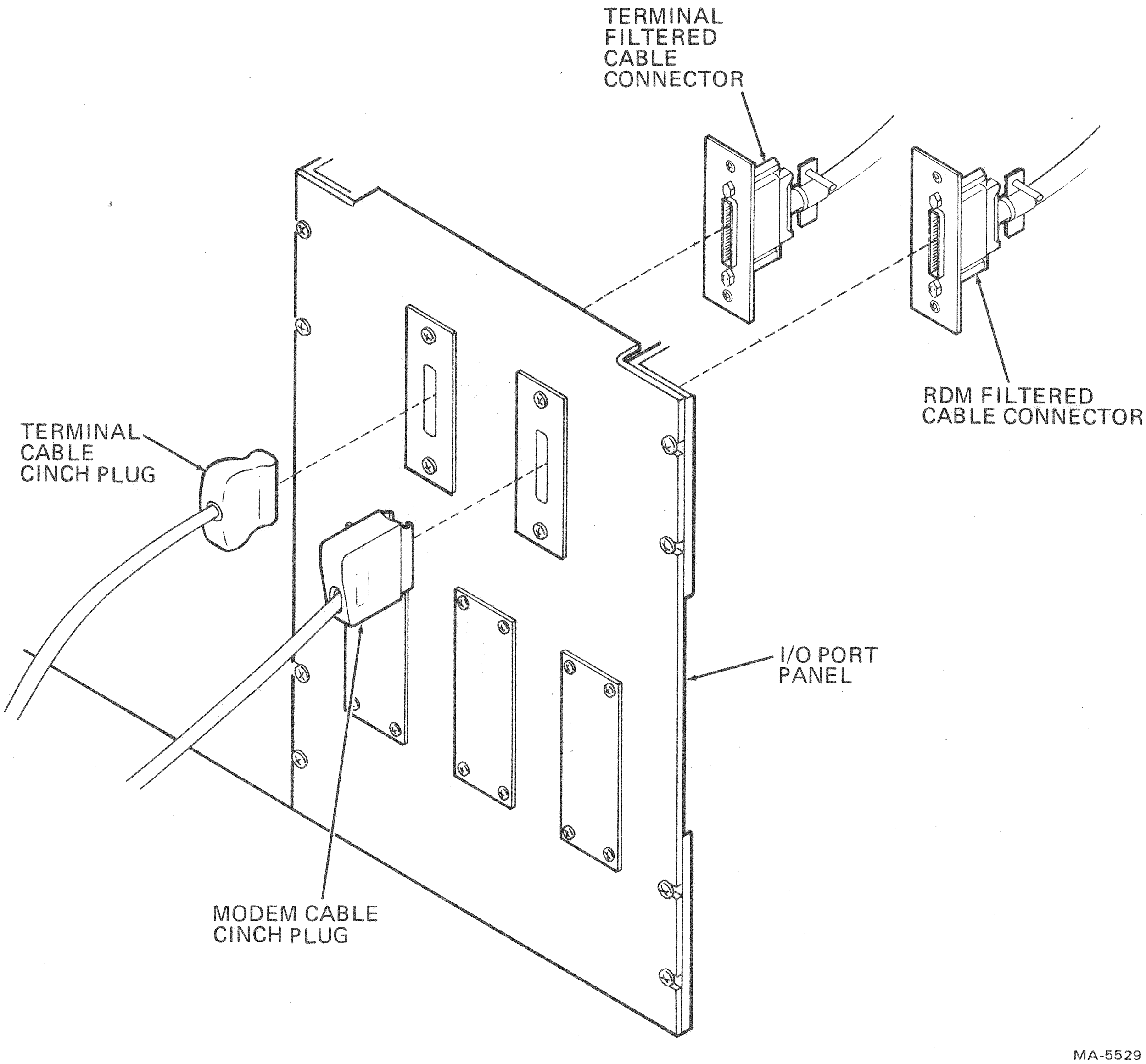


Figure 3-4 Filtered Cable Assembly Installation

6. Attach the filtered cable assembly to the I/O port panel (Figure 3-4), using the two 6-32 kep nuts provided with the CPU coverplate.

7. Install the asset tag around the filtered cable. Use the tie strap to secure the tag to the cable, and tighten to prevent movement of the tag.

8. Connect the modem cable cinch plug at the I/O port panel to the RDM filtered cable connector and secure.

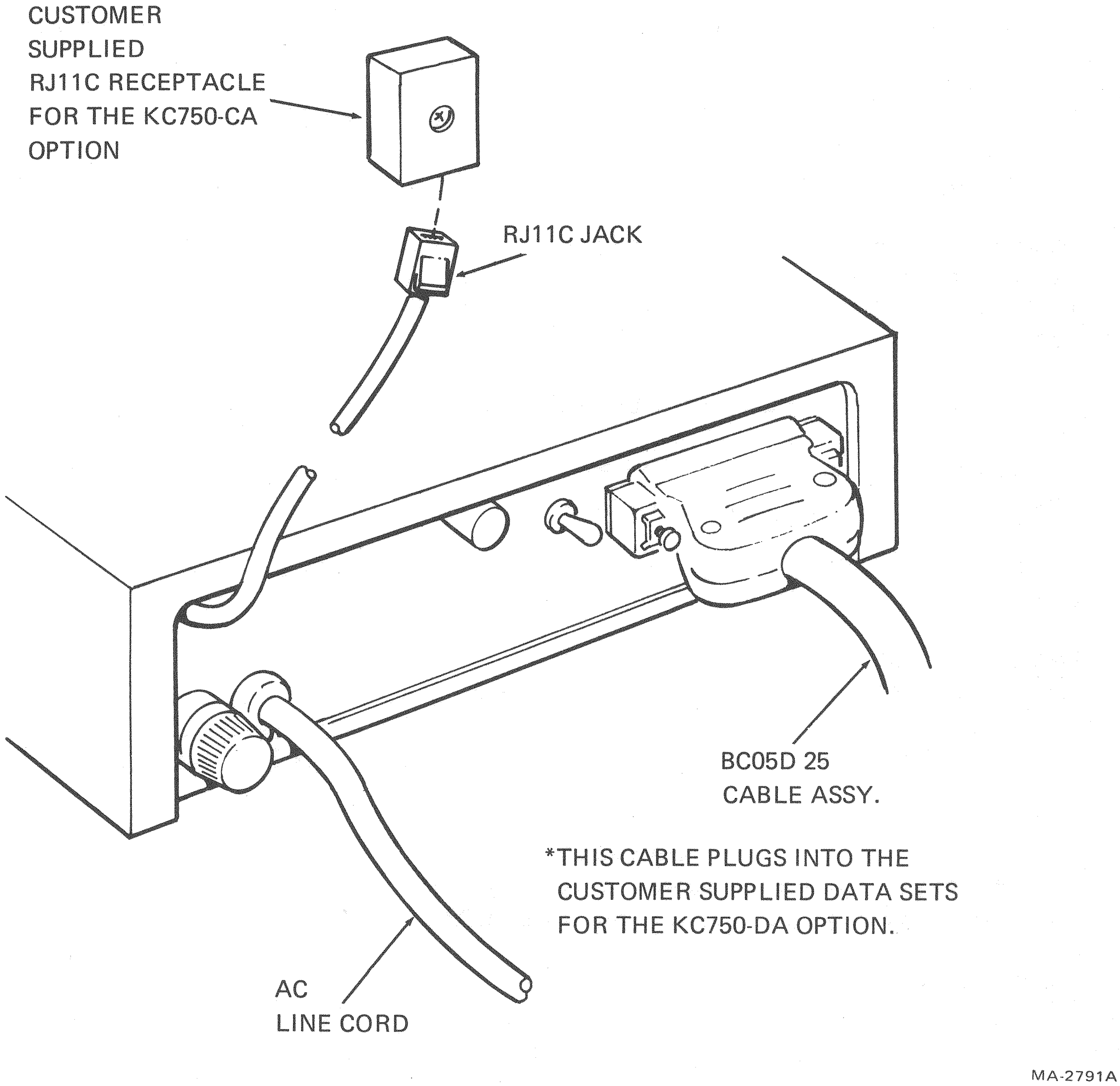


Figure 3-5 KC750-CA Modem Cabling

9. Route the modem cable as required and connect to the modem (Figure 3-5). The US modem is a freestanding unit that requires a 115 Vac power source.

CAUTION

Modem ac power should not come from the VAX-11/750 internal power distribution system, since this violates UL regulations and cabinet power integrity.

3.6 INSTALLATION INSPECTION

Verify KC75Ø option operation by performing the following procedure.

1. Mount diagnostic software and scratch media.

2. Power up system.

3. Observe RDM power-up self-test. (Fault indicator should turn on after power-up for about ten seconds. If fault indicator does not turn on or does not turn off, remove RDM.)

4. Turn console keyswitch to REMOTE.

5. Notify DRDC by phone that you want them to perform an installation verification.

6. Check with DRDC for verification results (by phone or terminal).

3.7 REMOVAL PROCEDURES

Removal of the KC75Ø option is the reverse procedure of installation (Paragraph 3.5). If removal will be temporary, only perform steps 1 through 4 in reverse. The added cables may be left in place.

WARNING

Always power down the system before installing or removing the RDM.

3.8 INSTALLATION OUTSIDE THE US

To install the KC75Ø option outside the U.S., follow the instructions in the preceding paragraphs except those marked "(US only)".

DIGITAL does not provide modems with KC75Ø options outside the U.S. You must follow the standard practice of a given country to obtain a modem there. Appendix E gives specifications for modems that work with the KC75Ø option.

In Europe, remove jumper W-3 on the RDM (Figure 3-6).

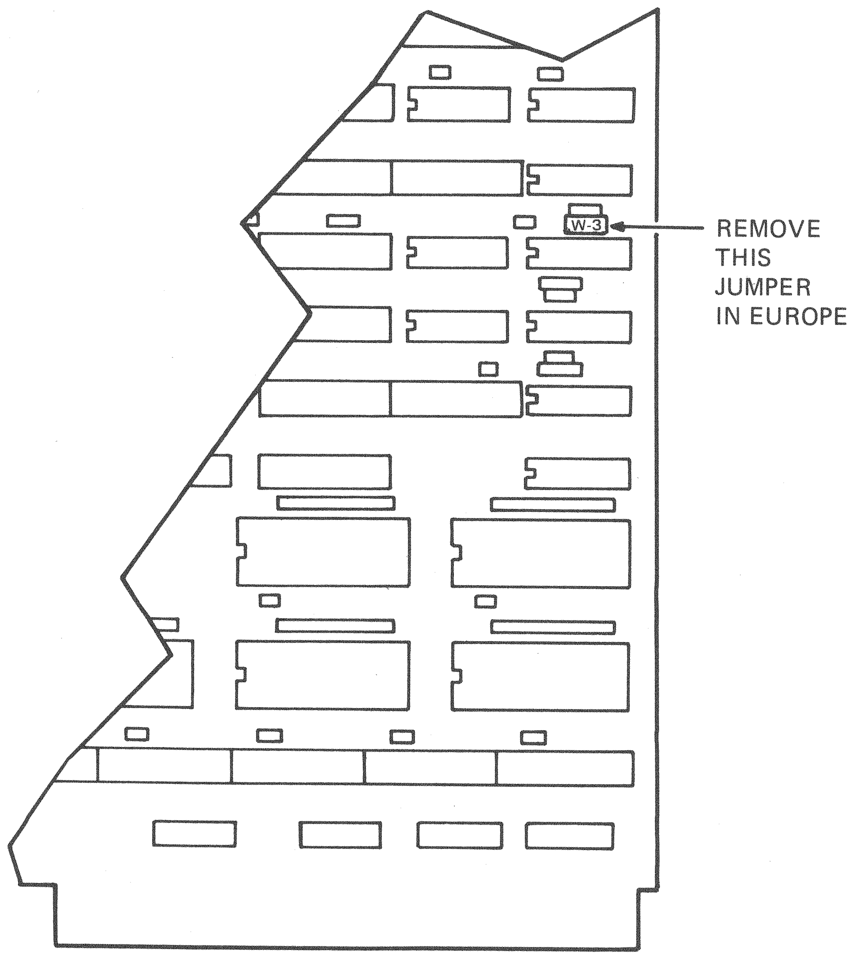


Figure 3-6 RDM Jumper Cut in Europe

CHAPTER 4
FUNCTIONAL DESCRIPTION

4.1 SCOPE

This chapter provides you, a Field Service engineer, with information about the internal RDM operations.

This background may make you more comfortable using the KC750 option.

The following paragraphs describe the general RDM functions, its major components, and how they work together. While reading this chapter, refer to Figure 4-1 (the RDM system block diagram).

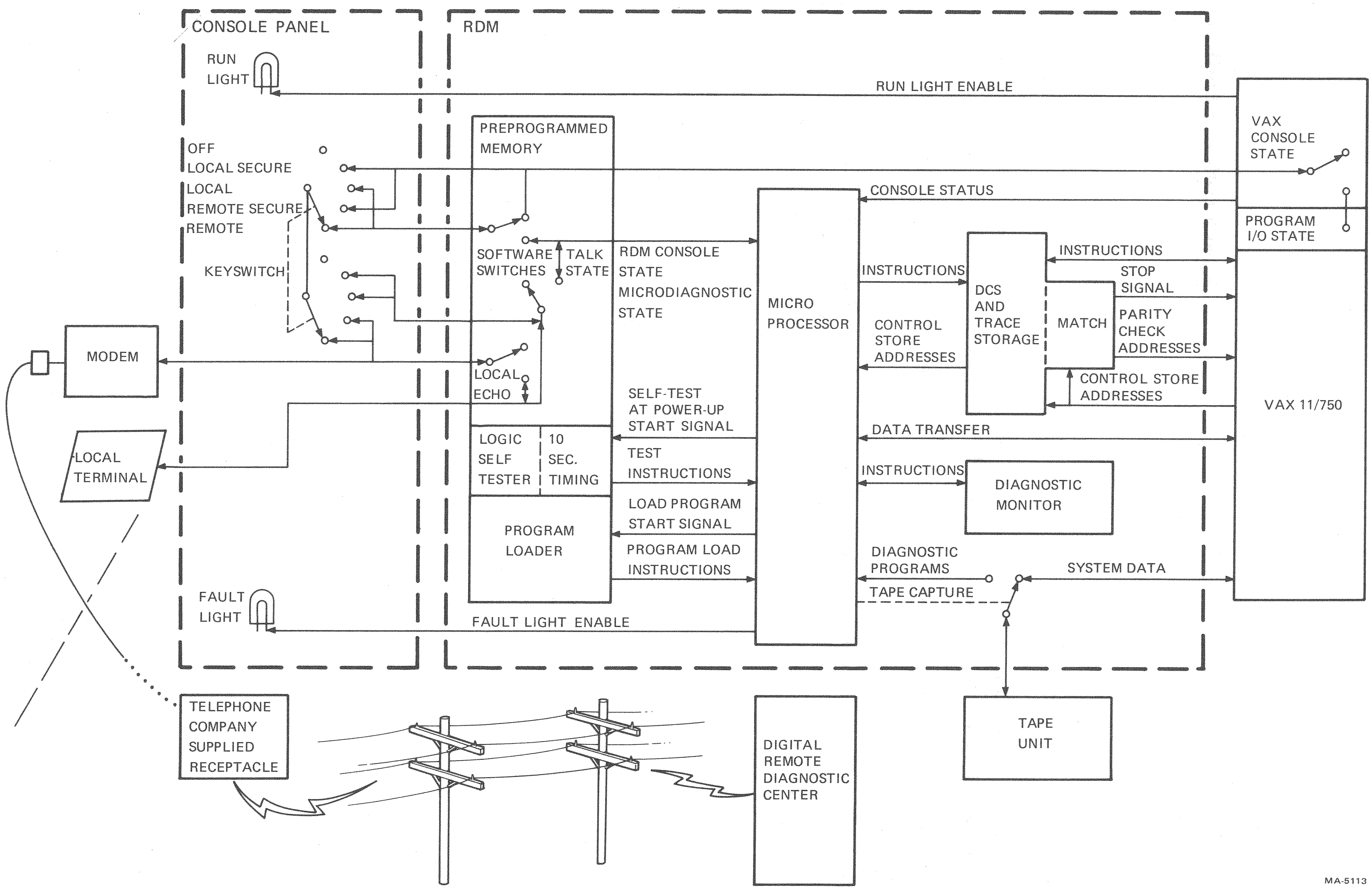


Figure 4-1 RDM System Block Diagram

4.2 GENERAL RDM FUNCTIONS

The following list describes the general functions of the remote diagnostic module.

1. Allows users to perform all regular processor console functions from either the local terminal or DRDC.

2. Allows remote and local control of the TU58 tape unit.

3. Supports DRDC's running of macrodiagnostic scripts. (Scripts are programs that call and run sequences of routines to test the CPU and system peripherals.)

4. Loads and reads VAX-11/750 main memory.

5. Loads and reads processor WBus in support of microdiagnostics.

6. Drives VAX-11/750 control lines in place of regular control store during microdiagnostics.

7. Traces and stores addresses of executed VAX-11/750 control store instructions.

NOTE

You cannot perform functions 6 and 7 at the same time.

8. Compares the addresses of instructions executed by the main processor with a preselected address, and stops the processor when a match occurs (at the micro break point).

9. Runs a parity check on the processor control store.

10. Conducts logic self-test of the RDM on power-up.

4.3 MAJOR RDM COMPONENTS

The major RDM components are a preprogrammed memory, a microprocessor, a diagnostic control store, and miscellaneous registers and control logic.

4.3.1 Preprogrammed Memory

Preprogrammed memory consists of three parts: software switches, logic self-tester, and program loader. Some RDM preprogrammed memory functions connect to preprogrammed memory functions in the VAX-11/750 console. Therefore, the following discussion includes appropriate VAX console functions to complete the descriptions.

4.3.1.1 Software Switches -- Software switches are instructions that change the RDM state and perform certain control functions. Each RDM state corresponds to a particular position of the software switches (Figures 4-2 through 4-5).

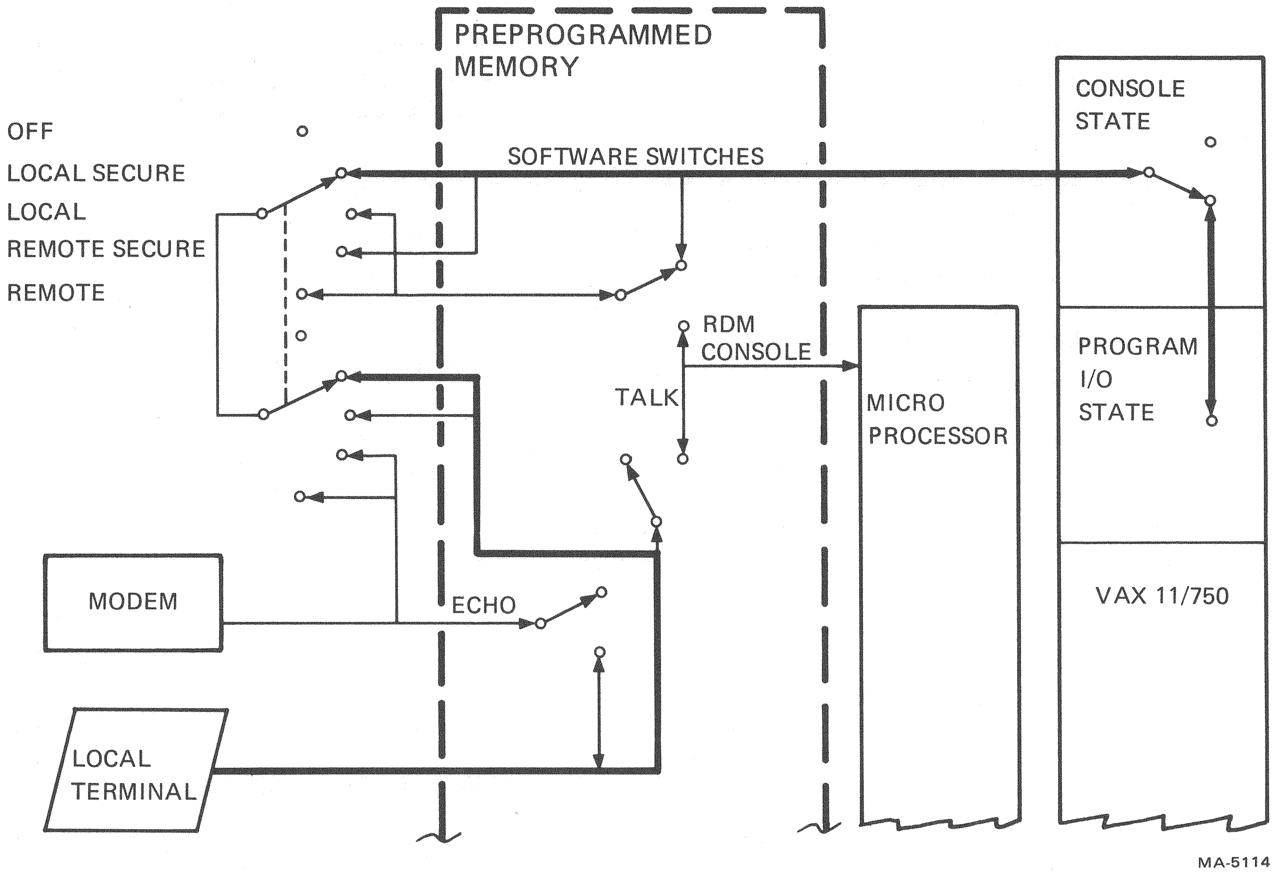


Figure 4-2 Local Secure Program I/O State

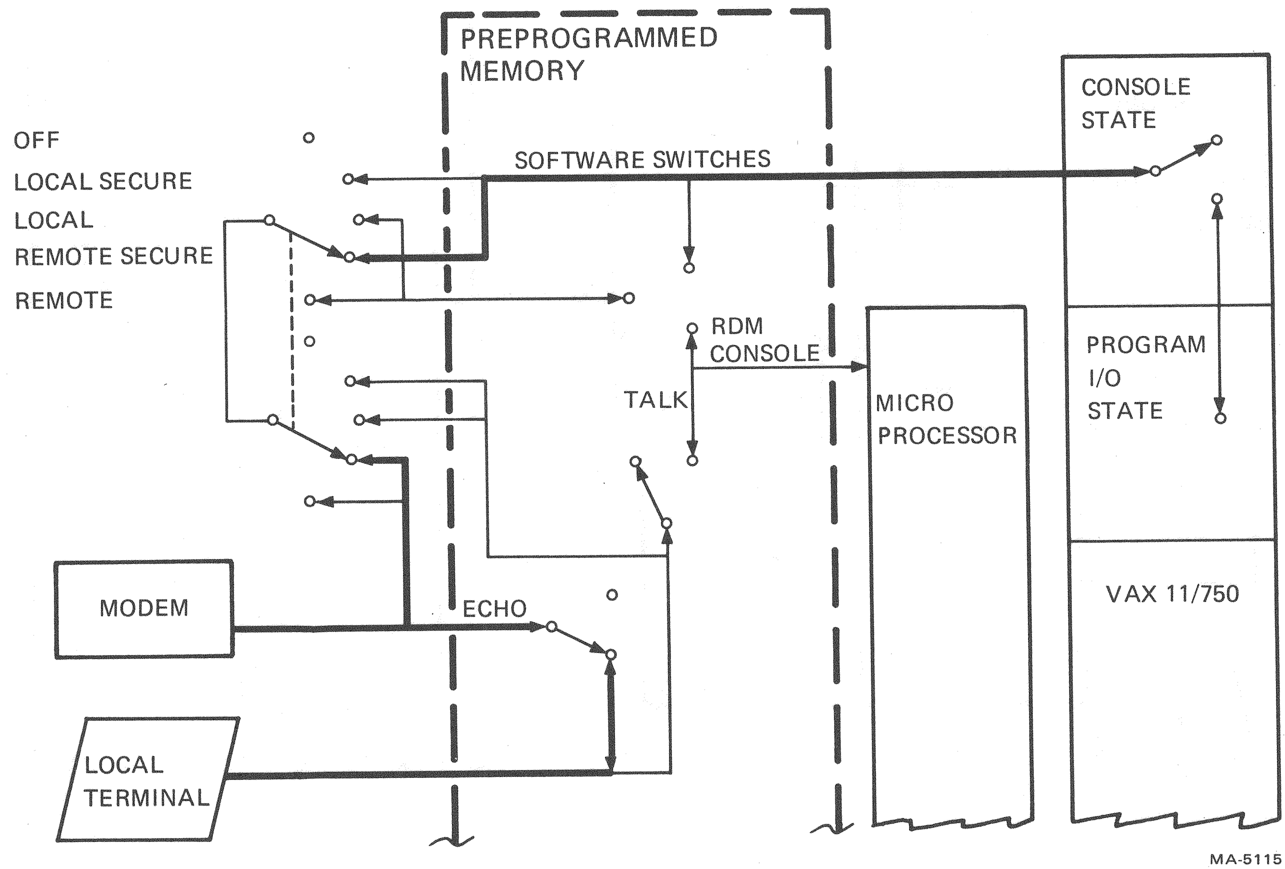


Figure 4-3 Remote Secure VAX Console State With Local Echo

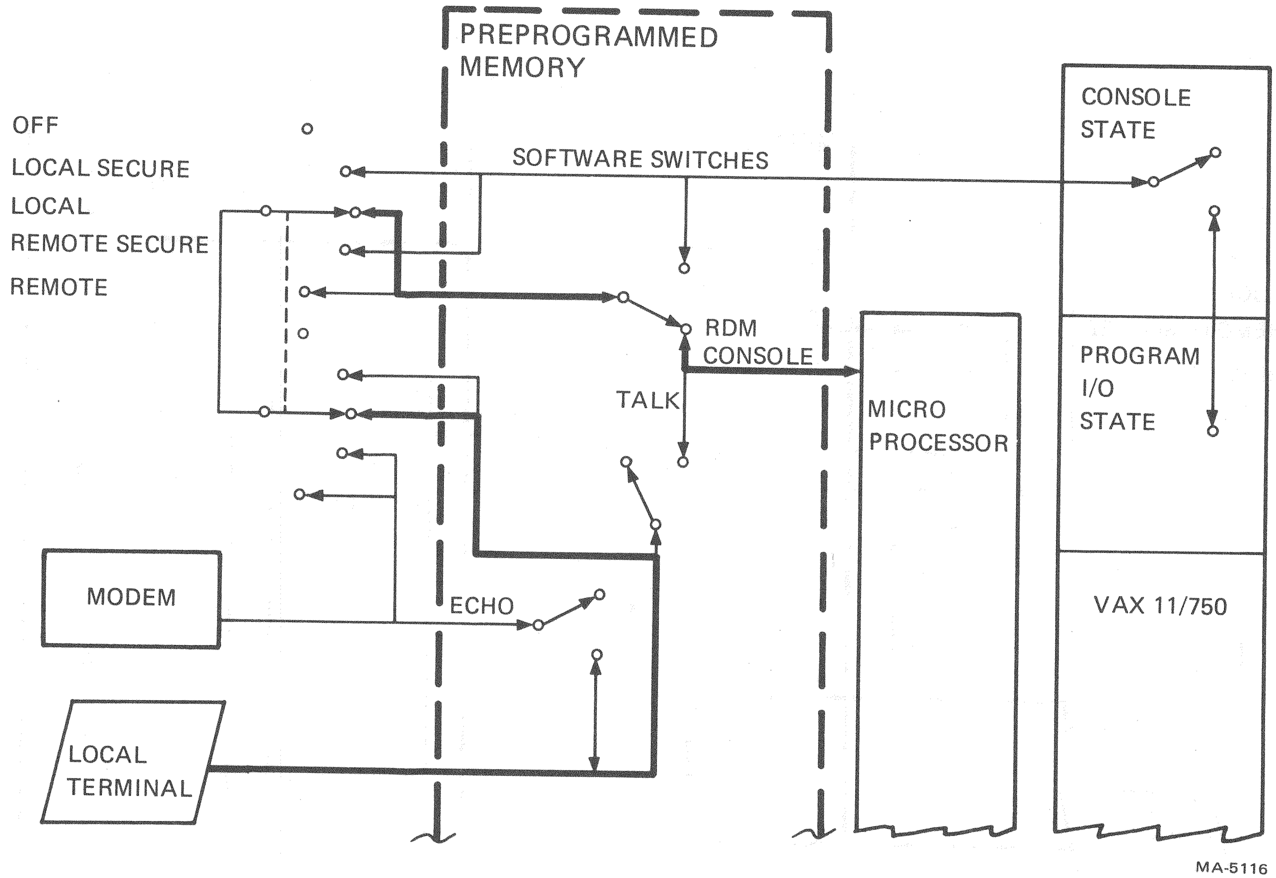


Figure 4-4 Local RDM Console State

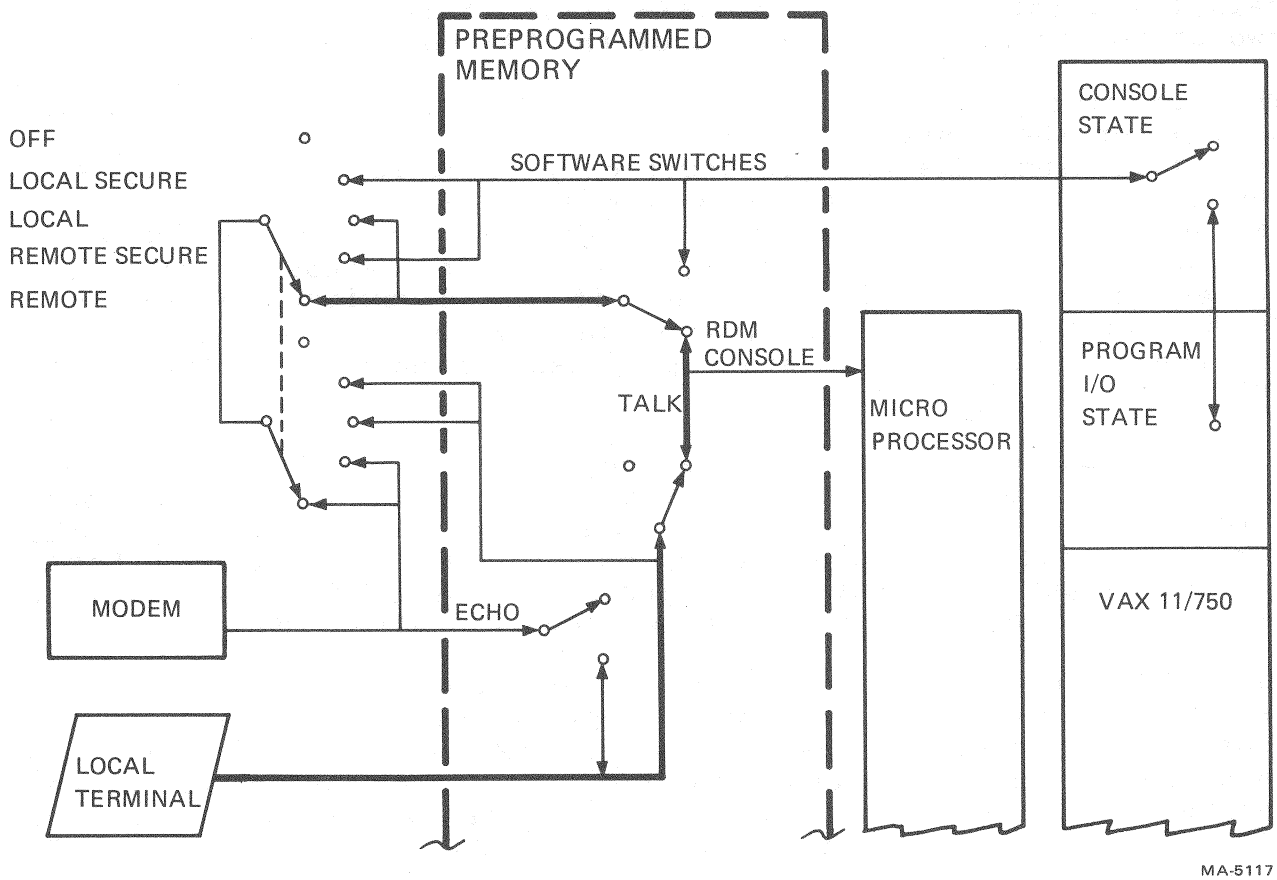


Figure 4-5 Remote Talk State

In program I/O state, the active terminal communicates with the VAX-11/750 at the program level. In VAX console state, the active terminal communicates with VAX-11/750 firmware. The VAX console and RDM monitor these lines for control characters that set software switches to change states. In program I/O state or VAX console state, the RDM's microprocessor plays no active role in relaying information to the VAX-11/750. Switching between these two states is the exclusive job of the VAX console.

The RDM microprocessor accepts commands from the active terminal only after the system switches to RDM console state. The RDM does this switching. The RDM processor only communicates directly with the active terminal when the system is in RDM console state. You may only enter the RDM console state (and other states entered via the RDM console state, such as the microdiagnostic and talk states) from the active terminal if the keyswitch is not in a SECURE position. RDM preprogrammed memory contains another switch for the talk state. This state provides communication between the DRDC and local terminal.

A similar switch exists for local copy. Local copy allows dialogue between the DRDC and RDM or VAX-11/750 to echo on the local terminal. The copy works in one direction only; that is, local terminal dialogue cannot echo on the DRDC terminal. (Local terminal input decodes as an interrupt at the DRDC. This interrupt alerts DRDC that the local operator requires attention.)

Chapter 2 describes all commands that switch system states and perform other control functions. Preprogrammed memory supports the control commands. The VAX console, RDM console, or both may recognize a given command. The current system state determines whether VAX or RDM supports a command. For example, VAX supports CTRL/O in either the program I/O or VAX console states; RDM supports CTRL/O in the RDM console or microdiagnostic states. However, only RDM supports the Talk command. VAX cannot support Talk even if the system is in the VAX console state.

4.3.1.2 Logic Self-Tester -- The logic self-tester is a group of preprogrammed instructions that verify RDM logic. On RDM power-up, the microprocessor signals the self-tester for instructions. These instructions tell the microprocessor to check various hardware parts and light the fault indicator. If the microprocessor does not find a hardware failure within 10 seconds, the microprocessor turns the fault indicator off. The logic self-test runs only at power-up.

4.3.1.3 Program Loader -- The program loader is a group of preprogrammed instructions called by the microprocessor to load the microdiagnostic monitor from the tape unit into the RDM's memory. The role of the microdiagnostic monitor is discussed in Paragraph 4.3.2.2. There is no hard connection between the tape unit and RDM. The RDM has no knowledge of the VAX's use of the tape unit. The RDM has a logical connection to the tape unit that the microprocessor uses to gain control of the tape unit from the main processor.

4.3.2 Microprocessor

The microprocessor (an Intel 8085) executes instructions in the preprogrammed memory and microdiagnostic monitor. Since the RDM has its own processor, preprogrammed memory, and stored program capacity, the microprocessor does not depend on the VAX processor to run tests. Only the VAX clock must operate.

4.3.2.1 Data Transfer Paths -- The microprocessor uses a data transfer line to read and write to two important areas in the VAX processor: the WBus and main memory. The WBus transfers information between the VAX-11/750's arithmetic logic unit (ALU) and general registers. Main memory stores programs and data to be manipulated later. The RDM tests for problems by writing and then reading known data to these areas. (The RDM's WD register and MD register hold data to be written to the WBus and memory, respectively. The RDM's WH register and MH register hold data read from the WBus and memory, respectively. The RDM's MA register holds the address of memory read from or written to.)

Another area in the VAX-11/750 directly accessed by the microprocessor is the VAX front panel console. The DRDC can therefore check with the RDM to read the front panel indicators or determine the keyswitch position.

4.3.2.2 Microdiagnostic Monitor -- The microdiagnostic monitor is the RDM's nonresident operating system. It directs microdiagnosis of the VAX-11/750. To load the monitor, the operator on the active terminal enters the Test command. The program loader instructs the microprocessor to load the monitor from the tape unit into an area of random access memory (RAM).

The Test command switches the RDM to the microdiagnostic state. (Before this happens, the keyswitch and software switch connections must be made, so that the system is in RDM console state.) Once in the microdiagnostic state, commands entered on the active terminal address the diagnostic monitor which controls the microprocessor.

4.3.3 Diagnostic Control Store

The following paragraphs discuss the function of the diagnostic control store (DCS) and its associated hardware.

4.3.3.1 Diagnostic Program Storage -- The diagnostic control store (DCS) is the area of memory where the microprocessor loads specific diagnostic routines. Once loaded, these diagnostic programs drive the VAX-11/750, in effect replacing the VAX control store. The microdiagnostic monitor loads the DCS and initiates driving of the VAX processor.

4.3.3.2 Trace Storage -- Trace is another function that occupies the DCS. The trace function continually records the 64 most recently executed control store addresses, as well as the address of the current microinstruction for the VAX to execute. Trace storage shares part of the memory used for storing diagnostic programs, so the two functions cannot perform at the same time.

Tracing can help determine what led the processor to stop. The microprocessor can relay control store addresses stored in memory to the active terminal for analysis.

In trace mode, the RDM continually and automatically clocks CS addresses from the CS address lines into a trap register, and from there into 64 DCS RAM locations. The RDM is in trace mode whenever it is not in the microdiagnostic state.

When the RDM loads a new CS address into the trap register, the previous CS address in the trap register moves into a DCS RAM location. The value of a DCS pointer determines the DCS RAM location to which the CS address moves. Every time an address moves from the trap register to RAM, the pointer increments by one. The RDM loads the trap register when the CPU latches a word.

The DCS pointer has a modulus of 64. After 64 words have been loaded into RAM, the pointer increments to the location loaded first. The DCS RAM, therefore, is a circular buffer in which new CS addresses from the trap register overwrite the oldest information.

Chapter 2 describes the Trace command. Reading out of the DCS is the reverse process of writing into the DCS. The microprocessor reads CS addresses out of the DCS and forwards them to the terminal for display.

The first address read is the address in the trap register, which is always the latest CS address. The next address is read from the current RAM location specified by the pointer.

4.3.3.3 Match Register -- Stop-on-micromatch is a third function associated with DCS. This function loads a match register with a specific control store address. It then compares the address with that of each control store instruction executed. When the address of the current control store instruction matches the selected address, the match logic stops the processor. Primarily, stop-on-micromatch permits you to trace the path the processor took to a particular microinstruction.

When the address in the match register matches the address on the CS address lines, a pulse generates on pin CØ681 of the CPU backplane. By deliberately making the CPU cycle on a matching address, you may use the match signal as a known pulse to trigger test equipment.

In addition, the match register helps the DCS gain control of the VAX processor. Under microdiagnostic monitor control, the microprocessor loads the match register with a DCS address. The match register places this DCS address on the control store address lines to force control of the VAX over to the DCS. Once the DCS has control, the particular microdiagnostic in the DCS drives the VAX processor control lines. The microdiagnostic is then responsible for making the CPU continue to select DCS as a source of instructions.

The match register also helps to check parity in the VAX control store. The microprocessor can make the match register increment through successive control store addresses. At each location, the VAX-11/750's own parity check logic is enabled to perform a parity check. The match register addresses each control store location until the register increments to an address in the CPU control store that holds a deliberately set parity error. (If the parity check uncovers an accidental parity error, the match register will not increment past that address.) Detecting the known parity error is a test of the VAX-11/750 parity check logic.

NOTE

When parity checking the writable control store (WCS), first load the WCS with data. WCS addresses are 2000 to 23FF.

CHAPTER 5
RUNNING MICRODIAGNOSTICS

5.1 SCOPE

The VAX-11/75Ø microdiagnostic program consists of two basic blocks: the microdiagnostic monitor and the test overlays.

When the microdiagnostic program starts, a ROM-based TU58 driver routine loads the microdiagnostic monitor (MICMON) from the TU58 tape cartridge into 8Ø85 RAM memory. The monitor remains in the 8Ø85 memory for the duration of the test session.

Once the monitor is loaded, the program initialization routine begins. When several passes are run, this routine executes at the beginning of each. The program initialization routine tracks program flow, initializes parameters, and invokes the TU58 driver routine to load the first test overlay for execution. Normally, the last instruction in each test overlay calls the TU58 driver routine to read in the next test overlay. The test overlays are thus read into memory one at a time.

Test overlays consist of pseudo instructions, test data, and microinstructions to execute from the diagnostic control store (DCS). The monitor reads the pseudo instructions and calls appropriate routines. These routines implement the required functions in 8Ø85 code. The pseudo instructions set up and control the execution of microinstructions in DCS.

You may use the monitor to set and clear flags, and issue other commands to control the program flow with precision.

5.2 RUNNING THE MICRODIAGNOSTIC PROGRAM

DIGITAL distributes the VAX-11/75Ø microdiagnostic program on a series of TU58 tape cartridges. Each tape cartridge contains the microdiagnostic monitor and a number of test overlays. The microdiagnostic monitor is identical on all tape cartridges, but the test overlays are unique. Each tape cartridge tests a portion of the CPU or memory. Normally, you should run the tapes in the following order: DPM, MIC, FPA, MEMORY.

To run the microdiagnostic program, perform the following procedure.

1. Insert the desired TU58 cartridge into the slot on the CPU's front panel.

2. Turn the POWER ON ACTION switch to HALT.

3. Turn the keyswitch to LOCAL to power up the CPU. If the console terminal executes microverify successfully it should type two percent signs, and then the console prompt >>>.

4. Type CTRL/D to invoke the RDM console command mode. The console terminal should respond with the RDM prompt RDM>.

5. Type TE to run the microdiagnostic program. If the test sequence on the tape cartridge runs to completion without error, the program types a series of messages on the console terminal confirming completion. The following example shows a typical printout with no errors.

Sample Dialogue:

```
RDM>TE<RET>                                       ! Test command.

ECKAA-VØ.22 MIC-VØØ.Ø4                            ! Program identification.
Ø1,Ø2,Ø3,Ø4,Ø5,Ø6,Ø7,Ø8,Ø9,ØA,ØB,ØC,ØD,           ! Test numbers typed
ØE,ØF,1Ø,11,12,13,14,15,16,17,18,19,1A,           ! at execution time.
1B,1C,1D,1E,1F,2Ø,21,22,23,                       ! Number of tests
END OF PASS Ø1                                    ! varies depending
                                                  ! on tape.

MIC>                                              ! Microdiagnostic monitor
                                                    prompt.
```

6. Replace the tape cartridge with the next microdiagnostic tape and type DI.

7. Type RET<CR> after the MICMON prompt to return to the console I/O mode.

8. Type RET<CR> after the RDM command mode prompt to return to the VAX console state.

5.3 MICRODIAGNOSTIC MESSAGES

The microdiagnostic program types messages on the console terminal to indicate the following items and functions.

- Load microdiagnostics
- Monitor name, version, and module under test
- Test number to execute next (current test)
- Error messages

If the program detects an error, it should halt test execution, type an error message, and pass control to the microdiagnostic monitor. Refer to the following example.

Sample Dialogue:

```
RDM>TE<RET>

ECKAA-VØ.22 MIC-VØØ.Ø4                   ! Program identification.
Ø1,                                      ! Test number.
?ERROR: ØØ3Ø TEST: Ø1 SUBTEST: Ø1

DATA:   AAAAAAAA
        AAAA8AAA                         ! MODULE NAME is listed
                                         ! only if it is not the
        ØØØØØØØ1                         ! module being tested.

FAILING GATE ARRAYS: ADK,
FAILING MODULES: MIC,

MIC>                                     ! Microdiagnostic monitor
                                         ! prompt.
```

The only test number printed is Ø1, indicating that the program started test Ø1 but did not reach test Ø2. The next line in the message confirms the location of the failure in test Ø1.

```
?ERROR: ØØ3Ø TEST: Ø1 SUBTEST: Ø1
```

The ØØ3Ø entry indicates the test program counter (PC) value of the failing pseudo instruction. The data patterns indicate the expected and received data and the loop count. If the program identifies failing gate arrays, they are listed at this point. The message then lists the names of modules that are probably causing the failure.

In most cases, you should attempt to repair the machine by turning off power, replacing the module or gate arrays called out, and rerunning the microdiagnostic program. If the message calls out several modules, replace them one at a time in the order listed and run the program after each replacement until the fault is repaired.

Replacing the parts called out may not repair the fault. Also, spare modules may not be available. In such cases, you may want to examine the program listing for the failing test. You may then use microdiagnostic monitor commands to set up scope loops and step through the failing test. The following chapters cover these topics.

6.1 SCOPE

Several commands control the microdiagnostic monitor. You may enter the monitor either directly or indirectly. To enter directly from the RDM console command mode, type TE/C (Test Command) instead of TE (Test). To enter directly during test execution, type CTRL/C. The microdiagnostic program enters the monitor indirectly (automatically) if it finds either a TU58 driver error, or a hardware error when the halt on error flag is set.

When the monitor is waiting for a command, it prompts with MIC>.

This list contains legal command op codes.

Diagnose
Loop
Set
Clear
Show
Return
Continue

This list contains legal command keyword arguments.

TEST
VBUS
LOOP
STEP
PASS
NER (no error report)
CYCLE (one microcycle)
BELL
TICK (one half microcycle)
FLAG
HALT
SA (signature analyzer)
SOMM (stop-on-micromatch)
INSTRUCTION
CONTINUE
CF (control file)
IB (inhibit burst)
QA (quality assurance)
TR (trace flag)

In the following command descriptions, an exclamation point means exclusive OR. In sample dialogues and examples, an exclamation point indicates that what follows on that line is a comment.

6.2 DIAGNOSE

This command takes three types of arguments (shown above).

```
Syntax:  DI TE:<test-number> [:<test-number> ! CO]
         DI PA:<pass-count>
```

The Diagnose command initializes the program control flags and starts program execution.

Flag	**Initial Setting**
LOOP	clear
NER	clear
BELL	clear
HALT	set
IB	clear
SA	clear
QA	clear
TR	clear

If you type Diagnose without any arguments (switches), the monitor executes all tests on the installed tape once. Control then returns to the RDM console command mode, if the program detects no errors.

Sample Dialogue:

```
MIC>DI                                 ! Diagnose command.

ECKAA-VØ.21 MIC-VØØ.Ø4

Ø1,Ø2,Ø3,Ø4,Ø5,Ø6,Ø7,Ø8,Ø9,ØA,ØB,ØC,ØD,ØE,ØF,1Ø,11,12,13,
14,15,16,17,18,19,1A,1B,1C,1D,1E,1F,2Ø,21,22,23,
END OF PASS Ø1

RDM> CLK STOPPED CSAD ØØØØ NEXT Ø8ØØ
RDM>
```

6.2.1 Diagnose Test

You may use the Test argument with the Diagnose command in three ways.

Diagnose Test: followed by a single test number executes the specified test indefinitely. Type CTRL/C to escape from the loop and return to the monitor.

Diagnose Test: followed by a test number and Continue executes the specified test and all following tests on the installed tape cartridge. Control then returns to the RDM console command mode (RDM>).

Diagnose Test: followed by two test numbers executes all tests between and including the two numbers specified. Control then returns to the monitor. If the first test number you type is out of range, the program types an error message and returns control to the monitor.

Sample Dialogue:

Example 1

```
    MIC>DI TE:2<RET>                       !  Loop on test 2
                                           !  indefinitely.
    ECKAA-V0.21 MIC-V00.04                 !  Monitor identifies itself.
                                           !  Test runs, looping
    CTRL/C                                 !  continuously.
    02^C                                   !  Test number is 2.
                                           !  CTRL/C to escape.
    MIC>                                   !  Microdiagnostic monitor
                                           !  prompt.
```

Example 2

```
    MIC>DI TE:2 CO<RET>                    !  Start with test 2 and
                                           !  run the remaining tests
                                           !  on the tape cartridge.
    ECKAA-V0.21 MIC-V00.04                 !  Monitor identifies itself.
    02,03,04,05,06,07,08,09,               !  Each test number is
    0A,0B,0C,0D,0E,0F,10,11,12,13,         !  typed as that test starts.
    14,15,16,17,18,19,1A,1B,1C,
    1D,1E,1F,20,21,22,23,
    END OF PASS 01
    MIC>                                   !  Prompt.
```

Example 3

```
    MIC>DI TE:3:6<RET>                     !  Execute tests 3,4,5,6.
    ECKAA-V0.21 MIC-V00.04                 !  Program identifies itself.
    03,04,05,06,                           !  Each test number is typed
                                           !  as that test starts.
    MIC>                                   !  Prompt.
```

6.2.2 Diagnose Pass

You may use the Pass argument with the Diagnose command to set the pass count to a specified number. The default pass count is 1. A pass count of 0 causes the program to run indefinitely. At the end of the specified number of passes, control returns to the RDM console command mode.

Sample Dialogue:

```
MIC>DI PA:2<RET>

ECKAA-VØ.21 MIC-VØØ.Ø4

Ø1,Ø2,Ø3,Ø4,Ø5,Ø6,Ø7,Ø8,Ø9,ØA,ØB,ØC,ØD,ØE,ØF,1Ø,11,12,13,
14,15,16,17,18,19,1A,1B,1C,1D,1E,1F,2Ø,21,22,23,
END OF PASS Ø1

Ø1,Ø2,Ø3,Ø4,Ø5,Ø6,Ø7,Ø8,Ø9,ØA,ØB,ØC,ØD,ØE,ØF,1Ø,11,12,13,
14,15,16,17,18,19,1A,1B,1C,1D,1E,1F,2Ø,21,22,23,
END OF PASS Ø2

RDM> CLK STOPPED CSAD ØØØØ NEXT Ø8ØØ      ! Stopped at control
RDM>                                      ! store address
                                          ! ØØØØ.
```

6.3 SHOW FLAGS

Syntax: SH FL

This command displays the current states of the program control flags listed in Table 6-1.

Sample Dialogue:

```
MIC>SH FL<RET>                            ! Show flags.

FLAGS SET: LO, NE, BE,
FLAGS CLEAR: HA, SA, IB, QA, TR

MIC>
```

Table 6-1 Program Control Flags

Flag	Description
HALT	Halt. Call the monitor when an error is detected and the error message has been typed.
LOOP	Loop on error. This flag is useful only in the event of a program-detected error. You may set the flag with the Loop command, or you may set it manually by typing SE FL LO. When this flag is set and the program detects an error, the test loops on the minimum amount of code necessary to re-create the error. However, when you set the LOOP flag manually, you must clear the HALT flag and set the NER flag to loop the program on error continuously.

Table 6-1 Program Control Flags (Cont)

Flag	Description
	The loop executed may include pseudo instructions (8Ø85 operations) and DCS microinstructions, running from the ERRLOOP instruction to the IFERROR instruction in the failing test. Or the loop may include DCS microinstructions only. If the IB flag is set or a microinstruction trap occurs, the program does not loop on the microinstructions in DCS. See Paragraph 7.17 (BRSTCLK) and Paragraph 6.15 (Loop command). Error messages are not inhibited while the loop is executing unless the NER flag is set. Type CTRL/C to escape from the loop and return to the monitor.
NER	No error report. If this flag is set, the program does not report errors.
BELL	Bell on error. If this flag is set, the program rings terminal bell on the first occurrence of an error and on every fifth occurrence.
SA	Signature analysis. If this flag is set and you type the Loop command, the program loops on the test in progress, between the BEGINSA and ENDSA pseudo instructions. The loop occurs whether or not the test detects errors. Set the SA flag when you use a signature analyzer to help diagnose faults. This flag provides two sync points on the backplane: a start/stop window (slot 6, pin C75) and a clock pulse (slot 6, pin C73). With the test looping and the signature analyzer connected to the sync points, the signature analyzer analyzes any test point that the probe samples. The signature analyzer displays a value (signature) if the signal pattern is steady. Compare this value with the corresponding value from a known good module to locate failures.
QA	Quality assurance. If this flag is set, the program responds to each test as if it had detected a failure.
IB	Inhibit burst flag. If this flag is set, the microcode in DCS does not return control to DCS address Ø.
TR	Trace flag. If this flag is set, the monitor types test names as well as test numbers.

6.4 SET FLAG

Syntax: SE FL <flag-name-list>

This command sets (enables) any of the program control flags.

Sample Dialogue:

```
MIC>SE FL HA                     ! Set HALT flag.
MIC>SE FL LO                     ! Set LOOP flag.
MIC>SE FL NE                     ! Set NO ERROR
                                 ! REPORT flag.
MIC>SE FL BE                     ! Set BELL flag.
MIC>SE FL SA                     ! Set SIGNATURE
                                 ! ANALYSIS flag.
MIC>SE FL QA                     ! Set QUALITY
                                 ! ASSURANCE flag.
MIC>SE FL IB                     ! Set INHIBIT BURST flag.
MIC>SE FL LO NE BE               ! Set LOOP, NO ERROR, and BELL
                                 ! flags.
```

6.5 CLEAR FLAG

Syntax: CL FL <flag-name-list>

This command clears any of the program control flags.

Sample Dialogue:

```
MIC>CL FL HA<RET>                ! Clear HALT flag.
MIC>
```

6.6 SET STOP-ON-MICROMATCH

Syntax: SE SO:<cs-address>

This command stops execution of code in DCS at the specified address. DCS addresses range from 1800 to 183F. Add 1800 to the desired DCS address. Type the Continue Command after the MIC> prompt to proceed from your current halt, after typing SE SO <CS-address> to get to that address.

Sample Dialogue:

```
MIC>SE SO:1803<RET>                    ! Stop at DCS address 3.
MIC>CO<RET>                            ! Continue.
MICRO BREAK MATCH, CS ADDR=1803, DCS ADDR=03.
MIC>
```

The function of the Set Stop-On-Micromatch command may be deceiving at times. Suppose you want to stop microcode execution at a location that follows a microinstruction specifying the NEXT field.

Example

```
DCS_DATA

;% ØØ   NOP
;% Ø1   MEMSCAR_R[TEMPØ]                  ; Physical/Virtual
                                          ; address to MEMSCAR.
;% Ø2   MEMSCR_R[TEMP1]                   ; Set MME off.
;% Ø3   MEMSCAR_R[TEMP2]                  ; Reference cache only
                                          ; to MEMSCAR.
;% Ø4   MEMSCR_R[TEMP3]                   ; Turn CMI off.
;% Ø5   PSL(PREV_CURM ISCURM_R[TEMP4])    ; Set mode to
                                          ; executive.
;% Ø6   STROBE.V.BUS,WB_RDM,PTE CHECK READ?,NEXT/18ØØ
                                          ; Perform PTE check.
;% Ø7   NOP,STOP.CLOCK.LATCH.CS.ADDR      ; Stop CPU clock.
```

In this example, typing SE SO:18Ø7 would not get you to DCS address 7. Instead, to get to DCS address 7, you must set stop-on-micromatch to stop at the address specified by the NEXT field in DCS address 6. This is necessary because Set Stop-on-Micromatch compares the control store address specified in the command with the address asserted on the NEXT control store lines.

Example

```
MIC>SE SO:18ØØ<RET>                  ; Stop at DCS address 7.
MIC:CO<RET>                          ; Continue.
MICRO BREAK MATCH, CS ADDR = 18ØØ, DCS ADDR = Ø7
MIC>
```

The NEXT address field never affects the DCS microcode. When DCS is active, the NEXT address field merely selects the DCS range of addresses (18ØØ to 183F). DCS address control resides in a counter. The counter can only increment or return to Ø. A microdiagnostic test specifies the NEXT field to test the microaddress generation function, not to create a microcode jump.

6.7 CLEAR STOP-ON-MICROMATCH

Syntax: CL SO[:<cs-address>]

Use this command to clear the stop-on-micromatch function. Add 1800 to the desired DCS address to create a scope sync pulse at that address. If you specify <cs-address>, a scope sync pulse is generated on slot 6, pin C81 when the current address matches <cs-address>. The pulse occurs with the M clock that marks the beginning of the specified microcycle.

Sample Dialogue:

```
MIC>CL SO:1803<RET>          ! Clear stop-on-micromatch
                             ! function at DCS address 3.
                             ! Load 1803 into the RDM match
                             ! register.
MIC>CO<RET>                  ! Continue.
```

6.8 SET CONTROL FILE

Syntax: SE CF:<dcs-address> <bit-number>

This command sets the specified <bit-number> in the control file of the specified <dcs-address>, bits <95:88>. See Paragraph 8.6. Table 6-2 lists the control file bit functions.

Sample Dialogue:

```
MIC>SE CF:3 4<RET>           ! Set control file bit
                             ! 4 (HRWBUS) in
                             ! DCS address 3.
MIC>
```

6.9 CLEAR CONTROL FILE

Syntax: CL CF: <dcs-address> <bit-number>

This command clears the specified bits in the control file of the <dcs-address>.

Sample Dialogue:

```
MIC>CL CF:3 4<RET>           ! Clear control file bit
                             ! 4 (HRWBUS) in
                             ! DCS address 3.
```

Table 6-2 Control File Bit Functions

Bit	Mnemonic	Function
Ø	VSTB	Strobe the visibility bus at the rising edge of the next M clock.
1	DCSACL	Clear the DCS address register.
2	STPCLK	Stop the VAX-11/75Ø CPU clock.
3	ENTRACE	Latch the trace register at the rising edge of the next M clock. When this bit is not set, the trace register monitors the control store address lines continuously.
4	HRWBUS	Read the WBus by strobing the WH register at the rising edge of the next M clock.
5	WBUSDR	Enable the WD register onto the WBus (write WBus).
6	STBCMI	Read CMI data, by strobing the MH register with CMI data at the rising edge of the next M clock.
7	SATRIG	Inhibit clocking the signature analyzer on the next M clock. This signal connects to a post on the module for use by the signature analyzer (backplane pin C73 on slot 6.)

6.10 SET STEP INSTRUCTION

Syntax: SE ST IN[:<test-pc>]

This command steps through the pseudo instructions (8085 code) in the current test. If you specify <test-pc>, the step function does not start until the instruction at the <test-pc> address is ready to execute. If you do not specify <test-pc>, the step function begins at the next pseudo instruction of the current test, following a Loop or Continue command.

In either case, when the program stops at the next pseudo instruction, press the space bar to continue stepping through the pseudo instructions. Type a carriage return to escape from the step mode back to the monitor.

Sample Dialogue:

Example 1

```
MIC>SE ST IN<RET>                    ! Set step instruction.
MIC>LO<RET>                          ! Start looping.
     TPC = 001F                      ! Test PC 1F.
                                     ! Space bar.
     TPC = 0024
     <RET>                           ! Carriage return.
MIC>
 .
 .
 .
```

Example 2

```
MIC>SE ST IN:2C<RET>                 ! Set step instruction
                                     ! starting at 2C.
MIC>CO<RET>                          ! Continue
     TPC = 002C                      ! Test PC is 2C.
                                     ! Space bar.
     TPC = 0038
                                     ! Space bar.
     TPC = 003E
     <RET>                           ! Carriage return.
MIC>
```

6.11 SET STEP CYCLE

Syntax: SE ST CY

This command steps through DCS microinstructions one CPU machine cycle at a time.

After typing in this command, you must type the Continue, Loop, or Diagnose commands to begin stepping. After the program stops at the first DCS address in the current test, press the space bar to step through the test. Type a carriage return to escape and return control to the monitor.

Sample Dialogue:

```
MIC>SE ST CY<RET>                    ! Set step cycle.
MIC>LO<RET>                          ! Loop.
     DCS ADDR = Ø1
                                     ! Space bar.
     DCS ADDR = Ø2
                                     ! Space bar.
     DCS ADDR = Ø3
     <RET>                           ! Carriage return.
MIC>
```

6.12 SET STEP TICK

Syntax: SE ST TI

This command steps through DCS microinstructions, stopping twice in every CPU machine cycle, as a function of the phase clock.

After typing this command, you must type the Continue, Loop, or Diagnose commands to begin stepping. After the program stops at the first DCS address in the current test, press the space bar to step through the test. Type a carriage return to escape and return control to the monitor.

Sample Dialogue:

```
MIC>SE ST TI<RET>                    ! Set step tick.
MIC>LO<RET>
     DCS ADDR = ØØ                   ! First half of first cycle.
                                     ! Space bar.
     DCS ADDR = ØØ                   ! Second half of first cycle.
                                     ! Space bar.
     DCS ADDR = Ø1                   ! First half of second cycle.
                                     ! Space bar.
     DCS ADDR = Ø1                   ! Second half of second cycle.
     <RET>                           ! Carriage return.
MIC>
```

6.13 SHOW VISIBILITY BUS

Syntax: SH VB

This command strobes the visibility bus (VBus) and prints out the current signal states. This command is most useful after executing the Initialize pseudo instruction, and when stepping through microcode in DCS. Show Visibility Bus displays the 4Ø bits of the visibility bus, with the low-order bit on the right.

Sample Dialogue:

```
MIC>SH VB<RET>
     VBUS= 1Ø1Ø1Ø11,Ø1ØØØØ1Ø,ØØ11ØØØØ,1Ø11Ø1Ø1,ØØ111111
MIC>
```

Table 6-3 lists the signal names, bit numbers, module names, and print set sheets for the visibility bus signals.

6.14 CONTINUE

Syntax: CO

This command continues testing after the program stops, following error detection or CTRL/C. This command does not modify any flags.

Sample Dialogue:

```
MIC>DI<RET>                               ! Run all tests.

ECKAA-VØ.22 MIC-VØØ.Ø4
Ø1,
?ERROR: ØØ3Ø TEST: Ø1 SUBTEST: Ø1         ! Error report.

DATA:    AAAAAAAA
         AAA8AAAA
         ØØØØØØØ1
FAILING MODULES: MIC,

MIC>CO<RET>                               ! Continue.
Ø2,Ø3,Ø4,Ø5,Ø6,Ø7,Ø8,Ø9,ØA,ØB,ØC,ØD,ØE,ØF,1Ø,11,12,13,14,
15,16,17,18,19,1A,1B,1C,1D,1E,1F,2Ø,21,22,23
END OF PASS Ø1

RDM> CLK STOPPED CSAD ØØØØ NEXT Ø8ØØ ! Stopped   at   control
RDM>                                 ! store address ØØØØ.
```

If you get an "unexpected clock stop" error message, you must type DI to resume testing. Typing CO simply returns you to the monitor.

Table 6-3 Visibility Bus Signals

Bit No. (Hex)	Bit No. (Decimal)	Signal Name	Module and Print Set Sheet No.
ØØ	Ø	UBIØ3 FORCE TB PE L	UBI14
Ø1	1	UBIØ3 FORCE CACHE PE L	UBI14
Ø2	2	CS HNEXT PAR H	UBI14
Ø3	3	UBIØ3 RTUT DINH L	UBI14
Ø4	4	UBIØ3 BUSF PAR H	UBI14
Ø5	5	UBI14 INT PEND L	UBI14
Ø6	6	UBI14 UB INT GRANT H	UBI14
Ø7	7	UBI11 CON HALT L	UBI14
Ø8	8	MICRO VECTOR 3 H	MICØ4
Ø9	9	MICRO VECTOR 2 H	MICØ4
ØA	1Ø	MICRO VECTOR 1 H	MICØ4
ØB	11	MICRO VECTOR Ø H	MICØ4
ØC	12	MICØ7 GEN DEST INH L	MICØ4
ØD	13	MICØ7 UTRAP L	MICØ4
ØE	14	MICØ4 LATCHED MBUS 15 L	MICØ4
ØF	15	MICØ4 PROC INIT L	MICØ4
1Ø	16	MICØ4 MSRC XB H	MICØ4
11	17	MICØ4 MEM STALL H	MICØ4
12	18	MICØ4 STATUS VALID H	MICØ4
13	19	MICØ5 UB REQ H	MICØ4
14	2Ø	MICØ7 CORR DATA INT L	MICØ4
15	21	MICØ7 WR BUS ERR INT L	MICØ4
16	22	Not Used, always Ø	MICØ4
17	23	Not Used, always Ø	MICØ4
18	24	DPM17 INSTR FETCH H	DPM22
19	25	DPM17 DO SRVC L	DPM22
1A	26	DPM16 IRD1 H	DPM22
1B	27	DPM14 UVCTR BRANCH H	DPM22
1C	28	DPM14 DISABLE HI NEXT H	DPM22
1D	29	DPM2Ø CS PARITY ERROR H	DPM22
1E	3Ø	DPM14 LD DSR L	DPM22
1F	31	DPM17 PSL CM H	DPM22
2Ø	32	DPM19 ISIZE Ø L	DPM22
21	33	DPM19 ISIZE 1 L	DPM22
22	34	DPM18 DST RMODE H	DPM22
23	35	DPM13 TIMER INT L	DPM22
24	36	DPM19 D SIZE Ø H	DPM22
25	37	DPM19 D SIZE 1 H	DPM22
26	38	DPM11 MCS TMP L	DPM22
27	39	UBI13 MSEQ INIT L	DPM22

6.15 LOOP

Syntax: LO

This command puts the program in an error loop after it detects and reports an error. If the SA flag is set, the loop will be a signature analysis loop. The command clears the HALT flag, sets the NER and LOOP flags, and performs a Continue command function.

Sample Dialogue:

```
?ERROR: ØØ3Ø TEST: Ø1 SUBTEST: Ø1

DATA:   AAAAAAAA
        AAAA8AAA                    ! Error report.
        ØØØØØØØ1
FAILING MODULES: MIC,

MIC>SE ST CY<RET>                   ! Set step cycle.

MIC>LO<RET>                         ! Loop.
        DCS ADDR= ØØ                ! First DCS address.
        DCS ADDR= Ø1                ! Press space bar to step.
        DCS ADDR= Ø2
        DCS ADDR= Ø3
        DCS ADDR= Ø4
        DCS ADDR= Ø5
        DCS ADDR= Ø6
        DCS ADDR= ØØ                ! Loop, and return to the
        DCS ADDR= Ø1                ! first DCS address. See
        DCS ADDR= Ø2                ! Paragraph 7.17 for
        DCS ADDR= Ø3                ! conditions under which
        DCS ADDR= Ø4                ! the DCS address is not
        DCS ADDR= Ø5                ! reset to Ø.
        <RET>                       ! Carriage return.
MIC>
```

6.16 RETURN

Syntax: RE

This command returns the monitor from the microdiagnostic program to RDM console command mode.

Sample Dialogue:

```
MIC>RE<RET>                        ! Return.
CLK STOPPED CSAD ØØØØ NEXT Ø8ØØ
RDM>                               ! RDM console command mode
                                   ! prompt.
```

CHAPTER 7
PSEUDO INSTRUCTIONS

7.1 SCOPE

In VAX-11/750 microdiagnostic programs, the monitor interprets and implements pseudo instructions. Functions are called pseudo instructions because they consist of many 8085 instructions. The rest of this chapter describes the pseudo instructions used in diagnosing the VAX-11/750.

7.2 INITIALIZE

Syntax: INITIALIZE

This pseudo instruction initializes the VAX-11/750 CPU as follows.

1. Cycles DC LO
2. Asserts PROCINIT (processor initialize)
3. Disables CMI

7.3 LOAD DCS

Syntax: LOADDCS <src-addr>, [src-index], <dcs-addr>, <wrd-cnt>

This pseudo instruction moves <wrd-cnt> microwords from the 8085 memory buffer to DCS, starting at <dcs-addr>. <Src-addr>, indexed by [src-index], specifies the starting 8085 memory address.

7.4 LOAD FIELD

Syntax: LDFIELD <src-addr>,[src-index],[data-type],<dst-addr>, <start-bit>, or <address-of-start-bit-table> <end-bit>, or <address-of-end-bit-table> [longlit], [start-bit-ind], [end-bit-ind], [no-parity], [NS]

This pseudo instruction acts only on RDM diagnostic control store (CS). It is normally used before the LOADDCS pseudo instruction to set up a microword before moving it into DCS address space. LDFIELD moves a field of length [data-type] indexed by [src-index], to a field in <dst-addr> specified by <start-bit> and <end-bit>.

If you also specify [start-bit-ind] and [end-bit-ind], the field is specified by items in a table. Those items are referenced by <address-of-start-bit>, indexed by [start-bit-ind] and <address-of-end-bit>, indexed by [end-bit-ind]. If you specify [longlit], data is inverted before being placed in the LONLIT field. If you specify [no-parity], the monitor does not calculate parity on that microword. Use the optional [NS] argument to load data unscrambled into the DCS. [NS] loads data other than microwords, which must be scrambled when loaded into the DCS.

Sample Dialogue:

```
1$:
;% NOP,NEXT/Ø                       ; Microinstruction.
2$:  .BYTE      ^X3F
     .
     .
     .
LDFIELD         2$,,B,1$,Ø,7,
LOADDCS         1$,,Ø,1
```

In the previous example, the LDFIELD instruction moves data at 2$, 3F, to the microinstruction stored in 8Ø85 memory at 1$. The 3F is inserted from bit Ø to bit 7. The microinstruction at 1$ becomes NOP,NEXT/3F. LOADDCS then moves this microinstruction to DCS address Ø. The next subtest may use the same microinstruction, but place a different value in any field with the LDFIELD pseudo instruction.

7.5 LOAD REGISTER

Syntax: LOADREG <reg-addr>,<src-addr>,[src-index],[data-type]

This pseudo instruction loads an RDM register from the 8Ø85 memory. LOADREG moves the contents of <src-addr>, indexed by [src-index], to the RDM register specified by <reg-addr>. If [data-type] is BYTE or unspecified, the instruction performs an 8-bit load. A [data-type] of WORD produces a 16-bit load, and a [data-type] of LONG produces a 32-bit load.

Sample Dialogue:

```
LOADREG         WDREG,1$,I,L

1$:             .LONG      ^XFFFFFFFF
```

In the previous example, the LOADREG instruction loads 32 bits of the WD register with ones. Assume that the index I equals 1.

7.6 LOAD INDEX

Syntax: LDINDEX <index-nam>,{<index-value>/
<index-value-add>,[index]}

This pseudo instruction initializes the specified index, <index-nam> (I,J, or K), to the specified value, <index-value>. The instruction also initializes the <index-nam> to <index-value-add> times [index].

7.7 SAVE INDEX

Syntax: SAVEINDEX <index-nam>,<dst-addr>

This pseudo instruction moves the current value of the specified index, <index-nam>, into <dst-addr> in the DCS.

7.8 INCREMENT INDEX

Syntax: INCINDEX <index-nam>

This pseudo instruction increments the current value of <index-nam> (I,J, or K).

7.9 STUCK AT ZERO OR ONE PATTERN

Syntax: SA01PAT <index-nam>,<dst-addr>,[longlit],[dst-addr-x]

This pseudo instruction selects one of the following patterns each time it is executed.

```
P1 10101010101010101010101010101010 (B)
P2 01010101010101010101010101010101 (B)
P3 00110011001100110011001100110011 (B)
P4 00001111000011110000111100001111 (B)
P5 00000000111111110000000011111111 (B)
P6 00000000000000001111111111111111 (B)
```

The pseudo instruction chooses a pattern according to the current value of <index-nam>. If you do not specify [longlit], it places the pattern in the test data region of the DCS at <dst-addr>. An <index-nam> of 1 selects pattern P1, and so on. The current value of <index-nam> must be less than or equal to 6.

If you specify [longlit], the pseudo instruction places the pattern in bits <62:31> of the microword starting at <dst-addr>. If you specify [dst-addr-x], the pattern is also placed at this address.

7.10 BEGIN SIGNATURE ANALYZER

Syntax: BEGINSA

If the SA flag is set, this pseudo instruction causes the program to loop between the BEGINSA and ENDSA pseudo instructions.

7.11 END SIGNATURE ANALYZER

Syntax: ENDSA

This pseudo instruction checks the SA flag. It marks the end of the signature analyzer loop.

If the SA flag is clear, the program executes the next pseudo instruction. See the example in Paragraph 8.3.

If you type CTRL/C, control returns to the monitor.

7.12 LOOP

Syntax: LOOP <loop-name>,<start-value>,<end-value>

This pseudo instruction initializes loop parameters for a specified index <loop name> (I,J, or K). If the <start-value> is less than the <end-value>, the loop is an incrementing loop. If the <end-value> is less than the <start-value>, the loop is a decrementing loop. Loops of this type are used to repeat a test or subtest, using a different data pattern indexed by <loop-name> each time.

7.13 END LOOP

Syntax: ENDLOOP <loop-name>

This pseudo instruction terminates a programmed loop. The program jumps to the pseudo instruction following the corresponding LOOP pseudo instruction, unless the required number of loops have been completed. A loop of this type functions whether or not an error has been detected.

7.14 ERROR LOOP

Syntax: ERRLOOP

This pseudo instruction saves the test PC. The instruction works with the IFERROR pseudo instruction; when the test detects an error and the LOOP flag is set, the program loops on the error.

7.15 IF ERROR

Syntax: IFERROR [data-cnt],[gate-array-list],[module-name]

This pseudo instruction checks the results of one of the three compare pseudo instructions: CMPREG, CMPREGMSK, and CMPVBUS.

Based on flag settings, the instruction executes a loop or continues with the program. The optional parameters specify the nature of the error messages. See the example in Paragraph 5.3.

[Data-cnt] specifies the number of RTEMP registers to type in the error message. For example, a [data-cnt] of 4 specifies RTEMPs Ø, 1, 2, 3. These registers contain test data as described in the error description in the test printout.

7.16 FETCH

Syntax: FETCH <dcs-addr>,[MIC-ADR-INH]

This pseudo instruction initiates a DCS microinstruction sequence. It loads the microword stored at <dcs-addr> into the control store latches. If you specify [mic-adr-inh], the instruction leaves the Micro Address Inhibit signal asserted after execution of the instruction to inhibit the processor from driving the control store lines.

7.17 BURST CLOCK

Syntax: BRSTCLK <stop-addr>, [INHIBIT]

This pseudo instruction bursts the VAX-11/75Ø CPU clock under normal circumstances. It checks to see that the microcode stopped at <stop-addr>. Control then passes to the next pseudo instruction.

If the following conditions exist, the instruction performs the functions set control file bit 1 (DCSACL) and clear control file bit 2 (STPCLK) at <stop-addr>.

- error occurs
- LOOP flag set
- [INHIBIT] argument not specified
- SA flag clear

In this way, BRSTCLK creates a DCS loop. Otherwise, control passes to the next pseudo instruction.

BRSTCLK also controls the step cycle, step tick, and stop-on-micromatch functions. See the Loop command description at Paragraph 6.15 and the LOOP flag description at Paragraph 6.3 for details of the looping function.

7.18 EXPECT MICROTRAP

Syntax: EXPUTRAP

This pseudo instruction enables the BRSTCLK function to recognize a microtrap. If a microtrap occurs as expected, the burst clock routine does not report an error and passes control to the next pseudo instruction.

If a microtrap occurs, the control store address lines are forced outside the range of DCS address space, 1800 to 183F. The RDM monitors the control store address lines and stops the CPU clock when they are forced outside the range of DCS.

If an EXPUTRAP pseudo instruction precedes a microtrap, the program handles the trap. Otherwise, the program prints an "unexpected clock stop" error message on the console terminal.

If a microtrap occurs, whether or not it is expected, you cannot create a DCS loop.

7.19 COMPARE REGISTER

Syntax: CMPREG <reg-addr>,<dst-addr>,[dst-index],[data-type],
[mode-type]

This pseudo instruction compares the contents of an RDM register <reg-addr> with <dst-addr>, indexed by [dst-index] in DCS.

If the [mode-type] is blank, the CMPREG pseudo instruction expects the data to be equal. If the [mode-type] is NE, it expects the data to be unequal.

On a comparison failure, the IFERROR pseudo instruction following CMPREG prints an error message.

7.20 COMPARE REGISTER MASKED

Syntax: CMPREGMSK <reg-addr>,<dst-addr>,[dst-index],<msk-addr>,
[msk-index],[data-type],[mode-type]

This pseudo instruction works exactly like CMPREG, except that the comparison is masked by <mask-addr>, and indexed by [mask-index]. Only bit positions where the mask equals 1 are compared, as shown in the following example.

```
Register Contents      1 1 1 1 1 1 1 0
Destination Contents   0 1 1 0 1 1 1 0
Mask                   0 1 1 1 0 0 0 0
```

Only these three bit positions are compared.

7.21 MASK

Syntax: MASK <src-addr>,[src-index],<msk-addr>,[msk-index]

This pseudo instruction masks bits in a byte of the DCS. Bits that are ones in the mask are preserved in the location at <src-addr>.

7.22 COMPARE VISIBILITY BUS

Syntax: CMPVBUS <data-table-ptr>,[index]

This pseudo instruction compares the values of bits on the visibility bus (VBus) with the expected values specified in the data table. If the comparison fails, the IFERROR pseudo instruction prints an error message.

Sample Dialogue:

```
        BRSTCLK   7                      ! End DCS program.
        CMPVBUS   2$,I                   ! Compare VBus results.
        IFERROR   ,<<ACV><ADK><UTR>>,    ! Possible failing array.

    2$: .BYTE     ^X4                    ! VBus data table.
        VBUSDATA  <^XØ8>,Ø               ! Iteration 1.
        VBUSDATA  <^XØ9>,Ø
        VBUSDATA  <^XØA>,1
        VBUSDATA  <^XØB>,1
```

In the preceding example the VBus data is compared with data in the table at 2$. .BYTE ^X4 at the top of the table indicates that the comparison involves four bits. VBUSDATA <^XØ8>, Ø indicates that bit 8 will be the first bit position compared, and the expected value is Ø.

7.23 NEW TEST

Syntax: NEWTEST <title>,[alt-test-number]

NEWTEST is always the first pseudo instruction in a test. It defines the beginning of the test and performs a variety of housekeeping functions.

7.24 SUBTEST

Syntax: SUBTEST [subtest number]

This pseudo instruction increments the current subtest number or takes the value given in the optional argument.

7.25 SKIP

Syntax: SKIP [address],[condition] [index_Ø] [index_1]

This pseudo instruction causes the program to skip to [address]. If you do not specify [address], the program skips to the ENDTEST pseudo instruction. If you specify [condition], it may be any of three types. If the [condition] is ONERROR, the program skips only when the error flag is set. If the [condition>] is NOERROR, the program skips only when the error flag is not set. If the [condition] is ONEQUAL, [index_Ø] and [index_1] are also specified. The SKIP pseudo instruction compares them and skips if they are equal.

7.26 END TEST

Syntax: ENDTEST

This pseudo instruction terminates each test. It tests various flags and acts accordingly.

If you type in DI TE:<test-number>, the program loops on the test. Otherwise, the instruction continues the program by loading the next test overlay from the TU58.

CHAPTER 8
MICRODIAGNOSTIC PROGRAM LISTINGS

8.1 SCOPE

In most cases, microdiagnostic program error messages provide enough information for you to isolate and repair faults in the processor. Error messages list the failing test number and call out suspected components and modules. However, if you change the specified parts and the same message prints after rerunning the program, the program listing provides a further troubleshooting resource. Use the microfiche card for the appropriate set of tests to locate the frame for the failing test, through the index in the lower right microfiche frame.

8.2 TEST OVERLAY LISTING FORMAT

Each test overlay consists of a test and associated data. A test may consist of several subtests. Although test overlays vary slightly, they use the same general format.

```
Test Title
Documentation
     Functional Description
     Test Algorithm
     Test Patterns
     Logic Description
     Assumptions
     Error Description
Subtest 1
     Pseudo instructions              ; comments
Subtest 2...
     Pseudo instructions              ; comments
Test data
     8Ø85 memory buffer data          ; comments
     DCS data                         ; comments
The monitor loads this directly into DCS.

     RTEMP data       ; Data loaded beginning at RTEMPØ.
     Cache data       ; Data loaded beginning at cache
                      ; address Ø.
```

NOTE

Before you analyze the test code and data, read the documentation carefully. It gives an overall description of the test and many useful details.

8.3 TEST EXECUTION FLOW

The execution flow for each subtest in a test consists of three functional steps.

1. Pseudo instructions set up test data and parameters. The 8Ø85 processor performs these functions.

2. Microcode in DCS executes.

3. Pseudo instructions check the results of microcode execution. If the check reveals no error, control passes to the next subtest or test. Flag settings, error conditions, and test structure cause the program to loop to the beginning of DCS, or to a loop point in the pseudo instructions, as appropriate.

Consider the test overlay listing in the following example.

I. Functional Description

This test consists of two subtests.

1. Cache Error Register
2. Cache Control Register

These are read/write registers in the CAK chip.

II. Test Algorithm

1. Modify DCS address 1 to select the appropriate temporary register for the subtest.

2. Set up a loop to select test patterns.

3. Load WD register (RDM) with test pattern.

4. Execute DCS program.

   a. Load memory status and control address register (MEMSCAR) with address in temporary register.

   b. Write test pattern in WDREG (RDM) to memory status and control register (MEMSCR).

   c. Read test pattern back to WHREG (RDM).

5. Compare results (expected and received).

6. If no error, go to step III for remaining test patterns.

III. Test Patterns

example of a test

Iteration	Test Pattern
1	A
2	5
3	3

IV. Logic Description

This test checks only the S/C registers that live in the CAK gate array. The registers are accessed via WBUS<27:24>. If on error, replacing the CAK does not work, try running the MEMSCAR test, which checks for a multiple addressing problem. If that fails, then one of the other gate arrays with S/C registers is causing trouble.

V. Assumptions

If SET STEP set we can single step thru' pseudo instructions

This test assumes that the WBus is operational.

VI. Error Description

EXPECTED MEMSCR DATA
RECEIVED MEMSCR DATA
LOOP COUNT

Error info

Explanation of pseudo-instruction in chapter 7.

```
ØØØ4   ;SUBTEST 1 Cache Error Register
ØØØ4   SUBTEST                              ; Cache error reg.
ØØØ6   INITIALIZE                           ; Initialize CPU.
ØØØ8   LOADDCS      4$,,Ø1,1                ; Modify DCS address Ø1.
ØØØF   BEGINSA                              ; Signature analyzer loop
                                            ; point.
ØØ11   LOOP         I,1,3                   ; Create test loop.
ØØ18   LOADREG      WDREG,1$,I,L            ; Test pattern to WD reg.
ØØ2Ø   ERRLOOP                              ; Error loop.
ØØ22   FETCH        Ø                       ; Start DCS program.
ØØ26   BRSTCLK      4                       ; End DCS program.
ØØ2A   CMPREGMSK    WHREG,1$,I,3$,,L,       ; Compare results.
ØØ36   IFERROR      ,<<CAK>>,               ; Possible failing array.
ØØ3C   ENDLOOP      I                       ; End test loop.
ØØ3E   ENDSA                                ; End signature analyzer
                                            ; loop.
ØØ41   ;SUBTEST 2 Cache Control Register

ØØ41   SUBTEST                              ; Cache control reg.
ØØ43   INITIALIZE                           ; Initialize CPU.
ØØ45   LOADDCS      2$,,Ø1,1                ; Change DCS address Ø1.
ØØ4C   BEGINSA                              ; Signature analyzer loop
                                            ; point.
ØØ4E   LOOP         I,1,3                   ; Create test loop.
ØØ55   LOADREG      WDREG,1$,I,L            ; Test pattern to WD reg.
ØØ5C   ERRLOOP                              ; Error loop.
ØØ5F   FETCH        Ø                       ; Start DCS program.
ØØ63   BRSTCLK      4                       ; End DCS program.
```

given to MICMON which sorts out an array no

```
0067 504  CMPREGMSK  WHREG,1$,I,3$,,L,   ; Compare results.
0073 505  IFERROR    ,<<CAK>>,           ; Possible failing array.
0079 506  ENDLOOP    I                   ; End test loop.
007C 507  ENDSA                          ; End signature analyzer
                                         ; loop.
007E 508  SKIP                           ; SKIP to line 568.
0084 513           BEGIN_TEST_DATA
        ;- - - - - - - - - - - - - - - - - - - - - - - - - - - -
0084 515 1$:          .LONG      ^X0A000000    ; Test pattern.
0088 516              .LONG      ^X05000000    ; Test pattern.
0086 517              .LONG      ^X03000000    ; Test pattern.
008E 518 2$:
008E 519 ;%     01    MEMSCAR_R[Temp1]         ; Rtemp for subtest 2.
009B 523 3$:          .LONG      ^X0F000000    ; Mask for CMPREGMSK
                                               ; pseudo op.
009F 524 4$:
009F 525 ;%     01    MEMSCAR_R[Temp0]         ; Rtemp for subtest 1.

00AA 532              BEGIN_CPU_DATA

0002 534              DCS_DATA

0001 536 ;%     00    NOP
000C 540 ;%     01    MEMSCAR_R[TEMP0]         ; Write add of S/C to
                                               ; MEMSCAR.
0017 544 ;%     02    WB_RDM,WCTRL/MEMSCR_WB   ; Write test pattern to
                                               ; MEMSCR.
0022 548 ;%     03    RDM_WB,WCTRL/MEMSCR      ; Read  test  pattern
                                               ; back to RDM.
002D 552 ;%     04    NOP,STOP.CLOCK           ; Stop CPU clock.

0038 557              RTEMP_DATA

0001 559              .LONG      ^X04000000    ; Cache error register
                                               ; address.
0005 560              .LONG      ^X06000000    ; Cache control
                                               ; Register address.

0009 562              END_CPU_DATA
00A2 566              END_TEST_DATA
        ;- - - - - - - - - - - - - - - - - - - - - - - - - - - - -

00A2 568              ENDTEST

00A2 569              NEWTEST ,\THIS_TEST      ; Terminate this test.
```

Drop thru to next sub-test if no fails.

Figure 8-1 illustrates the execution flow for the first subtest.

Note that FETCH gets control of the control store address lines. Then the microcode executes during BRSTCLK. The BRSTCLK pseudo instruction checks that the CPU clock stops at the DCS address specified (4). BRSTCLK then passes control to the next pseudo instruction.

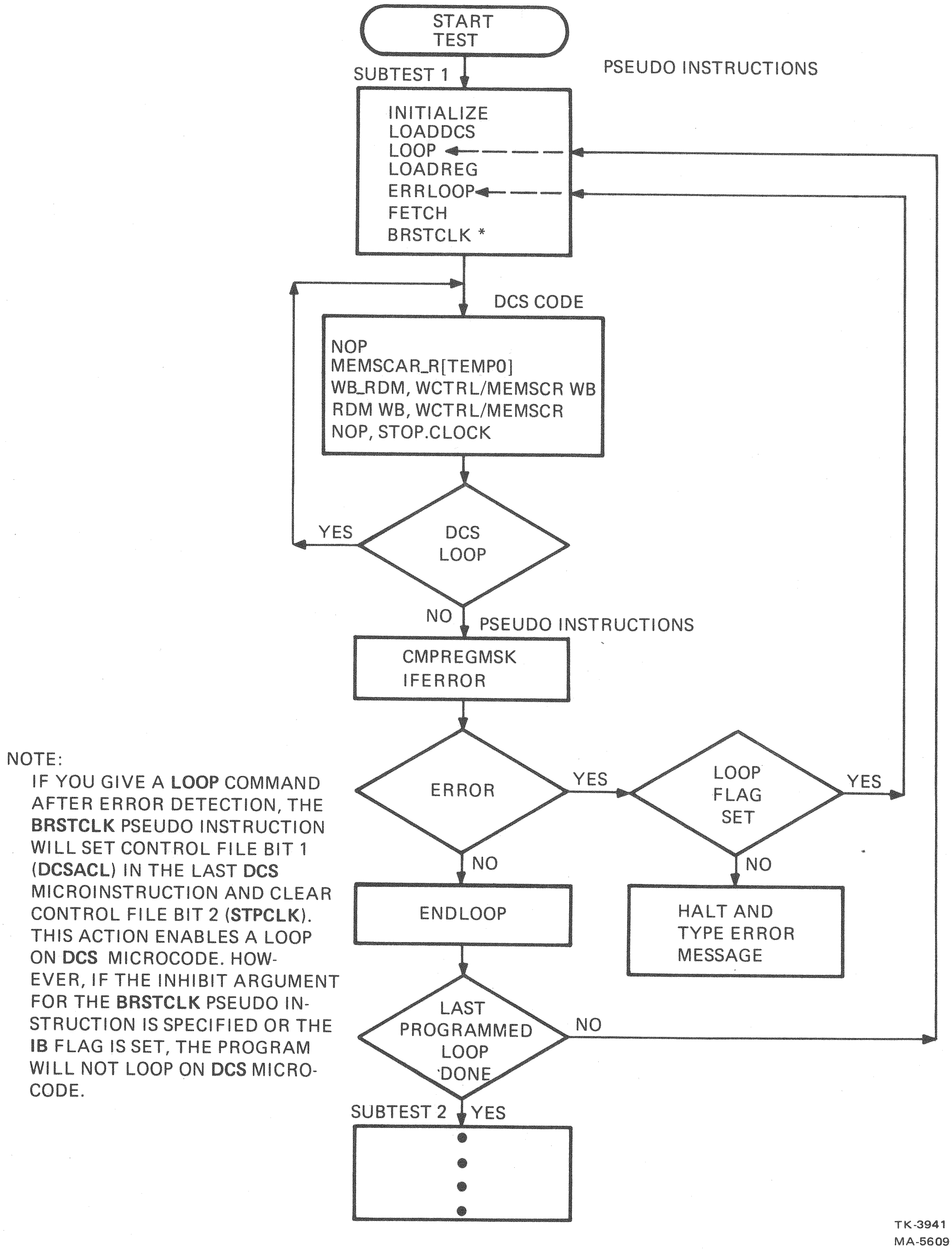


Figure 8-1 Sample Subtest Execution Flow

8.4 MICROINSTRUCTION REFERENCES

For detailed information on the microassembler, see the *MICRO2 User's Guide Reference Manual* (AA-H531A-TE).

For accurate interpretation of microorders, see the machine definitions (microorder definitions) and the macro definitions in the VAX-11/750 microcode listings. For special macros used only in microdiagnostics, see the microdiagnostic listings.

8.5 MICROINSTRUCTION INTERPRETATION TIPS

The DCS data in the listings consists of two types of statements: microorders, and microinstruction macros. Microorders take the form <field>/<value>, where <field> is a field of bits in the control store word and <value> is a hexadecimal number to be placed in that field. For example, WCTRL/MEMSCR, from line 548 of the example in Paragraph 8.3, puts the value identified by MEMSCR into the WCTRL field, bits <30:25>, of the control store word.

However, to find what the WCTRL field actually does with that value, you must check the machine definitions section of the VAX-11/750 microcode listings, under WCTRL. The comment in the listing for the WCTRL/MEMSCR microorder reads as follows.

;WBUS<27:24><-- Memory Status and Control Reg (@MEMSCAR)

The microorder therefore asserts bits <27:24> from the memory status and control register, addressed by MEMSCAR, on the WBus.

Macros generate the microcode for one or more microorders within one microinstruction. You can generally interpret the function of a macro by analyzing the macro name. Macros like NOP and STOP.CLOCK are obvious.

All macros that include an underscore symbol are register transfer macros. The transfer goes from the right term to the left term. However, when the underscore symbol is part of a microorder, it does not indicate the presence of a macro within the microorder.

Commas separate microorders and macros, when several are used to form one microinstruction.
Consider the DCS data from test 4 in the example in Paragraph 8.3.

Line 540: MEMSCAR_R[TEMP0]

Move the contents of the scratch pad register RTEMP0 to MEMSCAR, the memory status and control address register.

Line 544: WB_RDM,WCRTL/MEMSCR_WB

The WB_RDM macro moves data from the WDREG on the RDM to the WBus. WCTRL/MEMSCR_WB, according to the microorder definition in the VAX-11/750 microcode listing, moves WBus bits <27:24> to the memory status and control register. This is a write function.

Line 548: RDM_WB,WCTRL/MEMSCR

WCTRL/MEMSCR moves data bits <27:24> from MEMSCR to the WBus. The macro RDM_WB moves that data from the WBus back to the WHREG on the RDM. This is a read function.

8.6 DIAGNOSTIC CONTROL STORE AND CONTROL FILE

The diagnostic control store (DCS) contains 64 locations that are each 88 bits wide. The DCS address space runs from control store address 1800 to 183F. The first 80 bits in each word, bits <79:00>, correspond to the 80 bits of the VAX-11/750 microword. The upper eight bits in each word, bits <95:88>, make up the control file. They enable and disable control functions on the RDM. The intervening bit numbers, bits <87:80>, are unused. They are always zero.

The microdiagnostic tests control the eight bits of the control file with eight microinstruction macros (Table 8-1). These macro functions correspond to the set control file functions listed in Paragraph 6.8.

Table 8-1 RDM Control File Macros

Macro Name	Bit	RDMCF	Function
STROBE.V.BUS	88	0	Strobe the VBus at the rising edge of the next M clock.
DCS.ADDR_0	89	1	Clear the DCS address register. Return to DCS address 0.
STOP.CLOCK	90	2	Stop the CPU clock.
LATCH.CS.ADDR	91	3	Latch the trace register at the rising edge of the next M clock.
RDM_WB	92	4	Read data on the WBus into the WHREG on the RDM at the rising edge of the next M clock.
WB_RDM	93	5	Assert (write) the contents of the WDREG (on the RDM) on the WBus for the entire microcycle.
RDM_CMI	94	6	Read data on the CMI bus into the MHREG on the RDM at the rising edge of the next M clock.
INH.CLOCK.SA	95	7	Inhibit clocking the signature analyzer on M clock.

↑ Part of 88 bit wide U-insts
These additional 8 (80-87 is a gap)
used seperately only
with SH VB command.
See Para. 6.13 & 6.12

8.7 LOOPING AND STEPPING THROUGH TESTS

After you analyze the test description, code, and comments for the failing test, you may want to analyze certain signals in the processor. Locate points of interest in the circuit under test in the VAX-11/75Ø print set.

To loop on the failing test at machine speed, follow the command sequence shown in example 1.

Example 1

```
RDM>TE/C<RET>                            ! Load the microdiagnostic
                                         ! monitor.
LOADING MICRODIAGNOSTICS

MIC>DI TE:4<RET>                         ! Execute the failing
                                         ! test.
ECKAA-VO.22 MIC-VØØ.Ø4
Ø4,
?ERROR: ØØ36 TEST: Ø4 SUBTEST: Ø1

DATA:    ØAØØØØØØ                        ! Expected cache error register
                                         ! contents.
         Ø8ØØØØØØ                        ! Received cache error register
                                         ! contents.
         ØØØØØØØ1                        ! Loop count.
                                         ! See Error Description, example
                                         ! in Paragraph 8-3.
Failing Gate Arrays: CAK,
Failing Modules: MIC,
MIC>LO<RET>                              ! Loop on test 4.
```

To step through the entire test one pseudo instruction at a time, follow the steps shown in example 2. Note the flow of the subtest sequence in the TPC addresses.

Example 2

```
MIC>DI TE:4<RET>                         ! Execute the failing test.
ECKAA-VO.22 MIC-VØØ.Ø4
Ø4,
?ERROR: ØØ36 TEST: Ø4 SUBTEST: Ø1

DATA:    ØAØØØØØØ                        ! Expected data.
         Ø8ØØØØØØ                        ! Received data.
         ØØØØØØØ1                        ! Loop count.
Failing Gate Arrays: CAK,
Failing Modules: MIC,
MIC>SE ST IN                             ! Set step instruction.
```

```
MIC>CO<RET>
          TPC= 003C              ! Test continues at this point.
          TPC= 0018              ! Restart subtest 1, using
          TPC= 0020              ! the second test pattern, I = 2.
          TPC= 0022
          TPC= 0026
          TPC= 002A
          TPC= 0036
          TPC= 003C
          TPC= 0018              ! Restart subtest 1, using
          TPC= 0020              ! the third test pattern, I = 3.
          TPC= 0022
          TPC= 0026
          TPC= 002A
          TPC= 0036
          TPC= 003C
          TPC= 003E
          TPC= 0041              ! Start subtest 2, using
          TPC= 0043              ! the first test pattern, I = 1.
          TPC= 0045
          TPC= 004C
          TPC= 004E
          TPC= 0055
          TPC= 005C
          TPC= 005F
          TPC= 0063
          TPC= 0067
          TPC= 0073
          TPC= 0079
          TPC= 0055              ! Restart subtest 2, using
          TPC= 005C              ! the second test pattern, I = 2.
          TPC= 005F
          TPC= 0063
          TPC= 0067
          TPC= 0073
          TPC= 0079
          TPC= 0055              ! Restart subtest 2, using
          TPC= 005C              ! the third test pattern, I = 3.
          TPC= 005F
          TPC= 0063
          TPC= 0067
          TPC= 0073
          TPC= 0079
          TPC= 007C
          TPC= 007E
          TPC= 00AA
          TPC= 0004              ! Loop to restart subtest 1,
          TPC= 0006              ! using the first test pattern,
          TPC= 0008              ! I = 1.
CTRL/C    TPC= 000F^C            ! Type CTRL/C (or carriage
                                 ! return) to return to the
                                 ! microdiagnostic monitor.
MIC>
```

To step through the DCS code with a specific pattern in a specific subtest, step through the pseudo instructions to the proper point. Then type SE ST CY (set step cycle) to stop at each DCS instruction. Example 3 shows how you can step through the DCS code, using test pattern 2 in subtest 1.

Example 3

```
MIC>DI TE: 4<RET>                      ! Execute the failing test.
ECKAA-VO.22 MIC
Ø4,
?ERROR: ØØ34 TEST: Ø4 SUBTEST: Ø1

DATA:   ØAØØØØØØ                       ! Expected data.
        Ø8ØØØØØØ                       ! Received data.
        ØØØØØØØ1                       ! Loop count.
Failing Gate Arrays: CAK,
Failing Modules: MIC,
MIC>SE ST IN<RET>                      ! Set step instruction.

MIC>CO<RET>

        TPC= ØØ3C
        TPC= ØØ18                      ! Restart subtest 1, using
        TPC= ØØ2Ø                      ! the second test pattern.
        <RET>                          ! Carriage return.

MIC>SE ST CY<RET>

MIC>CO<RET>                            ! Step through DCS
                                       ! with the second test pattern.

        DCS ADDR= ØØ
        DCS ADDR= Ø1
        DCS ADDR= Ø2
        DCS ADDR= Ø3
        DCS ADDR= Ø4
MIC>
```

Note that control returns to the microdiagnostic monitor automatically, after execution of the instruction at DCS ADDR= Ø4.

Example 4 shows how you can force a loop in the DCS code with the Set Control File command. Set control file bit 1 and clear bit 2 in the last DCS word in the sequence.

Example 4

```
MIC>SE CF: 4 1<RET>                    ! Set CF bit 1 in DCS
                                       ! address 4.
MIC>CL CF: 4 2<RET>                    ! Clear CF bit 2 in DCS
                                       ! address 4.
MIC>CO<RET>                            ! Continue.
```

APPENDIX A
RDM CABLE CONNECTIONS AND TEST POINTS

Table A-1 describes the interconnection of the RDM to the modem. Table A-2 lists the backplane pin numbers for the local terminal, the TU58, and the front panel cables. Table A-3 lists backplane pin numbers for specific RDM test points.

Table A-1 RDM/Modem Signal Flow and Interconnection

RDM Pin	7016921 Cable Conn. Pin	EIA (RS-232C) Signal Name	EIA (Abbr.)	EIA Circuit
C0671	1	Protective Ground	GND	AA
C0672	7	Signal Ground	GND	AB
C0678	2	Transmitted Data	TXD	BA
C0680	3	Received Data	RXD	BB
C0690	4	Request to Send	RTS	CA
C0688	5	Clear to Send	CTS	CB
C0686	6	Data Set Ready	DSR	CC
C0682	20	Data Terminal Ready	DTR	CD
C0692	22	Ring Indicator	RI	CE
C0684	8	Carrier Detect	CD	CF
	Not used	Force Busy	FB	CN

Table A-2 Backplane Pins for Local Terminal, TU58, and Front Panel Cables

Pin	Signal Name
Local Terminal	
CØ624	Ground
CØ634	RDM terminal serial out
CØ632	RDM terminal serial in
CØ645	Console baud rate A L
CØ646	Console baud rate B L
CØ649	Console baud rate C L
CØ65Ø	Console baud rate D L
TU58	
BØ686	TU58 serial out (signals between TU58 and RDM)
BØ688	TU58 serial in (signals between TU58 and RDM)
BØ685	EIA TU serial out L (signals between RDM and CPU)
BØ687	EIA TU serial in L (signals between RDM and CPU)
Front Panel	

Front panel cable is plugged into SET A (top half), with cable pin 1 on top.

Table A-3 RDM-Specific Backplane Pins

Pin	Signal Name
CØ673	RDM SA CLK L (D)
CØ675	SA ST/SP L (D)
CØ674	RDM RESET L (R)
CØ681	RDM MATCH PULSE (D)

(D)=Drive, (R)=Receive.

The KC750-CA option usually comes with one of three modem types: General DataCom (GDC), Racal/Vadic, or DIGITAL. Figures B-1 through B-4 show jumper and switch settings for the different modems. These drawings are for information only. Read the manufacturer's manual before changing jumper or switch settings. Table B-1 lists jumper and switch settings for the DIGITAL DF02 modem.

CAUTION
It is an FCC violation to modify, replace, or repair the modem unless done by a duly authorized FCC representative.

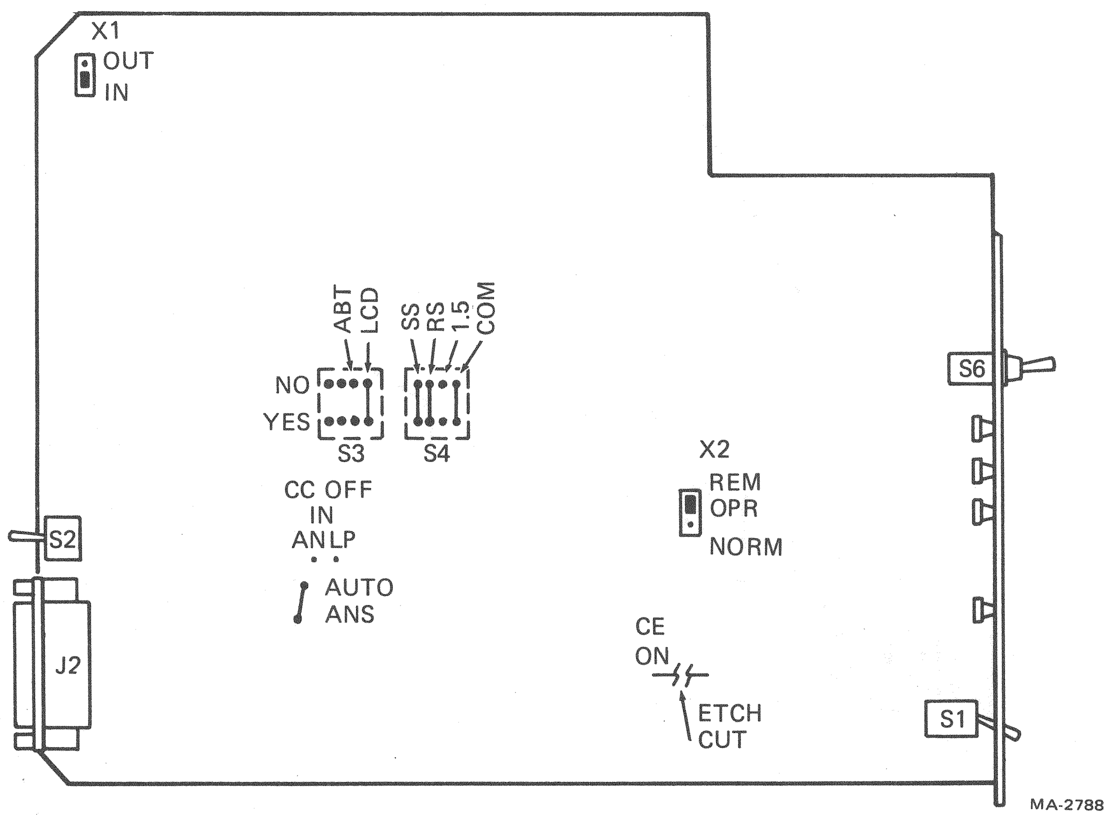


Figure B-1 GDC Jumper Configuration

RACAL/VADIC MODEM

OPTION JUMPER	JUMPER CONFIGURATION IN	OUT
A		*
B	*	
C	*	
D		*
E		*
F	*	
G	*	
H	*	

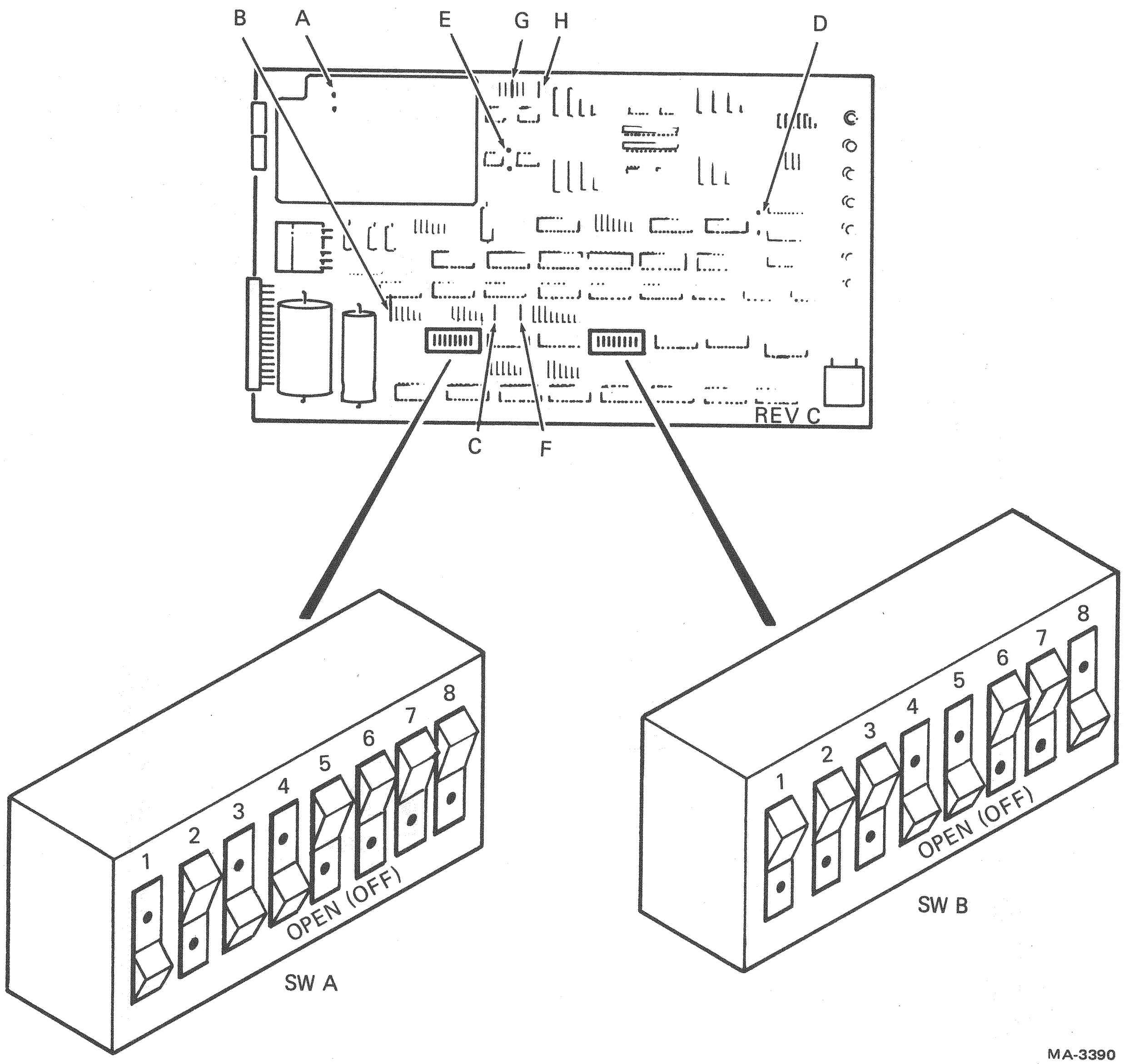


Figure B-2 Racal/Vadic Jumper Configuration and Switch Settings

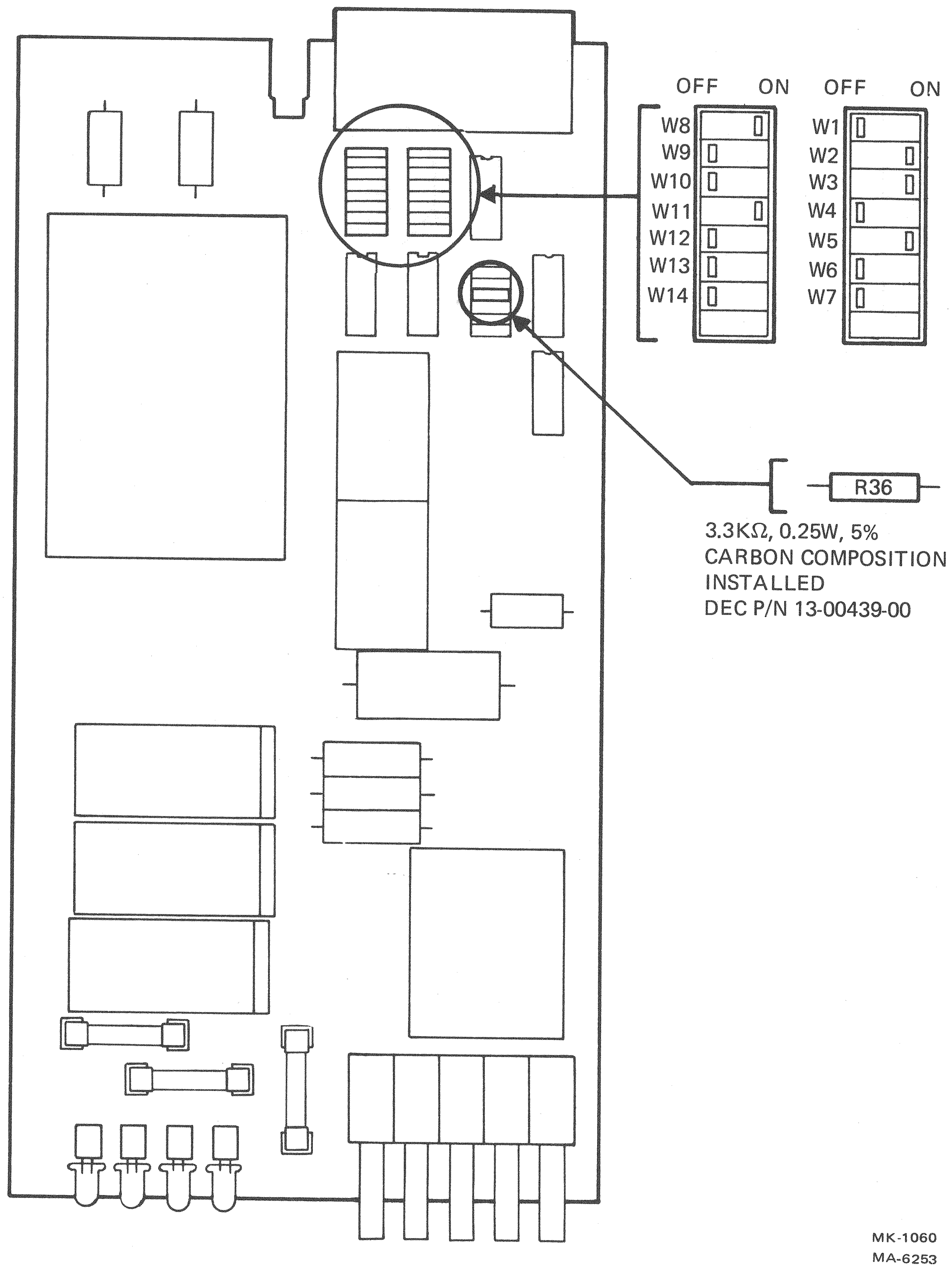


Figure B-3 DFØ2 Interface Board

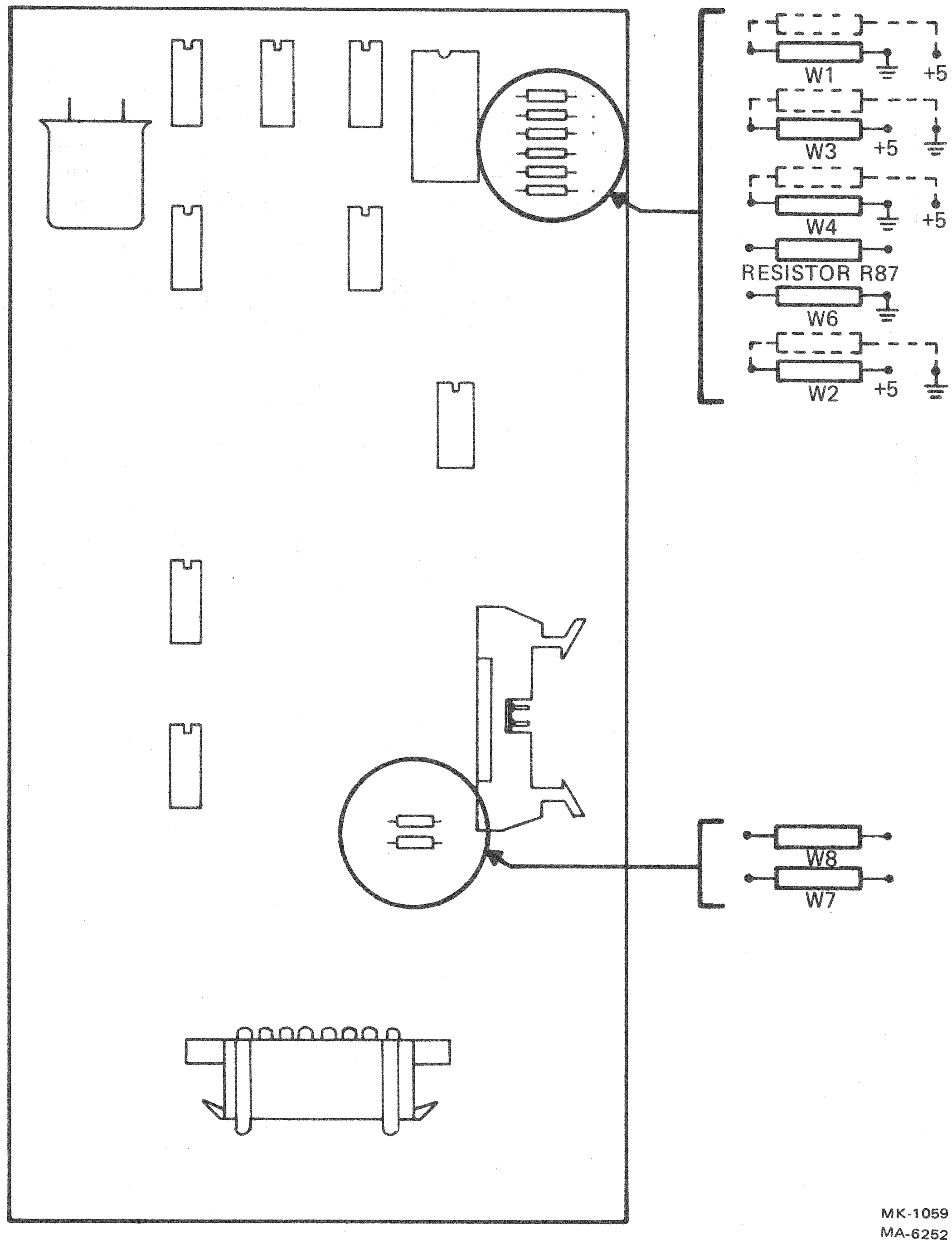


Figure B-4 DFØ2 Modem Board

Table B-1 DIGITAL DFØ2 Modem Jumper and Switch Settings

Jumper or Switch	Desired Setting
Interface Board	
W1	OFF
W2	ON
W3	ON
W4	OFF
W5	ON
W6	OFF
W7	OFF
W8	ON
W9	OFF
W1Ø	OFF
W11	ON
W12	OFF
W13	OFF
W14	OFF
Modem Board	
W1	Alternate setting (See Note.)
W2	Normal setting
W3	Normal setting
W4	Normal setting

NOTE

Alternate setting is shown by dotted lines in Figure B-4.

APPENDIX C
INSTALLATION ACKNOWLEDGEMENT

Figure C-1 shows an Installation Acknowledgement form. This form must be signed by the customer to ensure DIGITAL's full and free access to equipment.

digital

DIGITAL EQUIPMENT CORPORATION

VAX11/750 REMOTE DIAGNOSIS INSTALLATION ACKNOWLEDGEMENT

I, the undersigned, acknowledge receipt of Remote Diagnosis Kit (KC750).

Asset No. .... for use on VAX 11/750 ,

Serial No. ....

I recognize that this Remote Diagnosis Kit always remains the property of Digital Equipment Corporation, and agree not to remove or tamper with any part of it, nor to disclose or make any part of it available to a third party.

I further acknowledge that I will allow Digital full and free access for the purpose of removal of the Diagnosis Kit and restoration of the system to its original condition if for any reason the Field Service Agreement should be terminated.

Computer System to be located at: ____________________

Company ____________________

Address ____________________

City ____________________ *State* __________ *Zip Code* __________

By Authorized Representative ____________________ *Date* __________

Title ____________________

MA-5118

White—Customer Canary—Branch Pink—D.D.C.

Figure C-1 Installation Acknowledgement Form

APPENDIX D
RDM SPECIFICATIONS

Size:	31.12 cm (12.25 in) X 40.01 cm (15.75 in) extended hex
Pin Configuration:	3 X 94 - Standard L series
Power:	
Maximum	+5 V @ 12.9 A, +12 V @ 120 mA, -15 V @ 85 mA
Typical	+5 V @ 9.6 A, +12 V @ 60 mA, -15 V @ 30 mA
Technology:	TTL (Schottky, Low Power Schottky)
	MOS

Environmental standards (temperature, pressure, etc.) are the same as for the VAX 11/750.

APPENDIX E
MODEM OPERATING CHARACTERISTICS

E.1 INTRODUCTION

This appendix defines the required functional characteristics of modems used with DIGITAL remote diagnosis devices. Those devices operate with the following characteristics.

- 300 baud serial data communication
- Asynchronous (start/stop) character framing (one start bit, eight data bits, no parity bit, one stop bit)
- Full duplex
- Automatic answer
- Automatic disconnect

E.2 DEFINITIONS

Table E-1 defines the signals recognized by the RDM and modem. The table lists the signal name, followed by the EIA RS-232C circuit name, the CCITT V.24 circuit number, the position on a 25-pin D-type connector, and the signal definition.

Table E-1 Modem Signal Definitions

Name	RS-232C	V.24	25-pin	Definition
GND	AA	101	1	Protective ground. This signal provides a path between the remote diagnosis device and the modem for discharge of spurious potentials, such as static electricity.
GND	AB	102	7	Signal ground. This signal provides a reference level for the data and control signals that follow in this table.
TxD	BA	103	2	Transmit data (console to modem). This signal contains the serial bit stream to be sent from the console to the calling station.

Table E-1 Modem Signal Definitions (Cont)

Name	RS-232C	V.24	25-pin	Definition
RxD	BB	1Ø4	3	Receive data (modem to console). This signal contains the serial bit stream received by the modem from the calling station.
RTS	CA	1Ø5	4	Request to send (console to modem). This signal is asserted by the console, causing the carrier signal to be placed on the line. Called the station's answer mode carrier.
CTS	CB	1Ø6	5	Clear to send (modem to console). This signal is asserted by the modem to indicate that it has successfully placed its carrier signal on the line.
DSR	CC	1Ø7	6	Data set ready (modem to console). This signal indicates to the console that the telephone has been answered. (The telephone is "off hook".)
DTR	CD	1Ø8/2	2Ø	Data terminal ready (console to modem). This signal is asserted by the console, enabling telephone answering when a ring occurs.
RI	CE	125	22	Ring indicator (modem to console). This signal is monitored by the console to determine when a ring occurs (when a DDC host is attempting to call the system).
CD	CF	1Ø9	8	Carrier detect (modem to console). This signal is asserted by the modem to indicate that the calling station's carrier signal has been detected.

E.3 CALL REQUIREMENTS

The following paragraphs detail the procedure for successful completion of call answering and termination.

E.3.1 Automatic Call Answering

Use the following procedure to enable automatic call answering (Figures E-1 and E-2).

Set the front panel keyswitch to either REMOTE/SECURE or REMOTE. This clears all console device signals (DTR, RTS) and inhibits data line TxD (held at binary 1). It also clears all modem signals (RI, DSR, CTS, CD) and inhibits line RxD (held at binary 1). The console asserts the data terminal ready (DTR) signal and enables call answering.

When the console asserts data terminal ready, the modem attempts to answer the phone when it detects at least one ring indicator (RI) signal. When it answers the phone, the modem asserts data set ready (DSR). Once the console detects DSR, DTR may not deassert for five seconds, unless the console has also detected CD. The console must detect DSR assertion within 20 seconds, or the call terminates. Once it detects DSR, the console asserts the request to send (RTS) signal.

The modem responds to RTS assertion by placing its carrier signal or answer tone on the line. When the modem receives a carrier signal from the calling station, the modem responds by asserting the carrier detect (CD) signal and then asserting the clear to send (CTS) signal (700 ±300 ms later). Note, however, that whenever CD is off, RxD is held at binary 1. The console must detect CD and CTS within 20 seconds of RTS assertion, or the call terminates.

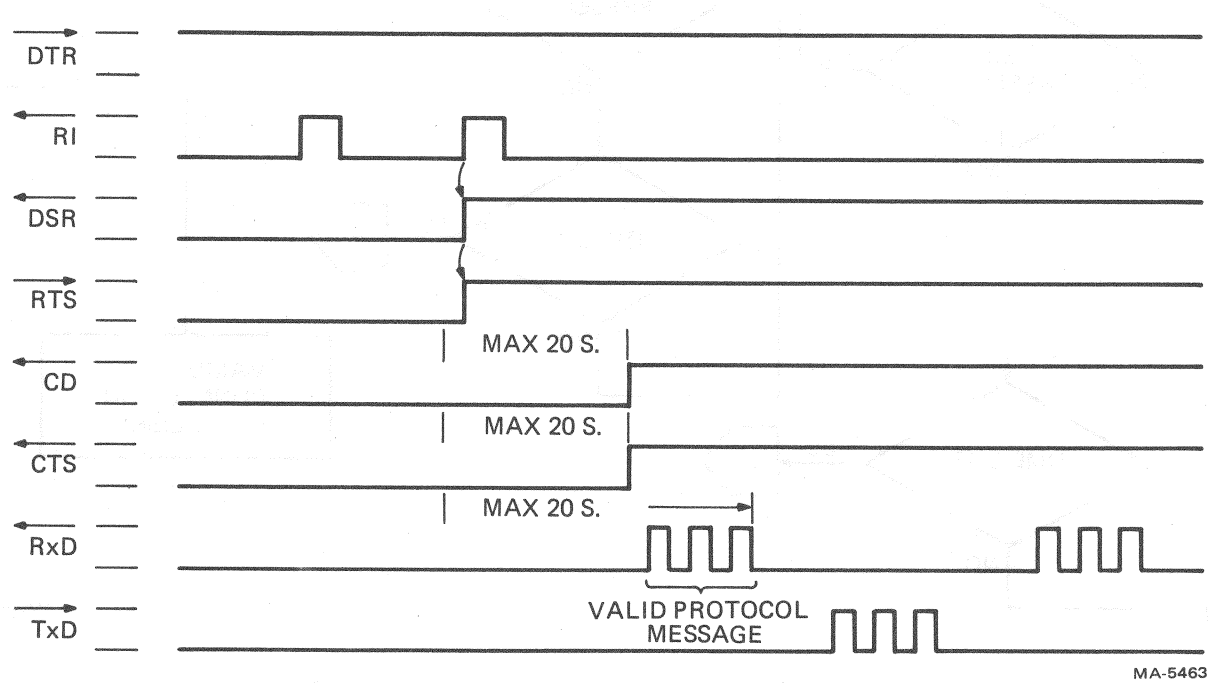


Figure E-1 Automatic Call Answering Sequence

Signal direction assumes console on left, modem on right. It is assumed throughout that the front panel keyswitch is in a REMOTE position.

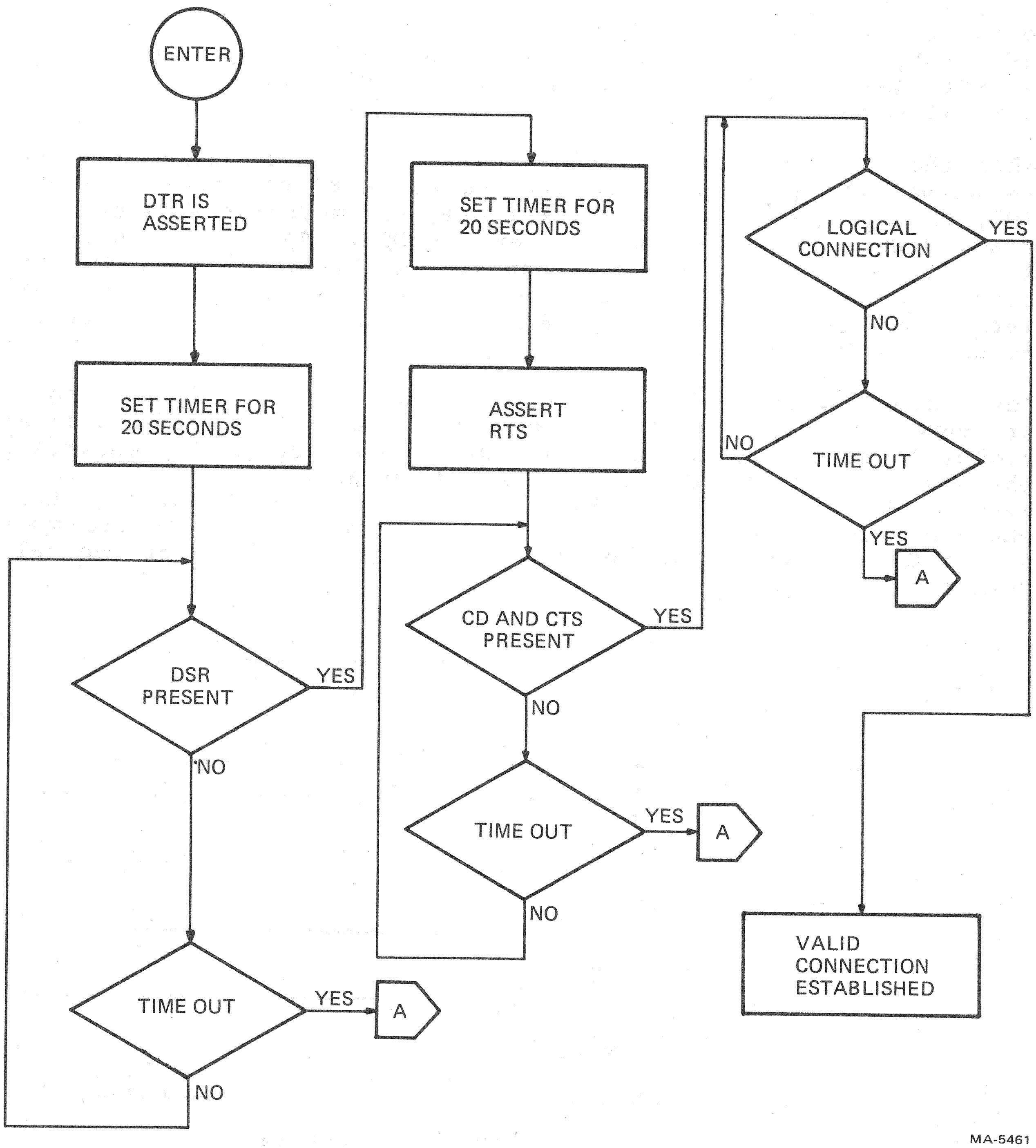


Figure E-2 Automatic Call Answering Flowchart

It is assumed, on entry, that the front panel keyswitch is in a REMOTE position, and that RI, DSR, CD, CTS, and RxD are clear.

Once the console receives CD and CTS, a dialogue initiates between the console and calling station. This dialogue, or logical connection (described in Paragraph E.5), consists of a series of data transfers via RxD and TxD; it must successfully complete before diagnostic operations can begin. TxD and RxD are subject to the following restrictions and conditions.

With signals RTS, CTS, DSR, and DTR on, data may transmit on TxD until any of these four signals goes off. While these signals are on, TxD is held in a binary 1 condition if data is not transmitting. If any one of the four signals RTS, CTS, DSR, or DTR go off, data flow is inhibited and TxD returned to a binary 1 state.

A logical connection must complete within 20 seconds of the console's detection of DSR or the call terminates. The call will also terminate during a logical connection if any one of signals DSR, CTS or CD is lost, or a SPACE condition (binary 0) on RxD is received for more than one second.

Once a logical connection completes, call answering is considered complete and a valid connection in effect. At this point, the diagnostic process may proceed.

Note that a call will terminate during call answering, but before logical connection, if any one of signals DSR, CD, or CTS is lost for more than 500 ms. Signal glitches less than 500 ms are ignored until a logical connection is established.

E.3.2 Establishing a Logical Connection

Once the modem asserts CD and CTS, the console recognizes data received on RxD only if the data is in the form of a protocol message. The protocol is a modified subset of the DIGITAL Data Communication Protocol (DDCMP) for ASCII serial communications. A logical connection is established if a valid protocol message packet is received from the calling station.

Successful completion of this transaction within 20 seconds after the modem asserts DSR establishes a logical connection.

E.3.3 Call Termination (Abort Sequence)

A call terminates during a call answering routine, if key signals are not detected within the established time frame, or are lost. The DRDC host can also terminate a call by removing its carrier signal, which results in the loss of CD. A local operator can terminate a call via the console by setting the keyswitch to LOCAL (Figures E-3 and E-4).

Once termination initiates, the console inhibits the data flow on TxD and deasserts DTR and RTS. A "Connection Lost" message prints on the system console terminal. The modem then monitors the states of signals RI, DSR, CTS, CD, and RxD. These must be detected as cleared to successfully terminate the call.

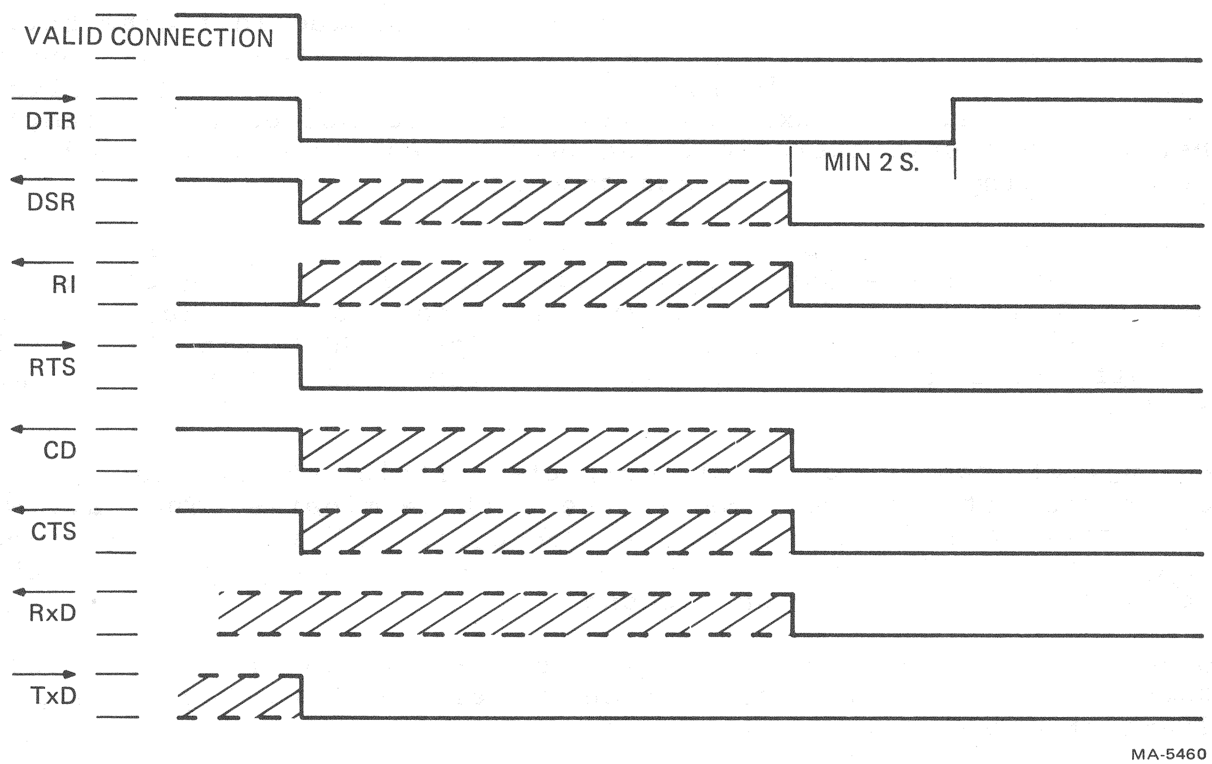


Figure E-3 Call Abort or Termination Sequence

It is assumed that the front panel keyswitch is in a REMOTE position.

DTR remains deasserted for two seconds following termination, before being reasserted to enable call answering. During this two-second delay period RTS must remain cleared, and data cannot be placed on TxD. Note that DSR and CTS must be cleared by the modem before DTR and RTS can be reasserted.

If you initiated the call termination by turning the front panel keyswitch to LOCAL, DTR and RTS are automatically cleared (as described above) but remain deasserted. Call answering is not enabled until you return the switch to REMOTE/SECURE or REMOTE, where DTR can be asserted.

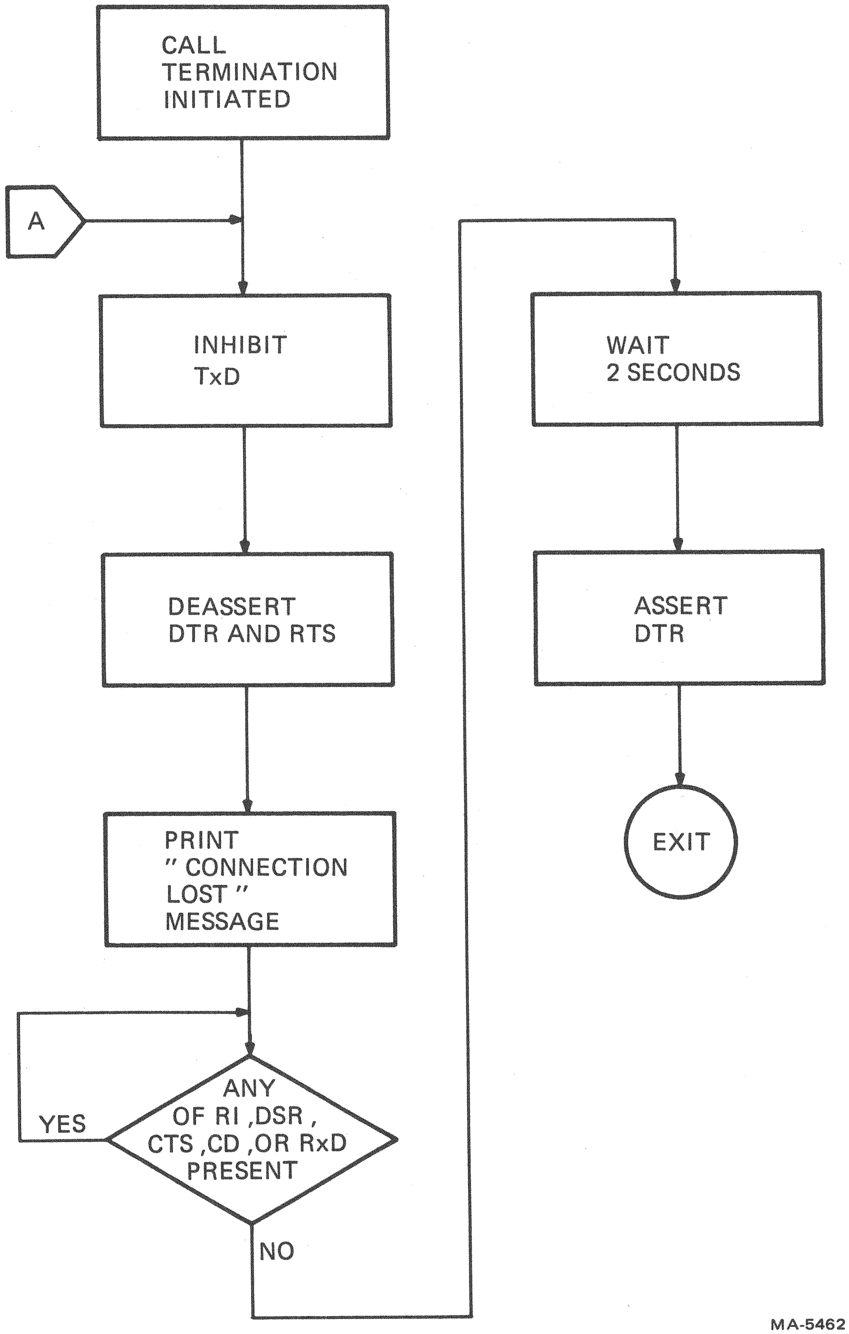


Figure E-4 Call Abort or Termination Flowchart

It is assumed that the front panel keyswitch is in a REMOTE position.

KC750 MICRODIAGNOSTICS
AND TECHNICAL MANUAL
EK-KC750-TM-001

Reader's Comments

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our publications.

What is your general reaction to this manual? In your judgement is it complete, accurate, well organized, well written, etc? Is it easy to use? ____________________

What features are most useful? ____________________

What faults or errors have you found in the manual? ____________________

Does this manual satisfy the need you think it was intended to satisfy? ____________________

Does it satisfy *your* needs? ____________ Why? ____________

☐ Please send me the current copy of the *Technical Documentation Catalog*, which contains information on the remainder of DIGITAL's technical documentation.

Name ____________ Street ____________
Title ____________ City ____________
Company ____________ State/Country ____________
Department ____________ Zip ____________

Additional copies of this document are available from:

Digital Equipment Corporation
Accessories and Supplies Group
Cotton Road
Nashua, NH 03060

Attention *Documentation Products*
Telephone 1-800-258-1710

Order No. EK-KC750-TM-001

Fold Here

Do Not Tear – Fold Here and Staple

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD, MA.

POSTAGE WILL BE PAID BY ADDRESSEE

Digital Equipment Corporation
Educational Services Development and Publishing
129 Parker Street, PK3-1/T12
Maynard, MA 01754

digital