# PDP-11

DIAGNOSTIC USER GUIDE
CZUGAA0

AH-E122A-MC

FICHE 1 OF 1

JUN 1978

digital

MADE IN USA

## IDENTIFICATION

|  |  |
|---|---|
| PRODUCT CODE: | AC-E121A-MC |
| PRODUCT NAME: | CZUGAAO PDP-11 DIAG USR GDE |
| PRODUCT DATE: | FEBRUARY 1, 1978 |
| MAINTAINER: | PDP-11 PRODUCT ENHANCEMENT GROUP |
| DIAGNOSTIC ENGINEER: | BARRY SUSSMAN |
| TECH WRITER: | MICHAEL A. MESROBIAN |
| PREPARED BY: | JANET WASIUK |

|  |  |  |  |
|---|---|---|---|
| DIGITAL | PDP | UNIBUS | MASSBUS |
| DEC | DECUS | DECTAPE | DECX/11 |

## PREFACE

The *Users' Guide For PDP-11 Diagnostic Software* is the first part of the two-part *MAINDEC Users' Manual For PDP-11 Diagnostic Software*.

Part I, this guide, presents the general operating procedures required to run diagnostic programs on PDP-11 computers.

Part II consists of a set of appendices pertaining to diagnostic options. Each appendix contains a step-by-step procedure for preparing, loading, and running the diagnostic for a specific option. The appendices are used to identify problem areas. If problems are encountered and more detailed diagnosis is required, the individual diagnostic listing and document must be consulted.

Used together, the guide and the appendices contain the information and procedures needed to run vitually any PDP-11 diagnostic. The procedures assume that the diagnostician is familiar with the hardware to be tested.

In those cases where the most detailed information is required, a diagnostician must refer to the individual diagnostic program documentation.

NOTE: PART II, THE SET OF APPENDICES DESCRIBING THE DIAGNOSTIC OPTIONS, IS NOT INCLUDED IN THIS DOCUMENT. THE APPENDICES ARE CURRENTLY IN PRODUCTION AND WILL BE RELEASED TO THE FIELD AS THEY ARE COMPLETED.

CONTENTS

## TABLES

## FIGURES

# CHAPTER 1

# INTRODUCTION & REFERENCES

This users' guide is intended for the person new to DEC, and especially for the person new to the diagnostic aspect of DEC's operations:

. the field service engineer.

. the manufacturing test person.

. the diagnostic programmer.

It should help in the training of new personnel and serve as a concise reference for more experienced people. Essentially, this guide presents to anyone an inclusive survey of the diagnostic software tools available for DEC PDP-11 computers.

The guide follows a design that builds on information presented in previous chapters. Chapter 2 presents two general approaches to running diagnostics. Chapter 3 gives directions for using a PDP-11 computer operating console. Chapter 4 tells you how to load a monitor into the machine; it assumes that you know how to use the operator's console (explained in Chapter 3), and that you have decided on which approach you want to use for running the diagnostics (explained in Chapter 2). The ensuing chapters describe the operations of the various monitors under which you can execute diagnostics.

The guide draws from and refers to many existing DEC documents, listed below in paragraph 1.1. There is also a short list of abbreviations in paragraph 1.2.

Used along with other diagnostic tools, this guide can help both the novice and the experienced diagnostician successfully accomplish his or her work at the machine being tested.

## 1.1 REFERENCES

ACT System Manager's Guide
ACT Users' Guide
APT System Manager's Guide
DECX/11 Users' Documentation and Reference Guide
Diagnostic Engineering Standards and Conventions
Microcomputer Handbook
PDP-11 Computer Programming Card
PDP-11 Peripherals Handbook
PDP-11/04/34/45/55 Handbook
XXDP/DECX/11 Programming Card
XXDP Users' Manual

## 1.2 ABBREVIATIONS AND MNEMONICS USED IN THIS GUIDE

| | |
|---|---|
| ACT | -- automated computer test |
| AINT | -- APT initialization utility |
| APT | -- automated product test |
| BR | -- bus request |
| CPU | -- central processing unit (same as processor) |
| DECX/11 | -- Digital Equipment Corporation UNIBUS exer- ciser for PDP-11 computers |
| DMA | -- direct memory access |
| ECO | -- engineering change order |
| EIS | -- extended instruction set |
| GPR | -- general purpose register |
| HW | -- hardware |
| I/O | -- input/output |
| LRT | -- long run-time |
| LSI | -- large-scale integration |
| MCO | -- manufacturing change order |
| MLE | -- multiple loading with error control (APT) |
| MOS | -- metal-oxide semiconductor |

| | |
|---|---|
| MTTD | -- mean-time-to-detect |
| MTTR | -- mean-time-to-repair |
| NPR | -- non-processor-request |
| ODT-11 | -- on-line debugging technique for PDP-11 computers |
| PC | -- program counter (register 7) |
| PSW | -- processor status word |
| Q/V | -- quick verify |
| RCSR | -- read control/status register |
| REV11 | -- the DMA-refresh, bootstrap, and terminator module for the LSI-11 CPU. |
| ROM | -- read-only-memory |
| SP | -- stack pointer (register 6) |
| SS | -- single step |
| SW | -- software |
| SWR | -- switch register |
| TSP | -- test-software-package utility (APT) |
| TST | -- time-sharing terminal (APT) |
| UUT | -- unit under test |
| XXDP | -- xx diagnostic package; "xx" replaced by device mnemonic |

# CHAPTER 2

# UNIT TESTING VS. THE SYSTEM APPROACH TO USING DIAGNOSTICS

A diagnostic is a program that comprises numerous individual tests arranged in a sequence. The tests detect malfunctions and help to locate their source. The results of any test determine which other tests in the sequence will be executed. In addition, a diagnostic program may *analyze* the test results and *identify* the failing component.

There are two testing approaches: *unit testing*, which tests single units; and *UNIBUS exercising*, which test several of the same or combinations of different units simultaneously.

You run *unit tests* when you decide that a malfunction comes from a particular unit: for example, a peripheral device, or the CPU, or memory. Unit tests usually operate under *controlled conditions* in order to *isolate* the malfunction. The testing sequence follows a logical, "building block" structure that tests the smallest part of the unit and proceeds upward to include the entire unit. Unit tests are not generally efficient at identifying problem areas in large systems, but are the best method for correctly identifying malfunctions in a given device.

You run a *UNIBUS exerciser* when you cannot determine the origin of a malfunction, or if unit testing does not substantiate your previous decision. A UNIBUS exerciser does not run under controlled conditions as compared to unit tests. Rather, it *creates* an operating environment that pushes the system to the limits of its specifications. The UNIBUS exerciser is DECX/11 (see Chapter 12 and the *DECX/11 Reference Manual*). A UNIBUS exerciser is designed to detect and report general malfunctions. Once the UNIBUS exerciser identifies the bad subsystem or unit, you can then run unit tests on that unit to determine the failing component. (Often, intermittent failures that are untraceable with unit tests appear when running a UNIBUS exerciser). UNIBUS exercising emphasizes *detection* over isolation. It uses testing time efficiently, but it may incorrectly identify the source of a malfunction in a multi-problem situation.

Note, too, that the multi-problem situation may occur where neither unit testing nor a UNIBUS exerciser can accurately identify a failing component. In this case, a field service engineer must rely on basic tools (oscilloscope, test meters, etc.) coupled with a thorough knowledge of the hardware and sound experience.

PDP-11 diagnostic software, therefore, can be categorized as two main types: unit diagnostics and UNIBUS exercisers.

To completely test a given unit, all unit diagnostics and the DECX/11 UNIBUS exerciser must be run.

## 2.1 UNIT DIAGNOSTICS

Unit diagnostics include tests for the units listed below:

modules
subassemblies
processors
peripherals
controllers
memories

Also, there may be several diagnostics for one unit, each diagnostic checking a particular function of the unit.

Diagnostics verify all logic that may be tested by program instructions with a single unit under processor control. There are no other units of similar or dissimilar design under test at the same time. The goal of the diagnostic is 100% coverage of the logic. In reality, the coverage is 80% to 95% of the logic. The remaining logic is tested by the UNIBUS exerciser.

Diagnostics are usually stand-alone: they require the processor to be dedicated to the unit under test (UUT).

The purposes for using diagnostics can be categorized as follows:

1. Verify that the H/W functions correctly.

2. Detect a H/W fault.

3. Isolate a H/W fault.

4. Verify that a repair has corrected a fault.

The proper use and execution sequence of diagnostics will reduce the mean-time-to-detect (MTTD) and the mean-time-to-repair (MTTR). The MTTD is the time necessary to detect that an error condition or fault exists. The MTTR is the time it takes to detect and isolate the fault, repair or replace the failing component, and verify that the repair has corrected the fault.

### 2.1.1 Error Detection Using Unit Diagnostics

Diagnostics are designed to detect the following classes of errors:

| ERROR CLASS | EXAMPLE OF CAUSE |
|---|---|
| 1. logical and functional errors | pin shorts; pins stuck high and low; media failures; mechanical failures. |
| 2. timing problems | capacitance; resistance; race conditions. |
| 3. intermittent failures | vibration; loose connections; dirty connections; power-up/power-down failures. |
| 4. heat problems | insufficient module cooling; dirty filters; defective fans; improper installation. |
| 5. noise problems | shielding deficiencies; UNIBUS termination; UNIBUS drivers and receivers; cable problems. |
| 6. marginal components | voltage variations. |

Once you determine the error class, you can decide which diagnostic or series of diagnostics to run. If you do not known the source of a problem, you can use a UNIBUS exerciser to locate the fault. (See paragraph 2.2).

There are two general ways to run unit diagnostics: the "quick-verify" mode (Q/V mode) and the "long-run-time" mode (LRT mode).

NOTE

For the most inclusive coverage, you should run each diagnostic in all modes of operation and with all applicable switch settings.

## 2.1.2 Running Diagnostics In Quick-verify Mode (Q/V Mode)

The "quick-verify" or Q/V mode of running a diagnostic performs the following services:

1. Verify that all the major components are present and functioning.

2. Test all the logic at least once and indicate faults.

3. Generally indicate that no "hard" errors exist, or that they no longer exist after a repair.

4. When there are many diagnostics for a single unit, provide a system that will isolate the failing component of the unit in the shortest possible time.

The Q/V mode may, but generally does not isolate intermittent failures, marginal components, heat problems, or noise problems. The first pass of a diagnostic is a Q/V pass. Unless iterations are surpressed, after the Q/V pass, all diagnostics enter LRT mode (see paragraph 2.1.3). All diagnostics should be run at least two passes -- a Q/V pass and one LRT pass.

Once the Q/V mode detects an error condition, you can use the LRT mode (below) to isolate the fault. (The diagnostics provide "scope loops".)

## 2.1.3 Running Diagnostics In Long-run-time Mode (LRT Mode)

The "long-run-time" or LRT mode of running a diagnostic detects the following:

1. noise problems.

2. heat problems.

3. marginal components.

4. timing problems.

5. intermittent failures.

6. vibration problems and bad connections.

The LRT mode also tests that a H/W repair holds up, and that the H/W that was repaired was the H/W at fault. If the UUT successfully completes at least two passes in LRT mode, then you can assume that you have verified its operation as much as possible in a stand-alone environment. However, additional successful passes will increase the confidence in the reliability of the hardware.

## 2.2 UNIBUS EXERCISERS

UNIBUS exercisers are designed to test all logic that is not tested or cannot be tested with unit diagnostics. UNIBUS exercisers detect faults that result when multiple units are in the system, and also when the given unit is in a system with other dissimilar units. UNIBUS exercisers test all the devices in the system simultaneously; the processor is not dedicated to only one unit.

UNIBUS exercisers are effective at isolating problems that arise when units interact with other units. The units may be the same type, or completely different in form and function.

The problems that arise from this interaction are noise, priority arbitration, timing, marginal components, and elusive and intermittent problems.

Typically the UNIBUS exerciser tests the units in a "worst-case" manner. This, in combination with the multiple units both similar and dissimilar on the system, creates an environment that cannot be duplicated by a stand-alone diagnostic. This is why UNIBUS exercisers can isolate interaction problems so well.

The design of a UNIBUS exerciser emphasizes the isolation of interaction problems. Therefore, they verify a large portion of the units' logic. However, this percentage is less than what can be verified by a stand-alone diagnostic. Additionally, the general constraints of the system environment limit which logic may be tested and to what degree. Careful design of the UNIBUS exerciser maximizes the coverage.

Fault isolation using a UNIBUS exerciser includes the following procedures:

1. Making a system inventory.

2. Detecting a fault.

3. Identifying the fault.

4. Isolating the fault.

5. Making and verifying the repair.

### 2.2.1 System Inventory

You make a system inventory to define the H/W configuration. Use the following recommended procedure:

1. Describe the CPU.

   - model (11/03, 11/34, 11/40, 11/45, 11/60, 11/70)

   - processor type (KD11-A, KD11-D)

   - ECO tally

   - options (floating point processor, memory management, cache)

2. Define memory.

   - size (4K, 8K, 16K, 32K, etc.)
   - type (core, MOS, bipolar)
   - parity controller
   - error correcting

3. Determine small options.

   - serial line units (DL11)
   - interprocessor links
   - DMA drivers (DR11B)
   - line clocks (KW11-P, KW11-L)
   - communications modules (DUP11)
   - DECwriters (LA30, LA36)
   - DECscopes (VT50, VT52)

   In all cases determine the applicable baud rates, line frequency, device addresses, vector addresses, bus request interrupt level (BR), relative position on the UNIBUS.

4. Determine large options.

   - mass storage devices (magnetic tape, DECtape, disk, floppy disk)

   Determine for all devices the unit numbers, device addresses, vector addresses, BR level, relative positon on the UNIBUS, and whether the device is DMA or proccessor-controlled.

5. Ensure that any equipment that is known to be down is still on the UNIBUS and is receiving power. If the equipment is not receiving power, you should remove it from the UNIBUS.

6. Ensure that the bus-grant-continuity cards are in place.

7. Ensure that the UNIBUS is properly terminated.

8. Determine whether any equipment is missing; for example, is a controller present with no drive? Or, is a BR plug missing?

9. Visually inspect the system: fuses, boards, cables.

## 2.2.2 Fault Detection

You start fault detection only after taking a system inventory. In other words, you must confirm that a reported problem on the system does, in fact, exist. If you observe the source of the failure without a doubt, then you can run the unit diagnostics directly, without a UNIBUS exerciser. However, when you cannot locate the source of failure, you must configure and run a UNIBUS exerciser. When the UNIBUS exerciser has detected the source of failure, it will point to the unit. This can be any of the following: a peripheral controller or drive, the CPU, memory, an option module such as floating-point or memory management. Now you have to isolate the failing component within the unit.

## 2.2.3 Fault Isolation

Use the following procedure to isolate faults.

1. See what diagnostics are available for the failing unit.
2. Determine which of these diagnostics to run, and in what order.
3. Run all the diagnostics for one Q/V pass. (This may be a chain file under the XXDP monitor. See paragraph 7.7.)
4. From the results of step 3, determine which diagnostic has detected any errors, provided the most information, tests the smallest logical block, and provides features such as "loop-on-test" or "loop-on-error".
5. Run the specific diagnostic resulting from step 4 in order to isolate the failing logic. You may have to use the diagnostic program listing, the device logic prints, and an oscilloscope.
6. For total coverage, run all diagnostics in all modes of operation, with all applicable switch settings.

## 2.2.4 Repair And Verify

After making the necessary repair or replacement, follow this procedure for verifying it.

1. Run all the unit diagnostics in the Q/V mode. A successful Q/V pass indicates that the diagnosis and repair were correct for the problem. Generally it also indicates that the H/W is complete and has no "hard" errors.

2. Run all unit diagnostics in LRT mode for several passes. This is "worst-case" testing of the repaired unit. Run the diagnostics in all modes of operation and with all combinations of applicable switch settings. Successful LRT passes indicate that the unit functions correctly in a stand-alone environment.

3. Run the UNIBUS exerciser to verify that the unit functions correctly in a "system" environment. Successful passes of the UNIBUS exerciser indicate that the repaired unit causes no system interference, or interaction and noise problems.

When you perform these three steps after making a repair, and receive successful end-of-pass messages, you can consider the unit correctly repaired and in good working order.

## 2.3 SPECIAL CONSIDERATIONS

1. In some rare instances, a UNIBUS exerciser may not detect a unit failure. Should this occur, the best method to detect the failure is to run all the standalone diagnostics for all the units in a system, using Q/V mode, followed by LRT mode. Start the testing with the diagnostics for the most suspect unit. After making the repair, verify it as described in paragraph 2.2.4.

2. If you replace a unit or module rather than repair it, you should run all the diagnostics for that unit. This will verify that you have replaced the correct failing unit, and that no other, different problems exist in the new unit. The verification process is the same one described in paragraph 2.2.4.

3. Here is the recommended order for running unit diagnostics:

    1. CPU diagnostics (basic instruction tests).

    2. memory diagnostics.

    3. CPU options (floating-point processor, memory management, cache, EIS).

    4. small options.

    5. large options.

    6. data reliability tests.

# CHAPTER 3

# OPERATOR'S CONSOLE

This chapter explains how to use the CPU operator's console. PDP-11 processors have three types of consoles.

1. those with a switch panel, discussed in paragraph 3.1.

2. those with a 20-button keypad, discussed in paragraph 3.2.

3. those with a separate terminal device, discussed in paragraph 3.3.

Identify the type of console on your processor and proceed to the appropriate paragraph. Also, consult the appropriate Processor Handbook.

## 3.1 SWITCH-PANEL CONSOLES

Switch-panel consoles have a switch panel, control switches, and indicators.

### 3.1.1 Switch Panel

The switch panel is a set of physical switches that correspond bit-by-bit to the hardware switch register, or HW/SWR (refer to chapter 6). When the CPU is running, the switch panel setting is the current value of the HW/SWR. When the CPU is not running, you can use the switch panel to define either the address number or the contents of any PDP-11 location, using the console control switches below. A PDP-11 location can be a memory word, a processor register, or a peripheral device register.

## 3.1.2 Control Switches

The control switches provide specific functions as described in Table 3-1. You must HALT the CPU before using them. Also, if your console has control knobs for address and data display, then turn these knobs to select the CONSOLE PHYSICAL and DATA PATHS positions.

Table 3-1. Console Control Switches

| SWITCH | FUNCTION | EXPLANATION |
|---|---|---|
| LOAD ADRS (depress) | load an address value | Transfers the value on the switch panel to the CPU and displays the address number. |
| EXAM (depress) | examine | Displays the contents of the location defined by a LOAD ADRS operation. Successive EXAM operations display the contents of sequential locations.<br><br>If you try to EXAM a nonexistent address, the operation fails; you must repeat LOAD ADRS with a legitimate address. |
| ENABLE/HALT (select) | enable/halt | ENABLE allows the CPU to execute instructions; HALT stops it after the current instruction (and any interrupts and traps). See also CONT. |
| CONT (depress & release) | continue | If the CPU is in ENABLE, then CONT causes it to continue operating from the point at which it stopped, without a system reset. IF the CPU is in HALT, repeated CONT operations execute the program instruction by instruction<br>CONT does not work in the RUN state. |

Table 3-1. (Continued)

| SWITCH | FUNCTION | EXPLANATION |
|---|---|---|
| DEP (raise & release) | deposit | Loads the value on the switch panel into the location defined by a LOAD ADRS operation. Successive DEP operations load sequential locations. If you try to DEP a non-existent location, the operation fails; you must repeat LOAD ADRS with a legitimate location. |
| START (depress & release) | start | CPU starts executing instructions at the location defined by a LOAD ADRS operation.<br><br>NOTE: START issues a system reset. Do not press START when the CPU is already in the RUN state. You may have to reload the program. The usual sequence for starting a program is: HALT<br>LOAD ADRS (starting address)<br>ENABLE<br>START |
| Some consoles also have the following control switches: | | |
| REG EXAM (depress) | examine GPR | Displays the contents of a GPR defined by a LOAD ADRS operation. |
| REG DEP (raise & release) | load GPR | Loads the value on the switch panel into the GPR defined by a LOAD ADRS op. |
| S/INST-S/BUS (select) | single-step | Modifies the operation of CONT (above) as follows: S/INST causes CONT to execute the next single instruction and stop; S/BUS causes CONT to complete the next bus operation and stop. |

### 3.1.3 Console Indicators

The console indicators vary among consoles, but here is a general list. For more information, refer to the PDP-11 Processor Handbook.

RUN -- glows when the CPU is executing instructions.

ADDRESS/DATA -- displays an address or the contents of an address, depending on the operation. Some consoles have separate indicators for address and data display.

PROC -- glows when the CPU has control of the bus.

CONSOLE -- glows when the CPU is not executing the stored program; i.e., the CPU is under manual control.

## 3.2 KEYPAD CONSOLES

Keypad consoles have a 20-button keypad, a LED digital display, and indicators.

### 3.2.1 Keypad

The keypad contains number buttons and control buttons. You use the number buttons to load numerical information into the CPU. The control buttons provide functions as described in Table 3-2. The keypad has a safety feature that prevents certain unintended control operations. This is the CNTRL button. You must depress CNTRL simultaneously with the following control buttons to make them work: INIT, BOOT START, CONT, HALT/SS (first hit only; you can single step without CNTRL).

Table 3-2. Console Control Buttons

| BUTTON | FUNCTION | EXPLANATION |
|---|---|---|
| CLR | clear | Clears the current keypad entry. There is no operation. |
| LAD | load address | Loads an address number (the address of a memory word, a processor register, or a peripheral device register) from the keypad to the CPU. |
| DIS-AD | display address | Displays the address number of the location the CPU is currently manipulating. |
| EXAM | examine | Displays the contents of the location the CPU is currently manipulating. |
| DEP | deposit | Loads a value from the keypad into the location the CPU is currently manipulating. |
| LSR | load switch register | Loads a value from the keypad to the hardware switch register (see paragraph 3.2.4 and Chapter 6). |
| The following control buttons require simultaneous pressing of CNTRL. | | |
| HLT/SS (CNTRL) | halt/single-step | Puts the CPU in HALT state. Also, if you continue to press this button without CNTRL, it provides single-step execution of a program. |
| CONT (CNTRL) | continue | Allows the CPU to continue program execution from the point at which it stopped. CONT does not issue a system reset. |

Table 3-2. (Continued)

| BUTTON | FUNCTION | EXPLANATION |
|---|---|---|
| START (CNTRL) | start | Issues a system reset and starts executing instructions at the address specified by a previous LAD operation. |
| BOOT (CNTRL) | bootstrap | Starts a bootstrap operation. |
| INIT (CNTRL) | initialize | Causes a system reset. |

## 3.2.2 LED Digital Display

The LED digital display that shows address/data values has a shift feature. The lights show the value you enter on the keypad up to the maximum digits on the display. If you enter more than the display can show, you lose the high-order digits; they shift off the display.

## 3.2.3 Indicators

The only indicators you need to know at this time are the RUN indicator, which glows when the CPU is executing macro instructions, and the SR DISP indicator, which tells you that the LED display is showing the contents of the hardware switch register. (The 11/40 has no SR DISP indicator and its RUN light is always on.)

## 3.2.4 Loading The Switch Register

Unlike CPU's with a switch panel, keypad consoles have no direct physical connection with the HW/SWR, at location 177570, or 777570 with memory management (see 3.1.1 and Chapter 6). On a keypad console you can load the HW/SWR at any time, but only by using the LSR button (load-switch-register). A LAD+DEP operation on the HW/SWR address will not load the HW/SWR. Similarly, a LAD+EXAM operation will not display the contents of the HW/SWR. The CPU displays the HW/SWR contents only at CNTL/START or CNTL/CONT. If you have cleared the display and need to know the HW/SWR value, depress CNTL/CONT.

## 3.3 TERMINAL DEVICE CONSOLES

Some PDP-11 processors do not have an operator's console for controlling the CPU. These CPU's do not have a HW/SWR either (See Chapter 6). You operate these processors with a separate _terminal device_ and a console emulation routine that is built into the processor. In addition, some PDP-11 processors with an operator's console may have a terminal device as well. You can operate these processors from either the operator's console or the terminal.

Several console emulators are available. However, this manual discusses only the REV11 and ODT-11 programs (for LSI-11 and 11/03) and the M9301 program (for other processors).

NOTE

Although the _run light_ is extinguished during any ODT-11 console emulation operations, the CPU is still executing the ODT console emulation routine. The processor, technically, _is running_.

### 3.3.1 Using ODT-11

ODT-11 is part of the LSI-11 hardware and requires no additional software to operate. To use ODT-11, prepare the CPU as follows:

1. Before power-up, put the HALT/ENABLE switch to HALT position.

2. Power-up the CPU.

3. ODT-11 will issue the prompt character, @, and accept any of the commands in Table 3-3. ODT accepts _only_ these command characters and the numbers 0-7 on a command line. The underlined characters in the examples are the ones ODT prints.

NOTE

For console ODT communication, the DLV11 must be configured for console bus addresses 177560 through 177566. These addresses are included in the LSI-11 processor's microcode and cannot be changed. If no device responds to the above addresses, the processor will go into loop. You can get out of this loop by cycling the power off and then on.

Table 3-3. ODT-11 Commands

| KEYBOARD CHARACTER | COMMAND | EXPLANATION & EXAMPLE |
|---|---|---|
| / | examine | Displays the contents of the specified location; if you do not specify a location, ODT operates on the last location used.<br><br>Example: examine location 100.<br><br>@100/025200 |
| CR (CARRIAGE RETURN) | deposit/ return to ODT command level | Deposits a value into the specified location and returns you to ODT command level. If you do not enter a value, you will not alter the location.<br><br>Example: deposit 7317270 in location 100, and verify the operation.<br><br>@100/025200 7317270CR<br>@/117270 (note high-order truncation) |

Table 3-3. (Continued)

| KEYBOARD CHARACTER | COMMAND | EXPLANATION & EXAMPLE |
|---|---|---|
| LF (line feed) | deposit/ examine next | Deposits a value into the specified location and examines the next higher location.<br><br>Example: put the numbers 1-5 into the five consecutive locations starting at location 100; then return to command level.<br><br>`@100/112220 1LF`<br><br>`000102∠000123 2LF`<br><br>`000104∠055200 3LF`<br><br>`000106∠132104 4LF`<br><br>`000110∠000000 5CR`<br><br>`@` |
| G | start (go) | Starts executing instructions at the specified location.<br><br>Example: start the program at address 200.<br><br>`@200G` (also, `@200;G`) |
| P | proceed | Continues program execution from the location in the PC. (See also: CONT, Table 3-1).<br><br>Example: resume execution.<br><br>`@P` (also, `@;P`) |

Table 3-3. (Continued)

| KEYBOARD CHARACTER | COMMAND | EXPLANATION & EXAMPLE |
|---|---|---|
| L | load bootstrap loader | Loads a program, in bootstrap loader format, from the device whose read control status register address is given. If RCSR is invalid, the CPU will hang; you must power off/on again.<br><br>Example: load a program from the console device.<br><br>@177560L |
| RO(rubout)<br>DEL(delete) | correct entry | Corrects one or more erroneous numerical entries. ODT prints a backslash, ( ), for RO or DEL. You cannot delete a command with this character. Also, it will not correct erroneous GPR numbers.<br><br>Example: deposit 167106 in location 100; then verify the operation.<br><br>@100/000120 5632  7106CR<br><br>@/162106 (note truncation) |
| R<br>$ | register designator | Used with an ODT command to operate on a GPR (specify 0-7) or the PSW (specify S).<br><br>Example: examine GPR 5, deposit zeroes; then examine PSW.<br><br>@R5/012345 0CR or, @$5/012345 0CR<br><br>@RS/000200 or, @$S/000200 |

## 3.3.2 REV11-A And REV11-C Options

REV11 is the DMA-refresh, bootstrap, and terminator module for the LSI-11 CPU To use REV11-A or REV11-C, prepare the CPU as follows:

1. Before power-up, put the HALT/ENABLE switch to ENABLE.

2. Power-up the CPU.

3. REV11 normally displays the prompt character, $, and accepts any of the commands in Table 3-4 below. The underlined characters are those printed by REV11.

NOTE

Certain processor/terminal jumper configurations may put the CPU in ODT-11 at power-up (see 3.3.1). You can manually start REV11 under ODT with the ODT go command, G. Specify address 173000 (REV11's starting address) as follows:

```
@173000G
$
```

The go command does not require a carriage return.

Refer to the <u>Microcomputer Handbook</u> for more information.

Table 3-4. REV11 Commands

| Command | Function | Explanation |
|---|---|---|
| OD | ODT-11 mode | Processor will respond to ODT commands discussed in 3.3.1. This command does not requiire a carriage return. To return to REV11, issue the ODT-11 command, P, if you have not changed the PC. If the PC has changed, issue the G command, and specify address 165006.<br><br>Example: under REV11, transfer control to ODT-11.<br><br>`$OD`<br>` pppppp      (the PC value)`<br>`@` |
| AL<cr> or<br>ALdddddd<cr> | absolute loader | Load the absolute loader program using the console paper tape read device (default) or the device whose RCSR address you specify, dddddd. The loaded program self-starts, or else a halt occurs at 165626. |
| AR<cr>, or<br>ARdddddd<cr> | absolute/relocated loader | Execute absolute loader program for relocated loading operation using console evice (default) or device RCSR = dddddd. Before using the AR command, you must prepare the CPU as follows: put the CPU in ODT mode, deposit the relocation address bias (nnnnnn) into R4 (the SW/SWR), and return control to REV11. Then you can issue AR.<br><br>`$OD       (transfer to ODT)`<br>` pppppp  (PC value)`<br>`@R4/xxxxxx nnnnnnCR`<br>`(deposit relocation`<br>`@P  (return to REV11) bias`<br>`$AR<cr>`<br><br>Successful load results in automatic program start. Otherwise, the program halts with 165412 display. |

Table 3-4. (Continued).

| Command | Function | Explanation |
|---|---|---|
| DX<cr>, or DXn<cr> | floppy disc bootstrap | Execute RXV11 floppy disk system bootstrap for disk 0 (default) or disk n (0 or 1). |
| DK<cr> DKn<cr> | RK05 bootstrap | Execute RK05 bootstrap for disk 0 (default) or disk n (0 or 1). |

NOTE

1. AL, AR, DX, and DK also execute a memory and processor diagnostic program before loading.
2. <cr> is a carriage return (octal code=015) command delimiter required by all commands except OD.
3. REV11-A and REV11-C ROM starting address is 173000, resulting in non-memory-modifying processor diagnostic test execution. Successful completion results in the $ prompt character being displayed.

## 3.3.3 The M9301 Console Emulator

The M9301 module contains a console emulator routine. When this routine is used in conjuction with the user's terminal, it generates functions quite similar to those found on the programmer's console of traditional PDP-11 family computers.

Summary of the Console Emulator Functions

| | |
|---|---|
| LOAD | - this function loads the address to be manipulated into the system. |
| EXAMINE | - allows the operator to examine the contents of the address that was loaded and/or deposited. |
| DEPOSIT | - allows the operator to write into the address that was loaded and/or examined. |
| START | - initializes the system and starts execution of the program at the address loaded. |
| BOOT | - allows the booting of a specified device by typing in a two character code and optional unit number. |

## CONSOLE EMULATOR OPERATION

The console emulator allows the user to perform load, examine, deposit, start, and boot functions by typing in appropriate code on the keyboard.

There are three ways of entering the console emulator:

1. Move the power switch to the on position.

2. Depress the boot switch.

3. Automatic entry on return from a power failure.

After the console emulator routine has started and the basic CPU diagnostics have all run successfully, a series of numbers representing the contents of R0,R4, SP and PC respectively, will be printed by the terminal. This sequence will be followed by a $ on the next line.

************************************************************

EXAMPLE--A TYPICAL PRINTOUT ON POWER UP:

```
XXXXXX          XXXXXX          XXXXXX          XXXXXX
$
   R0              R4              R6              PC
                                stack           program
prompt                          pointer         counter
character                         (sp)
```

NOTE

x signifies an octal number (0-7).
Whenever there is a power up routine, or the BOOT switch is released from the INIT position, the PC at this time will be stored. The stored value is printed out as above (noted as the PC).

************************************************************

## USING THE CONSOLE EMULATOR

Once the system has been powered up or booted, and R0, R4, SP, PC and $ have been printed, the console emulator routine can be used.

Keyboard input symbols--the discussion of keyboard input format uses THE FOLLOWING SYMBOLS:

1. space bar: (SB)

2. carriage return key: (CR)
   All commands are terminated with a carriage return

3. any number 0-7 (octal number) key: (x)

Keyboard input format--load, examine, deposit, start. All character keys shown in the following discussion respresent themselves with the exception of those in parentheses.

| FUNCTION | |
|---|---|
| LOAD ADDRESS | L (SB) (x) (x) (x) (x) (x) (x) |
| EXAMINE | E (SB) |
| DEPOSIT | D (SB) (x) (x) (x) (x) (x) (x) |
| START | S (CR) |

Order of significance of input keys--the first character that is typed will be the most significant character. Conversely, the last character that is typed is the least significant character.

Number of characters---the console emulator routine can accept up to six octal numbers in the range of 0-32k. If all six numbers are input, the most significant number should be a one or a zero.

Leading zeros--when an address or data word contains leading zeros, these zeros can be omitted when loading the address or depositing the data.

NOTE

Even addresses only--the console emulator routine will not work with odd addresses. Even numbered addresses must always be used.

```
*************************************************************

Example using the load, examine, deposit, and start
function--assume that a user wishes to:

1.  turn on power

2.  load address 700

3.  examine location 700

4.  deposit 777 into location 700

5.  examine location 700

6.  start at location 700


         USER                    TERMINAL DISPLAY
         ----                    ----------------

1.  turns on power          xxxxxx xxxxxx xxxxxx xxxxxx

2.  L (SB) 700 (CR)         $ L 700

3.  E (SB)                  $ E 000700 xxxxxx

4.  D (SB) 777 (CR)         $ D 777

5.  E (SB)                  $ E 000700 000777

6.  S (CR)                  $ S

**************************************************************
```

## SUCCESSIVE OPERATIONS

EXAMINE--successive examine operations are permitted. The address is loaded for the first examine only. Successive examines cause the address to increment by two and will display consecutive addresses along with their contents.

******************************************************************

Example of successive examine operations--examine addresses 500-506.

```
OPERATOR INPUT                    TERMINAL DISPLAY
--------------                    ----------------

L (SB) 500 (CR)                   $L 500
E (SB)                            $E 000500 xxxxxx
E (SB)                            $E 000502 xxxxxx
E (SB)                            $E 000504 xxxxxx
E (SB)                            $E 000506 xxxxxx
```

******************************************************************

DEPOSIT--successive deposit operations are permitted. The procedure is identical to that used with examine.

************************************************************

Example of successive deposit operations

```
Deposit:   60 into location 500
            2 into location 502
            4 into location 504

    OPERATOR INPUT                     TERMINAL DISPLAY
    --------------                     ----------------

    L (SB) 500 (CR)                     $L 500
    D (SB) 60 (CR)                      $D 60
    D (SB) 2 (CR)                       $D 2
    D (SB) 4 (CR)                       $D 4
```

*******************************************************

ALTERNATE DEPOSIT-EXAMINE OPERATIONS---this mode of operation will not increment the address. The address will contain the last data which was deposited.

******************************************************

Example of alternate deposit-examine operations---load address 500, deposit the following numbers with examines after every deposit: 1000, 2000, 5420.

```
    OPERATOR INPUT                     TERMINAL DISPLAY

    L (SB) 500 (CR)                     $L 500
    D (SB) 1000 (CR)                    $D 1000
    E (SB)                              $E 000500 001000
    D (SB) 2000 (CR)                    $D 2000
    E (SB)                              $E 000500 002000
    D (SB) 5420 (CR)                    $D 5420
    E (SB)                              $E 000500 005420
```

******************************************************

LIMITS OF OPERATION--The M9301 console emulator routine can directly manipulate the lower 28K of memory and the 4K I/O page. Refer to the PDP-11/34 Processor Handbook for a procedure to utilize the memory management unit to examine or deposit in expanded memory.

# CHAPTER 4

# BOOTSTRAP PROCEDURES

This chapter explains how to bootstrap a CPU. Bootstrapping is a procedure that starts a CPU. Essentially, you load and execute a very short program called a bootstrap loader, whose only function is to load and start a larger monitor program (for example, the absolute loader or an XXDP monitor).

Here are the general methods for bootstrapping a CPU:

1. key-in the bootstrap loader program manually and execute it.

2. load the starting address of a bootstrap ROM (a bootstrap loader program contained in read-only-memory) and start the CPU.

3. activate a "boot" button or switch on the console.

4. issue bootstrap commands under ODT-11 or REV11 (LSI-11 and 11/03).

This chapter assumes that you understand all the information in Chapter 3 that relates to your operator's console.

## 4.1 TOGGLE-IN THE BOOTSTRAP LOADER PROGRAM

If the CPU you are using has no bootstrap ROM and no bootstrap switch on the console, then you must manually key-in the bootstrap loader program into memory and execute it. This is called the toggle-in procedure. The bootstrap loader program that you toggle-in varies according to the device you are using to load the monitor (the monitor load device). The following are the actual toggle-in programs, arranged by monitor load device (controller/drive). The programs are also listed on the XXDP/DECX11 Programming Card.

### 4.1.1 PC11 and DL11

(high-speed paper tape reader; teletype console paper tape reader)

1. Halt the processor.

2. Choose a 3-digit number from Table 4-1, corresponding to the memory available.

Table 4-1. Address Prefixes for Paper Tape Bootstrap Loader

| MEMORY SIZE | PREFIX |
|---|---|
| 4K | 017 |
| 8K | 037 |
| 12K | 057 |
| 16K | 077 |
| 20K | 117 |
| 24K | 137 |
| 28K or greater | 157 |

3. Deposit the following bootstrap loader program into memory, using the 3-digit prefix from Table 4-1 for the first 3 digits of each address, and also for the first 3 digits of the contents of location xxx 766.

```
ADDRESS             CONTENTS
--------            --------
xxx 744             016 701
xxx 746             000 026
xxx 750             012 702
xxx 752             000 352
xxx 754             005 211
xxx 756             105 711
xxx 760             100 376
xxx 762             116 162
xxx 764             000 002
xxx 766             xxx 400
xxx 770             005 267
xxx 772             177 756
xxx 774             000 765
xxx 776             177 560 if load device is teletype reader.
                    177 550 if load device is PC11.
```

4. Place the absolute loader paper tape in the reader.

NOTE

The absolute loader is the only monitor for use with paper tape systems. You must have a copy of the absolute loader on paper tape to use this procedure. There is a separate absolute loader for CPU's that have HW/SWR's and for those with SW/SWR's.

1. Load address xxx 744.
2. Put the HALT/ENABLE switch to ENABLE.
3. Start the CPU.
4. You have loaded the absolute loader. It is not self-starting: it requires your intervention to load and execute programs. Proceed to Chapter 5.

### 4.1.2 RK11/RK05 (DECpack Disk Cartridge)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor DECpack volume on drive 0, start the device, and write-protect it.

3. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 010 000 | 012 737 |
| 010 002 | 000 005 |
| 010 004 | 177 404 |
| 010 006 | 000 001 |

4. Load address 010 000.

5. Put the HALT/ENABLE switch to ENABLE.

6. Start the CPU.

7. Wait one second and halt the CPU.

8. Load address 000 000.

9. Put the HALT/ENABLE switch to ENABLE.

10. Start the CPU again.

11. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

## 4.1.3 IC11/IU56 (DECtape)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor DECtape volume on drive 0, start the device, and write-protect it.

3. Deposit the value 004 003 at location 177 342. The tape will rewind and stop in the end zone. The "remote" light on the drive should remain lit.

4. Examine the current location.

5. Deposit the value 000 001 in the current location. The remote light should go out.

6. Deposit the following values at the locations specified:

| ADDRESS | CONTENTS |
|---|---|
| 000 216 | 012 737 |
| 000 220 | 000 005 |
| 000 222 | 177 342 |
| 000 224 | 000 777. |

7. Load address 000 216.

8. Put the HALT/ENABLE switch to ENABLE.

9. Start the CPU.

10. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

## 4.1.4 IM11/IU10 (7-track Magtape) And IM11/IS03 (9-track Magtape)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor magtape volume on drive 0, start the drive, and write-protect it.

3. Rewind drive 0 to "bot" and set "on-line".

4. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 010 000 | 005 137 |
| 010 002 | 172 524 |
| 010 004 | 012 737 |
| 010 006 | 060 011 |
| 010 010 | 172 522 |
| 010 012 | 000 777 |
| 010 014 | 012 737 |
| 010 016 | 060 003 |
| 010 020 | 172 522 |
| 010 022 | 105 737 |
| 010 024 | 172 522 |
| 010 026 | 100 375 |
| 010 030 | 000 137 |
| 010 032 | 000 000 |

5. Load address 010 000.

6. Put the HALT/ENABLE switch to ENABLE.

7. Start the CPU.

8. Wait one second and halt the CPU.

9. Load address 010 014.

10. Put the HALT/ENABLE switch to ENABLE.

11. Start the CPU again.

12. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

## 4.1.5 IM02/IU16 (9-track Magtape)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor magtape volume on drive 0, start the drive, and write-protect it.

3. Rewind drive 0 to "BOT" and set "on-line".

4. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 010 000 | 012 737 |
| 010 002 | 001 300 |
| 010 004 | 172 472 |
| 010 006 | 012 737 |
| 010 010 | 177 777 |
| 010 012 | 172 446 |
| 010 014 | 012 737 |
| 010 016 | 000 031 |
| 010 020 | 172 440 |
| 010 022 | 105 737 |
| 010 024 | 172 452 |
| 010 026 | 100 375 |
| 010 030 | 012 737 |
| 010 032 | 177 400 |
| 010 034 | 172 442 |
| 010 036 | 005 037 |
| 010 040 | 172 444 |
| 010 042 | 042 737 |
| 010 044 | 000 007 |
| 010 046 | 172 452 |
| 010 050 | 012 737 |
| 010 052 | 000 071 |
| 010 054 | 172 440 |
| 010 056 | 105 737 |
| 010 060 | 172 440 |
| 010 062 | 000 100 |
| 010 064 | 000 375 |
| 010 066 | 000 137 |
| 010 070 | 000 000 |

5. Load address 010 000.

6. Put the HALT/ENABLE switch to ENABLE.

7. Start the CPU.

8. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

### 4.1.6 TA11/TU60 (Cassette)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor cassette on drive 0, and write-protect it.

3. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 001 000 | 012 700 |
| 001 002 | 177 500 |
| 001 004 | 005 010 |
| 001 006 | 010 701 |
| 001 010 | 062 701 |
| 001 012 | 000 052 |
| 001 014 | 012 702 |
| 001 016 | 000 375 |
| 001 020 | 112 103 |
| 001 022 | 112 110 |
| 001 024 | 100 413 |
| 001 026 | 130 310 |
| 001 030 | 001 776 |
| 001 032 | 105 202 |
| 001 034 | 100 772 |
| 001 036 | 116 012 |
| 001 040 | 000 002 |
| 001 042 | 120 337 |
| 001 044 | 000 000 |
| 001 046 | 001 767 |
| 001 050 | 000 000 |
| 001 052 | 000 755 |
| 001 054 | 005 710 |
| 001 056 | 100 774 |
| 001 060 | 005 007 |
| 001 062 | 017 640 |
| 001 064 | 002 415 |
| 001 066 | 112 024 |

4. Load address 001 000.

5. Put the HALT/ENABLE switch to ENABLE.

6. Start the CPU.

7. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

### 4.1.7 RX11, RXV11/RX01 (Floppy Disk)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor floppy disc volume on drive 0 and start the device.

3. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 001 000 | 005 000 |
| 001 002 | 012 701 |
| 001 004 | 177 170 |
| 001 006 | 105 711 |
| 001 010 | 001 776 |
| 001 012 | 012 711 |
| 001 014 | 000 003 |
| 001 016 | 005 711 |
| 001 020 | 001 776 |
| 001 022 | 100 405 |
| 001 024 | 105 711 |
| 001 026 | 100 004 |
| 001 030 | 116 120 |
| 001 032 | 000 002 |
| 001 034 | 000 770 |
| 001 036 | 000 000 |
| 001 040 | 005 000 |
| 001 042 | 000 110 |
| 001 044 | 000 000 |
| 001 046 | 000 000 |
| 001 050 | 000 000 |

4. Load addcess 010 000.

5. Put the HALT/ENABLE switch to ENABLE.

6. Start the CPU.

7. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

## 4.1.8 RP11/RP02, RP03 (Multi-surface Disk Pack)

1. Halt the CPU.
2. Prepare the monitor load device: mount the monitor disk pack volume on drive 0, start the device, and write-protect it.
3. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 001 000 | 012 705 |
| 001 002 | 176 716 |
| 001 004 | 012 715 |
| 001 006 | 177 400 |
| 001 010 | 012 745 |
| 001 012 | 000 005 |
| 001 014 | 105 715 |
| 001 016 | 100 376 |
| 001 020 | 005 007 |

4. Load address 001 000.
5. Put the HALT/ENABLE switch to ENABLE.
6. Start the CPU.
7. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

4.1.9 RH11, RH20/RP04, RP05, RP06 (multi-surface Disk Pack)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor disk volume on drive 0, start the device, and write-protect it.

3. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 010 000 | 012 700 |
| 010 002 | 176 700 |
| 010 004 | 012 710 |
| 010 006 | 000 023 |
| 010 010 | 005 060 |
| 010 012 | 000 034 |
| 010 014 | 005 060 |
| 010 016 | 000 006 |
| 010 020 | 012 760 |
| 010 022 | 177 400 |
| 010 024 | 000 002 |
| 010 026 | 012 710 |
| 010 030 | 000 071 |
| 010 032 | 105 710 |
| 010 034 | 100 316 |
| 010 036 | 005 007 |

4. Load address 010 000.

5. Put the HALT/ENABLE switch to ENABLE.

6. Start the CPU.

7. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

## 4.1.10 RH11, RHZ0/RS03, RS04 (Single-disk, Fixed Head)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor disk volume on drive 0, start the device, and write-protect it.

3. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 001 000 | 012 705 |
| 001 002 | 172 044 |
| 001 004 | 012 745 |
| 001 006 | 177 400 |
| 001 010 | 012 745 |
| 001 012 | 000 071 |
| 001 014 | 032 715 |
| 001 016 | 100 200 |
| 001 020 | 001 775 |
| 001 022 | 100 762 |
| 001 024 | 005 007 |

4. Load address 001 000.

5. Put the HALT/ENABLE switch to ENABLE.

6. Start the CPU.

7. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

## 4.1.11 RK611/RK06 (Double High-density Disk Cartridge)

1. Halt the CPU.

2. Prepare the monitor load device: mount the monitor RK06 pack on drive 0, start the device, and write-protect it.

3. Deposit the following values at the specified locations:

| ADDRESS | CONTENTS |
|---|---|
| 010 000 | 012 737 |
| 010 002 | 000 003 |
| 010 004 | 177 440 |
| 010 006 | 012 737 |
| 010 010 | 177 000 |
| 010 012 | 177 442 |
| 010 014 | 012 737 |
| 010 016 | 000 021 |
| 010 020 | 177 440 |
| 010 022 | 000 001 |

4. Load address 010 000.

5. Put the HALT/ENABLE switch to ENABLE.

6. Start the CPU.

7. Wait one second and halt the CPU.

8. Load address 000 000.

9. Put the HALT/ENABLE switch to ENABLE.

10. Start the CPU again.

11. The monitor is loaded. Proceed to the chapter that describes the monitor you are using.

## 4.2 BOOTSTRAPPING WITH THE M9301 MODULE

1. Halt the CPU.

2. Load the M9301 start address: 173000, or 773000 under memory management.

3. If the CPU has HALT/ENABLE switch, place it in the ENABLE position.

4. Start the CPU. You should receive a prompt character, $.

5. Prepare the monitor load device: load the monitor volume on unit 0, start the device, and write-protect it.

6. Determine the two-character code for that device from Table 4-2.

7. Issue the two-character code as a command. (You can append a number from 0-7 to the command if you wish to specify a device unit.)

8. The monitor is loaded. Some monitors are self-starting; others, such as the absolute loader, require your intervention to load and execute programs. Proceed to the chapter that describes the monitor you are using.

Table 4-2. Bootstrap Device Codes Supported By M9301.

| CONTROLLER/DEVICE | DESCRIPTION | BOOT COMMAND |
|---|---|---|
| RK11/RK05 | DISK CARTRIDGE | DK |
| RP11/RP02,03 | RP02/03 DISK PACK | DP |
| TC11/TU56 | DECTAPE | DT |
| TM11/TU10,TS03 | 800 BPI MAGTAPE | MT |
| TA11/TU60 | MAGNETIC CASSETTE | CT |
| RX11,RXV11/RX01 | DISKETTE | DX |
| DL11/TTY | ASR-33 TELETYPE | TT |
| PC11 | PAPERTAPE | PR |
| RH11,RH70/RS03,04 | FIXED HEAD DISK | DS |
| RH11,RH70/RP04,05,06 | DISK PACK | DB |
| TM02/TU16 | MAGNETIC TAPE | MM |
| RK611/RK06 | DISK | DM |

NOTE

Be sure that your version of M9301 supports the load device you are using.

## 4.3 BOOTSTRAP LOADER ROM'S

Here is the procedure for loading a monitor with a bootstrap ROM.

1. Determine the bootstrap ROM included in your CPU.
2. Halt the CPU.
3. Prepare the monitor load device: load the monitor volume on unit 0, start the device, and write-protect it.
4. Determine the ROM start address for that device from Table 4-3.

NOTE
----

For multi-unit devices, the default load unit is unit 0. Consult the PDP-11 Programming Card for ROM start addresses for loading device units other than unit 0.

5. Load the ROM start address for the device into the CPU.
6. If the CPU has a HALT/ENABLE switch, put it to the ENABLE position.
7. Start the CPU.
8. The monitor is loaded. Some monitors are self-starting; others, such as the absolute loader, require your invention to load and execute programs. Proceed to the chapter that describes the monitor you are using.

Table 4-3. Bootstrap ROM Starting Addresses for Specific Devices

| CONTROLLER/DEVICE | BM873-YB | BM873-YA | MR11-DB | BM792 |
|---|---|---|---|---|
| RX11, RXV11/RX01 | n. s. | n. s. | n. s. | 773400 |
| RK11/RK05 | 773030 | 773010 | 773110 | 773100 (SWR=777406) |
| TC11/TU56 | 773070 | 773030 | 773120 | 773100 (SWR=777344) |
| TM11/TU10, TS03 | 773110 | 773050 | 773136 | n. s. |
| TM02/TU16 | 773150 | n. s. | n. s. | n. s. |
| TA11/TU60 | 773524 | 773230 | n. s. | 773300 |
| RP11/RP02, 03 | 773350 | 773100 | 773154 | n. s |
| RH11, RH70/RP04, 05, 06 | 773320 | n. s. | n. s. | n. s. |
| RH11, RH70/RS03, 04 | 773000 | n. s. | n. s. | n. s. |
| RF11 | 773136 | 773000 | 773100 | n. s. |
| RC11 | 773212 | 773144 | 773220 | n. s. |
| PC11 | 773620 | 773312 | n. s. | n. s. |
| KL11, DL11/TTY | 773510 | n. s. | n. s. | n. s. |

NOTE

When loading from RK11 (RK05 DECpack) and TC11 (DECtape) using the BM792 bootstrap ROM, you must also set the switch register to the values given in the table before you start the CPU.

n. s. means that the device is not supported by that ROM.

## 4.4 USING A BOOT SWITCH OR BOOT BUTTON

Some CPU's have a "boot" switch or button on the operator's console. The operation of this switch depends on the design of the system, but in any case, it is a fast convenient way to execute the program contained in a bootstrap ROM.

In some systems the ROM will display a prompt character, $. Here, you must direct the ROM to the loading device by specifying a two-character code (refer to Table 4-2). Then the ROM will load the monitor.

In other systems the switch or button starts a ROM that reads from a predetermined hardware-specified device. You have only to prepare that device and activate the switch or button: the ROM will then load the monitor automatically.

As with any of the other bootstrap procedures, some monitors will self-start; others, such as the absolute loader require your intervention to load and execute programs. When you have loaded the monitor, proceed to the chapter that describes the monitor you are using.

## 4.5 USING A ROM AT POWER-UP

Power-up bootstrap ROM's operate exactly like the "boot" switch procedures that display the prompt character, $. (Refer to 4.4 above). You direct the ROM to the monitor loading device by specifying a two-character code (see Table 4-2).

## 4.6 BOOTSTRAPPING UNDER ODT-11 (LSI-11 OR 11/03)

1. Ensure that the CPU is at ODT command level with a prompt character, @. Refer to 3.3.1.

2. Prepare the device from which you are loading the monitor.

3. Determine the read-control-status-register adaress (RCSR address) for that device.

4. Issue the ODT load command, L, using the device RCSR

   address: @nnnnnnL where nnnnnn is RCSR.

5. The monitor is loaded. Some monitors will self-start; others, such as the absolute loader require your intervention to load and execute programs. Proceed to the chapter that describes the monitor you are using.

## 4.7 BOOTSTRAPPING UNDER REV11 (LSI-11 OR 11/03)

1. Ensure that the CPU is at REV11 command level with the prompt character, $. Refer to 3.3.2.

2. Prepare the device from which you are loading the monitor.

3. Determine the RCSR address for that device.

4. Issue one of the following REV11 commands:
   - AL, if you are loading the absolute loader.
   - AR, if you are loading the absolute relocatable loader.
   - DX, if you are loading the monitor from floppy disc. This command does not require RCSR, but you may specify a device unit.
   - DK, if you are loading the monitor from RK05. This command does not require RCSR, but you may specify a device unit.

5. The monitor is loaded. Some monitors are self-starting; others, such as the absolute loader, require your intervention to load and execute programs. Proceed to the chapter that describes the monitor you are using.

| | | |
|---|---|---|
| Examples: | $AL | loads the absolute loader from the console teletype reader, the default device. |
| | $AL177550 | loads the absolute loader from PC11 paper tape reader. |
| | $AR | loads absolute relocatable loader from console teletype reader. |
| | $AR177550 | loads absolute relocatable loader from PC11 paper tape reader. |
| | $DX | loads monitor from floppy disk 0. |
| | $DX1 | loads monitor from floppy disk 1. |
| | $DK | loads monitor from RK05 disk 0. |
| | $DK1 | loads monitor from RK05 disk 1. |

For more information on REV11 commands, see paragraph 3.3.2.

# CHAPTER 5

# THE ABSOLUTE LOADER

The absolute loader is the monitor for paper-tape-based systems. The basic operating procedure is as follows:

1. load if necessary, and execute a **bootstrap loader** that will--

2. load the **absolute loader**, which in turn will enable you to--

3. load your **programs** from paper tape.

## 5.1 LOADING THE ABSOLUTE LOADER

If your system requires you to **toggle-in** the bootstrap loader, then follow the procedure given under 4.1.1.

If you are operating under an **M9301 module**, then bootstrap the monitor as described under 4.2.

If your system has a **bootstrap ROM**, then follow the instructions under 4.3.

When you have bootstrapped the absolute loader, proceed to paragraph 5.2.

## 5.2 LOADING DIAGNOSTIC PROGRAM TAPES

Once you have loaded the absolute loader, you can then load your diagnostic programs from paper tape.

Here is the normal procedure for loading program tapes:

1. Place the diagnostic paper tape into the paper tape reader. (Set HW/SWR to zeroes--all switches down.)

2. If your CPU console has a CONT control switch, activate it. The absolute loader should load the diagnostic paper tape. (Go to Step 5.) If it doesn't load it, go to Step 4.

3. If you are operating from a terminal device, issue the ODT proceed command, P. The absolute loader should load the diagnostic paper tape. (Go to Step 5.) If it doesn't, go to Step 4.

4. Some systems require you to start again after loading the absolute loader. In this case you must load the starting address of the absolute loader, xxx500 (xxx determined from Table 4-1), and start the CPU. (Under ODT you would issue the Go command, G, with the absolute loader starting address: @xxx500G.)

5. The diagnostic is loaded. The absolute loader will not automatically start execution of diagnostic programs. Proceed to paragraph 5.4.

NOTE

There is a separate absolute loader for CPU's with HW/SWR's and for those with SW/SWR's. The two absolute loaders function alike, but they access different locations to read switch settings. (See also Chapter 6)

If you are using the version of the absolute loader that reads HW/SWR's and your CPU has SW/SWR's, then you must make the following alterations to the absolute loader in the CPU before loading your diagnostic program tapes. (If the absolute loader version matches the CPU, then use the normal program loading procedure above.)

1. Load address xxx516 (xxx determined from Table 4-1). This location is the pointer to the CPU HW/SWR.

2. Deposit the value 000176 in location xxx516. Now the pointer contains the address of the SW/SWR.

3. Load address 000176 and deposit zeroes (unless you need a specific switch setting). You are setting the SW/SWR to zeroes.

4. Follow instructions 1, 4, and 5 of the normal procedure above.

## 5.3 OVERLAYING ANOTHER DIAGNOSTIC PROGRAM

You can use the procedure described in Step 4 of the normal procedure above to load in another diagnostic program at any time. It will overlay the one currently in memory. However, that program currently in memory must not have altered the absolute loader during its execution.

## 5.4 EXECUTING THE PROGRAM

The program loaded and the switch register set (Chapter 6), you are ready to begin program execution. Follow this procedure.

For CPU's **without terminals**:

1. HALT the CPU.
2. LOAD address 200.
3. ENABLE the CPU.
4. Activate START switch.

For CPU's **with terminals**:

1. Put CPU at terminal command level: ODT or M9301 command level. (See 3.3).
2. Under ODT issue a Go command, G, and specify address 200:

```
@200G
```

3. Under M9301 load address 200 and issue a start command:

```
$ L 200 <CR>

$ S
```

CPU is now under program control.

## 5.5 RESTARTING THE PROGRAM

Refer to the appropriate appendix for specific restart addresses. Restarting allows you to resume program execution without going through the whole start procedure again.

# CHAPTER 6

# SWITCH REGISTERS

A *switch register* is a set of switches that a program looks at in order to determine which paths of execution it should take.

## 6.1 *HARDWARE SWITCH REGISTER*

The *hardware switch register*, or HW/SWR, is a one-word hardware location at UNIBUS address 177570, or 777570 with memory management. You load the HW/SWR with an appropriate value, depending on what you want the program to do. If you do not know the specific switch settings for a program, then set the HW/SWR to *zeroes*. This is the standard setting. (If you load a HW/SWR with *all ones*, a diagnostic program will look instead at location 000176, the *software switch register*, for its directions. See paragraph 6.2).

Some processors allow you to load the HW/SWR at any time during program execution using the *console switch panel*. The switch panel corresponds bit-by-bit to the HW/SWR. If a panel switch is up, the corresponding bit in the HW/SWR is "on" or equal to 1; down is "off" or 0. When the CPU is *running*, the switch panel setting *is* the current value of the HW/SWR.

Processors with *keypad consoles* allow you to load the HW/SWR at any time by using the LSR button (load-switch-register). You can enter up to *six octal digits*; the CPU truncates to the low-order sixteen bits. You *cannot load* the HW/SWR on a keypad console with LAD + DEP operation. Likewise, you *cannot display* the contents of the HW/SWR with a LAD + EXAM operation. The CPU does display the HW/SWR contents at START or CONT time.

## 6.2 SOFTWARE SWITCH REGISTER

Some CPU's do not have a HW/SWR (e.g., LSI-11 or 11/03). In these CPU's diagnostic programs refer instead to location 000 176 called the software switch register, or SW/SWR. The SW/SWR provides the same function as the HW/SWR. However, the loading procedures are different. See paragraph 6.3.

NOTE

You can force a diagnostic to treat a HW/SWR CPU as though it were a SW/SWR CPU. There are two ways to do this:

1. Set the HW/SWR to all ones. Most diagnostics will refer to location 000176 for the switch register setting under this condition.

2. Halt the CPU after loading the diagnostic, but before starting it. Using the cross-reference table in the diagnostic's program listing, find the location labelled "SWR" or "SWREG". Change the contents of this location to 000176 (i.e., load its address and deposit 000 176). Return control to the CPU: either load address 200, enable and start the CPU; or, load the XXDP restart address given in the ready message, enable and start the CPU, and issue an XXDP start command, S. (See 7.4.3).

## 6.3 LOADING THE SOFTWARE SWITCH REGISTER

When you start a diagnostic, it will check to see whether the processor has a HW/SWR. If there is no HW/SWR (or if you have set it to all ones), the diagnostic will look at location 000 176 the software switch register, for the switch setting.

Some diagnostics allow you to modify the SW/SWR while they are executing. This is called dynamic switch register modification. Such a diagnostic will give you the following message:

```
SWR = nnnnnn         NEW SWR =
```

and expect you to enter the value you need; nnnnnn is the current setting. At any time thereafter, you may change the SW/SWR by typing a G (or CNTRL G) at the terminal. The value you enter can be up to six octal digits; the diagnostic will truncate the number to the low-order sixteen bits in case of overflow. If you do not enter a new value, the old SW/SWR setting remains. You must complete the operation with a carriage return.

In addition, if you enter a erroneous value, you can delete it before you have done a carriage return by typing U (or CNTRL U). The diagnostic will then accept the corrected switch register setting.

NOTE

> All CNTRL operations ( ) mean that you press the CNTRL-key ( -key) simultaneously with a letter key.

If the diagnostic does not provide dynamic switch register modification, then you must set the SW/SWR before starting the diagnostic program. You do this by halting the CPU, loading address 000 176, and depositing the switch register value that the diagnostic requires. Then you load either the program's start address (usually location 200) or the monitor's restart address (given in the ready message); then enable and start the CPU. If you are returning to the monitor, then you must issue a start command, S, to start the diagnostic program (see 7.4.3).

## 6.4 STANDARD SWITCH SETTINGS

Most diagnostics use switches to set test parameters. The following are standard switch settings. A switch is set when it equals one.

Table 6-1 Switch Register Settings

| HW SWITCH | KEYPAD CPU & SW/SWR SETTING | OPTION |
|---|---|---|
| 15 | 100 000 | halt on error |
| 14 | 040 000 | loop on test |
| 13 | 020 000 | inhibit error typeouts |
| 12 | 010 000 | inhibit trace trap |
| 11 | 004 000 | inhibit iterations |
| 10 | 002 000 | bell on error |
| 9 | 001 000 | loop on error |
| 8 | 000 400 to<br>000 777 | loop on test<br>specified in bits 0-7 of SWR. |

NOTE

The keypad and SW/SWR settings are accumulative. If you want more than one option, then add the SWR values and enter the sum as the SWR setting. For example, if you want to designate the first four options in the list, add the first four SWR settings. Then enter 170 000 as the SWR setting. Likewise, choosing the first, third, fifth and seventh options will give a SWR setting of 125 000.

SWITCH 15 - HALT ON ERROR

This switch is checked in the error routines and when found to be on a one the program will halt.

SWITCH 14 - LOOP ON TEST

When set, this switch will cause the diagnostic to loop on the test presently being executed.

SWITCH 13 - INHIBIT ERROR TYPEOUTS

When set, error messages will not be typed (output) on the console.

SWITCH 12 - INHIBIT TRACE TRAP

Normally a program will run with the "T" bit set on alternate passes of the program. If this switch is set (equal to one) the "T" bit will not be set.

SWITCH 11 - INHIBIT ITERATIONS

If this switch is set, the subtest being performed will be iterated only one time.

SWITCH 10 - BELL ON ERROR

If set, and error will cause the TTY bell to ring.

SWITCH 09 - LOOP ON ERROR

If this switch is set, the diagnostic will loop on the test that caused the error. If the error goes away during the looping, the diagnostic will proceed to the next test.

SWITCH 08 - LOOP ON TEST IN SWR<07:00>

if this switch is set, the diagnostic will loop on the test number specified by bits <07:00> of the switch register (SWR).

NOTE
----

Many diagnostics do not use this convention. Such diagnostics need specific input parameters and therefore need special switch settings.

# CHAPTER 7

## LOADING & RUNNING DIAGNOSTICS UNDER XXDP

XXDP is a group of PDP-11 diagnostic software monitors that includes the following:

| | |
|---|---|
| TCDP | - TC11 diagnostic package (DECTAPE). |
| RKDP | - RK11 diagnostic package (DECPACK). |
| TMDP | - TM11/TM02 diagnostic package (7 or 9 track MAGTAPE). 9 track can be loaded from TU10 or TU16 |
| TADP | - TA11 diagnostic package (TM11 Cassettes). |
| RXDP | - RX11 diagnostic package (Floppy Disk). |
| RPDP | - RP11 diagnostic package. |
| RBDP | - RH11/RP04 diagnostic package. |
| RSDP | - RH11/RS03 diagnostic package. |
| RMDP | - RK06 diagnostic package. |

The XXDP packages contain the diagnostic monitors, diagnostic utilities, and diagnostic programs on media other than paper tape. XXDP packages have the following advantages:

1. Easy and convenient means of loading programs under keyboard control.

2. Means are provided for updating and modifying programs.

3. Possible to sequentially run a series of programs through use of the "chain mode" feature. Programs must be chainable. See paragraph 7.7.

All XXDP packages require:

1. PDP-11 processor with at least 8K storage.
2. Console terminal device.
3. One of the diagnostic package media:
    1. TC11 dectape control and TU56 transport or,
    2. RK11 disk control and RK03 or RK05 drive or,
    3. TA11 control and TU60 cassette drive or,
    4. TM11 magtape control and TU10 magtape drive or,
    5. TM02 magtape control unit and TU16 drive or,
    6. RX11/RXV11 floppy control unit and RX01 floppy drive or,
    7. RP11 disk controller and RP03 drive or,
    8. RH11/RH70 disk controller and RP04/RP05/RP06 drive or,
    9. RH11/RH70 disk controller and RS03/RS04 drive or,
    10. RK611 disk controller and RK06 drive.

The above requirements are for loading and running diagnostic programs already stored on one of the diagnostic package media.

## 7.1 LOADING AN XXDP MONITOR

You load an XXDP monitor as described under Chapter 4 Bootstrap Procedures.

First, determine your processor configuration: does it have bootstrap ROM's or M9301? a "boot" button or switch? must you toggle-in? etc.

Then, depending on the load medium you are using, refer to the appropriate paragraph to see how to bootstrap from that device.

When you have successfully loaded the monitor, it gives you the following message:

```
aaaaa-a          dd-mmm-yy       xxxx-xxxx MONITOR    nnK
RESTART ADDRESS:  rrrrrr
BOOTED VIA UNIT#:   b
TO ABORT THE FOLLOWING HELP MESSAGE TYPE CTRL C ( C)

TYPE:
F<CR> TO SET CONSOLE FILL COUNT.
D<CR> FOR DIRECTORY ON CONSOLE.
D/F<CR> FOR SHORT DIRECTORY ON CONSOLE.
D/L<CR> FOR DIRECTORY ON LINE PRINTER.
D/L/F<CR> FOR SHORT DIRECTORY ON LINE PRINTER.
R COPY<CR> TO RUN COPY PROGRAM.
R FILENAME<CR> TO RUN ANY OTHER PROGRAM.
L FILENAME<CR> TO LOAD A PROGRAM ONLY.
S<CR> TO START THE PROGRAM JUST LOADED.
S ADDR<CR> TO START THE PROGRAM AT SPECIFIC ADDRESS.
C FILENAME<CR> TO RUN A CHAIN.
C FILENAME/QV<CR> TO RUN A CHAIN IN QUICK VERIFY MODE.
REFER TO XXDP MANUAL MD-11-DZQXA FOR ADDITIONAL HELP.
.
```

| | |
|---|---|
| aaaaa-a | -- the name of the MAINDEC program module. |
| dd-mmm-yy | -- the module release date. |
| xxxx-xxxx | -- the name of the monitor. |
| nnK | -- the system storage up to 28K. |
| rrrrrr | -- the monitor's restart address (see 7.2). |
| b | -- the device unit or device drive number from which you bootstrapped the monitor; the default drive or unit is 0. |
| . | -- the monitor is ready to accept commands. |

## 7.2 HOW TO USE AN XXDP MONITOR

XXDP issues a prompt character, the period (.), when it is loaded and ready to accept XXDP commands. A description of these commands appears below, paragraph 7.4.

However, while operating under XXDP, you can at any time put the CPU at the console command level. Use the HALT/ENABLE switch, or equivalent means, to stop processing. (See also paragraph 3.3, "Terminal Device Consoles"). This will allow you to modify memory locations; or "toggle-in" a patch to a program; or manually start or restart a program; or reset the software switch register; etc.

After you have halted the CPU and modified memory as needed, you can then return control to whichever program you want -- the XXDP monitor or a diagnostic program. To resume or start execution of a program, not the XXDP monitor, simply load the start address (usually location 200), or the restart address of that program and start the CPU.

If you are returing control to the XXDP monitor, then load its restart address (given in the ready message), and start the CPU. NOTE: If you or an executing program have destroyed any of the XXDP monitor in memory, the results of this restart are unpredictable.

Also remember that when you reload an XXDP monitor, your memory modifications are destroyed. More specific examples of these procedures appear below.

**CAUTION**

When running diagnostics that test the XXDP monitor load device, be sure to write-protect the monitor volume or the drive that is holding it; otherwise, you may accidentally destroy the monitor on the medium. Also, if you need to test the particular drive or unit that holds the monitor volume, then remove the monitor volume from that drive and mount a "scratch" volume in its place.

```
          ------------------
          !                !
          !    LOAD THE    !
          ! XXDP MONITOR   !
          ------------------    See paragraph 7.1.
                !
                !
                !                         ---------------
        ---------------                   !             !
        !             !    NO             !    START    !    YES
        !   MODIFY    !-------------------->   AT 200?  !-----------------
        !   MEMORY?   !                   !             !                !
        !             !                   ---------------                !
        ---------------                          !                       !
        YES     !                         NO     !                       !
                !                                !                       !
                v                                v                       v
        ---------------                   ---------------         ---------------
        !             !                   !             !         !             !
        ! ISSUE XXDP  !                   ! ISSUE XXDP  !         ! ISSUE XXDP  !
        !   "LOAD"    !                   !   "LOAD"    !         !    "RUN"    !
        !   COMMAND   !See 7.4.2.         !   COMMAND   !See 7.4.2!   COMMAND   ! See 7.4.1
        !             !                   !             !         !             !
        ---------------                   ---------------         ---------------
                !                                !                       !
                !                                !                       !
                !                                !                       !
        ---------------                          !                       !
        !             !                          !                       !
        !  HALT CPU   !                          !                       !
        !             !                          !                       !
        ---------------                          !                       !
                !                                !                       !
                !                                !                       !
        ---------------                          !                       !
        !             !                          !                       !
        !   MODIFY    !                          !                       !
        ! MEMORY AS   !                          !                       !
        ! NEEDED      !                          !                       !
        ---------------                          !                       !
                !                                !                       !
                !                                !                       !
                v                                v                       v
```

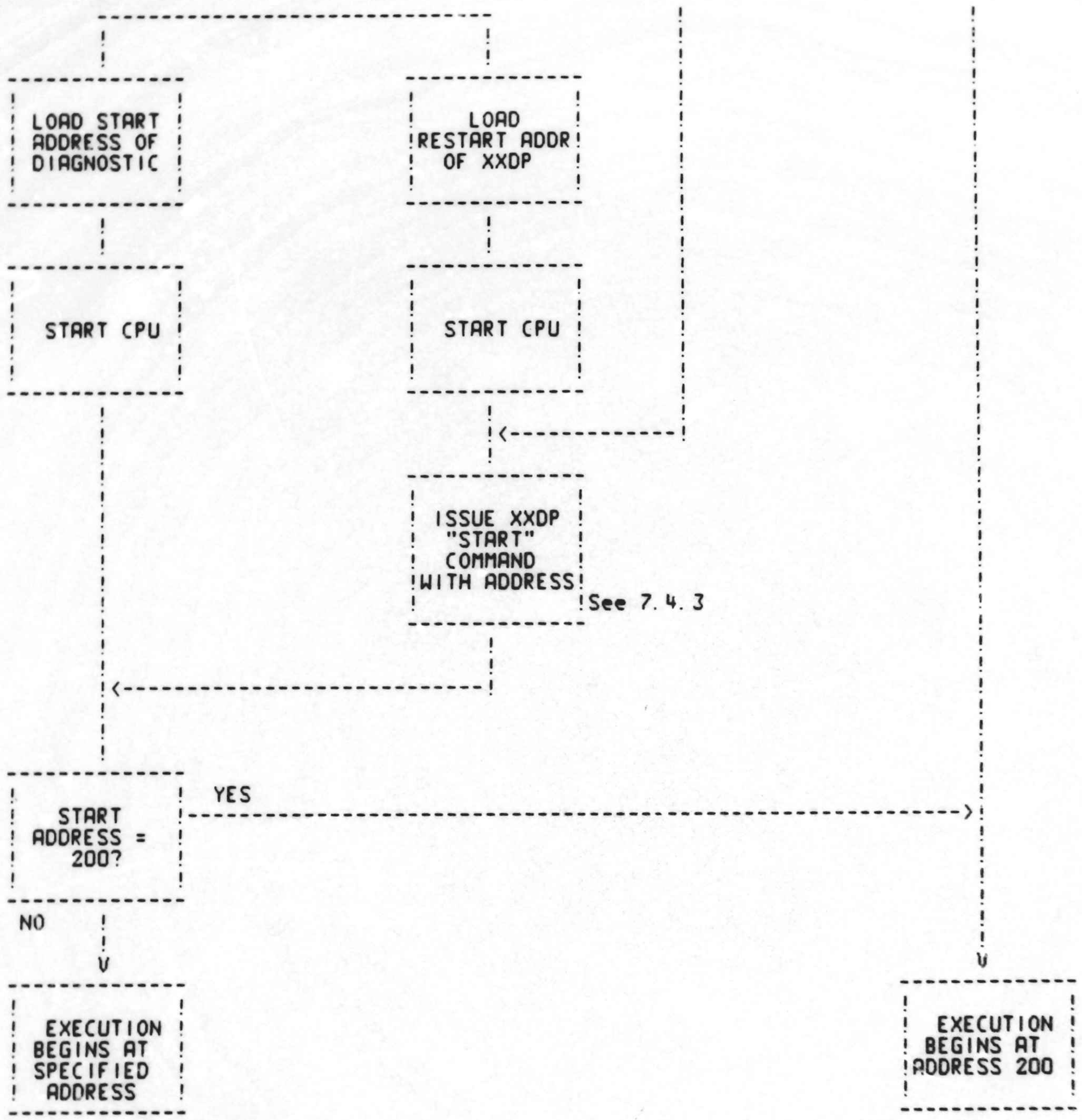
LOAD START ADDRESS OF DIAGNOSTIC
LOAD RESTART ADDR OF XXDP
START CPU
START CPU
ISSUE XXDP "START" COMMAND WITH ADDRESS
See 7.4.3
START ADDRESS = 200?
YES
NO
EXECUTION BEGINS AT SPECIFIED ADDRESS
EXECUTION BEGINS AT ADDRESS 200

FIGURE 7-1. GENERAL PROCEDURE FOR RUNNING PROGRAMS UNDER XXDP.

## 7.3 LOADING AND RUNNING PROCEDURES

There are two alternatives for loading and running programs under XXDP:

1. Load and start a program without modifying memory.

2. Load a program, halt the CPU, modify memory, and start the CPU.

Here is the general procedure, also illustrated in Figure 7-1. A more detailed example appears at paragraph 7.4.6.

### 7.3.1 RUNNING A DIAGNOSTIC WITHOUT MODIFYING MEMORY

1. Issue the XXDP run command, R (see paragraph 7.4.1).

2. Diagnostic execution begins at location 200, the standard diagnostic start address.

OR,

1. Issue the XXDP load command, L (see paragraph 7.4.2).

2. Issue the XXDP start command, S, and specify an address (see paragraph 7.4.3). If you do not specify an address, XXDP starts execution at location 200.

3. Diagnostic execution begins at specified address.

### 7.3.2 MODIFYING MEMORY AND THEN RUNNING A DIAGNOSTIC

1. Issue the XXDP load command, L (see paragraph 7.4.2).

2. Halt the CPU.

3. Modify memory as needed: Set the SW/SWR, "patch" a program, etc. Then continue following procedure "a" or "b" below.

4a. Load the address at which you want to begin execution; location 200 is the standard starting address.

5a. Start the CPU.

6a. Diagnostic execution begins at address specified in step 4a.

OR

4b. Load the XXDP restart address given in the ready message.

5b. Start the CPU.

6b. Issue the XXDP start command, S. If you do not specify an address, XXDP starts execution at location 200.

7b. Diagnostic execution begins at address specified in step 6b.

## 7.4 XXDP COMMANDS

Here are the XXDP monitor commands. They apply to all XXDP monitors. In addition, the TADP monitor requires slightly modified operation. Refer to paragraph 7.6 for this description.

NOTE

<cr> -- press "carriage return" key.

```
CNTRL C -- press CNTRL-key and C-key simultaneously;
 C         puts you at XXDP command level ( . ) at any time.

DEL     -- press key to delete erroneous
RUBOUT     keyboard entry.
```

Underlined characters in formats and examples are XXDP monitor output.

## 7.4.1 RUN Command

Here is the format for the XXDP run command, R.

```
_R filnam <cr>
```

The filnam is the filename of the program (up to six characters), without the extension. Do not type the extension. XXDP will load and run the program.

Example: run the diagnostic program CHECKIT.BIN.

1. Load the appropriate XXDP monitor as described under paragraph 7.1.

2. Set the switch register (see Chapter 6.0) as required by CHECKIT.BIN.

3. Issue the run command, R, as follows:

   ```
   _R CHECKIT <cr>
   ```

4. XXDP will run CHECKIT.BIN.

5. To run another program, halt the CPU, load the XXDP restart address, and start the CPU. XXDP will accept commands if the previous diagnostic has not destroyed it in memory. If you have lost the monitor then you must reload it, as described under paragraph 7.1.

## 7.4.2 The LOAD Command

Here is the format for the XXDP **load** command, L.

```
_L filnam <cr>
```

The **filnam** is the filename of the program, **without the extension**. Do not type the extension. XXDP will load the program into memory.

Example: load the diagnostic program CHECKIT.BIN.

1. Load the appropriate XXDP monitor as described under 7.1.
2. Issue the XXDP **load** command, L, as follows:

```
_L CHECKIT <cr>
```

3. XXDP will **load** CHECKIT.BIN from the XXDP volume into memory and await further commands.

## 7.4.3 The START Command

Before starting a **loaded** program, you may want to set the **switch register** (see Chapter 6.0); you may also want to modify other memory locations (see paragraph 7.2).

Here is the format for the XXDP **start** command, S.

```
_S <address> <cr>
```

The S command starts program execution at the address you specify. If you do not specify an address execution begins at 200. When the diagnostic has finished, and provided **it has not overlaid** the XXDP monitor in memory, you can run another diagnostic by **halting** the CPU, **loading** the XXDP **restart address**, and **starting** the CPU again. If you **have** lost the monitor, then you must reload it as described under paragraph 7.1. Remember that you lose all your modifications when you reload the monitor.

## 7.4.4 Directory Commands

The directory commands print or display directories of file-structured XXDP volumes either on the terminal device or the line printer. A directory contains the following information:

| | |
|---|---|
| filnam.ext -- | the file name (up to six characters) and extension. The only valid XXDP extensions are .BIN for a binary file, .BIC for a chainable binary file, .SAV for a core image file, and .CCC for an ASCII chain file. |
| length -- | a decimal number designating the number of blocks in a disc or tape file. |
| start -- | an octal number designating the starting block number of a disc or tape file. |
| date -- | the date of file creation. |

The directory commands have four forms:

```
     D -- gives a directory on the terminal.
   D/F -- gives a fast directory on the terminal.
          (Does not include length, start, or date).
   D/L -- gives a directory on the line printer.
 D/L/F -- gives a fast directory on the line printer.
          (Does not include length, start, or date).
```

**NOTE**

Line printer must be present; no check is made for it. The processor will trap to location 000004.

Example: give a short directory on the line printer.

```
_D/L/F <cr>

_
```

The directory will appear on the line printer.

### 7.4.5 Set Fill Count

The "set-fill-count" command, F, allows you to change the number of fill characters inserted after a carriage return operation. The LA30S terminal requires fillers to operate properly, so the filler default value is octal 14. Other terminals do not require these fillers, and by issuing an F command, you can delete the annoying, time-consuming fill operations. The F command displays the current fill count, and expects you to enter the new fill count.

Example: change the fill count to 0.

```
_F <cr>

000014 0 <cr>

_
```

### 7.4.6 Example Of Loading And Running A Program Using XXDP Commands

Run the diagnostic program CHECKIT.BIN. Before execution, take a short directory reading; put the value 17 into location 17530; and set the SW/SWR to 007100. Start execution at location 000214.

1. Load the appropriate XXDP monitor as described in paragraph 7.1.

2. Issue a "fast directory" command:

   ```
   _D/F <cr>
   ```

3. Issue the XXDP "load" command:

   ```
   _L CHECKIT <cr>
   ```

4. Halt the CPU.

5. Load address 017530.

6. Deposit 000017.

7. Load address 000176 (the address of the SW/SWR).

8. Deposit 007100.

Continue following procedure "a" or "b" below.

9a. Load address 000214.

10a. Put the HALT/ENABLE switch to ENABLE.

11a. Start the CPU.

12a. CPU begins executing CHECKIT at address 000214.

OR,

9b. Load the restart address of the XXDP monitor, given in the ready message.

10b. Put the HALT/ENABLE switch to ENABLE.

11b. Start the CPU.

12b. Issue the XXDP "start" command:

```
_S 000214 <cr>
```

13b. XXDP transfers CPU control to CHECKIT program at address 000214.

## 7.5 XXDP ERROR MESSAGES

| | |
|---|---|
| INVCMD/SW | Invalid command and/or switch. Check command just given. |
| DEVERR | Device error on input device. |
| EOM | End of medium. Occurs during input operations when the program attempts to input and the file is at an end. Serious problem. File in storage is probably wiped out. |
| INVADR | Invalid address. Must be even and within existing memory limits. Must not be within update program. |
| CKSMER | Checksum error during "load" command. |
| POFLO | Program too large to load within existing memory. |
| INVNAM | Invalid character typed for file name. |
| NEXFIL | Non-existent file. If in chain mode the program to be run does not have .BIC extension. |

## 7.6 TADP MONITOR EXCEPTIONS

The TADP package cassettes are packaged according to the following schemes:

1. One TADP cassette contains the TADP monitor and XXDP utilities (UPD1, UPD2, etc).

2. Several diagnostic cassettes containing the diagnostic programs.

When using TADP, the TADP cassette must be mounted on drive 0 (left hand drive) of the TA11; the diagnostic cassette is mounted on drive 1 (right hand drive).

Because the TADP package is a two drive system, two additional commands are provided that control the drive that is to be accessed:

```
          E 0<cr>            ;Enables access to drive 0.

          E 1<cr>            ;Enables access to drive 1.
```

When the TADP monitor is first loaded it defaults to drive 0. At that point all commands given to the monitor apply to drive 0 only.

Typing E 1<cr> enables access to drive 1 with all monitor commands applying to drive 1. To return to access drive 0 the E 0<cr> command is given.

Examples;

```
E 0<cr>              ;Enables drive 0 access.
D<cr>                ;Obtains drive 0 directory.
R UPD2               ;Runs UPD2 after loading from drive 0.

E 1<cr>              ;Enables drive 1 access.
D/F<cr>              ;Fast directory from drive 1.
L ZTCAA0<cr>         ;Loads ZTCAA0 from drive 1.
S 200<cr>            ;Starts ZTCAS0.
E 0<cr>              ;Reenables drive 0 access.
```

When the "D" (directory) command is given and drive 1 is enabled drive 0 will be accessed first in order to load the non-resident directory routine from the TADP monitor on drive 0. Then drive 1 is accessed to obtain drive 1 directory.

In chain mode (see 7.7) the chain file is always accessed from whatever drive was enabled when the "C" command was given, even if the chain file itself causes another drive to be assigned.

Example;

```
E 0<cr>                 ;Drive 0 enabled.
C CHAIN<cr>             ;Run chain from CHAIN.CCC (drive 0).

     ASSUME CHAIN.CCC CONTAINS THE FOLLOWING:

E 1<cr>
                        ;Enable drive 1.
R T1/10<cr>
                        ;Run T1 10 times.
R T2/10<cr>
                        ;Run T2 10 times.
R T3<cr>
                        ;Run T3
R T4<cr>
                        ;Run T4
R T5<cr>
                        ;Run T5
"
"
"
"
"
R T90<cr>
                        ;Run T90
E 0<cr>
                        ;Enable drive 0.
```

NOTE

In ASCII chain files, comments may not appear on command lines.

The CHAIN.CCC file will be accessed from drive 0. All the test programs will be accessed from drive 1. At completion of chain drive 0 will be enabled.

Note that with TADP, chain files do not have to be in the same cassette as the test programs.

When in doubt as to what drive is available the user just has to give the command that enables the drive he/she wishes to use.

## 7.7 CHAIN MODE OPERATION

Chain mode operation consists of the sequential execution of programs without operator intervention. Only programs that have been modified to run in chain mode can be chained. Chainable programs are identified in the directory by the extension .BIC.

To run chain mode, the XXDP monitor requires a file indicating the programs to run, and the number of times each program must execute before going on to the next program in the table. (A chain file may be generated by using the XTECO text editor, and the user must put a .CCC extension on the chain file. Refer to the XXDP USERS MANUAL for information about XTECO text editor.)

To Summarize:

1. Chain mode runs chainable programs only. (.BIC extensions).

2. A chain file indicates the programs to run and their pass counts.

3. Only programs resident on the same medium drive can be chained.

4. The chain file must be on the same medium with a .CCC extension.

NOTE: The .CCC extension indicates a chain file.

Chain mode is entered by typing:

```
C filename<cr>                  (While in monitor mode).
```

Where:
C is the "CHAIN" command
Filename is the name of the ASCII file that contains the monitor commands to be executed. The file must have a ".CCC" extension.

## 7.7.1 Making A Chain ASCII File

The chain ASCII file may be created by running the XTECO program and using the text editor to create the ASCII chain file. The chain file must contain only the commands supported under the XXDP monitor. The commands in the ASCII file are executed in the order in which they are entered and run as a batch mode.

NOTE

In ASCII chain files, comments may not appear on command lines.

Example of a chain file: run T1 1000 times; run T2 1000 times; run T3 1000 times; run T4 1000 times; run T5 1000 times; run T6 1000 times; run T7 1000 times; run T8 1000 times; run T9 1000 times; run T10 1000 times; run T11 1000 times; run T12 1000 times; load T13; start it and run 1000 times; resubmit the chain file again.

```
;CPU.CCC
;This chain file exercises the XYZ processor with T1-T13.
;
R DOAA/1000
R DOBA/1000
R DOCA/1000
R DODA/1000
R DOEA/1000
R DOFA/1000
R DOGA/1000
R DOHA/1000
R DOJA/1000
R DOKA/1000
R DOLA/1000
R DOMA/1000
L DONA
S/1000<cr>
C CPU
```

### 7.7.2 Running A Chain

To execute a chain file the user types:

```
C filnam<cr>            or            C filnam/QV<cr>
```

The pass count specified in the chain file determines the number of times XXDP will execute each program. The optional /QV switch provides a "quick verify" operation by executing each diagnostic program in the chain only once, despite the pass count specified.

The chain file to be executed must have an extensin of .CCC.

The chain file and the objective programs to be run must reside in the same XXDP medium and must be mounted on drive 0 of XXDP device.

When in chain mode switch register or software switch register should be set to 000000.

The XXDP monitor will type each command that it evaluates and then proceed to execute it.

If the monitor encounters a program that does not have a .BIC extension it types "NEXFIL". Then if the error resulted from a R (RUN COMMAND) only, it will continue with the chain file command, otherwise it terminates the chain operation.

When the last command other than another "C" command has been executed the XXDP monitor terminates chain mode and types a DOT(.), ready to accept another command from the console.

If the user wishes to terminate chain mode before its normal termination he may do so by repeatedly typing CTL C ( C) at the console until the monitor accepts it at the end of a program pass.

## 7.8 UPD1 AND UPD2

Each XXDP package contains two update programs, called UPD1.BIN and UPD2.BIN. These programs are used to add, delete, rename, and patch programs in the XXDP packages, and also to provide general file maintenance services.

This section describes three update commands. For more information on UPD1 and UPD2 consult the XXDP Users Manual.

**WARNING**

Do not use the XXDP UPDATE utility to load diagnostics. UPDATE does not write-protect the monitor volume: You may inadvertantly destroy the monitor. Instead use the XXDP load command, L, or the run command, R.

You run UPD1 and UPD2 by issuing the XXDP run command, R, as follows:

```
    _R UPD1 <cr>        or        _R UPD2 <cr>
```

Update responds and asks for the date, formatted dd-mmm-yy. Then it gives it relocation and restart addresses and issues a ready signal, *. Update will now accept commands.

**NOTE**

Running an update program destroys the XXDP monitor in memory If you need the monitor, you must reload it. Use the BOOT command for this. (See 7.8.3 below).

## 7.8.1 The Load Command

The Load command loads files from a device to memory. The format of the load command follows:

```
        *LOAD  ddn:filnam.ext     <cr>

          dd -- the load device name from Table 7-1.
           n -- the unit or drive number for that device;
                does not apply to paper tape devices.
 filnam.ext -- the name of the file on the device; omit if
                the load medium is paper tape.
```

The load command gives the starting address of the program loaded and the memory limits of the program.

Example: Load a program from paper tape, using the highspeed paper tape reader.

```
    *LOAD PR:         <cr>
     XFRADR:  000050       CORE:  000000, 017670
    *
```

Example: load a program from floppy disk drive 1 called MYFILE.BIN.

```
    %LOAD DX1:MYFILE.BIN      <cr>
     %ERADR:  000200      CORE:  000000, 012522
    %
```

## 7.8.2 The Dump Command

The dump command puts the contents of memory onto a specified device. The format of the dump command follows:

```
         %DUMP    ddn:filnam.ext     <cr>
           dd -- the name of the device on which you want to
                 dump memory; see Table 7-1.

  filnam.ext -- the name you want to give the file; omit
                if you are dumping to paper tape.
```

Example: dump a file from memory to RK05 disk drive 0; call the file MYFILE.BIN.

```
         %DUMP DK0:MYFILE.BIN      <cr>
```

Example: dump a file from memory to paper tape, using the high-speed paper tape punch.

```
         %DUMP PP      <cr>
```

## 7.8.3 The Boot Command

The boot command allows you to return CPU control to the XXDP monitor. The command format follows:

```
           %BOOT ddn:    <cr>

    dd -- the name of the device from which you are bootstrapping
          XXDP; see Table 7-1.

     n -- the unit or drive number of the device.
```

Example: bootstrap the TADP monitor from cassette drive 0.

```
            %BOOT CT0:   <cr>
```

Bootstrap RKDP monitor from disk drive 0.

```
            %BOOT DK0:    <cr>
```

Table 7-1. PERIPHERALS SUPPORTED BY UPDATE PROGRAMS

| DEVICE CODE | DEVICE NAME | REMARKS |
|---|---|---|
| PR: | PC11 HIGH SPEED PAPER TAPE READER | (UPD1, UPD2) |
| PP: | PC11 HIGH SPEED PAPER TAPE PUNCH | (UPD1, UPD2) |
| KB: | TTY KEYBOARD, OR LOW SPEED READER | (UPD1, UPD2) |
| PT: | TTY PRINTER AND PUNCH | (UPD1, UPD2) |
| DTN: | TC11 DECTAPE | (UPD1 N=0 OR 1), (UPD2, N=0-3) |
| DKN: | RK11/RK05 DISK | (UPD1 N=0 OR 1), (UPD2, N=0-3) |
| MTN: | TM11/TU10 MAGTAPE 7/9 TRACK | (UPD2, ONLY, N=0-3) |
| CTN: | TA11 CASSETTE | (UPD1, N=0 OR 1), (UPD2, N=0 OR 1). |
| DXN: | RX11/RX01 FLOPPY DISK | (UPD1 N=0 OR 1) (UPD2, N=0 OR 1) |
| MMN: | TM02/TU16 MAGTAPE | (UPD2 ONLY, N=0-3) |
| DPN: | RP11C/RP02/RP03 | (UPD2 ONLY, N=0 OR 1) |
| DBN: | RH11, RH70/RP04, RP05, RP06 DISK | (UPD2 ONLY, N=0 OR 1) |
| DSN: | RH11, RH70/RS03, RS04 DISK | (UPD2 ONLY, N=0 OR 1) |
| RMN: | RK611/RK06 DISK | (UPD2 ONLY, N=0-3) |

# CHAPTER 8

# DIAGNOSTIC PROGRAM ERROR REPORTING

In most cases an error will be reported via a message output on the console (TTY, etc).

The standard format for an error message is:

```
MESSAGE

TEST        P.C      H/W (OPTIONAL)
```

Where:

MESSAGE is a description of the type of error.

P.C. is the address in the diagnostic where the failure was detected.

H/W is the status of the hardware under test when the failure was detected. This is optional.

If an output console is unavailable, the error information will be stored in a location in memory. This address may be examined after the processor is halted.

Each appendix should be consulted for a description of the following:

1. Error messages.

2. Error types.

3. Information contained in error message.

4. Location of error message.

5. Error numbers.

For each diagnostic, the listings and cross references should be consulted for the address of the error information.

# CHAPTER 9

# DEVICE ADDRESSES

All options have standard device addresses, interrupt vector addresses, and interrupt priority (BR) levels. Many options can e configured, by means of jumpers, to have addresses and BR levels that are different from these standards. However, most diagnostics are written assuming the standara addresses and BR levels.

For example, the LP11 line printer has the following addresses and BR level:

| | | |
|---|---|---|
| PRINTER STATUS REGISTER | - | 777514 |
| PRINTER DATA BUFFER REGISTER | - | 777516 |
| INTERRUPT VECTOR ADDRESS | - | 200 |
| PRIORITY LEVEL | - | BR4 |

Usually the diagnostic will reference these values by referencing a label, i.e.:

```
     LPS     :  177514  ;  STATUS

     LPB     A  177516  ;  BUFFER

     INTVEC  :     200  ;  VECTOR ADDRESS

     PRLUL   :       4  ;  PRIORITY
```

Should the device be configured for values other than these, the llocations indicated by the labels must be changed to the proper values. The location of these labels will be found in the cross reference table that accompanies the diagnostic's listing.

NOTE

These locations must be changed after the diagnostic has been loaded into memory. Failure to change these locations will result in a trap to location 000004 when the diagnostic executes and references the registers.

# CHAPTER 10

# AUTOMATED PRODUCT TEST SYSTEM (APT)

The automated product test system, APT, loads and monitors one or more PDP-11 diagnostics into a PDP-11 computer known as the unit under test, or UUT. The mechanism for loading the diagnostic into the UUT memory is a specialized I/O controller designed by test engineering.

A diagnostic executing under APT runs pass after pass until halted, either externally by the operator, or by APT when running a script file. On an error condition, the diagnostic may either loop or halt; it is the diagnostic's job to indicate the error and then take the appropriate path of execution.

APT must communicate with the UUT diagnostic to ensure that the diagnostic is executing correctly. APT uses a polling mechanism to do this: APT periodically reads an eight-word block of memory called the mailbox. Based on the values it finds there, APT performs particular services, such as verifying the proper execution of a diagnostic, or ensuring that no error conditions exist. When it completes these services, APT sends to the mailbox, indicating to the diagnostic that the requested services are complete.

## 10.1 LOG IN PROCEDURE

1. Log into APT on an APT system control terminal by typing:

   ```
   HELLO <cr>
   ```

2. APT gives a message and prompts you for an account number with a #.

   a. Enter account 1,10 (or 1/10) if you want to create or edit a program or a script file.

**OR,**

b. Enter account 3,0 (or 3/0) if you want to load and execute a program or a script file.

Complete the account number prompt with a carriage return, <cr>.

3. APT asks for a **password**, to which you respond: APT.

```
        # PASSWORD:  <cr>
```

The APT password is **not visible** at the terminal.

4. APT issues messages describing current system usage and any new operating features. You can cancel these messages by typing CNTRL-C ( C). Then APT issues a ready message:

```
            READY
```

Now you can proceed to the pararaph that discusses the operations you want to perform under APT: paragraph 10.2 for creating program and script files; or paragraph 10.3 for loading and running program and script files.

5. You log off the APT system with the following command:

```
        BYE/F  <cr>          (for "fast good-bye").
```

## 10.2 CREATING AND EDITING FILES UNDER TSP

The test software package utility, TSP, allows you to create and edit files for use by the APT system. If you have logged into APT under account 1,10, then you need only give the following command after the APT ready signal:

```
        RUN TSP  <cr>
```

APT will transfer control to TSP, or else give the message "TSP IN USE". Note that TSP is available to only one user at a time; you may have to wait to use it.

**NOTE**

If you have logged into APT under account 3,0 and you want to run TSP, then specify TST for the AREA? prompt in the AINT program dialogue. This action logs you out of account 3,0. Then you must log in again under account 1,10 and issue RUN TSP when you get a ready message. See paragraph 10.3.

When you have succeeded in calling TSP, you will receive this message:

```
        TEST SOFTWARE PACKAGE/      TSP      REV:

           COMMAND <HE>:
```

The characters following the word COMMAND indicate the current TSP command in use, in this case, the HELP command, abbreviated HE.

NOTE

Throughout the operation of TSP, any characters contained within < >'s are the default value. If you issue a carriage return without giving a value, then TSP uses the string contained by the < >.

All TSP commands use interactive dialogue to obtain information. For a detailed explanation of the command dialogues, consult the APT General Specification and the APT System Manager's Guide.

## 10.2.1 TSP Command, CREATE

The CREATE command lets you build an APT file. This may be a program or a script or a master script.

NOTE

In these examples, do not confuse <cr> (which means carriage return) with <CR> (which is TSP's abbreviated default notation for the CREATE command).

Type in CREATE at TSP command level after the colon.

```
           COMMAND <HE>:  CREATE <cr>
```

This will begin a dialogue that asks you for information concerning the file you are building. See paragraph 10.2.3 for a sample of this dialogue.

### 10.2.2 Returning Control To APT

When you want to leave TSP, issue the OFF command.

```
                COMMAND (xx):   OFF <cr>
```

This puts you back to APT ready level. You can log-off using BYE/F.

### 10.2.3 Sample TSP Dialogue

Here is an example in which a user logs into APT, calls TSP, builds a file called MEZZ, then leaves TSP and logs off APT. All user- supplied information in this example is underlined. This differs from the usual format used throughout this manual. Text contained in *'s is explanatory, and not part of the control terminal printout.

```
HELLO <cr>          *APT log in command*

RSTS V06A-02 APT-D-RB-03NOV  JOB 34  KB3  05-JAN-78  09:23
#1/10 <cr>          *account number prompt*
PASSWORD:      <cr> *enter password APT;  will not appear
                     at terminal*
JOB(S) 12 ARE DETACHED UNDER THIS ACCOUNT
JOB NUMBER TO ATTACH TO?  <cr> *do not specify a job
                                number*
4 OTHER USER(S) ARE LOGGED IN UNDER THIS ACCOUNT



READY

RUN TSP <cr> *first attempt to run TSP utility*
TSP IN USE       *TSP not available*

READY

RUN TSP <cr> *try again*

                 *this try successful*

TEST SOFTWARE PACKAGE/ TSP       REV: 07
```

```
        COMMAND <HE>:     <cr> *issuing <cr> gives default command,
                                HELP*

        COMMANDS ARE:           *HELP command output*

        CREATE
        EDIT
        LIST
        RENAME
        LOCATE
        DELETE
        UPDATE
        PATCH
        COPY
        HELP
        OFF

        COMMAND <HE>:  CREATE <cr>  *issue TSP command CREATE*
        COMPUTER TYPE (8,10,11,VS)  <11>: 11 <cr>  *since 11 was
                                                    default value,
        SOFTWARE TYPE (P,S,M) <P>: <cr>             user could
                                                    have issued
                                                    just <cr>*
        APT,ACT,PRE-ACT,SPVR-DIAG PROGRAM <APT>: PRE-ACT <cr>
        LOAD DEVICE (PR,RK,MT,DT,MM,DM,MC) <PR>:  <cr>
        PR UNIT NUMBER (0-7) <0>: <cr>
        IS PATCHING REQUIRED (Y/N)  <N>:  <cr>
        PROGRAM NAME:  MEZZ <cr>

        LOAD AND READY THE PAPER TAPE READER.
        TYPE <CR> WHEN READY.  <cr>  *both <CR> and <cr> here mean
                                      carriage return*
        WORKING....

        REVISION < >: A <cr>
        MCO < >: 0   <cr>
        TITLE < >: MEZZ  <cr>

        COMMAND <CR>: OFF <cr>  *leave TSP; note default
                                 command is now CREATE
                                 (abbreviated CR), the
                                 last command used*

        READY

        BYE/F  <cr>    *log off APT*
```

## 10.3 LOADING AND RUNNING PROGRAMS UNDER APT

You must log into APT under account 3,0 (or 3/0) in order to load and run program files and script files under APT. See paragraph 10.1.

When you receive the APT ready message, issue the following command to run the APT initialization utility, AINT.

```
RUN AINT <cr>
```

This will start a dialogue. After each response you give during the dialogue, press the carriage return key to enter the information. Use the following values:

```
BADGE NO?  = 1
    AREA?  = MLE  (or TST; see NOTE and
                   paragraph 10.3.1).
    UNIT?  = <cr>
```

An example of this dialogue appears in paragraph 10.3.2.

NOTE

If you are on line under account 3,0 and you want to run TSP (which requires account 1,10), then specify TST for the AREA? prompt. Then log in under account 1,10 as described in paragraph 10.1.

### 10.3.1 Returning Control To APT

Issue the OFF command in response to a dialogue prompt. This will put APT back on line. The only way to log off from here is to specify TST for the AREA? prompt; then give the BYE/F command after the ready signal.

### 10.3.2 Sample MLE Dialogue

In the following example, a user logs into APT account 3,0, unsuccessfully runs two program files (MEZZ, created in paragraph 10.2.3, and ACM15, created another time) and a script file (RONU), and then logs off. All user-supplied information in this example is underlined. This differs from other example formats used in this manual. Text contained in *'s is explanatory and not part of the control terminal printout.

```
HELLO 320 <cr>  *quick log in:  give account number with
                       HELLO*
PASSWORD:  <cr>  *password APT does not appear at terminal*
18 OTHER USER(S) ARE LOGGED IN UNDER THIS ACCOUNT

READY

RUN ATNI <cr> *APT initialization necessary to run programs*
APT ON LINE

05-JAN-78   09:54

BADGE NO?  1 <cr>

AREA?  MLE <cr>

UNIT?  <cr>
APT READY
UUT LINE #?  1397 <cr> *this number obtained from listings in
                          APT machine room*
COMPUTER?  1105 <cr> *computer type you are testing*
INTERFACE?  G50888 <cr> *varies according to interface you are
                          using*
PROGRAM NAME:   MEZZ <cr>
RCVD        09:55
HUNG-LOAD       UUT LINE#:   1397       TIME:  10:13     DATE:
05-JAN-78
SCRIPT:         PROGRAM:  MEZZ    ERROR:  00000 00000  ET.:
0:00:00
PROGRAM NAME:   RONU <cr> *this is a script file*
RCVD        09:58
UUT LINE #?  1367 <cr> *APT will accept only legitimate UUT
                          line #*
UUT LINE #?
MESSAGE(S) ARE PENDING FOR LINE(S)
1397
RESULTS NOW (N/ALL/LINE #)?  1397 <cr>
HUNG LOAD       UUT LINE #:  1397     TIME:  09:59   DATE 05-JAN-78
SCRIPT: RONU    PROGRAM: T01    ERROR:  00000 00000  ET.:  0:00:00
DELETE MESSAGE(S) (Y/N)?   Y <cr>
UUT LINE #? OFF    <cr>   *put APT back on line*
APT ON LINE

05-JAN-78     10:02

BADGE NO? 1  <cr>

AREA? TST <cr> *you must specify TST in order to log off*
THIS TERMINAL CAN BE USED FOR TIMESHARING

BYE/F   *this BYE/F issued by APT*

READY

BYE/F <cr>
```

## 10.4 APT ERROR MESSAGES

Most error messages printed on control terminals related to program loading problems. A few of these error messages are described below.

1. Request lost - this indicates that the last user's program load request hasn't been processed. It tells the operator to try again to load the program into the UUT.

2. Hung load - this indicates that there is some hardware malfunction between the interface and the UUT to be loaded.

3. Hung diagnostic - this indicates that when APT monitored the diagnostic it was not running correctly in the UUT.

NOTE

This error can be caused if the system manager specified "first pass run time", "maximum pass run time" or " longest test time" values that are too short.

4. Diagnostic error - this indicates that when the diagnostic was running in the UUT it detected a hardware error in the equipment being tested.

5. Line error - this indicates that a parity error was detected on the line connecting APT to the UUT.

NOTE

The line testing utility (LNTSTU) should be run on lines that frequently generate line errors.

# CHAPTER 11

# AUTOMATED COMPUTER TEST SYSTEM, ACT

The automated computer test system, ACT, provides three basic services that aid DEC's manufacturing areas in testing PDP-11 computers.

1. Load and run a diagnostic into a unit under test, UUT, as if it were loaded and run manually; called ACT "dump" mode.

2. Automatically load, run, and monitor a single diagnostic or sequence of diagnostics through one or more iterations; called ACT "auto-accept" mode; includes "quick verify" mode.

3. Directly perform a variety of UUT memory tests; called ACT "station test" mode.

An ACT system comprises a central computer and from one to thirty-two test stations. The central computer, also called "mother", consists of a PDP-11 CPU, one to eight RK disk drives, a high-speed paper tape reader, and a console terminal. (Additional peripherals supported by RKDP monitor are available). Each test station, also called a "daughter", consists of a station console, the UUT, and the UUT's console terminal. You instruct "mother" by setting switches on the "daughters'" consoles. (Refer to the ACT Users' Guide for the exact settings of these switches when using ACT). The UNIBUS of the UUT is connected to the "daughter" station, and all control of the UUT is through the UNIBUS.

## 11.1 LOADING AND STARTING THE ACT-11 MONITOR, OR "MOTHER"

1. ACT-11 resides only on Manufacturing RKDP disk volume #1. Mount this volume on RK drive 0.

2. Bootstrap the RK drive as described in Chapter 4 of this guide.

3. RKDP identifies itself and gives a ready signal, the period (.).

4. Call ACT-11 by issuing the following command:

```
_R ACT<cr>
```

5. ACT-11 starts a dialogue to get information (restart? suppress messages? time-of-day?) and then issues a ready signal,

```
READY
```

6. The ACT-11 monitor, or "mother", can now service the "daughter" stations.

## 11.2 ACT-11 "DUMP" MODE

Operating ACT-11 in "dump" mode allows you to load and execute a single diagnostic into the UUT as if under manual control.

1. Ready the ACT-11 monitor. See paragraph 11.1.

2. Ensure that the daughter station is powered-up and on-line (the "on-line" switch is up).

3. Determine the number of the diagnostic you are running; this will be either an ACT number or an RKDP number.

4. Set the daughter station switch panel according to the diagnostic number and the load device. Refer to the ACI Users' Guide for the exact switch settings.

5. Set the daughter station control switches. Refer again to the ACI Users' Guide for the control switch settings.

6. Activate the "initialize" switch on the daughter station console.

7. ACT will load the diagnostic into the UUT and start its execution.

## 11.3 ACT-11 "AUTO-ACCEPT" MODE

ACT "auto-accept" mode automatically loads and executes a set of diagnostic programs in the UUT. ACT reads a list called the sequence table to determine which programs to run and in what order to run them. (Refer to the ACI System Manager's Guide for instructions on building sequence tables). You can instruct ACT to complete only one run of the sequence table entries, also called a quick verify pass; or you can specify many repetitions of the table. When ACT has completed the list, it sends a message to the mother console and to the UUT console.

### 11.3.1 Running A List Of Diagnostics

1. Ready the ACT-11 monitor. See paragraph 11.1.

2. Ensure that the daughter station is powered-up and on-line ("on-line" switch is up).

3. Set the daughter station switch panel according to the UUT memory size, UUT options, and whether you want messages on the UUT terminal. Refer to the ACI Users' Guide for the exact switch panel settings.

4. Set the daughter station control switches; this includes specifying the sequence table number. See the ACI Users' Guide.

5. Activate the daughter station "initialize" switch.

6. ACT will load and execute the set of diagnostics as defined by the sequence table. ACT issues a "pass complete" message upon successful completion of all the diagnostics listed in the table.

### 11.3.2 Running A Single Diagnostic In "Quick-Verify" Mode

You can instruct ACT to load and run a single program automatically. One run of the program is called a "quick-verify" or Q/V pass.

1. Ready the ACT monitor. See paragraph 11.1.

2. Ensure that the daughter station is powered-up and on-line.

3. Determine the number of the diagnostic you are running. This will be either an ACT number or an RKDP number.

4. Set the daughter station switch panel to indicate the program number, and that you want a Q/V pass. Refer to the ACT Users' Guide for exact switch settings.

5. Set the daughter station control switches; this includes specifying the maximum run-time for the program. See the ACT Users' Guide for Q/V mode control switch settings.

6. Activate the daughter station "initialize" switch.

7. ACT will load and execute the program defined on the switch panel. ACT issues a "pass complete" message upon successful execution of the diagnostic.

## 11.4 ACT-11 "STATION TEST" MODE

When running ACT-11 in station test mode, you can perform the following tests. Consult the ACT-11 Users' Guide for the step-by-step procedures.

1. Lamp test -- verifies the operation of a daughter station's indicator lights.

2. UUT word test -- verifies the read/write accuracy of any UUT memory word.

3. UUT field test -- verifies the read/write accuracy of any 4K block of UUT memory.

4. NPR interaction test -- verifies that "non-processor-request" transfers to and from the UUT are good.

# CHAPTER 12

# DECX/11 UNIBUS EXERCISER

DECX/11 is the UNIBUS exerciser for the PDP-11 computer family. UNIBUS exercising provides a means of testing the expected reliability of a particular system within a specified period of time. Here are the basic components of a DECX/11 package:

DECX/11 monitors (standard monitor, short monitor, and 11/70 monitor).

DECX/11 option/device test modules.

DECX/11 configurator/linker program.

DECX/11 documentation.

You use the monitor, the test modules, and the configurator/linker program to create a UNIBUS exerciser. This exerciser includes the monitor and those test modules required by the system you are testing. This chapter assumes that you have already created a UNIBUS exerciser, and provides an overview only for running it. For a complete description and instructions on creating a UNIBUS exerciser, consult the DECX/11 Reference Guide.

## 12.1 LOADING THE UNIBUS EXERCISER

If you are loading a UNIBUS exerciser from paper tape, use the absolute loader. See Chapter 5 of this guide.

If you are loading a UNIBUS exerciser from other media, it must be a named file (with the extension .BIN or .BIC) on an XXDP monitor volume.

1. Load the XXDP monitor as described in Chapter 4.

2. Issue the XXDP load command, L, giving the filename of the UNIBUS exerciser. Do not type the extension when giving the filename.

   Example: load the UNIBUS exerciser DECX1.BIN.

   ```
   _L DECX1    <cr>
   ```

3. You have loaded the exerciser. Proceed to paragraph 12.2.

OR,

1a. Load the XXDP monitor as described in Chapter 4.

2a. Issue the XXDP run command, R, and the UNIBUS exerciser will self-start after loading. Using the same exerciser DECX1.BIN:

   ```
   _R DECX1    <cr>
   ```

3a. The exerciser will start at address 000200. It gives a message and a ready signal, the period (.). Proceed to paragraph 12.3.

WARNING

Do not use the XXDP UPDATE utility to load a UNIBUS exerciser. UPDATE does not write-protect the monitor volume: you may inadvertently destroy the monitor. Instead use the XXDP load command, L, or the run command, R, as described above.

## 12.2 STARTING THE UNIBUS EXERCISER

1. To start the exerciser issue the XXDP start command, S (see paragraph 7.4.3).

2. The monitor issues a message and a ready signal, the period (.). Proceed to paragraph 12.3.

OR,

1a. Halt the processor.

2a. Load address 000200.

3a. Put the HALT/ENABLE switch to ENABLE.

4a. Start the processor.

5a. The monitor issues a message and a ready signal, the period (.). Proceed to paragraph 12.3.

## 12.3 OPERATING THE UNIBUS EXERCISER

You control the exerciser by using the switch register (see also Chapter 6) and the DECX/11 commands, paragraph 12.3.2.

### 12.3.1 Switch Register Settings

The following switch register settings apply only when running a DECX/11 UNIBUS exerciser. A switch is set when it equals one.

Table 12-1 DECX/11 Switch Register Settings.

| PANEL SWITCH | SW/SWR SETTING (ALSO KEYPAD CPU SETTING) | OPTION |
|---|---|---|
| 15 | 100 000 | Drop module after error: the failing module stops executing; "MODULE DROPPED" message. |
| 14 | 040 000 | Inhibit module drop after error; if SW14=0, the failing module stops executing after 20 errors; "MODULE DROPPED" message. |
| 13 | 020 000 | Inhibit error and module messages. |
| 12 | 010 000 | Enable "END OF PASS" message. |
| 10 | 002 000 | Report all data errors; if SW10=0, report first three errors of any block transfers. |
| 9 | 001 000 | Inhibit "RELOCATED TO" message. |

NOTE

The Keypad and SW/SWR settings are accumulative: if you want more than one option, then add the SWR values and enter the sum as the SWR setting. For example, if you want to designate the second and third options on the list (panel switches 13 and 14 aa), then enter the value 060 000 as the SWR setting.

If your CPU does not have a hardware switch register (for example, the PDP-11/04), then you must make the following patch before running the exerciser module:

1. Find the address of the location labelled FAKESR in the DECX/11 cross-reference listing. FAKESR is the DECX/11 software switch register.

2. Using the DECX/11 MOD command (see Table 12-2), load the address number of FAKESR into the location labelled SR, also found in the DECX/11 cross-reference listing.

3. The DECX/11 monitor will now read location FAKESR for switch register settings. You load location FAKESR with the appropriate values from Table 12-1, depending on the options you select.

## 12.3.2 DECX/11 Monitor Commands

You enter a command by terminating it with a carriage return, <cr>. You can use the RUBOUT key to delete individual characters, and the combination CNTRL/C (or C) to delete a whole line.

Table 12-2. DECX/11 Monitor Commands

| COMMAND | DESCRIPTION |
|---|---|
| MAP | Types map of modules in the UNIBUS exerciser. Indicates module starting address, and module status. |
| SEL | Selects all modules for execution. |
| DES | Deselects all modules. |
| SEL modulename | Selects specified module. |
| DES modulename | Deselects specified module. |
| MOD addr | Types contents of address specified. Operator can enter new value if desired. |
| MOD modulename addr | Types contents of address relative to starting address of module specified. User can enter new value if desired. |
| RUN addr | Starts execution of exerciser. Only selected modules run. If optional address is used and relocation is allowed, starts program at nearest (lower) 4K boundary. |
| RUNL addr | Starts exerciser execution at specified address. Address must be on 4K boundary. (20000, 40000, etc. ). The program stays "locked" there. It does not relocate. |
| KTON | Turns on memory management (KT11). |
| KTOFF | Turns off KT11. |
| PON | Enables parity memory. |
| POFF | Disables parity memory. |
| FILL | Sets the fill count. |

Table 12-2. (Continued)

| COMMAND | DESCRIPTION |
|---|---|
| ROTON | Enables write buffer rotation and types range. |
| ROTOFF | Disables write buffer rotation. |
| CON | Turns on the cache memory (PDP-11/70 only). |
| COFF | Turns off the cache memory (PDP-11/70 only). |
| MON | Enables the MAP box (PDP-11/70 only). |
| MOFF | Disables the MAP box (PDP-11/70 only). |
| LPON | Directs all TTY output to the line printer. NOTE: Cannot be used if line printer module is selected. |

## 12.4 DECX/11 ERROR MESSAGES

KBUF OFLO -- You entered too many characters at some point in a command line.

INVALID COMMAND -- You have given an invalid command, or have entered a valid command incorrectly. (Module names *always* have five characters.)

MODULE NAME NOT FOUND -- You have specified a nonexistent module, or have entered the name of an existing one incorrectly. (Module names *always* have five characters).

INVALID ADDR/DATA -- You have specified either an odd-numbered address, or a nonoctal address.

# CHAPTER 13

# OUTLINE FOR DIAGNOSTIC OPTION APPENDICES

This chapter describes the type of information contained in the individual appendices for diagnostic options.

Each option appendix is a procedural, step-by-step method of setting up an option to be tested, loading the diagnostic, and executing the diagnostic.

Wherever possible, the appendix will reference the standard procedures described in this User's Guide. If the field service representative is unfamiliar with the standards, he/she should consult the appropriate sections of the guide.

All appendices will contain the following sections:

1. ABSTRACT--This section will state the options to be tested by the diagnostics.

2. HARDWARE--This section will give the minimum hardware requirements necessary to run the diagnostics:

   . Central Processor Options

   . Memory

   . I/O Devices

   . Interfaces

   . Controllers

   . Units

   . Accessories

3. TEST SEQUENCE--This section will list the diagnostics available for the option and the order in which they must be run. Running the diagnostics in the given order will isolate the hardware faults in the shortest possible time.

4. TEST DESCRIPTION--This section wil' describe what each diagnostic tests. The description will be by function and/or hardware as applicable.

5. CAUTIONS--This section will describe all procedures necessary to prevent accidental erasure of system files and diagnostics.

6. OPERATING PROCEDURE--This section will describe the detailed, step-by-step procedure for running the diagnostics. There will be a separate operating procedure for each diagnostic.

NOTE

These procedures assume the field service representative is familiar with the following:

1. The hardware to be tested.

2. The individual PDP-11 Processor.

3. The Users' Guide.

The operating procedure will be structured as follows:

. TITLE--The five-letter code for the diagnostic.

. SYSTEM INITIALIZATION--All procedures for the entire system that must be performed prior to running the diagnostic.

. LOADING PROCEDURE--Where applicable, reference will be made to the standard practices described in the Users' Guide. If the diagnostic has its own unique method of loading, it will be described here.

. STARTING PROCEDURE--Where applicable, reference will be made to the standard practices described in the Users' Guide. If the diagnostic has its own unique starting procedure, it will be described here.

. SWITCHES--The switches used by the diagnostic will be described.

. INITIAL PRINTOUT--This will be the initial message the diagnostic outputs on the available i/O console. Output of this message indicates the successful loading and starting of the diagnostic.

- OPERATOR INTERROGATION--This section will describe the messages output by the diagnostic and the proper operator response.

- EXECUTION TIME--This is how long it takes the diagnostic to complete one pass successfully.

- END OF PASS INDICATOR--This will describe the method the diagnostic uses to signal a successful pass.

- ERROR REPORTS--This will describe types of messages the diagnostic will output on the available I/O Console when an error is detected. The information contained in the error message will also be described. Whenever possible, reference will be made to the standard practices described in the Users' Guide.

- RESTART ADDRESSES--If the diagnostic has restart addresses they will be listed here. The listings for the individual diagnostics must be consulted to determine the usefulness of a given restart address. If the diagnostic has no restart address, the expression "Starting Address" will appear.

NOTE

PART II, THE SET OF APPENDICES DESCRIBING THE DIAGNOSTIC OPTIONS, IS NOT INCLUDED IN THIS DOCUMENT. THE APPENDICES ARE CURRENTLY IN PRODUCTION AND WILL BE RELEASED TO THE FIELD AS THEY ARE COMPLETED.

Send all suggestions concerning this document to:

BARRY SUSSMAN

DIAGNOSTIC ENGINEERING

ML21-4/E10

DIGITAL EQUIPMENT CORPORATION

MAYNARD, MASS.

(617) - 493-4875

DTN : 223-4875

You should submit all software trouble reports via the AIDS problem reporting system.