pdp11

M9301
bootstrap/terminator module maintenance and operator's manual

digital

EK-M9301-TM-001

M9301
bootstrap/terminator module maintenance and operator's manual

digital equipment corporation • maynard, massachusetts

1st Edition, June 1977

Copyright © 1977 by Digital Equipment Corporation

The material in this manual is for informational purposes and is subject to change without notice.

Digital Equipment Corporation assumes no responsibility for any errors which may appear in this manual.

Printed in U.S.A.

This document was set on DIGITAL's DECset-8000 computerized typesetting system.

The following are trademarks of Digital Equipment Corporation, Maynard, Massachusetts:

DEC	DECtape	PDP
DECCOMM	DECUS	RSTS
DECsystem-10	DIGITAL	TYPESET-8
DECSYSTEM-20	MASSBUS	TYPESET-11
		UNIBUS

CONTENTS

CONTENTS (CONT)

CONTENTS (CONT)

CONTENTS (CONT)

CONTENTS (CONT)

FIGURES

FIGURES (CONT)

TABLES

PREFACE

This manual describes the M9301 Bootstrap/Terminator module and its various versions. Complete understanding of the manual requires that the user have a general knowledge of digital circuitry and a basic understanding of PDP-11 computers. The following related documents may be valuable as references.

PDP-11 Peripherals Handbook
PDP-11 Processor Handbooks
PDP-11/34, 11/04 User's Manual
PDP-11/70 Systems Manual
Drawing Directory B-DD M9301-0 (for ROM listings)

CHAPTER 1
INTRODUCTION

1.1 GENERAL DESCRIPTION

The M9301 contains a complete set of Unibus termination resistors along with 512 words of read-only memory which can be used for bootstrap programs, diagnostic routines, and the console emulator routine. The module also provides circuitry for initiating bootstrap programs either on power-ups or from an external or logic level switch closure. See Figure 1-1 for a photo of the module.

1.2 FEATURES

- Available features depend on the variation used.
- The M9301 combines Unibus termination and bootstrap capability on one double height module.
- Can be used in PDP-11 machines which can handle an extended length module in the terminator slots. (The M9301-YC can be used only in a PDP-11/70 system.)
- Bootstrap programs may be initiated by the following means:

  1. Direct program jumps to the memory space occupied by the bootstrap.
  2. Programmer's console load address and start sequence.
  3. Power restarts (Paragraph 2.3).
  4. External boot switch closure.

- Provides capability of enabling or disabling boots on power restarts.
- Provides 512 words of user memory space which can be programmed by the user or purchased with standard patterns provided by DEC.

The first two chapters of this manual describe features of the M9301 which apply to all versions. Chapters 3, 4, and 5 do not pertain to the -YC and -YH versions.

The remaining chapters deal with the programming information and specific functions for each version. One chapter is devoted entirely to each version, so that the reader need not search through irrelevant information to find what he needs.

1.3 PHYSICAL DESCRIPTION

The M9301 is a double height extended, 21.6 × 14 cm (8-1/2 × 5-1/2 in) Flip-Chip module which plugs into the A and B terminator slots on the PDP-11 backplane.

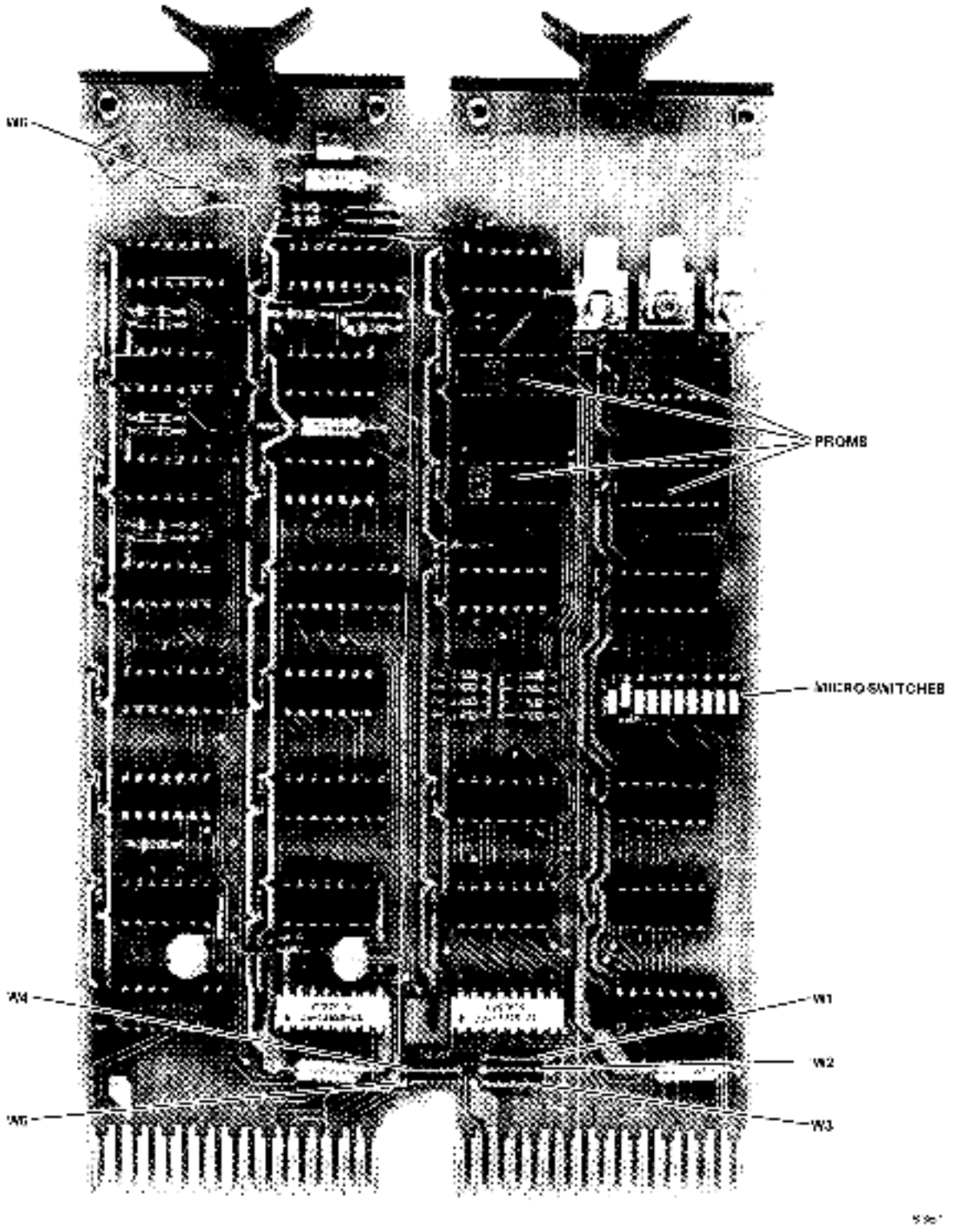

Figure 1-1 Bootstrap/Terminator Module

1.4 ELECTRICAL SPECIFICATIONS

Power Consumption +5 Vdc - 2.0 A typical

Electrical Interfaces The Unibus interface is standard using 8837 and 8640 receivers and 8881 drivers.

1.4.1 External Electrical Interfaces

The external interface consists of three Faston tabs (TP1, TP2, and TP3) provided at the handle end of the module. The following loading and usage constraints apply.

TP1 Represents one standard TTL load with a 1K ohm pull-up. Input should be stable during a power-up. Refer to Paragraph 2.4 for TP1 usage.

TP2 Represents two standard TTL loads with a 1 kilohm pull-up resistor. Input signals should be limited to a 100 ns minimum pulse width with all switch bounce noise restricted to a 5 ms maximum duration. Note that triggering is initiated upon release of an input low (logic 0) pulse. On all power-ups, triggering is disabled until approximately 100 ms after power returns (Paragraph 2.7), assuming that +5 Vdc will be available from the power supply within 20 ms. It is also important to realize that this input has no high voltage protection capability and adequate filtering must be provided when locating this input outside the standard DEC computer enclosure. Refer to Paragraph 2.5 for TP2 usage.

TP3 Should be used as a ground return for external switches attached to TP1 and TP2. Note that there is no protection for large voltage spikes on this input so proper filters should be externally installed to guarantee adequate isolation.

1.4.2 Electrical Prerequisites

Refer to Chapter 6 and following for system constraints on the specific version being used.

Power and Ground Pinouts (Refer to Table 1-1 for pin assignments.)
+5 Vdc: Pins AA2, BA2
GND: Pins AC2, AT1, BC2, BT1

1.4.3 Timing

Figure 1-2 shows important timing constraints for the M9301. Values shown are typical.

1.4.4 Operating Environmental Specifications

Temperature Range 5° C to 50° C

Relative Humidity 20% to 95% (without condensation)

1.5 INSTALLATION

As a universal bootstrap/terminator module, the M9301 in its various versions, can be adapted by the user to meet a variety of boot requirements and system configurations. Major factors that must be considered during installation concern the address offset microswitches and the external boot switch. Figure 1-3 shows the relationship between the switches and the lower nine bus address bits.

Table 1-1 M9301 Modified Unibus Pin Assignments

Pin	Signal	Pin	Signal
AA1	BUS INIT L	BA1	SPARE
AA2	POWER (+5 V)	BA2	POWER (+5 V)
AB1	BUS INTR L	BB1	SPARE
AB2	TEST POINT	BB2	TEST POINT
AC1	BUS D00 L	BC1	BUS BR 5 L
AC2	GROUND	BC2	GROUND
AD1	BUS D02 L	BD1	BAT BACKUP +5 V
AD2	BUS D01 L	BD2	BR 4 L
AE1	BUS D04 L	BE1	INT SSYN
AE2	BUS D03 L	BE2	PAR DET
AF1	BUS D06 L	BF1	BUS ACLO L
AF2	BUS D05 L	BF2	BUS DCLO L
AH1	BUS D08 L	BH1	BUS A01 L
AH2	BUS D07 L	BH2	BUS A00 L
AJ1	BUS D10 L	BJ1	BUS A03 L
AJ2	BUS D09 L	BJ2	BUS A02 L
AK1	BUS D12 L	BK1	BUS A05 L
AK2	BUS D11 L	BK2	BUS A04 L
AL1	BUS D14 L	BL1	BUS A07 L
AL2	BUS D13 L	BL2	BUS A06 L
AM1	BUS PA L	BM1	BUS A09 L
AM2	BUS D15 L	BM2	BUS A08 L
AN1	PI	BN1	BUS A11 L
AN2	BUS PB L	BN2	BUS A10 L
AP1	PO	BP1	BUS A13 L
AP2	BUS BBSY L	BP2	BUS A12 L
AR1	BAT BACKUP +15 V	BR1	BUS A15 L
AR2	BUS SACK L	BR2	BUS A14 L
AS1	BAT BACKUP -15 V	BS1	BUS A17 L
AS2	BUS NPR L	BS2	BUS A16 L
AT1	GROUND	BT1	GROUND
AT2	BUS BR 7 L	BT2	BUS C1 L
AU1	+20 V	BU1	BUS SSYN L
AU2	BUS BR 6L	BU2	BUS CO L
AV1	+20 V	BV1	BUS MSYN L
AV2	+20 V	BV2	-5 V

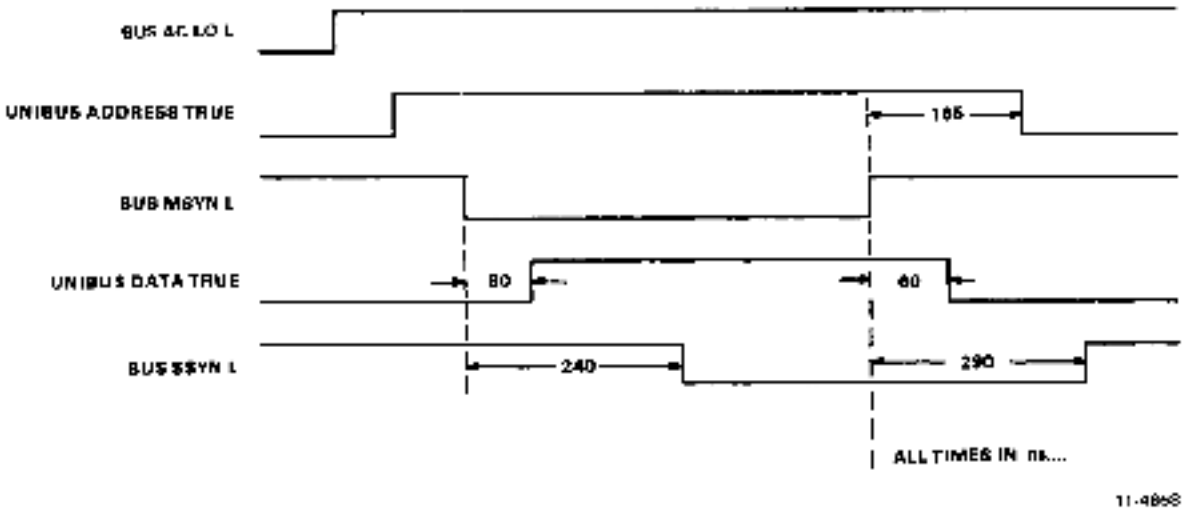

Figure 1-2 M9301 Timing

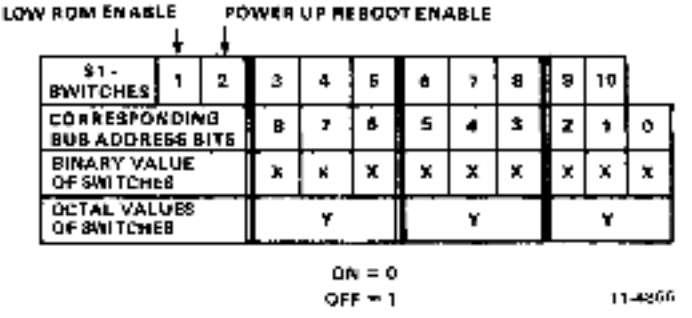

Figure 1-3 Offset Switches and Corresponding Bus Address Bits

1.5.1 Power-Up Reboot Enable

Automatic booting on all power-ups can be enabled or disabled using the POWER UP REBOOT ENAB switch (S1-2). If this dip switch is set to the OFF position, the processor will power up normally, obtaining a new PC from memory location 24_8 and a new PSW from location 26_8. When the switch is in the ON position during a power-up, the processor will obtain its new PC and PSW from locations 173024 and 173026 respectively. The address of BOOT SELECT switches S1-3 through S1-10 is 173024.

The function performed by the REBOOT ENABLE switch (S1-2) can be controlled by an external switch using the Faston tabs on the M9301 module. This external switch closure should be made to ground (enabling boot) using Faston tab TP3 as a ground return. In the PDP-11/04 and 34, when MOS memory is present with battery backup, a battery status indication signal is generated by the power supply. This signal should be attached to the POWER UP REBOOT ENAB input (TP1) on the M9301. If this status signal goes low, it indicates that the contents of the MOS memory are no longer valid and must be reloaded, usually from some mass storage device. The M9301, sensing the status of the memory, forces a boot on power-up, allowing new data to be written into memory. Note that grounding TP1 will override S1-2 if the switch is off.

If the battery status input is high (logic 1), the M9301 will not automatically boot on power-up, and execution will begin at the address specified by location 24.

1.5.2 Low ROM Enable

This dip switch (S1-1), when set to the OFF position, prevents the M9301 from responding to the Unibus address range 765000 through 765777. See address maps for the specific versions to see how each version is affected by S1-1.

1.5.3 Boot Switch Selection

In order to select bootstraps to be run on power-ups and external boot enables, eight dip switches (S1-3 through S1-10) are provided on the M9301. Tables in Chapter 7 and following chapters show switch configurations for the various options supplied by each version.

1.5.4 External Boot Switch

A device can be externally booted using the EXTERNAL BOOT input on Faston tab TP2. If this input is brought to ground, BUS AC LO L will be activated causing the processor to perform a POWER FAIL. Upon returning to a logic 1, this input will deactivate BUS AC LO L initiating a power-up sequence in the CPU. Faston tab TP3 should be used as a ground return for the external boot switch. Note that the power-down trap vectors to locations 24_8 and 26_8.

CHAPTER 2
HARDWARE DESCRIPTION

2.1 GENERAL

All versions of the M9301 are physically and electrically identical with the exception of the ROMs that are used; and the hardware description which follows applies to each of them. Various portions of the circuitry will be analyzed separately for clarity. M9301 circuit schematics (CS M9301-0-1) will be referenced throughout the description.

2.2 DEFINITION OF TERMS

Bootstrap Program

A bootstrap program is any program which loads another (usually larger) program into computer memory from a peripheral device.

Bootstrap

Bootstrap and bootstrap program are used interchangeably.

Boot

Boot is a verb which means to initiate execution of a bootstrap program.

NOTE

Some processors have a feature which enables them to select power fail vectors. A vector of 24_8 will be used in this discussion for the sake of simplicity.

2.3 POWER-UP SEQUENCE

Typically all PDP-11 computers perform a power-up sequence each time power is applied to their CPU module(s). This sequence is as follows:

1. +5 Vdc comes true
2. BUS DC LO L unasserted by power supply
3. BUS INIT asserted
4. BUS AC LO L released by power supply
5. Processor accesses memory location 24_8 for new PC
6. Processor accesses memory location 26_8 for new PSW
7. Processor begins running program at new PC contents.

With an M9301 Bootstrap/Terminator in the PDP-11 computer system, on power-ups the user can optionally force the processor to read its new PC from a ROM memory location (Unibus location 773024_8) and offset switch bank on the M9301 on power-up. Switch S1-2 on the M9301 can enable or disable this feature. A new PSW will also be read from a location (Unibus location 773026_8) in the M9301 memory. This new PC and PSW will then direct the processor to a program (typically a bootstrap) in the M9301 ROM (Unibus memory locations 773000 through 773776).

If switch S1-2 is turned off, disabling power-up boots, an external switch or logic level can be used to force the processor to execute a boot program.

2.4 POWER-UP BOOTING LOGIC

The status of the PDP-11 power supply is described by the two Unibus control lines BUS AC LO L and BUS DC LO L. The condition of these two lines in relation to the +5 V output of the power supply is defined by Unibus specifications as summarized in Figure 2-1.

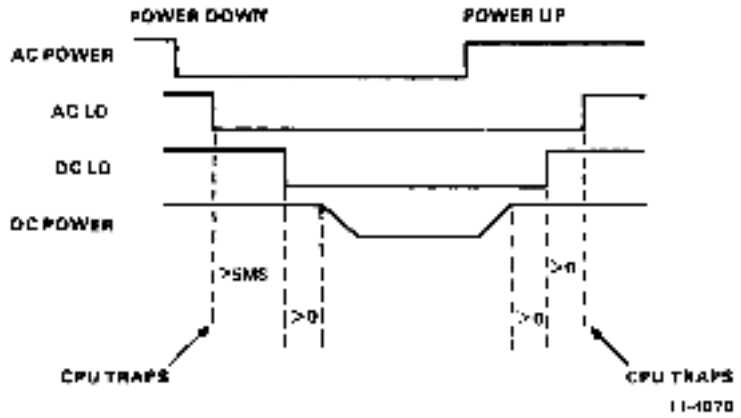

Figure 2-1 Power Fail Sequence

2.4.1 Power-Up and Power Down

On the M9301, power-up sequences are detected by the circuitry shown in Figure 2-2. When +5 V first becomes true, both BUS AC LO and BUS DC LO are asserted low. Assuming the POWER UP REBOOT ENABLE switch (S1-2) is closed on, flip-flop E29 will then be set. When BUS DC LO L goes high followed by BUS AC LO L, this flip-flop is then cleared, generating a low-to-high transition on the output of E20 (pin 13). This transition triggers the one-shot E21 which asserts Unibus address lines BUS A09 L, BUS A10 L, and BUS A12 L through BUS A17 L for up to 300 ms.

2.4.2 Processor Reads New Program Counter

During the 300 ms time-out of E21, the central processor will be performing its power-up sequence. When the processor attempts to read a new program counter (PC) address from memory location 24_8, the address bits enabled by the one-shot are logically ORed to generate the address 773024_8. This location happens to be an address in the M9301 ROM space which contains the starting address of a specific boot routine.

2.4.3 Processor Reads New Status Word

Having obtained a new PC from location 773024_8, the processor then attempts to read a new processor status word (PSW) from memory location 26_8. The address bits enabled by one-shot E21 are logically ORed to generate the address 773026_8 which is also in the M9301 ROM address space. Once this transfer is completed, the removal of MSYN from the bus will generate an ADDR CLR L signal which clears the one-shot (E21) time-out, removing address bits BUS A09, BUS A10, and BUS A12 through BUS A17. The 300 ms time-out length of E21 was chosen to guarantee enough time for any PDP-11 processor to complete the two memory transfers described before releasing the address lines.

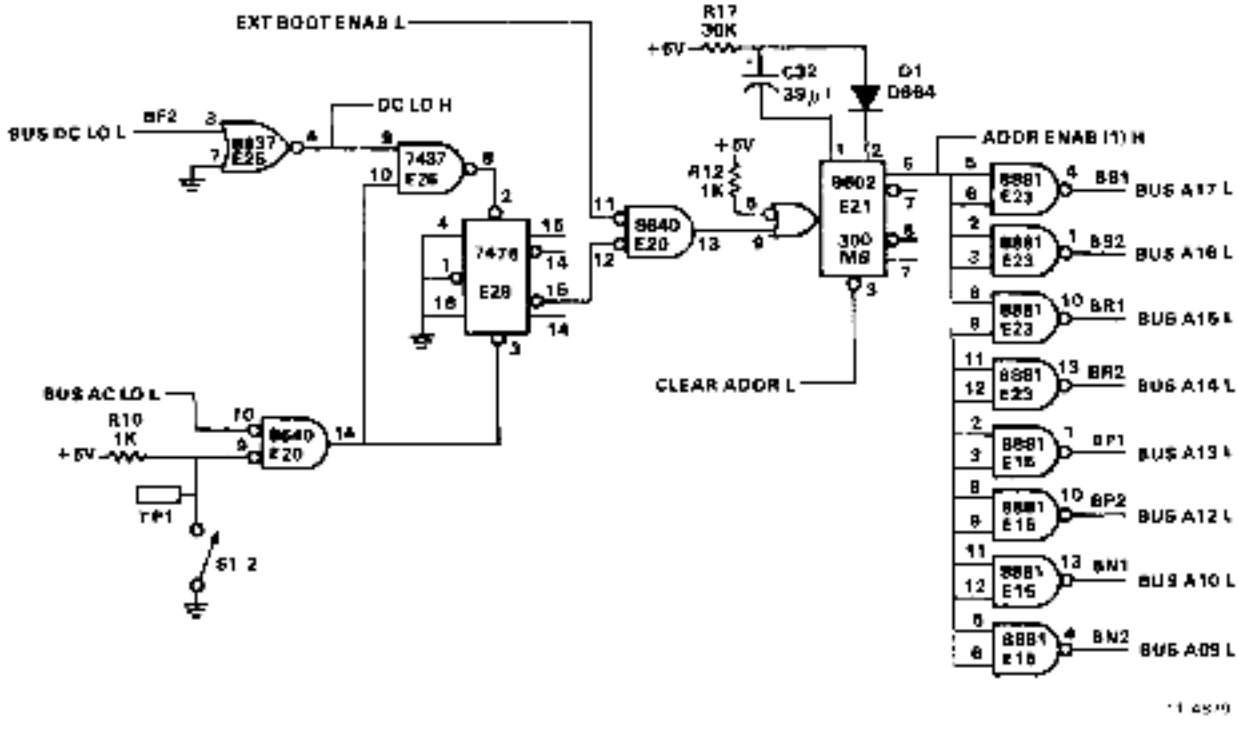

Figure 2-2 Power-Up Boot Logic

2.4.4 Power-Up Reboot Enable Switch

The POWER UP REBOOT ENABLE switch (S1-2) can be used to disable the logic shown in Figure 2-2. With this switch open, the clear input to flip-flop E29 will always be low, preventing it from ever being set on power restarts. Faston tab TP1 is provided to allow switch S1-2 to be remotely controlled. Note that when an external switch is used, S1-2 must be left in the OFF position.

2.5 EXTERNAL BOOT CIRCUIT

The processor can be externally activated by grounding the Faston tab TP2 input, as shown in Figure 2-3. This low input sets flip-flop E13 which then generates a BUS AC LO L signal on the Unibus. Upon seeing this signal the PDP-11 processor will begin a power-down routine, anticipating a real power failure. After completing this routine, the processor will wait for the release of BUS AC LO L, at which time it will perform a power-up sequence through locations 24_8 and 26_8.

When the EXTERNAL BOOT input switch is released, the one-shot E21 is triggered, causing an 8 ms time-out, and the set input to flip-flop E13 is removed (Figure 2-4). At the end of the one-shot time-out, flip-flop E13 is clocked low, releasing the BUS AC LO L line and firing the 300 ms one-shot (E21). The processor then is forced to read its new PC and PSW from location 773024 and 773026 respectively.

2.6 POWER-UP TRANSFER DETECTION LOGIC

After BUS AC LO L and BUS DC LO L have performed their power-up sequence, the logic shown in Figure 2-5 counts the first two DATI transfers on the Unibus and generates a 75 ns pulse on the CLR ADDR L line. The two Unibus transfers performed will be to obtain a new PC and PSW as previously described. The CLR ADDR L pulse resulting will be used to clear the one-shot E21 shown in Figure 2-2, releasing bus address line BUS A09, BUS A10, and BUS A12 through BUS A17.

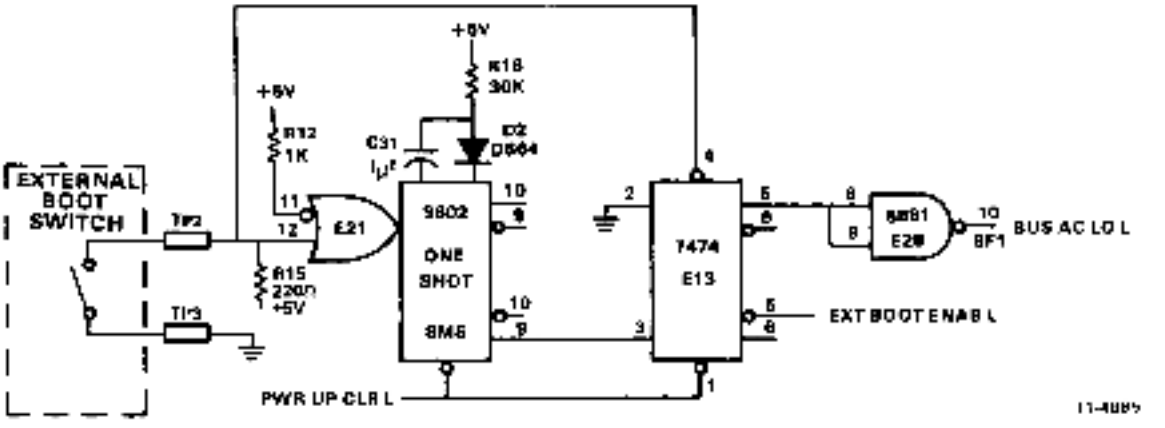

Figure 2-3 External Boot Circuit

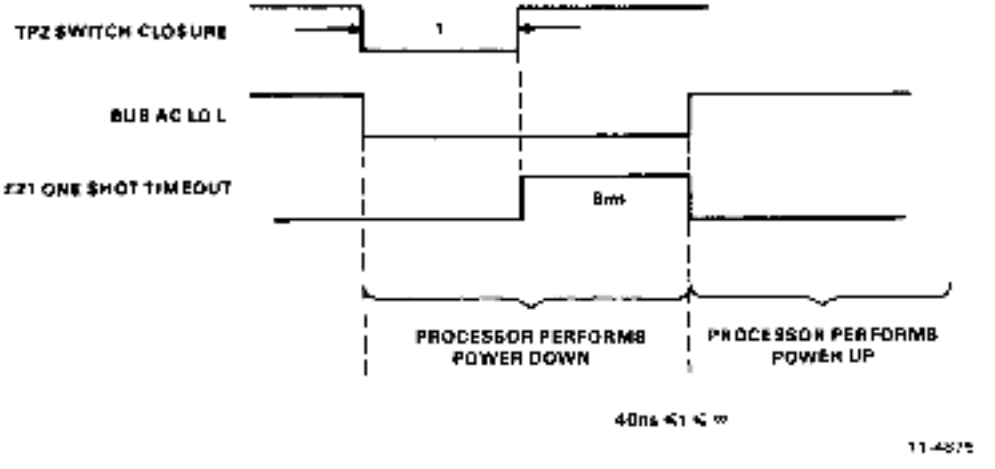

Figure 2-4 External Boot Timing

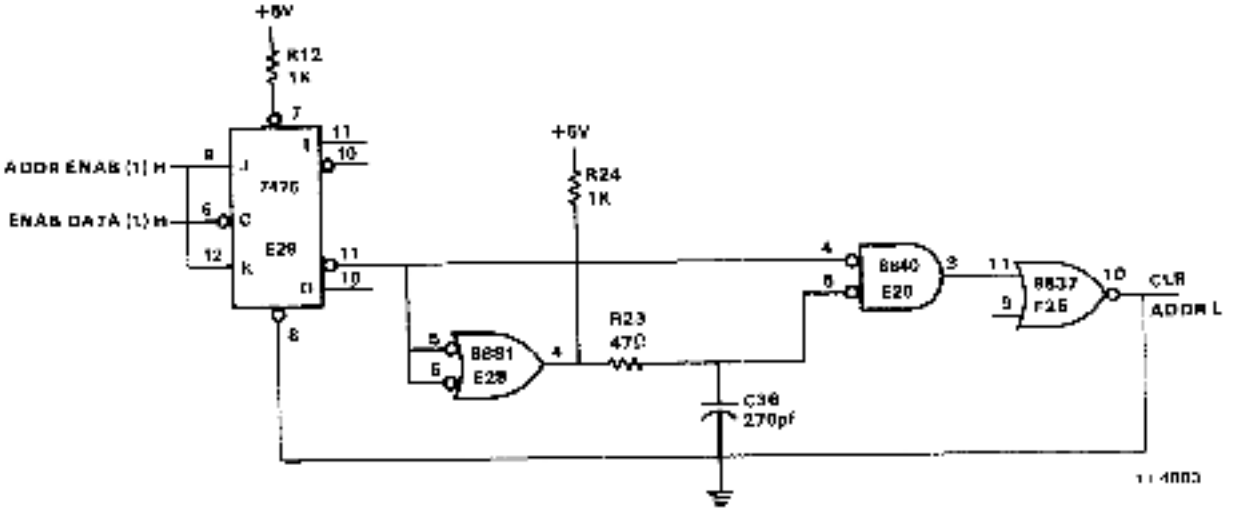

Figure 2-5 Transfer Detection Logic

2.7 POWER-UP CLEAR

The circuit shown in Figure 2-6 is included on the M9301 to guarantee that specific storage elements on the module are cleared when power is first applied. The PWR CLR L signal will be held low for approximately 70 ms after the +5 V has returned, assuming that the +5 V supply has a rise time of less than 20 ms. The exact period of time for holding PWR CLR L is a function of the rise time of the +5 V power supply.

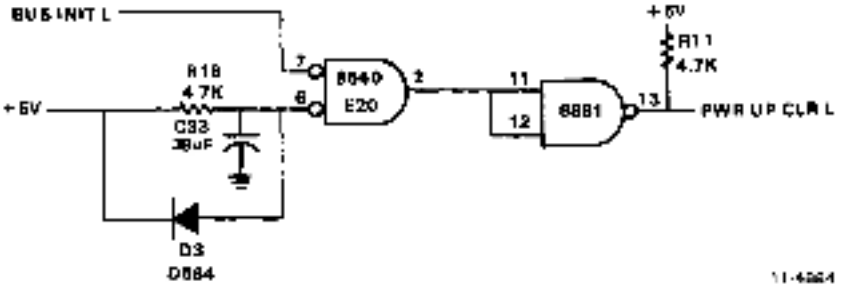

Figure 2-6 Power-Up Clear Logic

2.8 ADDRESS DETECTION LOGIC

2.8.1 M9301 Address Space

Figure 2-7 shows the complete Unibus address detection logic on the M9301. The purpose of this circuitry is to detect Unibus addresses within the address space of the M9301: 773000_8 – 773777_8 and 765000_8 – 765777_8, and to recognize the specific addresses 773024_8 and 773026_8 for the power-up circuit previously described.

2.8.2 M9301 Memory Access Constraints

The circuitry shown in Figure 2-7 determines when the M9301 ROM address space is being accessed. Upon receiving a recognized Unibus address, and BUS MSYN, the ROM data outputs are enabled onto the Unibus data lines (BUS D00 L – BUS D15 L), and BUS SSYN L is enabled 200 ns later. Conditions which must be met before enabling the ROM data and returning BUS SSYN are as follows:

1. Detection of the Unibus address 765XXX (dependent on position of the L ROM ENABLE switch S1-1) or 773XXX.

2. Transfer being performed is a DATI operation where BUS C1 L is not asserted.

3. A BUS MSYN L control signal has been obtained.

2.8.3 Low ROM Enable Switch

The LOW ROM ENABLE switch (S1-1) shown in Figure 2-7 allows the user to disable the M9301 detection of Unibus addresses 765000 through 765777. Disabling the detection of these addresses (S1-1 is set to OFF position) becomes essential when that memory space is being used by other peripheral devices in the system. For M9301 modules containing standard DEC programs, users should note what program features will be eliminated by disabling M9301 address locations 765000 through 765777.

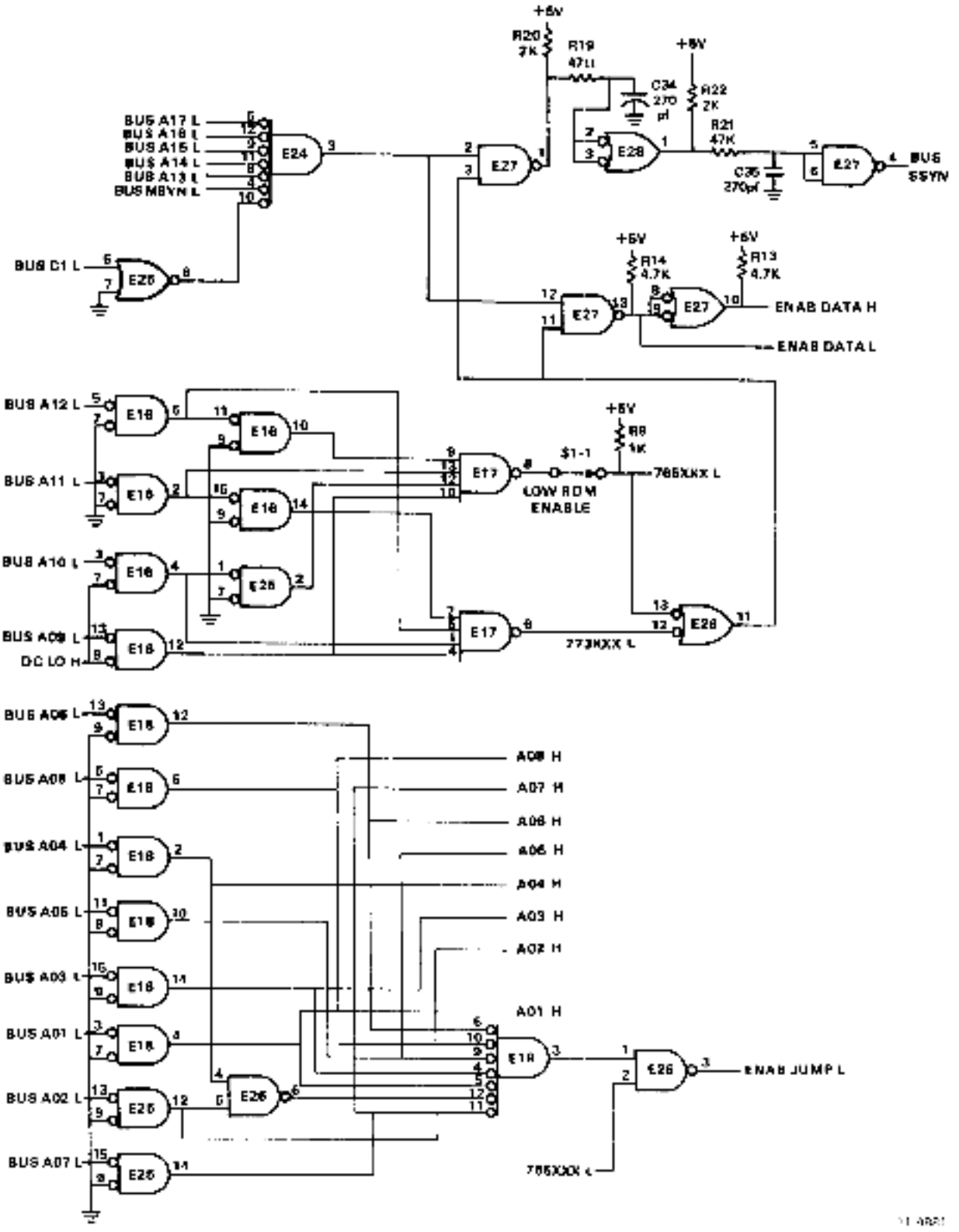

Figure 2-7 Address Decoder Logic

2.8.4 ROM Address Generation

Logic shown in the lower half of Figure 2-7 performs two functions. First it receives the nine address inputs for the M9301 ROM memory (765XXXL and A01H through A08H). Second it detects the Unibus address 773024 and generates the offset switch enable signal ENAB JUMP L.

2.9 ADDRESS OFFSET SWITCH BANK

As previously mentioned, on power-up, the M9301 processor obtains its new PC from location 773024_8 instead of location 24_8. When the M9301 address detection logic decodes address 24_8, it enables (via ENAB JUMP L) the address offset switch bank (Figure 2-8). The contents of these switches, combined with the contents of the specified address in M9301 ROM memory, produce a new PC for the CPU. This new PC will point the processor to the starting address of a specific program (usually a bootstrap routine) in the M9301 memory. Several programs can be included in the M9301 memory with any one being user selectable through the address offset switch selection.

Example:

M9301 ROM address 773024 contains	173000
Offset switch bank contains	254
New PC read by CPU	173254

NOTE
A switch in the ON position = 0.
A switch in the OFF position = 1.

See Figure 1-3 for the relationship of the address offset switches to the bus address bits.

2.10 ROM MEMORY

The heart of the M9301 is the 512-word ROM shown in Figure 2-9. It is composed of four 512- × 4-bit tristate ROMs organized in a 512- × 16-bit configuration. All four units share the same address lines and produce 16-bit PDP-11 instructions for execution by the processor.

All four sets of ROM outputs are always enabled, so that any change in address inputs will result in a change in the Unibus data lines when the ENAB DATA signals in Figure 2-7 are enabled. For further information on M9301 compatible ROMs, consult Chapter 6.

M9301 users that program their own PROMs should note the following programming constraints.

1. There is no address or data output translation required.

2. Unibus address locations 773000_8 through 773776_8 are located in the lower 256 words of the PROM. These addresses constitute the upper portion of the PROM, as far as the rest of the system is concerned, however. Unibus address locations 765000_8 through 765776_8 are resident in the upper 256 words. These addresses consitute the lower PROM addresses with relation to the system, however, and are enabled and disabled by S1-1, the LOW ROM ENABLE switch.

3. When coding PROM patterns, data bits D01 through D08 must always contain the inverse of the data required, to compensate for the extra inversion logic included in the M9301 offset switch circuitry.

4. PROM location 773024 bits D01 through D08 must be logical 1s for the offset switches to work correctly.

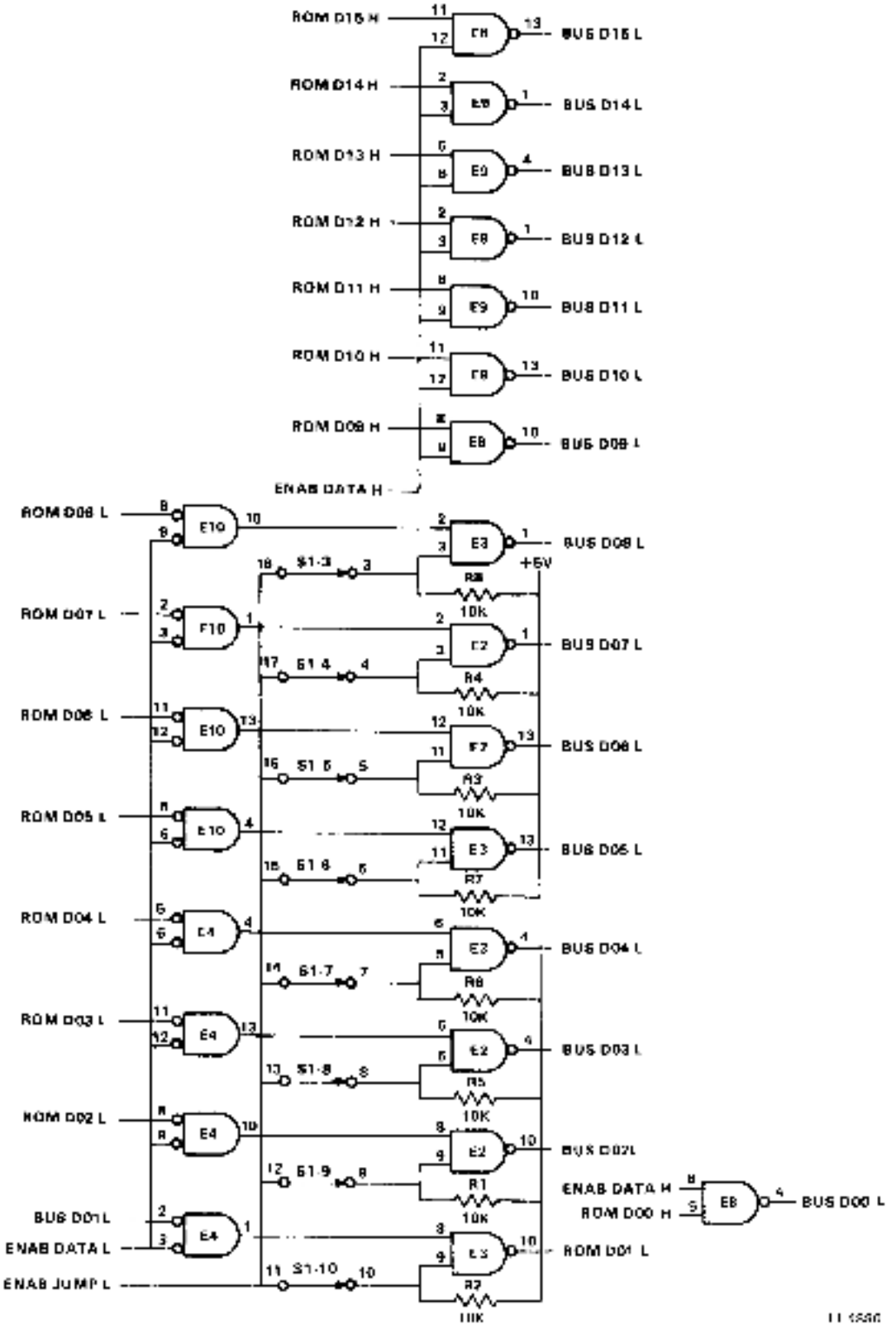

Figure 2-8 Address Offset Switch Bank

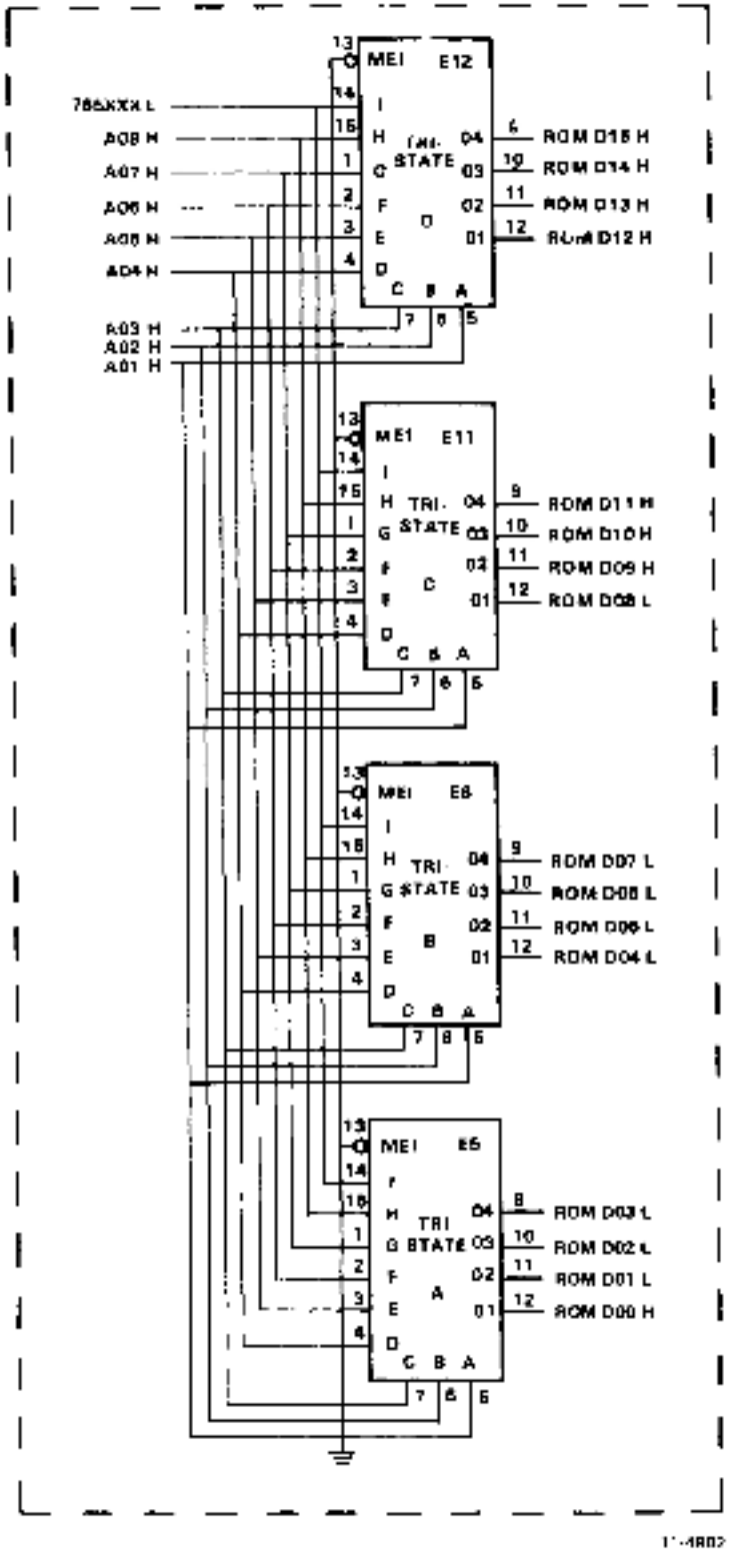

Figure 2-9 M9301 ROM Memory

2.11 M9301 TERMINATOR

The terminator section of the M9301 consists of four resistor pack circuits, each containing the required pull-up and pull-down resistors for proper Unibus termination. Since PDP-11/04 and PDP-11/34 computers incorporate BUS GRANT pull-up resistors on the processor modules, space has been left on the M9301 for five jumpers (W1 through W5) which allow the user to decide whether or not to include BUS GRANT pull-up resistors. The M9301 BUS GRANT jumpers must be installed on all modules which are used to replace an M930.

Caution should be taken when inserting M9301 modules in various PDP-11 computers. If the processor in question does not have BUS GRANT pull-ups on the CPU, jumpers W1 through W5 on the M9301 should be inserted or the M9301 should be positioned at the end of the Unibus furthest from the CPU.

A good rule of thumb is that the jumpers should be installed when the M9301 is used in place of an M930 or equivalent.

CHAPTER 3
THE CONSOLE EMULATOR

3.1 GENERAL
The console emulator routine is a standard part of the firmware for the M9301-YA, -YB, -YE, -YF, and -YJ options. A simplified operator's flowchart is presented in Figure 3-1 to give the reader a unified picture of the routine.

3.2 SYMBOLS
A list of the symbols used follows.

Rectangle: indicates an automatic operation performed by the machine.

Diamond: indicates a decision, an automatic operation which can take either of two paths depending on how the question stated within the diamond is answered.

Trapezoid: indicates operator action, the moving of a switch or the typing of keys.

3.3 USING THE CONSOLE EMULATOR
With switches S1-1 through S1-10 set in the ON position the system will execute a normal console emulator power-up routine when power is supplied or the boot switch is pressed. Diagnostic tests 1–5 will run first. An error will cause the processor to loop. Completion of these diagnostic tests will be followed by the register display routine. R0, R4, R6, and R5 will be printed out on the terminal. A $ sign will be printed at the beginning of the next line on the terminal, indicating that the console emulator routine is waiting for input from the operator. In the discussion that follows, the following symbols will be used.

(SB): Space bar
(CR): Carriage return key
X: Any octal number 0–7

The four console functions can be exercised by pressing keys, as follows:

Function	Keyboard Strokes
Load Address	L (SB) X X X X X X (CR)
Examine	E (SB)
Deposit	D (SB) X X X X X X (CR)
Start	S (CR)

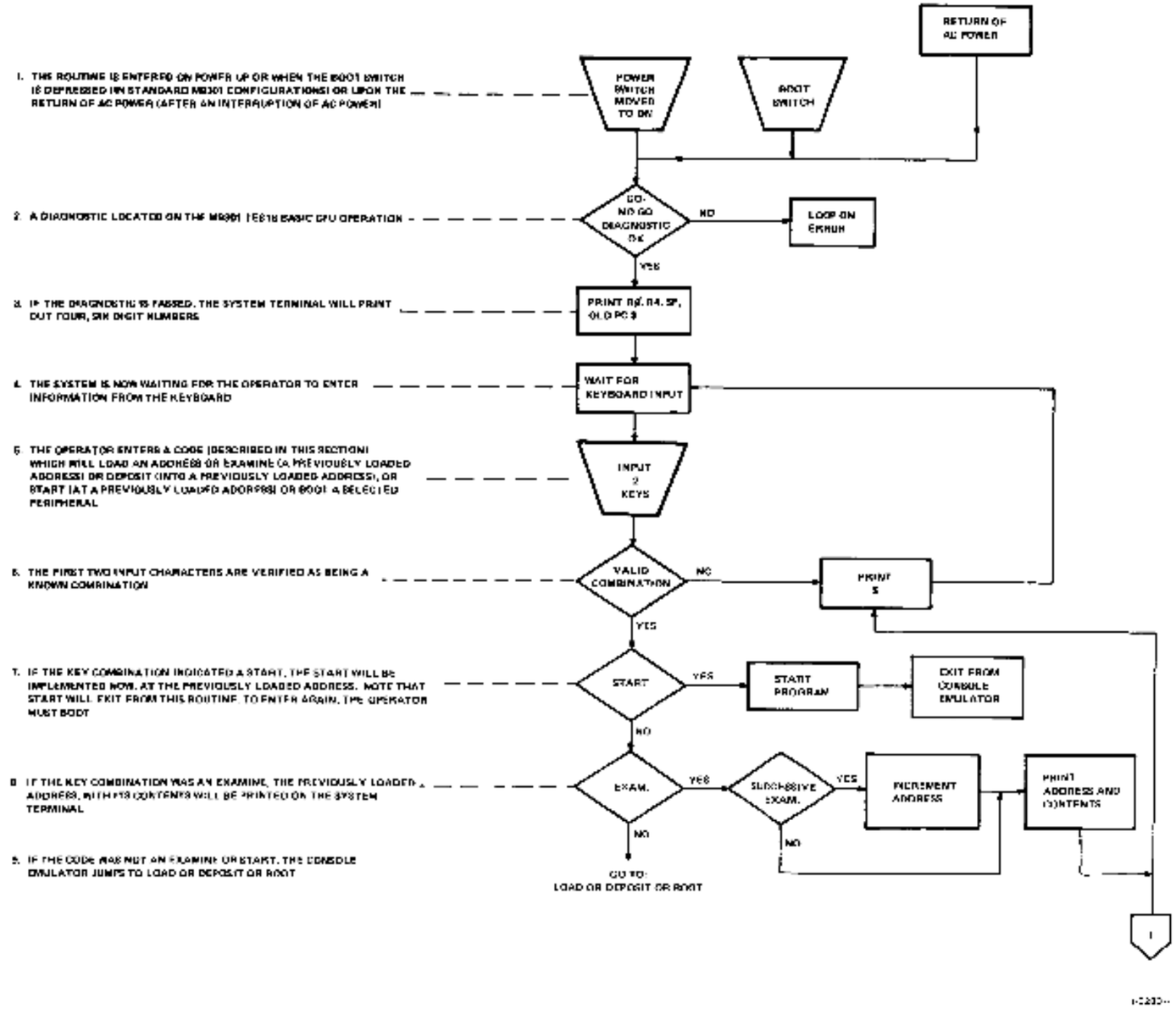

Figure 3-1 Console Emulator Routine Flowchart (Sheet 1 of 2)

LOAD SEQUENCE	DEPOSIT SEQUENCE	BOOT SEQUENCE
10. SEQUENCE IS ENTERED	SEQUENCE IS ENTERED	SEQUENCE IS ENTERED
11. ADDRESS IS ENTERED MOST SIGNIFICANT DIGIT IS FIRST	DATA TO BE DEPOSITED IS ENTERED, MOST SIGNIFICANT DIGIT FIRST	UNIT NUMBER IS ENTERED. DEFAULT NUMBER IS ZERO
12. A CHECK IS MADE TO INSURE THAT THE NUMBER WHICH WAS ENTERED WAS AN OCTAL NUMBER 0-7. IF IT ISN'T, A ? IS PRINTED SIGNIFYING AN ERROR.		
13. ENTERING (CR) SIGNIFIES THE END OF THE LOAD SEQUENCE	TEST TO SEE IF THE COMBINATION INPUT, IMMEDIATELY PRECEDING THIS ONE WAS A DEPOSIT	ENTERING (CR) SIGNIFIES THAT THE OPERATOR WISHES TO BOOT NOW
14. ADDRESS IS LOADED INTO AN INTERNAL REGISTER AND $ IS PRINTED SIGNIFYING SYSTEM IS READY FOR NEXT KEYBOARD INPUT	IF PREVIOUS OPERATION WAS A DEPOSIT, ADDRESS WILL INCREMENT	PRIOR TO BOOTING, A MEMORY DIAGNOSTIC IS RUN. IF IT FAILS, RUN LIGHT REMAINS ON AND PERIPHERAL IS NOT BOOTED
15.	(CR) INDICATES THAT THE OPERATOR WISHES TO DEPOSIT NOW	IF MEMORY DIAGNOSTIC PASSED, THE DEVICE IS NOW BOOTED
16.	DATA IS DEPOSITED FOLLOWED BY A $ PRINTOUT, SIGNIFYING THE SYSTEM IS READY FOR THE NEXT KEYBOARD INPUT	MACHINE EXITS FROM THE CONSOLE EMULATOR. TO RE-ENTER THE OPERATOR MUST DEPRESS THE BOOT/INIT SWITCH

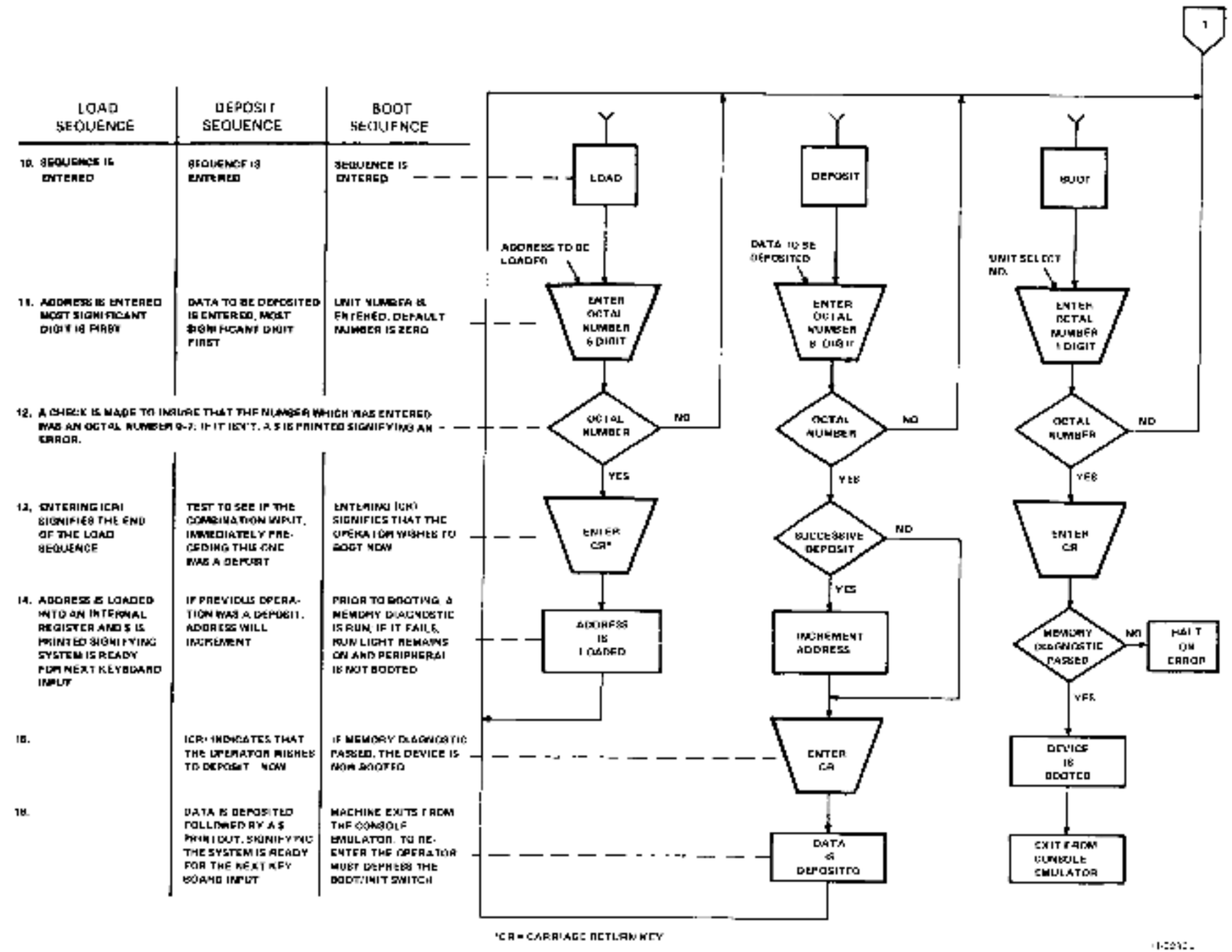

Figure 3-1 Console Emulator Routine Flowchart (Sheet 2 of 2)

The first digit typed will be the most significant digit. The last digit typed will be the least significant. If an address or data word contains leading zeros, these zeros can be omitted when loading the address or depositing the data. An example using the load, examine, deposit, and start functions follows. 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.

To accomplish this, the following procedure must be followed.

Operator Input		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

NOTE

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

3.3.1 Successive Operations

3.3.1.1 Examine – Successive examine operations are permitted. The address is loaded for the first examine only. Successive examine operations cause the address to increment and will display consecutive addresses along with their contents.

For example, to 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

3.3.1.2 Deposit – Successive deposit operations are permitted. The procedure is identical to that used with examine.

For example, to deposit 60 into location 500, 2 into location 502, and 4 into location 504:

Operator Input	Terminal Display
L(SB) 500 (CR)	SL 500
D(SB) 60 (CR)	$D 60
D(SB) 2 (CR)	$D 2
D(SB) 4 (CR)	$D 4

3.3.1.3 Alternate Deposit-Examine Operations

This mode of operation will not increment the address. The address will contain the last data which was deposited.

For example, to load address 500 and deposit 1000, 2000, and 5420 with examine operations after every deposit:

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

3.3.1.4 Alternate Examine-Deposit Operations

If an examine is the first instruction after a load sequence, and is alternately followed by deposits and examines, the address will not be incremented, and the address will contain the last data which was deposited. The above example applies to this operation, and with the exception of the order of examine and deposit, the end result is the same.

3.3.2 Limits of Operation

The M9301 console emulator routine can directly manipulate the lower 28K of memory and the 4K I/O page. See Chapter 5 for an explanation of techniques required to access addresses above the lower 28K.

3.4 BOOTING FROM THE KEYBOARD

Once the $ symbol has been displayed in response to system power-up, or pressing of the boot switch, the system is ready to load a bootstrap from the device which the operator selects. The procedure is as follows.

1. Find the 2-character boot command code in the appropriate chapter that corresponds to the peripheral to be booted.
2. Load paper tape, magtape, disk, etc. into the peripheral to be booted if required.
3. Verify that the peripheral indicators signify that the peripheral is ready (if applicable).
4. Type the 2-character code obtained from the table.
5. If there is more than one unit of a given peripheral, type the unit number to be booted (0–7). If no number is typed, the default number will be 0.
6. Type (CR); this initiates the boot.

Before booting, always remember:

1. The medium (paper tape, disk, magtape, cassette, etc.) must be placed in the peripheral to be booted prior to booting.

2. The machine will not be under the control of the console emulator routine after booting.

3. The program which is booted in must:
   a. Be self starting or
   b. Allow the user to load another program by using the CONT function or
   c. Be startable from the console emulator after having been booted in.

4. Actuating the boot switch will always abort the program being run. The contents of the general registers (R0–R7) will be destroyed. There is no way to continue with the program which was aborted. Some programs are designed to be restartable.

3.4.1 Booting the High-Speed Reader Using the Console Emulator

To load the CPU diagnostic for an 11/34 computer system with a high-speed reader, perform the following procedure.

1. Place the HALT/CONT switch in the CONT position.

2. Obtain a $ by either turning on system power or actuating the boot. (R0, R4, SP, and old PC will be printed prior to the $.)

3. Place the absolute loader paper tape (coded leader section) in the high-speed reader.

4. Type: PR (CR)
   The absolute loader tape will be loaded, and the machine will halt.

5. Remove the absolute loader and place the leader of the program, in this case a CPU diagnostic, in the reader.

6. Move the HALT/CONT switch to HALT and then return it to CONT. The diagnostic will be loaded and the machine will halt (normal for this program; non-diagnostic programs may or may not be self-starting).

7. If program is not self-starting, activate the BOOT/INIT switch. This will restart the console emulator routine.

8. Using the console emulator, deposit desired functions into the software switch register (a memory address) location. (See the diagnostic for the software switch register's actual location and significance.)

9. Using the console emulator, load the starting address, and start the program as described earlier in this section.

3.4.2 Booting a Disk Using the Console Emulator

To boot the system's RK11 disk, which contains the CPU diagnostic that you want to run, perform the following procedure.

1. Verify that the HALT/CONT switch is in the CONT position and the write lock switch on the RK11 peripheral is in the ON position.

2. Turn on system power or press the console boot switch. The system terminal displays R0, R4, SP, and old PC which are octal numbers, followed by a $ on the next line.

3. Place the disk pack in drive 0.

4. When the RK05 load light appears, the system is ready to be booted.

5. Type: DK (CR)
   This causes the loading of the bootstrap routine into memory and the execution of that routine.

6. The program should identify itself and initiate a dialogue (which will not be discussed here).

3.5 RECOVERING FROM ERRORS IN THE CONSOLE EMULATOR ROUTINE

Table 3-1 describes the effects of entering information incorrectly to the console emulator routine. The following symbols are used in the table.

(9) – Represents a non-octal number (8 or 9)

(Y) – Represents:

1. All keys (other than numerics) which are unknown

2. Keys which are known but do not constitute a valid code in the context which they are entered.

Refer to previous sections for a discussion of the correct method operating the console emulator routine.

Escape Route

If an entry has not been completed and the user realizes that an incorrect or unwanted character has been entered, press the rub out or delete key. This action will void the entire entry and allow the user to try again.

Table 3-1 Deposit Errors: Useful Examples

Error	Result	Remedy	Operator	Terminal
L was followed by a key other than (SB)	Terminal display will immediately return a S to signify an unknown code. No address is loaded.	Try again	L(Y)	SL S
An illegal (non octal) number (8 or 9) is typed after the correct load entrance, within an otherwise valid number.	Upon receipt of the illegal number, the Console Emulator will ignore the entire address and return a S.	Try again	L(SB)XXX9	SL XXX9 S
An incorrect alphabetic key is typed after the correct load entrance within an otherwise valid number.	Same as illegal number.	Try again	L(SB)XXXY	SL XXXY S
The most significant octa. number in a six bit address is greater than one.	An address will be loaded. However, the state of the most significant address bit will be determined by bit 15 only: 2 = 0 3 = 1 4 = 0 5 = 1 6 = 0 7 = 1	Try again if required	S(SB) 6XXXXX	SS 0XXXXX S
An unwanted but legal octal number is loaded.	Address will be loaded but unchanged.	Try again		
An extra (seventh) octal number is typed.	The loaded number will be incorrect. The system will accept any size word but will only remember the last six characters typed in.	Try again	L(SB)1XXXXXX	SL 1XXXXXX S Actually Load XXXXXX
A memory location higher than the highest memory location available in the machine is loaded.	No errors will result unless a deposit, examine, or start is attempted.	Try again	L(SB)1XXXXX (CR)	L 1XXXXX

Table 3-1 Deposit Errors: Useful Examples (Cont)

Error	Result	Remedy	Operator	Terminal
Load address and number were entered correctly, (CR) was not entered.	Machine will wait indefinitely for (CR). $ will not be returned.	Type (CR)	L(SB)XXXXX (CR)	$L XXXXX $XXXXX(CR)
E or S is followed by a key other than space.	Terminal display will immediately return a $ to signify an unknown code.	Try again	E(Y) or S(X)	SY $ SS $
Examine or start is attempted to a memory location which is higher than the highest available memory location in the machine (I/O page can be examined) or to an odd memory location.	The system will hang up when (SB) is executed.	Depress the boot switch	L(SB) or S(SB)	SL or SS
Examine is performed without loading an address prior to first examine.	An examine operation of an unknown address will be performed. It is possible that the machine may attempt to examine an address which does not exist. The system may hangup in a program loop.	Try again or boot if system hangs up		
Start is performed without loading an address prior to starting.	Start at an unknown location will occur.	Reload program and try again.		
D was followed by a key other than space.	Terminal display will immediately return a $ to signify an unknown code.	Try again	DY	$D $
Deposit is attempted to a memory location which is higher than the highest available memory location in the machine (with the exception of the I/O page).	The system will hang up when (SB) is executed	Activate the boot switch to reboot the system.		

Table 3-1 Deposit Errors: Useful Examples (Cont)

Error	Result	Remedy	Operator	Terminal
Deposit is performed without loading an address or knowing what address has been previously loaded.	a. Data will be written over and lost.	a. Immediately following the error, perform an examine to determine the location which was accessed. Restore original contents if known.		
	b. Machine might hang up in a program loop.	b. Reboot machine.		
Deposit into an odd address is attempted.	The system will hang up when (CR) is executed.	Depress the boot switch.		

CHAPTER 4
BOOTSTRAPPING – M9301-YA, -YB, -YE, -YF, -YJ

The routines to bootstrap a device typically read in the first sector, block, or 512 words from the device into location 0 through 512 of memory. The exception to this rule is the paper tape boot. The paper tape boot is unique in that it can do no error checking and the secondary bootstrap (the absolute loader, for example) is read into the upper part of memory. The actual locations loaded by the paper tape boot are partially determined by the secondary bootstrap itself and by the size routine which determines the highest available memory address within the first 28K. The flexible disk (or floppy) reads sector 1 on track 1 into consecutive locations starting at 0. The magnetic tape boots read the second block into consecutive locations starting at 0. If no errors are detected in the device, the bootstraps normally transfer control to location 0 in order to execute the secondary bootstrap just loaded. The only exception to this starting address concerns the paper tape boots. They transfer control to location XXX374, where XXX is determined initially by the size routine to be at the top of memory. This is where the absolute loader has just been loaded.

If a device error is detected, a reset will be executed and the bootstrap will try again. The bootstrap will be retried indefinitely until it succeeds without error unless the user (operator) intervenes. The advantage of retrying the boot is that if a particular device being booted is not on-line or not loaded, perhaps because of a power failure, the boot will give the device a chance to power up (for disks this is essential).

A magnetic tape transport, however, will not automatically reload itself after a power failure and restart. This situation requires user intervention. The user must reload the magtape and bring it back on-line, at which time the magtape bootstrap, which will have been continually attempting to boot the tape, will succeed.

Note that the only way to bootstrap a device drive (unit or transport) other than drive 0 is by entering the console emulator to specify the drive number desired. Otherwise the bootstraps will default to drive 0. This means that only drive 0 of a device can be bootstrapped without operator intervention.

CHAPTER 5
EXTENDED ADDRESSING

5.1 GENERAL

This chapter applies to use of the M9301-YA, -YB, -YE, -YF, and -YJ in PDP-11 systems which have no console. When the memory of a PDP-11 system is extended beyond 28K, the processor is able to access upper memory through the memory management system. However, the console emulator normally allows the user to access only the lower 28K of memory. This chapter provides an explanation of the method by which the user can gain access to upper memory in order to read or modify the contents of any location. The reader should be familiar with the concepts of memory management in the KD11-E processor.

5.2 VIRTUAL AND PHYSICAL ADDRESSES

Addresses generated in the processor are called virtual addresses, and will be 16 bits in length. Physical addresses refer to actual locations in memory. They are asserted on the Unibus and may be up to 18 bits in length (for 128K memories).

5.3 ADDRESS MAPPING WITHOUT MEMORY MANAGEMENT

With memory management disabled (as is the case following depression of the boot switch), a simple hardware mapping scheme converts virtual addresses to physical addresses. Virtual addresses in the 0 to 28K range are mapped directly into physical addresses in the range from 0 to 28K. Virtual addresses on the I/O page, in the range from 28K to 32K (160000=177776), are mapped into physical addresses in the range from 124K to 128K.

5.4 ADDRESS MAPPING WITH MEMORY MANAGEMENT

With memory management enabled, a different mapping scheme is used. In this scheme, a relocation constant is added to the virtual address to create a physical or "relocated" address.

Virtual address space consists of eight 4K banks where each bank can be relocated by the relocation constant associated with that bank. The procedure specified in this section allows the user to:

1. Create a virtual address to type into the load address command.
2. Determine the relocation constant required to relocate the calculated virtual address into the desired physical address.
3. Enable or disable the memory management hardware.

5.5 CREATION OF A VIRTUAL ADDRESS

The easiest way to create a virtual address is to divide the 18-bit physical address into two separate fields – a virtual address and a physical bank number. The virtual address is represented by the lower 13 bits and the physical bank by the upper 5 bits. The lower 3 bits of the physical bank number (bits 13, 14, 15) represent the virtual bank number (Table 5-1). Thus if bits 13, 14, and 15 are all 0s, the virtual bank selected is 0. The user should calculate the relocation constant according to Table 5-2. He can then deposit this constant in the relocation register associated with virtual bank 0 (Table 5-1).

Table 5-1 Unibus Address Assignments

Virtual Address	Virtual Bank	Relocation Register	Descriptor Register
160000-177776	7	172356	172316
140000-157776	6	172354	172314
120000-137776	5	172352	172312
100000-117776	4	172350	172310
060000-077776	3	172346	172306
040000-057776	2	172344	172304
020000-037776	1	172342	172302
000000-017776	0	172340	172300

Table 5-2 Relocation Constants

Physical Bank Number	Relocation Constant	Physical Bank Number	Relocation Constant
37	007600	17	003600
36	007400	16	003400
35	007200	15	003200
34	007000	14	003000
33	006600	13	002600
32	006400	12	002400
31	006200	11	002200
30	006000	10	002000
27	005600	7	001600
26	005400	6	001400
25	005200	5	001200
24	005000	4	001000
23	004600	3	000600
22	004400	2	000400
21	004200	1	000200
20	004000	0	000000

One relocation register exists for each of the eight virtual banks. In addition to the relocation registers, each bank has its own descriptor register which provides information regarding the types of access allowed (read only, read or write, or no access).

The memory management logic also provides various forms of protection against unauthorized access. The corresponding descriptor register must be set up along with the relocation register to allow access anywhere within the 4K bank.

For example, assume a user wishes to access location 533720_8. The normal access capability of the console is 0 to 28K. This address (533720) is between the 28K limit and the I/O page (760000–777776), and consequently must be accessed as a relocated virtual address, with memory management enabled. The virtual address is 13720 in physical bank 25 and is derived as follows.

All locations in bank 25 may be accessed through the virtual addresses 000000–017776. The relocation and descriptor registers in the processor are still accessible since their addresses are within the I/O page. (Note that access to the I/O page is not automatically relocated with memory management, while access to the I/O page is automatically relocated when memory management is not used.)

The relocation constant for physical bank 25 is 005200. This constant is added in the relocation unit to the virtual address, as shown, yielding 533720.

013720	Virtual address
520000	Relocated constant (Table 5-2)
533720	Physical address

The Unibus addresses of the relocation registers and the descriptor registers are given in Table 5-1. The relocation constant to be loaded into the relocation register for each 4K bank is provided in Table 5-2. The data to be loaded in the descriptor register to provide read/write access to the full 4K is always 077406.

The Unibus address of the control register to enable memory mangement is 177572. This register is loaded with the value 000001 to enable memory management, and loaded with 0 to disable it.

To complete the example previously described (accessing location 533720), the console routine would be as follows:

```
$L      172340      /Access relocation register for virtual bank 0
$D        5200      /Deposit code for physical bank 25.
$L      172356      /Access relocation register for virtual bank 7.
$D        7600      /Deposit code for the I/O page.
$L      172300      /Access descriptor register, virtual bank 0.
$D       77406      /Deposit code for read/write access to 4K.
$L      172316      /Access descriptor register, virtual bank 7.
$D       77406      /Deposit code for read/write access to 4K.
$L      177572      /Access control register.
$D           1      /Enable memory management.
$L       13720      /Load virtual address of location desired.
$E                  /Examine the data in location 533720.
                    /Data will be displayed.
```

5.6 CONSTRAINTS

Loading a new relocation constant into the relocation register for virtual bank 0 will cause virtual addresses 000000–017776 to access the new physical bank. A second bank can be made accessible by loading the relocation constant and descriptor data into the relocation and descriptor registers for virtual bank 1 and accessing the location through virtual address 020000–037776. Seven banks are accessible in this manner, by loading the proper constants, setting up the descriptor data, and selecting the proper virtual address. Bank 7 (I/O page) must remain relocated to physical bank 37 as it is accessed by the CPU to execute the console emulator routine.

Memory management is disabled by clearing (loading with 0s) the control register 177572. It should always be disabled prior to typing a boot command.

The start command automatically disables memory management and the CPU begins executing at the physical address corresponding to the address specified by the previous load address command. Pressing the boot switch automatically disables memory management. The contents of the relocation registers are not modified.

The HALT/CONT switch has no effect on memory management.

CHAPTER 6
M9301-0

6.1 GENERAL.

The M9301-0 has been created as a universal bootstrap device which allows the user to program and install customized 512- × 4-bit PROMs. This module version comes with four 16-pin IC (integrated circuit) sockets in place of the ROMs normally inserted on other module versions. When configuring PROM bit patterns for the M9301-0, care should be taken to arrange them to meet the address and data output pinouts on the module (Paragraph 2.10).

The switches for this version are hardwired in the same way as the switches for the other versions (Paragraphs 2.4, 2.8, and 2.9). However, the user will have to determine how he wants to set the switches, depending on the programs he blasts into the PROMs and the program starting locations he selects.

6.2 PROM PROGRAMMMING PROCEDURE

The PROM is manufactured with logic level 0s in all storage locations. In order to program a logic level 1 at a specified bit, the ROM programmer must electrically alter a bit at logic level 0 to logic level 1. There are 2048 bits which are organized as 512 words of 4 bits each. Figure 6-1 shows a PROM pin diagram. Figure 6-2 is a block diagram showing the logic internal to the PROM.

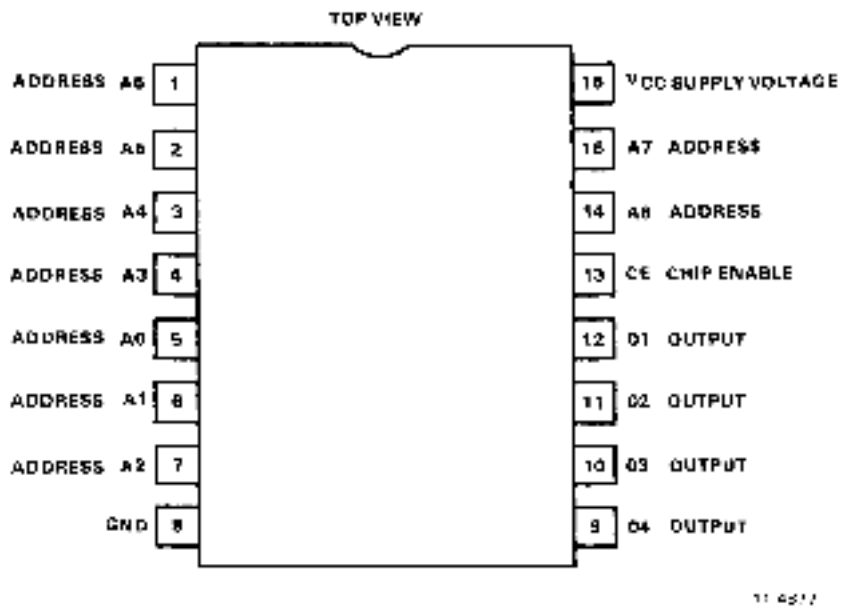

Figure 6-1 PROM Pin Connection Diagram

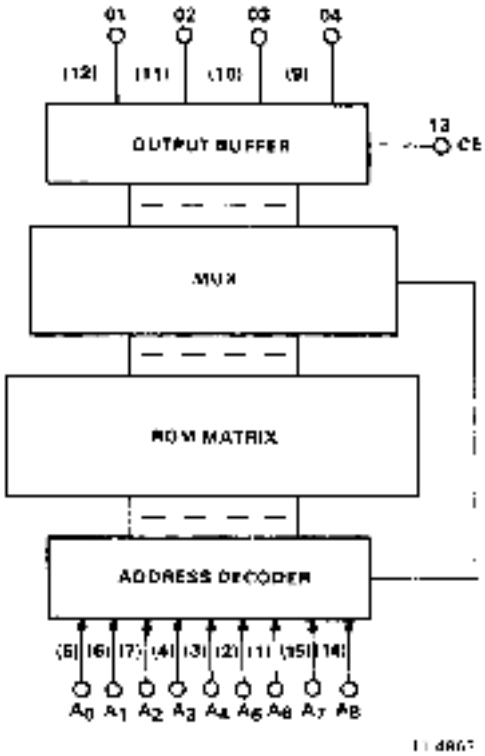

Figure 6-2 PROM Internal Logic

CHAPTER 7
M9301-YA

7.1 INTRODUCTION

The M9301-YA is designed specifically for PDP-11/04 and PDP-11/34 OEM systems. The ROM routines include basic CPU and memory GO-NO GO diagnostics, a console emulator program, and a specific set of bootstrap programs.

The only physical difference between the M9301-YA and the M9301-0 module is that specially programmed tristate 512- × 4-bit ROMs are inserted in the 16-pin dip sockets of the M9301-0. Figure 7-1 is a program memory map of the M9301-YA module, and it lists the nature of each diagnostic test in the ROMs. A program listing for the ROMs may be found in the M9301 engineering drawings.

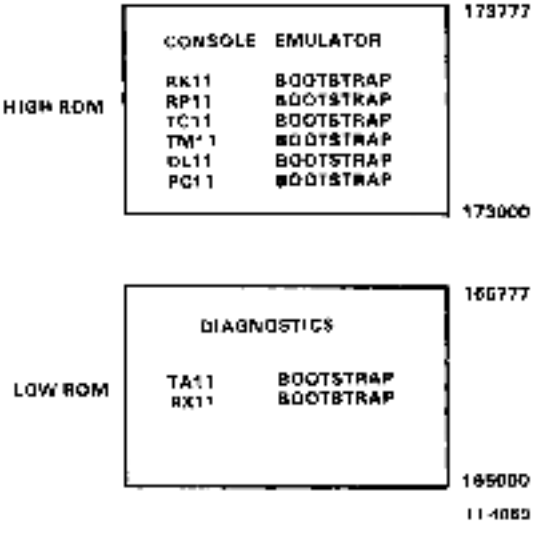

Figure 7-1 Program Memory Map for the M9301-YA

7.2 BOOTSTRAP PROGRAMS

The commands which can be used to call the bootstrap programs from the console emulator are listed in Table 7-1

Table 7-1 Bootstrap Routine Codes for the M9301-YA

Device	Description	Command
RK11	Disk Cartridge	DK
RP11	RP02/03 Disk Pack	DP
TC11	DECtape	DT
TM11	800 bpi Magtape	MT
TA11	Magnetic Cassette	CT
RX11	Diskette	DX
DL11	ASR-33 Teletype	TT
PC11	Paper Tape	PR

An explanation of the functions performed by the various bootstrap programs follows.

RX11 Diskette Bootstrap – Loads the first 64 words (200_8 bytes) of data from track one, sector one into memory locations 0–176 beginning at location 0. If location 0 does not contain 240, the boot is restarted. Restarts will occur 2000 times before the machine is halted automatically.

TA11 Cassette Bootstrap – This bootstrap is identical to that of the RX11 except that data is loaded from the cassette beginning at the second block.

PC11 Paper Tape Reader Bootstrap – Loads an absolute loader formatted tape into the upper memory locations XXX746 to XXX777 (XXX is dependent on memory size). Once loading is completed, the boot transfers operation to a routine beginning at location XXX752. In systems containing an M9301-YA which is set up not to run diagnostic test 1 through test 5, XXX will become 017, not the upper part of memory.

Disk and DECtape Bootstraps (excluding RX11) – Load 1000_8 words (2000_8 bytes) of data from the device into memory locations $0–1776_8$.

Magtape Bootstraps

TM11 – Loads second record (2000_8 bytes maximum size) from the magtape into memory location $0–1776_8$.

7.3 MICROSWITCH SETTINGS

A set of ten microswitches is located on the M9301 module. They determine which ROM routines are selected on power-up and give the user automatic access to any function. Switch S1-1 should be on for normal operation in order to allow access to the low ROM addresses (765000–765777). The primary activating processes for the M9301-YA are the power-up sequence and the enabling of the console boot switch. Switch S1-2 must be in the ON position in order to enable activation of the M9301-YA on a power-up. If switch S1-2 is off, then the processor will trap to location 24 (as normal) to execute the user power-up routine. When switch S1-2 is ON, the other switches, S1-3 through S1-10, determine what action the M9301-YA will take on power-up.

If the system includes a console boot switch, then any time that switch is pressed the M9301-YA will be activated. Note that some processors will have to be halted for this switch to have any effect. Enabling the console boot switch causes the processor to enter a ROM routine by creating a fake power-down/power-up sequence. The user should note that the position of switch S1-2 is irrelevant when the console boot switch is used.

Pushing the console boot switch thus results in a normal power-up sequence in the processor. Prior to the power-up sequence, the M9301-YA asserts 773000 on the Unibus address lines. This causes the new PC to be taken from ROM location 773024 instead of location 000024. The new PC will be the logical OR of the contents of ROM location 773024 and the eight microswitches on the M9301-YA module. A switch in the ON position is read as a 0, while a switch in the OFF position is a 1. In this way all the M9301-YA program options are accessible.

Each program option is given a different starting address. Note that microswitch S1-10 is ORed with bit 1 of the data in ROM location 773024, S1-9 is ORed with bit 2, etc. No switch is provided for combination with bit 0, because an odd address could result when going through the trap sequence. Figure 7-2 shows the relationship of the switches to the bus address bits.

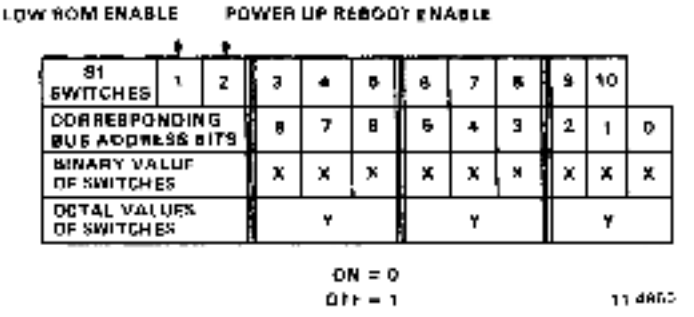

Figure 7-2 Offset Switches and Corresponding Bus Address Bits

7.4 PROGRAM CONTROL THROUGH THE MICROSWITCHES

The microswitches on the M9301-YA enable the user either to start a bootstrap operation or to enter the console emulator, simply by pressing the boot switch. Note that a momentary power failure will have the same effect as pushing the boot switch. The user should also note that he can select any function with or without diagnostics. Adding 2 to the appropriate octal code in the switches will disable the diagnostics.

Figure 7-3 shows a program flowchart for the M9301-YA. Notice that the choice and sequence of routines is entirely dependent on the offset switch settings.

Table 7-2 shows the options with their corresponding switch settings and octal codes.

7.5 DIAGNOSTIC PROGRAMS

An explanation of the eight CPU and memory diagnostic tests follows. Three types of tests are included in the M9301-YA diagnostics:

1. Primary CPU tests (1–5)
2. Secondary CPU tests (6, 7)
3. Memory test (8)

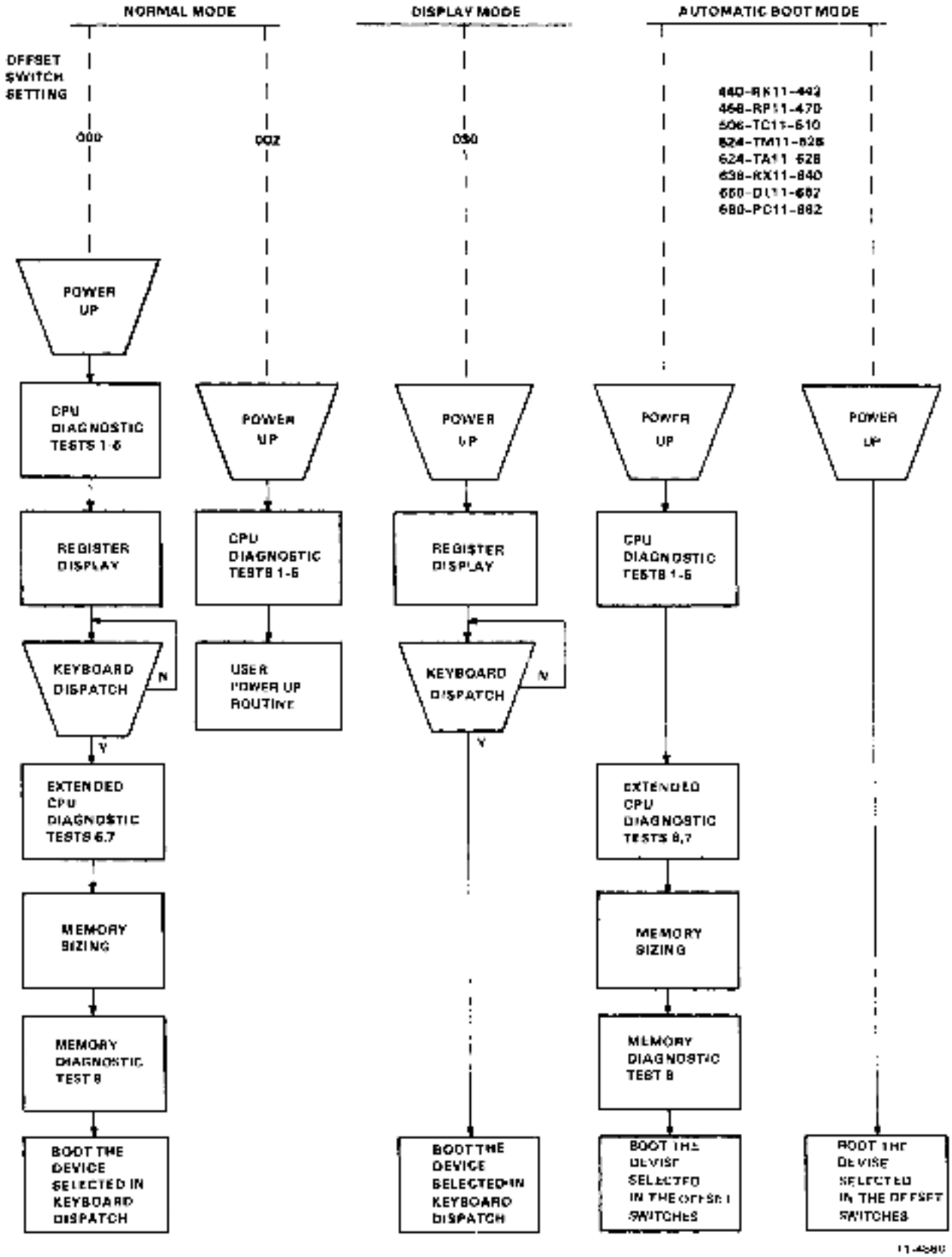

Figure 7-3 M9301-YA Program Flowchart

Table 7-2 Options and Corresponding Switch Settings

Function	S3	S4	S5	S6	S7	S8	S9	S10	Octal Code
CPU Diagnostics with Console Emulator	ON	ON	ON	ON	ON	ON	ON	ON	000
CPU Diag. Vector through location 24 Console Emulator	ON	ON	ON	ON	ON	ON	ON	OFF	002
without Diag.	ON	ON	ON	ON	OFF	OFF	ON	ON	030
CPU Diag. Boot RK11	OFF	ON	ON	OFF	ON	ON	ON	ON	440
Boot RK11 (without Diag.)	OFF	ON	ON	OFF	ON	ON	ON	OFF	442
CPU Diag. Boot RP11	OFF	ON	ON	OFF	OFF	ON	OFF	OFF	466
Boot RP11 (without Diag.)	OFF	ON	ON	OFF	OFF	OFF	ON	ON	470
CPU Diag. Boot TC11	OFF	ON	OFF	ON	ON	ON	OFF	OFF	506
Boot TC11 (without Diag.)	OFF	ON	OFF	ON	ON	OFF	ON	ON	510
CPU Diag. Boot TM11	OFF	ON	OFF	ON	OFF	ON	OFF	ON	524
Boot TM11 (without Diag.)	OFF	ON	OFF	ON	OFF	ON	OFF	OFF	526
CPU Diag. Boot TA11	OFF	OFF	ON	ON	OFF	ON	OFF	ON	624
Boot TA11 (without Diag.)	OFF	OFF	ON	ON	OFF	ON	OFF	OFF	626
CPU Diag. Boot RX11	OFF	OFF	ON	ON	OFF	OFF	OFF	OFF	636
Boot RX11 (without Diag.)	OFF	OFF	ON	OFF	ON	ON	ON	ON	640
CPU Diag. Boot DL11	OFF	OFF	ON	OFF	ON	OFF	ON	ON	650
Boot DL11 (without Diag.)	OFF	OFF	ON	OFF	ON	OFF	ON	OFF	652
CPU Diag. Boot PC11	OFF	OFF	ON	OFF	OFF	ON	ON	ON	660
Boot PC11 (without Diag.)	OFF	OFF	ON	OFF	OFF	ON	ON	OFF	662

Note: ON = Logic 0, OFF = Logic 1

7.5.1 Primary CPU Tests

The primary CPU tests exercise all unary and double operand instructions with all source modes. These tests do not modify memory. If a failure is detected, a branch-self (BR.) will be executed. The run light will stay on, because the processor will hang in a loop, but there will be no register display. The user must use the halt switch to exit from the loop. If no failure is detected in tests 1–5, the processor will emerge from the last test and enter the register display routine portion of the console emulator program.

TEST 1 – SINGLE OPERAND TEST

This test executes all single operand instructions using destination mode 0. The basic objective is to verify that all single operand instructions operate; it also provides a cursory check on the operation of each instruction, while ensuring that the CPU decodes each instruction in the correct manner.

Test 1 tests the destination register in its three possible states: zero, negative, and positive. Each instruction operates on the register contents in one of four ways:

1. Data will be changed via a direct operation, i.e., increment, clear, decrement, etc.

2. Data will be changed via an indirect operation, i.e., arithmetic shifts, add carry, and subtract carry.

3. Data will be unchanged but operated upon via a direct operation, i.e., clear a register already containing zeros.

4. Data will be unchanged but examined via a non-modifying instruction (TEST).

NOTE

When operating upon data in an indirect manner, the data is modified by the state of the appropriate condition code. Arithmetic shift will move the C bit into or out of the destination. This operation, when performed correctly, implies that the C bit was set correctly by the previous instruction. There are no checks on the data integrity prior to the end of the test. However, a check is made on the end result of the data manipulation. A correct result implies that all instructions manipulated the data in the correct way. If the data is incorrect, the program will hang in a program loop until the machine is halted.

TEST 2 - DOUBLE OPERAND, ALL SOURCE MODES

This test verifies all double operand, general, and logical instructions, each in one of the seven addressing modes (excludes mode 0). Thus, two operations are checked: the correct decoding of each double operand instruction, and the correct operation of each addressing mode for the source operand.

Each instruction in the test must operate correctly in order for the next instruction to operate. This interdependence is carried through to the last instruction (bit test) where, only through the correct execution of all previous instructions, a data field is examined for a specific bit configuration. Thus, each instruction prior to the last serves to set up the pointer to the test data.

Two checks on instruction operation are made in test 2. One check, a branch on condition, is made following the compare instruction, while the second is made as the last instruction in the test sequence.

Since the GO-NO GO test resides in a ROM memory, all data manipulation (modification) must be performed in destination mode 0 (register contains data). The data and addressing constants used by test 2 are contained within the ROM.

It is important to note that two different types of operations must execute correctly in order for this test to operate:

1. Those instructions that participate in computing the final address of the data mask for the final bit test instruction.

2. Those instructions that manipulate the test data within the register to generate the expected bit pattern.

Detection of an error within this test results in a program loop.

TEST 3 – JUMP TEST MODES 1, 2, AND 3
The purpose of this test is to ensure correct operation of the jump instruction. This test is constructed so that only a jump to the expected instruction will provide the correct pointer for the next instruction.

There are two possible failure modes that can occur in this test:

1. The jump addressing circuitry will malfunction causing a transfer of execution to an incorrect instruction sequence or non-existent memory.

2. The jump addressing circuitry will malfunction in such a way as to cause the CPU to loop.

The latter case is a logical error indicator. The former, however, may manifest itself as an after-the-fact error. For example, if the jump causes control to be given to other routines within the M9301, the interdependent instruction sequences would probably cause failure to eventually occur. In any case, the failing of the jump instruction will eventually cause an out of sequence or illogical event to occur. This in itself is a meaningful indicator of a malfunctioning CPU.

This test contains a jump mode 2 instruction, which is not compatible across the PDP-11 line. However, it will operate on any PDP-11 within this test, due to the unique programming of the instruction within test 3. Before illustrating the operation, it is important to understand the differences of the jump mode 2 between machines.

On the PDP-11/20, 11/05, 11/15, and 11/10 processors, for the jump mode 2 [JMP(R)+], the register (R) is incremented by 2 prior to execution of the jump. On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70, (R) is used as the jump address and incremented by 2 after execution of the jump.

In order to deal with this incompatibility, JMP (R)+ is programmed with (R) pointing back on the jump itself. On 11/20, 11/05, 11/10, and 11/15 processors, execution of the instruction would cause (R) to be incremented to point to the following instruction, effectively continuing a normal execution sequence.

On PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, the use of the initial value of (R) will cause the jump to loop back on itself. However, correct operation of the autoincrement will move (R) to point to the next instruction following the initial jump. The jump will then be executed again. However, the destination address will be the next instruction in sequence.

TEST 4 – SINGLE OPERAND, NON-MODIFYING BYTE TEST
This test focuses on a unique single operand instruction, the TST. TST is a special case in the CPU execution flow since it is a non-modifying operation. Test 4 also tests the byte operation of this instruction. The TSTB instruction will be executed in mode 1 (register deferred) and mode 2 (register deferred, autoincrement).

The TSTB is programmed to operate on data which has a negative value most significant byte and a zero (not negative) least significant byte.

In order for this test to operate properly, the TSTB on the low byte must first be able to access the even addressed byte and then set the proper condition codes. The TSTB is then reexecuted with the autoincrement facility. After the autoincrement, the addressing register should be pointing to the high byte of the test data. Another TSTB is executed on what should be the high byte. The N bit of the condition codes should be set by this operation.

Correct execution of the last TSTB implies that the autoincrement recognized that a byte operation was requested, thereby only incrementing the address in the register by one, rather than two. If the correct condition code has not been set by the associated TSTB instruction, the program will loop.

TEST 5 – DOUBLE-OPERAND, NON-MODIFYING TEST
Two non-modifying, double-operand instructions are used in this test – the compare (CMP) and bit test (BIT). These two instructions operate on test data in source modes 1 and 4, and destination modes 2 and 4.

The BIT and CMP instructions will operate on data consisting of all ones (177777). Two separate fields of ones are used in order to utilize the compare instructions, and to provide a field large enough to handle the autoincrementing of the addressing register.

Since the compare instruction is executed on two fields containing the same data, the expected result is a true Z bit, indicating equality.

The BIT instruction will use a mask argument of all ones against another field of all ones. The expected result is a non-zero condition (Z).

Most failures will result in a one instruction loop.

On successful completion of test 5, the register display routine is enabled, provided the console emulator has been selected in the microswitches. This routine prints out the octal contents of the CPU registers R0, R4, SP, and old PC on the console terminal. This sequence will be followed by a prompt character ($) on the next line.

An example of a typical printout follows.

	XXXXXX	XXXXXX	XXXXXX	XXXXXX
$				
Prompt Character	R0	R4	R6 (Stack Pointer)	R5 (Old PC)

NOTES:

1. Where X signifies an octal number (0–7).
2. Whenever there is a power-up routine or the boot switch is released on PDP-11/04 and PDP-11/34 machines, the PC at this time will be stored in R5. The contents of R5 are then printed as the old PC shown in the example.
3. The prompting character string indicates that diagnostics have been run and the processor is operating.

7.5.2 Secondary CPU and Memory Tests
The secondary CPU tests modify memory and involve the use of the stack pointer. The JMP and JSR instructions and all destination modes are tested. If a failure is detected, these tests, unlike the primary tests, will execute a halt.

Secondary CPU and memory diagnostics are run immediately after test 5 when they have been evoked by means other than the console emulator, provided that the correct microswitches have been set. If the console emulator has been entered at the completion of test 5, the secondary CPU and memory diagnostics will be run when the appropriate boot command is given.

Note that the user can execute the secondary CPU and memory diagnostics without running a bootstrap program. A false boot command (an invalid device code followed by a carriage return) will cause execution of tests 6, 7, and 8 before the attempt to boot is made. If these tests are executed successfully, the device will be called but not found. The processor will trap to location 4, which has been set up by the M9301-YA. A new PC is obtained at location 4, causing control to return to the console emulator. The readings for the four registers are now available, and the old PC is the highest location in memory. A failure in one of the tests will, of course, cause a halt.

TEST 6 – DOUBLE OPERAND, MODIFYING BYTE TEST

The objective of this test is to verify that the double-operand, modifying instructions will operate in the byte mode. Test 6 contains three subtests:

1. Test source mode 2, destination mode 1, odd and even bytes
2. Test source mode 3, destination mode 2
3. Test source mode 0, destination mode 3, even byte.

The move byte (MOVB), bit clear byte (BICB), and bit set byte (BISB) are used within test 6 to verify the operation of the modifying double-operand functions.

Since modifying instructions are under test, memory must be used as a destination for the test data. Test 6 uses location 500 as a destination address. Later, in test 7 and the memory test, location 500 is used as the first available storage for the stack.

Note that since test 6 is a byte test, location 500 implies that both 500 and 501 are used for the byte tests (even and odd, respectively). Thus, in the word of data at 500, odd and even bytes are caused to be all 0s and then all 1s alternately throughout the test. Each byte is modifed independently of the other.

TEST 7 – JSR TEST

The JSR is the first test in the GO-NO GO sequence that utilizes the stack. The jump subroutine command (JSR) is executed in modes 1 and 6. After the JSR is executed, the subroutine which was given control will examine the stack to ensure that the correct data was placed in the correct stack location (500). The routine will also ensure that the link back register points to the correct address. Any errors detected in this test will result in a halt.

TEST 8 – MEMORY TEST

Although this test is intended to test both core and MOS memories, the data patterns used are designed to exercise the most taxing operation for MOS. Before the details of the test are described, it would be appropriate to discuss the assumptions placed upon the failure modes of the MOS technology.

This test is intended to check for two types of problems that may arise in the memory.

1. Solid element or sense amp failures

2. Addressing malfunctions external to the chip.

The simplest failure to detect is a solid read or write problem. If a cell fails to hold the appropriate data, it is expected that the memory test will easily detect this problem. In addition, the program attempts to saturate a chip in such a way as to cause marginal sense amp operation to manifest itself as a loss or pick-up of unexpected data. The 4K- × 1-chip used in the memory consists of a 64- × 64-matrix of MOS storage elements. Each 64-bit section is tied to a common sense amplifier. The objective of the program is to saturate the section with, at first all 0s and one 1-bit. This 1-bit is then floated through the chip. At the end, the data is complemented, and the test repeated.

For external addressing failures, it is assumed that if two or more locations are selected at the same time, and a write occurs, it is likely that both locations will assume the correct state. Thus, prior to writing any test data, the background data is checked to ensure that there was no crosstalk between any two locations. All failures will result in a program halt as do failures in tests 6 and 7.

NOTE

If the expected and received data are the same, it is highly probable that an intermittent failure has been detected (i.e., timing or margin problem). The reason the expected and received data can be identical is that the test program rereads the failing address after the initial non-compare is detected. Thus, a failure at CPU speed is detected, and indicated by the reading of the failing address on a single reference (not at speed) operation.

7.6 TROUBLESHOOTING THROUGH REGISTER DISPLAY

When a halt occurs, the user should reboot the system by pressing the BOOT/INIT switch. The registers R0, R4, R6, and R5 will be displayed on the terminal in that order.

R0	Expected data
R4	Received data
R6	Failing address (SP)
R5	Old PC

The diagnostic program in the M9301-YA will cause the processor to jump to one of four addresses: 165316, 165346, 165370, or 165534. The user should consult the diagnostic program listing to find the failing test and begin troubleshooting. Possible causes of the failure include bus errors, a bad M9301 module, and a bad CPU.

CHAPTER 8
M9301-YB

8.1 INTRODUCTION

The M9301-YB is designed specifically for PDP-11/04 and PDP-11/34 end user systems. The ROM routines include basic CPU and memory GO-NO GO diagnostics, a console emulator program, and a specific set of bootstrap programs. Figure 8-1 is a program memory map for the M9301-YB.

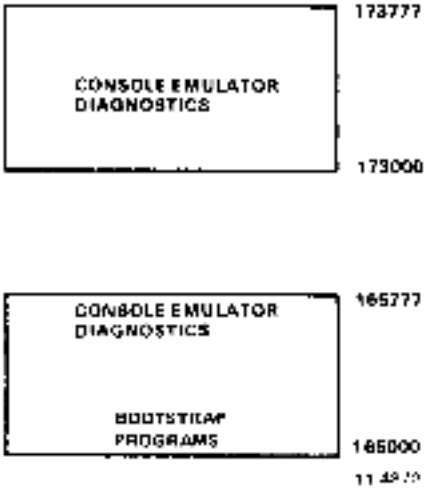

Figure 8-1 Program Memory Map for the M9301-YB

8.2 BOOTSTRAP PROGRAMS

The commands used to call the bootstrap programs from the console emulator are listed in Table 8-1.

An explanation of the functions performed by the various bootstrap programs follows.

RX11 Diskette Bootstrap – Loads the first 64 words (200_8 bytes) of data from track one, sector one into memory locations 0–176 beginning at location 0. Once loaded the content of location 0 is checked. If it contains 240, operation is transferred to the routine beginning in location 0. If location 0 does not contain 240, the boot is restarted. Restarts will occur 2000 times before the machine is halted automatically.

TA11 Cassette Bootstrap – This bootstrap is identical to that of the RX11 except that data is loaded from the cassette beginning at the second block.

Table 8-1 Bootstrap Routine Codes for the M9301-YB

Device	Description	Command
RK11	Disk Cartridge	DK
RP11	RP02/03 Disk Pack	DP
TC11	DECtape	DT
TM11	800 bpi Magtape	MT
TA11	Magnetic Cassette	CT
RX11	Diskette	DX
DL11	ASR-33 Teletype	TT
PC11	Paper Tape	PR
RJS03/04	Fixed Head Disk	DS
RJP04	Disk Pack	DB
TJU16	Magnetic Tape	MM

PC11 Paper Tape Reader and DL11 Bootstraps – Load an absolute loader formatted tape into the upper memory locations XXX746 to XXX777 (XXX is dependent on memory size). Once loading is completed, these boots transfer operation to a routine beginning at location XXX752.

Disk and DECtape Bootstraps (excluding RX11) – Load 1000_8 words (2000_8 bytes) of data from the device into memory locations 0–1776.

Magtape Bootstraps

TM11 – Loads second record (2000_8 bytes maximum size) from the magtape into memory location 0–1776_8.

TJU16 – Load second record (2000_8 bytes maximum size) from magtape into memory locations 0–1776. (Note that the first record contains the magtape directory.)

8.3 MICROSWITCH SETTINGS

A set of ten microswitches is located on the M9301 module. They determine which ROM routines are selected and give the user automatic access to any function.

The primary activating processes for the M9301-YB are the power-up sequence and the enabling of the console boot switch. Switch S1-1 (low ROM enable) must be in the ON position in order to enable activation of the M9301-YB console emulator or diagnostics. Switch S1-2 (POWER UP REBOOT ENABLE) must be on to activate the ROM on power-up. If switch S1-2 is off, then the processor will trap to location 24 (as normal) to execute the user power-up routine. When switch S1-2 is on, the other switches, S1-3 through S1-10, determine what action the M9301-YB will take on power-up.

If the system includes a console boot switch, then any time that switch is pressed the M9301-YB will be activated. Note that some processors will have to be halted for this switch to have any effect. Enabling the console boot switch causes the processor to enter a ROM routine by creating a fake power-down/power-up sequence. The user should note that the position of switch S1-2 is irrelevant when the console boot switch is used.

Pushing the console boot switch thus results in a normal boot on power-up sequence in the processor. Prior to the power-up sequence, the M9301-YB asserts 773000 on the Unibus address lines. This causes the new PC to be taken from ROM location 773024 instead of location 000024. The new PC will be the logical OR of the contents of ROM location 773024 and the eight microswitches on the M9301-YB module. A switch in the ON position is read as a 0. Likewise, a switch in the OFF position is a 1. In this way all the M9301-YB options are accessible.

Each option is given a different address. Note that microswitch S1-10 is ORed with bit 1 of the data in ROM location 773024. S1-9 is ORed with bit 2, etc. No switch is provided for combination with bit 0, because an odd address could result when going through the trap sequence. Figure 8-2 shows the relationship of the switches to the bus address bits.

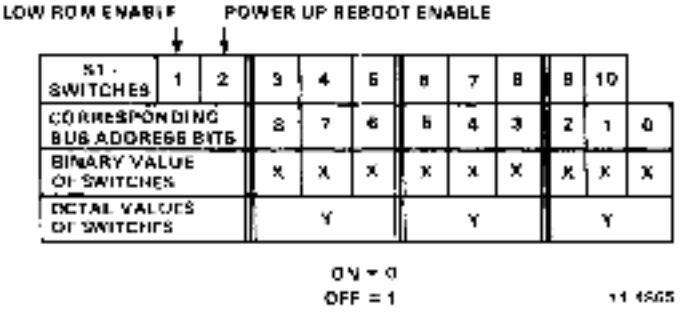

Figure 8-2 Offset Switches and Corresponding Bus Address Bits

The microswitches on the M9301-YB also enable the user to enter the console emulator, simply by pressing the boot switch.

Figure 8-3 shows a program flowchart for the M9301-YB. Notice that the choice and sequence of routines is entirely dependent on the offset switch settings.

Table 8-2 shows the options and corresponding switch settings and octal codes. The user should note that the M9301-YB does not provide for default booting from a peripheral.

Table 8-2 Options and Corresponding Switch Settings

Function (on Power-up)	S1-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	Octal Code
Vector through Location 24.	–	OFF	–	–	–	–	–	–	–	–	–
GO-NO GO Console Emulator	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	000
GO-NO GO Vector through Location 24.	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF	002
Console Emulator	ON	ON	ON	OFF	ON	ON	OFF	ON	OFF	OFF	226

NOTE

The above functions are the same for a boot initiated by the boot switch except that S1-2 may be ON or OFF.

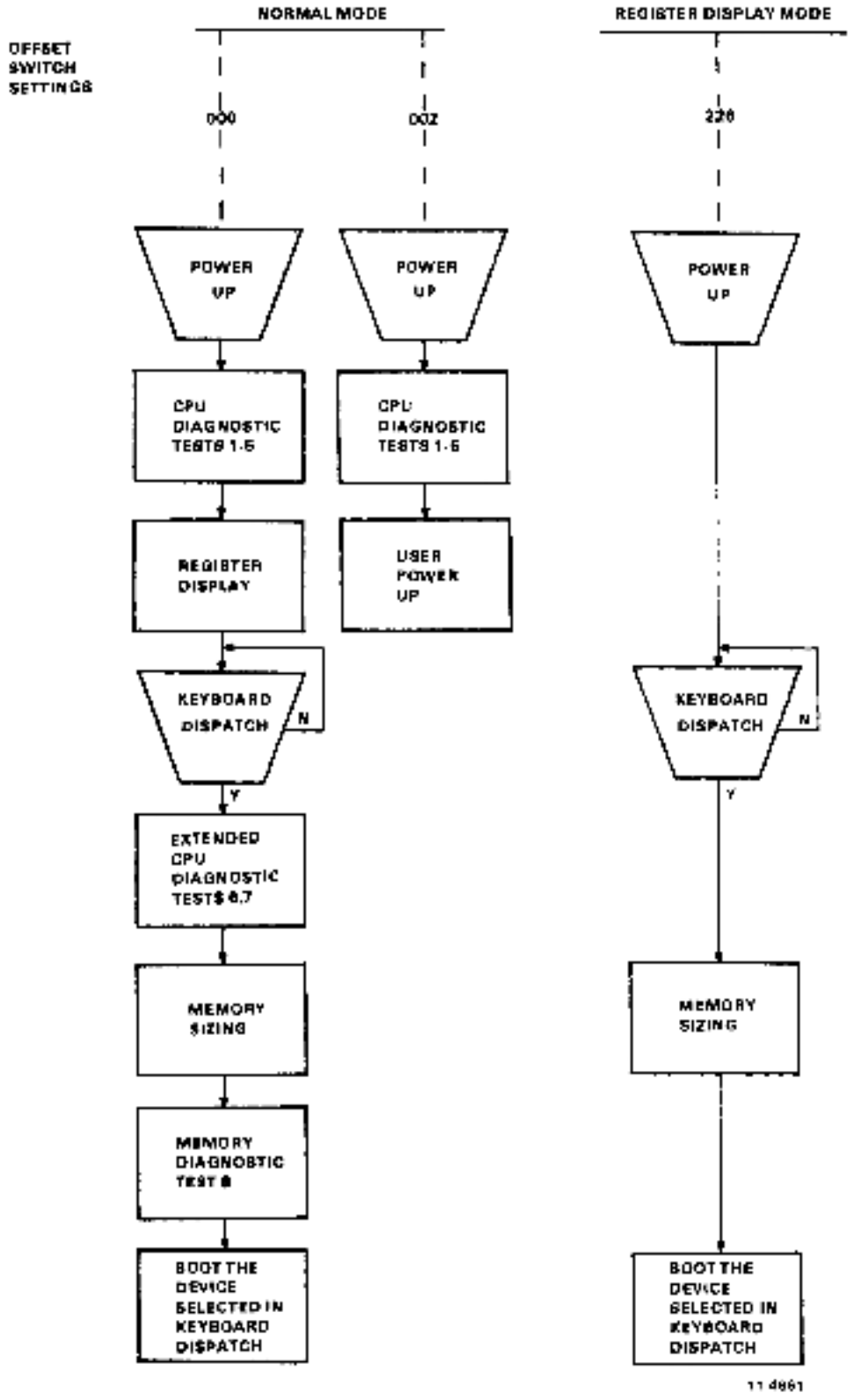

Figure 8-3 M9301-YB Program Flowchart

8.4 DIAGNOSTICS

An explanation of the eight CPU and memory diagnostic tests follows. Three types of tests are included in the M9301-YB diagnostics:

1. Primary CPU tests (1–5)
2. Secondary CPU tests (6, 7)
3. Memory test (8)

8.4.1 Primary CPU Tests

The primary CPU tests exercise all unary and double operand instructions with all source modes. These tests do not modify memory. If a failure is detected, a branch-self (BR.) will be executed. The run light will stay on, because the processor will hang in a loop, but there will be no register display. The user must use the halt switch to exit from the loop. If no failure is detected in tests 1–5, the processor will emerge from the last test and enter the register display routine (console emulator).

TEST 1 – SINGLE OPERAND TEST

This test executes all single operand instructions using destination mode 0. The basic objective is to verify that all single operand instructions operate; it also provides a cursory check on the operation of each instruction, while ensuring that the CPU decodes each instruction in the correct manner.

Test 1 tests the destination register in its three possible states: zero, negative, and positive. Each instruction operates on the register contents in one of four ways:

1. Data will be changed via a direct operation, i.e., increment, clear, decrement, etc.

2. Data will be changed via an indirect operation, i.e., arithmetic shifts, add carry, and subtract carry.

3. Data will be unchanged, but operated upon, via a direct operation, i.e., clear a register already containing zeros.

4. Data will be unchanged but examined via a non-modifying instruction (TEST).

NOTE

When operating upon data in an indirect manner, the data is modified by the state of the appropriate condition code. Arithmetic shift will move the C bit into or out of the destination. This operation, when performed correctly, implies that the C bit was set correctly by the previous instruction. There are no checks on the data integrity prior to the end of the test. However, a check is made on the end result of the data manipulation. A correct result implies that all instructions manipulated the data in the correct way. If the data is incorrect, the program will hang in a program loop until the machine is halted.

TEST 2 – DOUBLE OPERAND, ALL SOURCE MODES

This test verifies all double operand, general, and logical instructions, each in one of the seven addressing modes (excludes mode 0). Thus two operations are checked: the correct decoding of each double operand instruction, and the correct operation of each addressing mode for the source operand.

Each instruction in the test must operate correctly in order for the next instruction to operate. This interdependence is carried through to the last instruction (bit test) where, only through the correct execution of all previous instructions, a data field is examined for a specific bit configuration. Thus, each instruction prior to the last serves to set up the pointer to the test data.

Two checks on instruction operation are made in test 2. One check, a branch on condition, is made following the compare instruction, while the second is made as the last instruction in the test sequence.

Since the GO-NO GO test resides in ROM memory, all data manipulation (modification) must be performed in destination mode 0 (register contains data). The data and addressing constants used by test 2 are contained within the ROM.

It is important to note that two different types of operations must execute correctly in order for this test to operate:

1. Those instructions that participate in computing the final address of the data mask for the final bit test instruction.

2. Those instructions that manipulate the test data within the register to generate the expected bit pattern.

Detection of an error within this test results in a program loop.

TEST 3 – JUMP TEST MODES 1, 2, 3

The purpose of this test is to ensure correct operation of the jump instruction. This test is constructed such that only a jump to the expected instruction will provide the correct pointer for the next instruction.

There are two possible failure modes that can occur in this test:

1. The jump addressing circuitry will malfunction causing a transfer of execution to an incorrect instruction sequence or non-existent memory.

2. The jump addressing circuitry will malfunction in such a way as to cause the CPU to loop.

The latter case is a logical error indicator. The former, however, may manifest itself as an after-the-fact error. For example, if the jump causes control to be given to other routines within the M9301, the interdependent instruction sequences would probably cause a failure to eventually occur. In any case, the failing of the jump instruction will eventually cause an out of sequence or illogical event to occur. This in itself is a meaningful indicator of a malfunctioning CPU.

This test contains a jump mode 2 instruction which is not compatible across the PDP-11 line. However, it will operate on any PDP-11 within this test, due to the unique programming of the instruction within test 3. Before illustrating the operation, it is important to understand the differences of the jump mode 2 between machines.

On the PDP-11/20, 11/05, 11/15, and 11/10 processors, for the jump mode 2 [JMP(R)+], the register (R) is incremented by 2 prior to execution of the jump. On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70, (R) is used as the jump address and incremented by 2 after execution of the jump.

In order to avoid this incompatability, the JMP (R)+ is programmed with (R) pointing back on the jump itself. On 11/20, 11/05, 11/10, and 11/15 processors, execution of the instruction would cause (R) to be incremented to point to the following instruction, effectively continuing a normal execution sequence.

On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, the use of the initial value of (R) will cause the jump to loop back on itself. However, correct operation of the autoincrement will move (R) to point to the next instruction following the initial jump. The jump will then be executed again. However, the destination address will be the next instruction in sequence.

TEST 4 - SINGLE OPERAND, NON-MODIFYING BYTE TEST

This test focuses on one unique single operand instruction, the TST. TST is a special case in the CPU execution flow since it is a non-modifying operation. Test 4 also tests the byte operation of this instruction. The TSTB instruction will be executed in mode 1 (register deferred) and mode 2 (register deferred, autoincrement).

The TSTB is programmed to operate on data which has a negative value most significant byte and a zero (not negative) least significant byte.

In order for this test to operate properly, the TSTB on the low byte must first be able to access the even addressed byte and then set the proper condition codes. The TSTB is then reexecuted with the autoincrement facility. After the autoincrement, the addressing register should be pointing to the high byte of the test data. Another TSTB is executed on what should be the high byte. The N bit of the condition codes should be set by this operation.

Correct execution of the last TSTB implies that the autoincrement recognized that a byte operation was requested, thereby only incrementing the address in the register by one, rather than two. If the correct condition code has not been set by the associated TSTB instruction, the program will loop.

TEST 5 - DOUBLE-OPERAND, NON-MODIFYING TEST

Two non-modifying, double-operand instructions are used in this test - the compare (CMP) and bit test (BIT). These two instructions operate on test data in source modes 1 and 4, and destination modes 2 and 4.

The BIT and CMP instructions will operate on data consisting of all ones (177777). Two separate fields of ones are used in order to utilize the compare instructions, and to provide a field large enough to handle the autoincrementing of the addressing register.

Since the compare instruction is executed on two fields containing the same data, the expected result is a true Z bit, indicating equality.

The BIT instruction will use a mask argument of all ones against another field of all ones. The expected result is a non-zero condition (Z).

Most failures will result in a one instruction loop.

On successful completion of test 5, the register display routine is enabled, provided the console emulator has been selected in the microswitches. This routine prints out the octal contents of the CPU registers R0, R4, SP, and old PC on the console terminal. This sequence will be followed by a prompt character (S) on the next line.

An example of a typical printout follows.

	XXXXXX	XXXXXX	XXXXXX	XXXXXX
$				
Prompt Character	R0	R4	R6 (Stack Pointer)	R5 (Old PC)

NOTES:

1. Where X signifies an octal number (0–7).

2. Whenever there is a power-up routine or the boot switch is released on PDP-11/04 and PDP-11/34 machines, the PC at this time will be stored in R5. The contents of R5 are then printed as the old PC shown in the example.

3. The prompting character string indicates that diagnostics have been run and the processor is operating.

8.4.2 Secondary CPU and Memory Tests

The secondary CPU tests modify memory and involve the use of the stack pointer. The JMP and JSR instructions and all destination modes are tested. If a failure is detected, these tests, unlike the primary tests, will execute a halt.

Secondary CPU and memory diagnostics are run immediately after test 5 when they have been evoked by means other than the console emulator, provided that the correct microswitches have been set. If the console emulator has been entered at the completion of test 5, the secondary CPU and memory diagnostics will be run when the appropriate boot command is given.

Note that the user can execute the secondary CPU and memory diagnostics without running a bootstrap program. A false boot command (an invalid device code followed by a carriage return) will cause execution of tests 6, 7, and 8 before the attempt to boot is made. If these tests are executed successfully, the device will be called but not found. The processor will trap to location 4, which has been set up by the M9301-YB. A new PC is obtained at location 4, causing control to return to the console emulator. The readings of the four registers are now available, and the old PC is the highest location in memory. A failure in one of the tests will, of course, cause a halt.

TEST 6 – DOUBLE OPERAND, MODIFYING BYTE TEST

The objective of this test is to verify that the double-operand, modifying instructions will operate in the byte mode. Test 6 contains three subtests:

1. Test source mode 2, destination mode 1, odd and even bytes
2. Test source mode 3, destination mode 2
3. Test source mode 0, destination mode 3, even byte.

The move byte (MOVB), bit clear byte (BICB), and bit set byte (BISB) are used within test 6 to verify the operation of the modifying double-operand functions.

Since modifying instructions are under test, memory must be used as a destination for the test data. Test 6 uses location 500 as a destination address. Later, in test 7 and the memory test, location 500 is used as the first available storage for the stack.

Note that since test 6 is a byte test, location 500 implies that both 500 and 501 are used for the byte tests (even and odd, respectively). Thus, in the word of data at 500, odd and even bytes are caused to be all 0s and then all 1s alternately throughout the test. Each byte is modified independently of the other.

TEST 7 – JSR TEST
The JSR is the first test in the GO-NO GO sequence that utilizes the stack. The jump subroutine command (JSR) is executed in modes 1 and 6. After the JSR is executed, the subroutine which was given control will examine the stack to ensure that the correct data was placed in the correct stack location (500). The routine will also ensure that the line back register points to the correct address. Any errors detected in this test will result in a halt.

TEST 8 – MEMORY TEST
Although this test is intended to test both core and MOS memories, the data patterns used are designed to exercise the most taxing operation for MOS. Before the details of the test are described, it would be appropriate to discuss the assumptions placed upon the failure modes of the MOS technology.

This test is intended to check for two types of problems that may arise in the memory.

1. Solid element or sense amp failures
2. Addressing malfunctions external to the chip.

The simplest failure to detect is a solid read or write problem. If a cell fails to hold the appropriate data, it is expected that the memory test will easily detect this problem. In addition, the program attempts to saturate a chip in such a way as to cause marginal sense amp operation to manifest itself as a loss or pick-up of unexpected data. The 4K- × 1-chip used in the memory consists of a 64- × 64- matrix of MOS storage elements. Each 64-bit section is tied to a common sense amplifier. The objective of the program is to saturate the section with, at first all 0s and one 1-bit. This 1-bit is then floated through the chip. At the end, the data is complemented, and the test repeated.

For external addressing failures, it is assumed that if two or more locations are selected at the same time, and a write occurs, it is likely that both locations will assume the correct state. Thus, prior to writing any test data, the background data is checked to ensure that there was no crosstalk between any two locations. All failures will result in a program halt as do failures in tests 6 and 7.

NOTE

If the expected and received data are the same, it is highly probable that an intermittent failure has been detected (i.e., timing or margin problem). The reason the expected and received data can be identical is that the test program rereads the failing address after the initial non-compare is detected. Thus, a failure at CPU speed is detected, and indicated by the reading of the failing address on a single reference (not at speed) operation.

8.5 TROUBLESHOOTING THROUGH REGISTER DISPLAY

When a halt occurs, the user should reboot the system by pressing the BOOT/INIT switch. The registers R0, R4, R6, and R5 will be displayed on the terminal in that order.

R0	Expected data
R4	Received data
R6	Failing address (SP)
R5	Old PC

The diagnostic program in the M9301-YB will cause the processor to jump to one of four addresses: 165316, 165346, 165370, or 165534. The user should consult the diagnostic program listing to find the failing test and begin troubleshooting. Possible causes of the failure include bus errors, a bad M9301 module, and a bad CPU.

CHAPTER 9
M9301-YC

9.1 INTRODUCTION

The M9301-YC is used primarily in the PDP-11/70. It contains basic CPU, cache, and memory diagnostics. It can boot from any one of 8 devices and to any one of the 16 lowest 32K banks of memory (0–512K).

The PDP-11/70 requires pull-up resistors on the bus grant lines, so W1 through W5 must be installed. Refer to ECO M9301-00005 to determine if W6 must be installed.

The ROM routine causes a device code, drive number, and desired bank to be obtained from the console switch register after load and start. If the SWR is 0, then device code and drive number are obtained from the M9301 microswitches (32K bank 0 is used). For instructions on how to start, refer to Paragraph 9.6.

Figure 9-1 shows a memory map for the M9301-YC ROMs. MAINDEC-11-DEKBH is a listing of the code. A copy of this may be found in the *PDP-11/70 Systems Manual*.

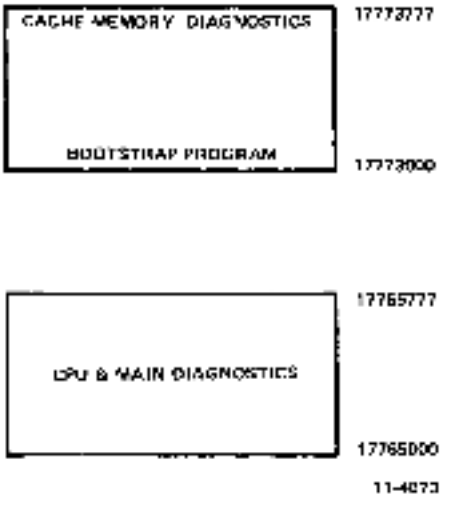

Figure 9-1 Program Memory Map for the M9301-YC

9.2 DIAGNOSTIC TEST EXPLANATION

The diagnostic portion of the program will test the basic CPU, including the branches, the registers, all addressing modes, and most of the instructions in the PDP-11 repertoire. It will then set the stack pointer to kernel D-space PAR 7. It will also turn on, if requested, memory management and the Unibus map, and will check memory from virtual address 100 to 157776. After main memory has been verified, with the cache off, the cache memory will be tested to verify that hits occur properly. Main memory will be scanned again to ensure that the cache is working properly throughout the 28K of memory to be used in the boot operation.

If one of the cache memory tests fails, the operator can attempt to boot the system anyway by pressing continue. This will cause the program to force misses in both groups of the cache before going to the bootstrap section of the program.

A listing of the M9301-YC diagnostic tests follows.

TEST 1	This test verifies the unconditional branch
TEST 2	Test CLR, MODE 0, and BMI, BVS, BHI, BLOS
TEST 3	Test DEC, MODE 0, and BPL, BEQ, BGE, BGT, BLE
TEST 4	Test ROR, MODE 0, and BVC, BHIS, BHI, BNE
TEST 5	Test BIH, BLT, and BLOS
TEST 6	Test BLE and BGT
TEST 7	Test register data path and modes 2, 3, 6
TEST 10	Test ROL, BCC, BLT, and MODE 6
TEST 11	Test ADD, INC, COM, and BCS, BLE
TEST 12	Test ROR, BIS, ADD, and BLO, BGE
TEST 13	Test DEC and BLOS, BLT
TEST 14	Test COM, BIC, and BGT, BGE, BLE
TEST 15	Test ADC, CMP, BIT, and BNE, BGT, BEQ
TEST 16	Test MOVB, SOB, CLR, TST and BPL, BNE
TEST 17	Test ASR, ASL
TEST 20	Test ASH, AND SWAB
TEST 21	Test 16 KERNEL PARs
TEST 22	Test and load KIPDRs
TEST 23	Test JSR, RTS, RTI, and JMP
TEST 24	Load and turn on memory management and the Unibus map
TEST 25	Test main memory from virtual 100 to 28K
TEST 26	Test cache data memory
TEST 27	Test virtual 28K with cache on

9.3 DIAGNOSTIC TEST DESCRIPTIONS

TEST 1 - This test verifies the unconditional branch.
The registers and condition codes are all undefined when this test is entered and they should remain that way upon the completion of this test.

TEST 2 -- Test CLR, MODE 0, and BMI, BVS, BHI, BLOS.
The registers and condition codes are all undefined when this test is entered. Upon completion of this test the SP (R6) should be zero and only the Z flip-flop will be set.

TEST 3 – Test DEC, MODE 0, and BPL, BGE, BGT, BLE.
Upon entering this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 000000.
Upon completion of this test the condition codes will be: N = 1, Z = 0, V = 0, and C = 0.
The registers affected by the test are: SP = 177777.

TEST 4 - Test ROR, MODE 0, and BVC, BHIS, BHI, BNE.
Upon entering this test the condition codes are: N = 1, Z = 0, V = 0, and C = 0.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 177777.
Upon completion of this test the condition codes will be: N = 0, Z = 0, V = 1, and C = 1.
The registers affected by the test are: SP = 077777.

TEST 5 - Test BH1, BLT, and BLOS.
Upon entering this test the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 077777.
Upon completion of this test the condition codes will be: N = 1, Z = 1, V = 1, and C = 1.
The registers are unaffected by the test.

TEST 6 - Test BLE and BGT.
Upon entering this test the condition codes are: N = 1, Z = 1, V = 1, and C = 1.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 077777.
Upon completion of this test the condition codes will be: N = 1, Z = 0, V = 1, and C = 1.
The registers are unaffected by the test.

TEST 7 - Test register data path and modes 2, 3, 6.
When this test is entered the condition codes are: N = 1, Z = 0, V = 1, and C = 1.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 077777.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are left as follows: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 125252, R4 = 125252, R5 = 125252. SP = 1225252. and MAPL00 = 125252.

TEST 10 - Test ROL. BCC. BLT, and MODE 6.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 125252, R4 = 125252, R5 = 125252, SP = 125252, and MAPL00 = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are left unchanged except for MAPL00 which should now equal 052524.

TEST 11 - Test ADD, INC, COM, and BCS, BLE.
When this test is entered the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 125252, R4 = 125252, R5 = 125252, SP = 125252, and MAPL00 = 052524.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are left unchanged except for R3 which now equals 000000, and R1 which is also 000000.

TEST 12 - Test ROR, BIS, ADD, and BLO, BGE.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 030000, R4 = 125252, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are left unchanged except for R3 which should be modified back to 000000, and R4 which should now equal 052525.

TEST 13 - Test DEC and BLOS, BLT.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 1, Z = 0, V = 0, and C = 0.
The registers are left unchanged except for R1 which should now equal 177777.

TEST 14 – Test COM, BIC, and BGT, BLE.
When this test is entered the condition codes are: N = 1, Z = 0, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 177777, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are left unchanged except for R0 which should now equal 052525, and R1 which should now equal 052524.

TEST 15 – Test ADC, CMP, BIT, and BNE, BGT, BEQ.
When this test is entered the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are: R0 = 052525, R1 = 052524, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are now: R0 = 052525, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.

TEST 16 – Test MOVB, SOB, CLR, TST and BPL, BNE.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 052525, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
R0 is decremented by an SOB instruction to 000000; R1 is cleared and then incremented around to 000000.

TEST 17 – Test ASR, ASL.
When this test is entered the condition codes are:: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 0, V = 0, and C = 0.
The registers are left unchanged except for R0 which is now equal to 000000, R1 which is now 000001, and R2 which is now 000000.

TEST 20 – Test ASH, and SWAB.
When this test is entered the condition codes are: N = 0, Z = 0, V = 0, and C = 0.
The registers are: R0 = 000000, R1 = 000001, R2 = 000000, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 1.
The registers are left unchanged except for R1 which should now equal 000000.

TEST 21 – Test 16 KERNEL PARs.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 1.
The registers are: R0 = 000000, R1 = 000000, R2 = 000000, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 00, and C = 0.
The registers now equal: R0 = 172400, R1 = 000000, R2 = 000000, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
All KERNEL PARs = 125252.

TEST 22 – Test and load KIPDRs.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 172400, R1 = 000000, R2 = 000000, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers that are modified are: R0 = 172300, R1 = 000000, and R2 = 077406.
All KERNEL I-SPACE PDRs (172300 – 172316) = 077406.

TEST 23 – Test JSR, RTS, RTI, JMP.
This test first sets the stack pointer to KDPAR7 (172376), and then verifies that JSR, RTS, RTI, and JMP all work properly.

On entry to this test the stack pointer (SP) is initialized to 172376 and is left that way on exit.

TEST 24 – Load and turn on memory management and the Unibus mapp.
This test is only executed if the upper 4 bits (<15:12>) of the switch register are non-zero. The test will load memory management to relocate to the 32K block number specified. It will also set up the Unibus addresses correctly. (If bits <15:12> specify block number 3, the user should boot into memory from 96K to 128K. The KIPARs will be loaded as follows: KIPAR0 = 006000, KIPAR1 = 006200, KIPAR2 = 006400, KIPAR3 = 006600, KIPAR4 = 007000, KIPAR5 = 007200, KIPAR6 = 007400. KIPAR7 will always equal 177600.)
The Unibus map registers will then be set as follows: MAPL0 = 000000, MAPH0 = 03, MAPL1 = 020000, MAPH1 = 03, MAPL2 = 040000, MAPH2 = 03, MAPL3 = 060000, MAPH3 = 03, MAPL4 = 100000, MAPH4 = 03, MAPL5 = 120000, MAPH5 = 03, MAPL6 = 140000, MAPH6 = 03.

TEST 25 – Test main memory from virtual 1000 to 28K.
This test will test main memory with the cache disabled, from virtual address 001000 to 157776. If the data does not compare properly, the test will halt at either 165740 or 165776. If a parity error occurs, the test will halt at address 165776, with PC + 2 on the stack which is in the KERNEL D-SPACE PARs.

In this test the registers are initialized as follows: R0 = 001000, R1 = DATA READ, R2 = 067400, R3 = 0001000, R4 = 067400, R5 = 177746 (control register), SP = 172376.

The following two tests are cache memory tests. If either fails to run successfully, the program will come to a halt in the M9301 ROM. If you desire to try to boot your system or diagnostic, you can press continue and the program will force misses in both groups of the cache and go to the bootstrap that has been selected.

TEST 26 – Test cache data memory.
This test will check the data memory in the cache, first group 0 and then group 1. It loads 052525 into an address, complements it twice and then reads the data. It then checks that address to ensure the data was a bit. The sequence is repeated on the same address with 125252 as the data. All cache memory data locations are tested in this way. If either group fails and the operator presses continue, the program will try to boot with the cache disabled.

The registers are initialized as follows for this test: R0 = 001000 (address), R1 = 000002 (count), R2 = 001000 (count), R3 = 001000 (count), R4 = 125252 (pattern), R5 = 177746 (control register), SP = 172374 (flag of zero pushed on stack).

TEST 27 – Test virtual 28K with cache on.
This test checks virtual memory from 001000 through 157776 to ensure that you can get hits all the way up through main memory. It starts with group 1 enabled, then tests group 0, and finally checks memory with both groups enabled. If any one of the three passes fail, the test will halt at CONT + 2. Then if the operator presses continue, the program will try to boot with the cache disabled.

Upon entry the registers will be set up as follows: R0 = 001000 (address), R1 = 000003 (pass count), R2 = 067400 (memory counter), R3 = 001000 (first address), R4 = 067400 (memory counter), R5 = 177746 (control register), SP = 172374 (pointing to code for control register).

Upon completion of this test, main memory from virtual address 001000 through 157776 will contain its own virtual address.

9.4 PDP-11/70 BOOTSTRAP

The bootstrap portion of the program (beginning at 17773000) looks at the lower byte of the console switch register to determine which one of eight devices and which one of eight drives to boot from. Switches <02:00> select the drive number (0–7), and switches <07:03> select the device code (1–11) to be used. If the lower byte of the switch register is zero, the program will read the set of microswitches on the M9301 to determine the device and drive number. These switches can be set to select a default boot device (provided the console SWR is 0).

If the bootstrap operation fails as a result of a hardware error in the peripheral device, the program will do a RESET instruction, jump back to the test that sets up memory management, and will attempt to boot again.

9.5 MICROSWITCHES

Switch S1-1 is the low ROM enable switch. It is used to inhibit (when OFF) the M9301-YC from responding to addresses 17765000–17765777. Switch S1-2 is the power-up reboot enable switch. It enables the M9301-YC ROM on power-up. Switches S1-3 through S1-10 correspond to bus data bits 8–1 respectively. They are read when location 17773024 is accessed (Figure 9-2).

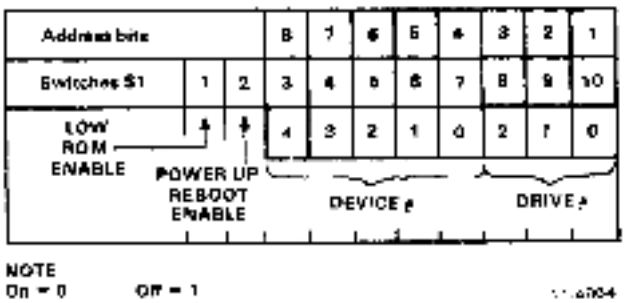

Figure 9-2 Microswitch Assignment

Since the external boot switch and boot on power-up capabilities are available on the M9301-YC (ECO M9301-00005), if all required ECOs are included and microswitch S1-2 is ON, the PDP-11/70 will boot (dependent on console switch register and/or the microswitches) on a power-up. If booting on power-up is not desired, switch S1-2 should be OFF.

If the operator wishes to load address 17765000 and start (or boot on power-up), switch S1-1 should be on. However, some peripheral devices in the system may require bus addresses in the range of 17765000–17765777. In this case switch S1-1 must be OFF. With switch S1-1 OFF, the M9301-YC must be started at location 17773000 or an SSYN time-out will occur. Microswitches S1-3 through S1-10 are used whenever 17765000 or 17773000 is loaded and the processor is started from the console, and the console switches are set equal to 0 (all ON). Microswitches S1-8 through S1-10 correspond to drive numbers. Microswitches S1-3 through S1-7 correspond to device codes. For example, the RK05 drive 1 is selected through the following configuration:

S1-1	S1-2	S1-3	S1-4	S1-5	S1-6	S1-7	S1-8	S1-9	S1-10
ON/OFF	ON/OFF	ON	OFF	OFF	ON	OFF	ON	ON	OFF

Note that only eight devices can be selected (1–4 and 6–11). Device code 5 is reserved. Use of this code will result in errors.

Figure 9-3 shows a program flowchart for the M9301-YC. Notice that the choice of routines is determined by jumpers and microswitch settings as well as console switch register settings.

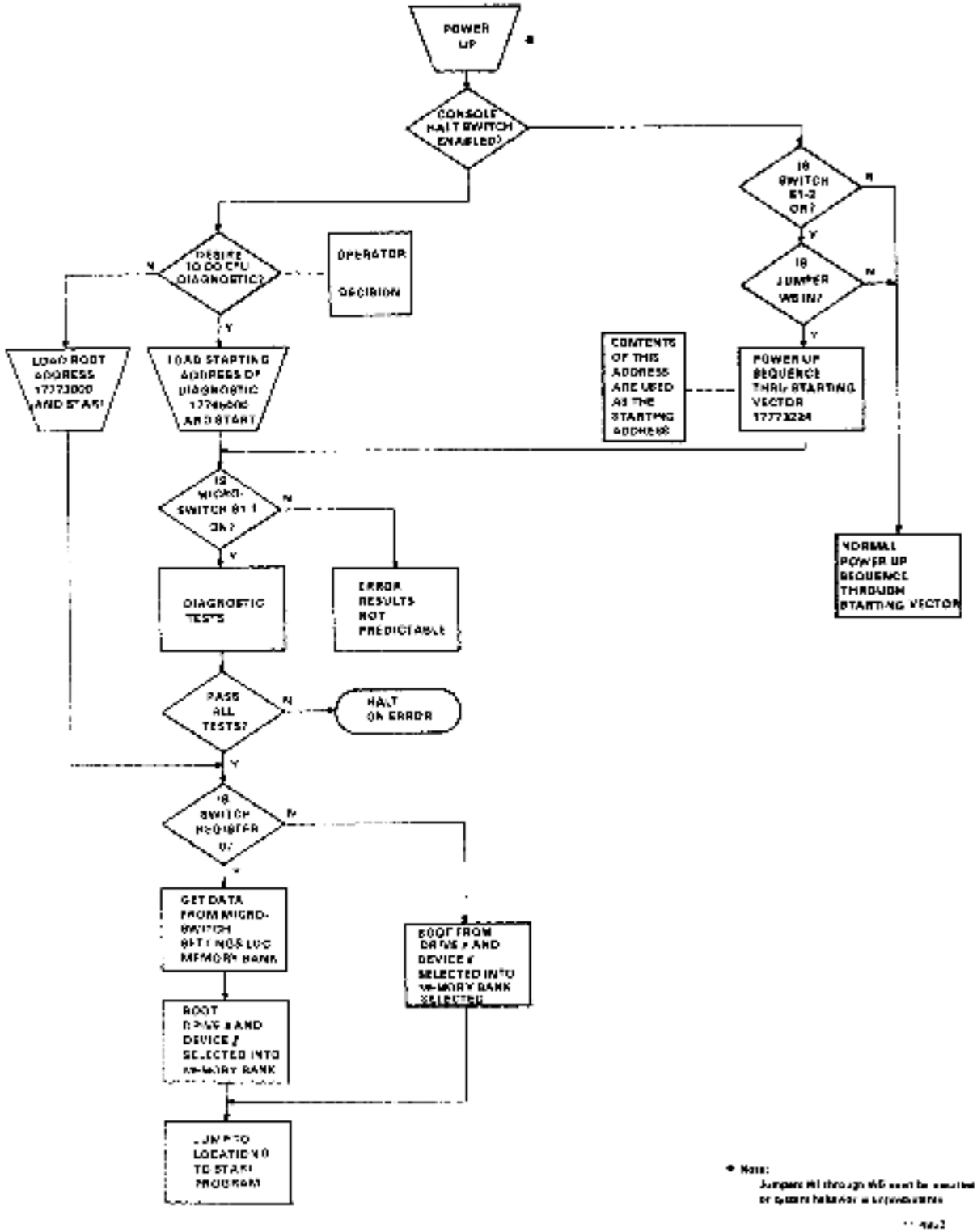

Figure 9-3 M9301-YC Program Flowchart

9.6 STARTING PROCEDURE

9.6.1 Console Switch Settings

The lower byte of the console switch register should be set to have the drive number (0-7) in switches <02:00>, and the device code (1–11) in switches <07:03>.

The upper byte of the switch register should be set to have the bank number of the 32K block of memory for the bootstrap operation (0–17) in switches <15:12>.

The memory blocks are as follows:

0 Physical Memory 0–28K
1 Physical Memory 32K–60K
2 Physical Memory 64K–92K
3 Physical Memory 96K–124K
4 Physical Memory 128K–156K
.
.
10 Physical Memory 256K–284K
.
14 Physical Memory 384K–412K
15 Physical Memory 416K–444K
16 Physical Memory 448K–476K
17 Physical Memory 480K–508K

9.6.2 Starting Addresses

The normal starting address for the diagnostic and boot routine is 17765000 (microswitch S1-1 must be ON). If the diagnostic portion of this program fails and the operator wants to attempt to boot anyway, he must follow these steps:

1. Set up memory management if booting into other than the lower 28K of memory. BUS INIT (console start) initializes memory management to 0–28K of main memory.

2a. If the device is on the Massbus, set stack pointer to a valid address and load that address with the memory bank number put into switches <15:12>.

2b. If the device is on the Unibus, set up Unibus map registers 0 through 6 to map to the same memory as memory management. BUS INIT (console start) initializes the map to 0–128K of main memory.

3. Deposit address 173000 into the PC.

4. Set the device code and drive number in the lower byte of the switch register.

5. Press continue.

Examples:

A. RP04 – Set stack pointer to 40000,
load 000000 into address 40000,
load 173000 into the PC (17777707),
set 000070 into switches (RP04 drive 0),
press continue.

B. RK05 – Load 173000 into the PC (17777707),
set 000030 into switches (RK05 drive 0),
press continue.

9.6.3 Operator Action

If the diagnostic portion of the ROM routine fails, record PC of the instruction and refer to the listing to find out which test in the program failed.

9.7 ERRORS

Table 9-1 is a list of error halts indexed by the address displayed.

Table 9-1 Error Halts

Address Displayed	Test Number and Subsystem Under Test
17765004	Test 1, Branch Test
17765020	Test 2, Branch Test
17765036	Test 3, Branch Test
17765052	Test 4, Branch Test
17765066	Test 5, Branch Test
17765076	Test 6, Branch Test
17765134	Test 7, Register Data Path Test
17765146	Test 10, Branch Test
17765166	Test 11, CPU Instruction Test
17765204	Test 112, CPU Instruction Test
17765214	Test 13, CPU Instruction Test
17765222	Test 14, COM Instruction Test
17765236	Test 14, CPU Instruction Test
17765260	Test 15, CPU Instruction Test
17765270	Test 16, Branch Test
17765312	Test 16, CPU Instruction Test
17765346	Test 17, CPU Instruction Test
17765360	Test 20, CPU Instruction Test
17765374	Test 20, CPU Instruction Test
17765450	Test 21, KERNAL PAR Test
17765474	Test 22, KERNAL PAR Test
17765510	Test 23, JSR Test
17765520	Test 23, JSR Test
17765530	Test 23, RTS Test
17765542	Test 23, RTI Test
17765550	Test 23, JMP Test
17765760	Test 25, Main Memory Data Compare Error
17766000	Test 25, Main Memory Parity Error (no recovery possible from this error)
17773644	Test 26, Cache Memory Data Compare Error
17773654	Test 26, Cache Memory No Hit (Pressing continue here will cause boot attempt forcing misses)
17773736	Test 27, Cache Memory Data Compare Error
17773746	Test 27, Cache Memory No Hit (Pressing continue here will cause boot attempt forcing misses)
17773764	Test 25 OR 26, Cache Memory Parity Error (Pressing continue here will cause boot attempt forcing misses)

Error Recovery

Most of the error halts are hard failures (there is no recovery). If main memory halts are the fault, try booting to another bank (if the program is relocatable) or swap modules from another bank with bank 0.

If the processor halts in one of the two cache tests, the error can be recovered from. Press continue to finish the test (if at either 17773644 or 17773736) or force misses in both groups of the cache and attempt to boot the system monitor with the cache disabled (if at either 17773654, 17773746, or 17773764).

If this program fails in an uncontrolled manner, it might be due to an unexpected trap to location 000004. If this is suspected, then load a 000006 into address 000004 and a 000000 into location 000006. This will cause all traps to location 000004 to halt with 00000010 in the address lights. The operator can then examine the CPU error register at location 17777766.

The bits in the CPU error register are defined as follows:

Bit 02 = Red zone stack limit
Bit 03 = Yellow zone limit
Bit 04 = Unibus time-out
Bit 05 = Nonexistent memory (cache)
Bit 06 = Odd address error
Bit 07 = Illegal halt

CHAPTER 10
M9301-YD

10.1 INTRODUCTION

The M9301-YD is programmed to provide transparent pass-through of data between a terminal on a satellite computer and an asynchronous serial line on the same computer. It also contains all of the necessary instructions for requesting a secondary mode program load and accepting a down-line load on its serial line from another machine using DDCMP protocol. These features enable a PDP-11 computer to be a satellite in a REMOTE-11 system.

The M9301-YD ROM will run on any PDP-11 computer. Two DL11s are required with the following addresses and vectors:

Function	Address	Vector
Local Terminal	177560	60
Interprocessor	175610	400

The ROM may be activated by placing the starting address for the desired type of bootstrap in the switch register and pressing LOAD ADDRESS, START, or by utilizing the power-up bootstrap feature of the M9301-YD with proper setting of the address offset microswitches.

Figure 10-1 shows the relation of the microswitches to the starting address. Figure 10-2 is the M9301-YD power-up flowchart.

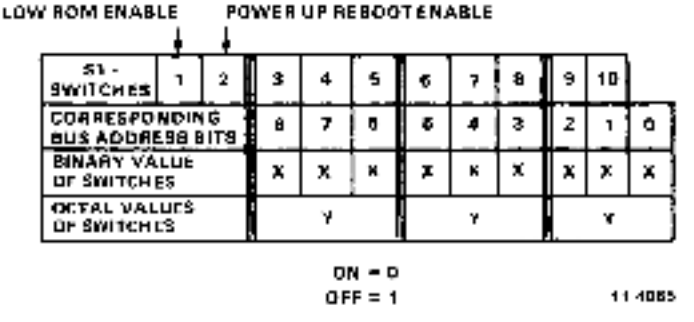

Figure 10-1 Offset Switches and Corresponding Address Bits

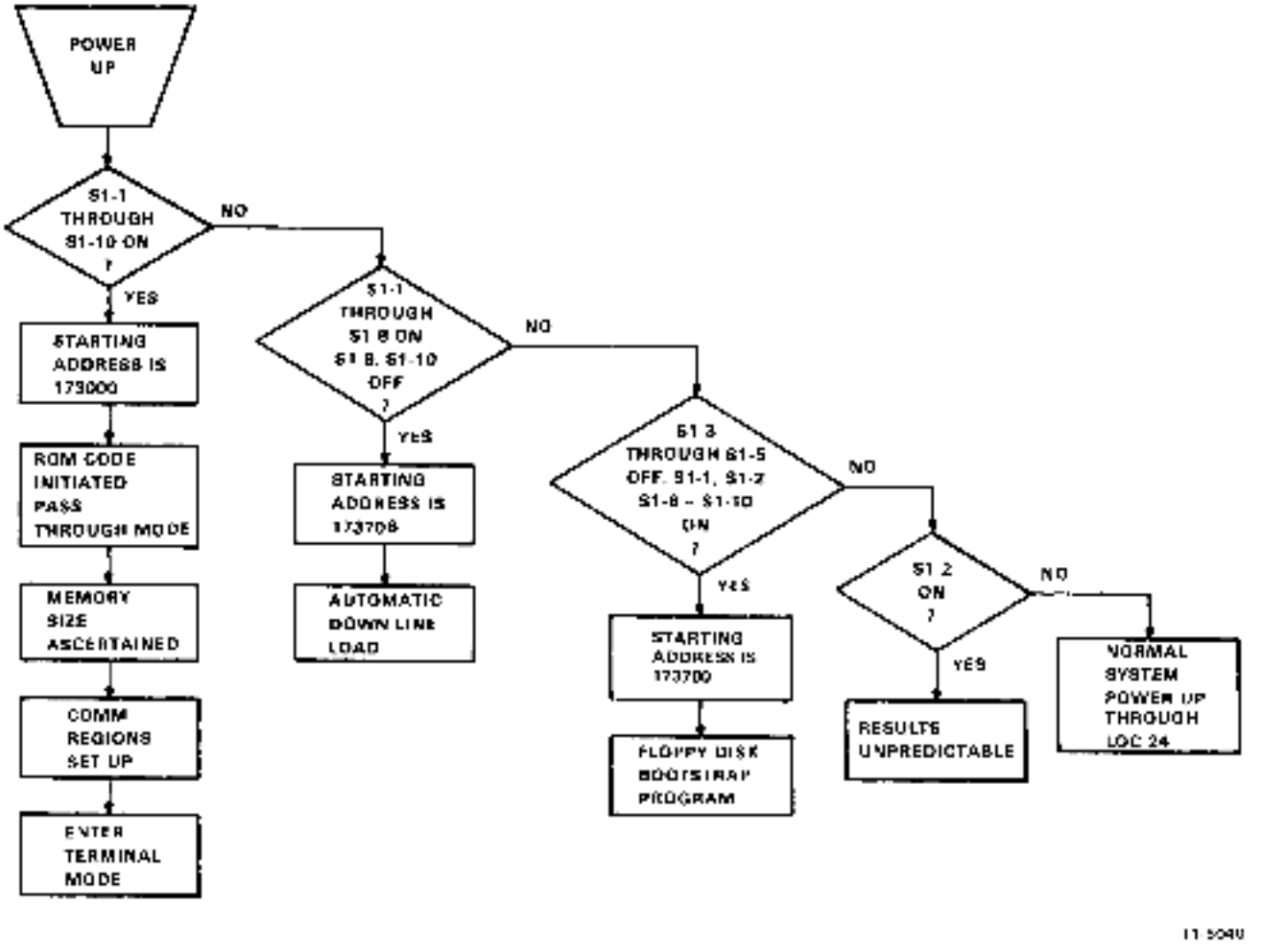

Figure 10-2 M9301-YD Program Flowchart

10.2 NORMAL BOOTSTRAP (173000)

The starting address is 173000. Switches S1-1 through S1-10 are on for power-up bootstrap. Ordinarily, when the ROM code is initiated, the satellite comes up in pass-through mode. First, the memory size of the computer is ascertained. Then, necessary communications regions are set up and the OSOP message "Enter terminal mode" is sent on the serial line. All further communication is in ASCII with the ROM program polling the ready flags in the four control and status registers to determine the next action. XON (Q) and XOFF (S) are supported within the satellite computer, and transmission is assumed to be full duplex.

The only way the terminal mode may be discontinued is through the receipt of a DDCMP message. The ROM accepts only the following messages: "Program load without transfer address," "Program load with transfer address," and "Enter terminal mode."

10.3 SECONDARY MODE BOOTSTRAP (173006)

Switches S1-1 through S1-8 are ON for use of the power-up bootstrap function. S1-9 and S1-10 are OFF. The alternate starting address 173006 causes an automatic down line load. The "Enter terminal mode" message is replaced by "Request secondary mode program load." The host machine should respond to this message with a program load so that terminal mode is only transiently activated. Bus addresses 177560–177566 will be addressed during this time. The secondary mode program load message may be used for loading programs into satellites without terminals or for loading a scroller into a machine with a graphics terminal. All other secondary mode actions are identical to those of the normal bootstrap.

10.4 FLOPPY BOOTSTRAP (173700)

The starting address 173700 should be used for initiating a boot load from a floppy disk. The switches should be set as follows: S1-1, S1-2 ON; S1-3 through S1-5 OFF; S1-6 through S1-10 ON.

10.5 SENDING DDCMP MESSAGES

A callable subroutine within the ROM will send DDCMP boot mode messages. These messages should be already loaded in memory. Location 162 is loaded by the M9301-YD with the address of this subroutine. To use it, call from the user program with:

```
CLR     R2              ;indicate no completion routine
MOV     #MSG,R5
JSR     PC,@162
```

where MSG has the following format:

Bytes 0 and 1 contain the number of bytes in the message
Byte 2 has an OSOP code
Bytes 3 through n have the data (if any).

The message is sent using interrupt driven code, but control is not returned to the user program until the message has been completely sent out.

10.6 RECEIVING DDCMP MESSAGES

A program which has been booted down line from the host to the satellite will automatically receive DDCMP messages if properly initialized and if the interrupt for the serial line receive register is enabled. The program listing which follows shows the routine which handles the messages. It should be included in the booted program.

```
MOV     @162,-(SP)      ;get RECDDC address
SUB     #62,(SP)
MOV     #-9.,R4         ;MSG size
CLR     R3              ;CRC initialize
CLR     R2              ;completion (move)
JSR     PC,@(SP)+       ;call RECDDC
BCS     ERROR
Add     #8.,R0          ;point to OSOP Code
```

To restart this receiving routine, execute JMP@164.

The M9301-YD uses locations 150–167 for communications and 170–171 for temporary data during program loads. It also loads location 54 with the high usable memory address. The address in 54 has been adjusted to leave room for buffers and a stack for use by the ROM.

CHAPTER 11
M9301-YE

11.1 INTRODUCTION

The M9301-YE provides DECnet bootstrapping capabilities for the PDP-11 system with or without the console switch register. In addition, the M9301-YE includes a console emulator routine, a paper tape bootstrap, and some basic CPU and memory GO-NO GO diagnostic tests.

This module has been designed for maximum flexibility of operation. Its function may be initiated in any of four ways: automatically at power-up, by pressing the console boot switch, by a LOAD ADDRESS and START sequence, or by using the console terminal while running the console emulator routine.

11.2 PROGRAM MEMORY MAP

Figure 11-1 is a program memory map for the M9301-YE.

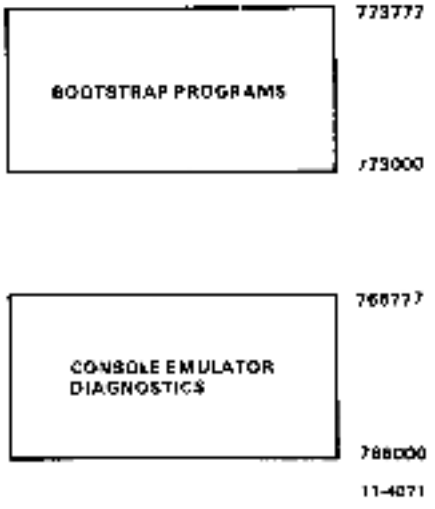

Figure 11-1 M9301-YE Program Memory Map

The lower addresses 765000–765777 can be disabled by putting switch S1-1 in the OFF position. This would free addresses 765000–765777, allowing the user to occupy this address space. Note that if switch S1-1 is OFF, the console emulator, the low-speed paper tape boot, and diagnostics are no longer available.

Note also that the code in the M9301-YE is written with the assumption that the processor will trap to Unibus location 24 on power-up if the default boot on power-up feature is to be used. It is also assumed that a power-up trap to Unibus location 24 will be invoked by the assertion and then negation of AC LO if the pushbutton boot feature is used.

11.3 MICROSWITCH FUNCTIONS

A set of ten microswitches is located on the M9301 module. They determine which ROM routines are selected and give the user automatic access to any function.

The primary activating processes for the M9301-YE are the power-up sequence and the enabling of the console boot switch. Switch S1-1 must be in the ON position in order to enable activation of the M9301-YE console emulator and diagnostics. Switch S1-2 is the power-up reboot enable switch. It must be ON to enable to M9301-YE on power-up. If switch S1-2 is OFF, then the processor will trap to location 24 (as normal) to execute the user power-up routine. When switch S1-2 is ON, the other switches, S1-3 through S1-10, determine what action the M9301-YE will take on power-up.

If the system supports a console boot switch, then any time that switch is pressed, the M9301-YE will be activated. Enabling the console boot switch causes the processor to enter a ROM routine by creating a fake power down/power-up sequence. The user should note that the position of switch S1-2 is irrelevant when the console boot switch is used. Pushing the console boot switch thus results in a power-up sequence in the processor through an M9301-YE ROM routine.

Prior to the power-up sequence, the M9301-YE asserts 773000 on the Unibus address lines. This causes the new PC to be taken from ROM location 773024 instead of location 000024. The new PC will be the logical OR of the contents of ROM location 773024 and the eight microswitches on the M9301-YE module. A switch in the ON position is read as a 0, and a switch in the OFF position is a 1. In this way all the M9301-YE options are accessible. Each option is given a different address. Note that microswitch S1-10 is ORed with bit 1 of the data in ROM location 773024; S1-9 is ORed with bit 2, etc. No switch is provided for combination with bit 0, because an odd address could result when going through the trap sequence.

The microswitches on the M9301-YE enable the user either to start a bootstrap operation or to enter the console emulator, simply by pressing the boot switch. Note that a momentary power failure will have the same effect as pushing the boot switch.

Figure 11-2 shows the relationship of the switches to the bus address bits.

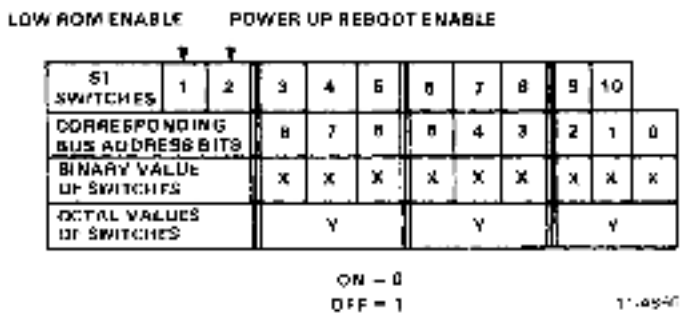

Figure 11-2 Offset Switches and Corresponding Bus Address Bits

11.4 BOOTSTRAPPING PAPER TAPE

The paper tape boot is unique in that it can do no error checking and that the secondary bootstrap (e.g., the absolute loader) is read into the upper part of memory. The actual locations loaded by the paper tape boot are partially determined by the secondary bootstrap itself and by the size routine which determines the highest available memory address within the first 28K. The paper tape boot transfers control to location XXX374, where XXX is determined initially by the size routine to be at the top of memory: this is where the absolute loader will have just been loaded.

Note that the only way to bootstrap a device other than unit 0 is by entering the console emulator to specify the drive number desired. Otherwise the bootstraps will default to unit 0. This means that only unit 0 of a device can be bootstrapped without operator intervention.

11.5 DECNET BOOTSTRAPPING: DU11, DL11-E, DUP11

The DECnet bootstrap routines load memory in a satellite computer over a communication link, using the maintenance operation protocol (MOP) within the DECnet system. The entire bootstrapping process involves three separate bootstrap programs.

11.5.1 Primary Bootstrap

The primary bootstrap is a mode associated with the communications device within the M9301-YE ROMs. It requests a secondary boot from the host computer. The primary bootstrap routine will wait approximately 9 seconds for a reply and then make another request for the secondary boot. If, after eight tries, the host has not responded, the satellite will hang up the phone for approximately one half second. Data terminal ready (DTR) is turned OFF and then ON again. After interrupting the communication link, the primary bootstrap will initiate another series of requests for a secondary boot. The M9301-YE supports two MOP codes, 8 and 0, as follows. The primary boot sends out a MOP 8 message. It expects a MOP 0 message back. The MOP 0 is the secondary boot which loads the tertiary boot with transfer address.

If the primary boot detects any errors, it will make the request again.

11.5.2 Secondary Bootstrap

The secondary bootstrap program, once loaded, requests a tertiary boot. These intermediate steps are important because the boot program which handles the actual loading of the operating system is too long and complex to be stored in the ROM.

11.5.3 Tertiary Bootstrap

Once the tertiary boot has been requested and loaded, it relocates in upper memory and then begins the down-line loading of the operating system. Figure 11-3 is a flowchart showing the relation of the three bootstrap programs.

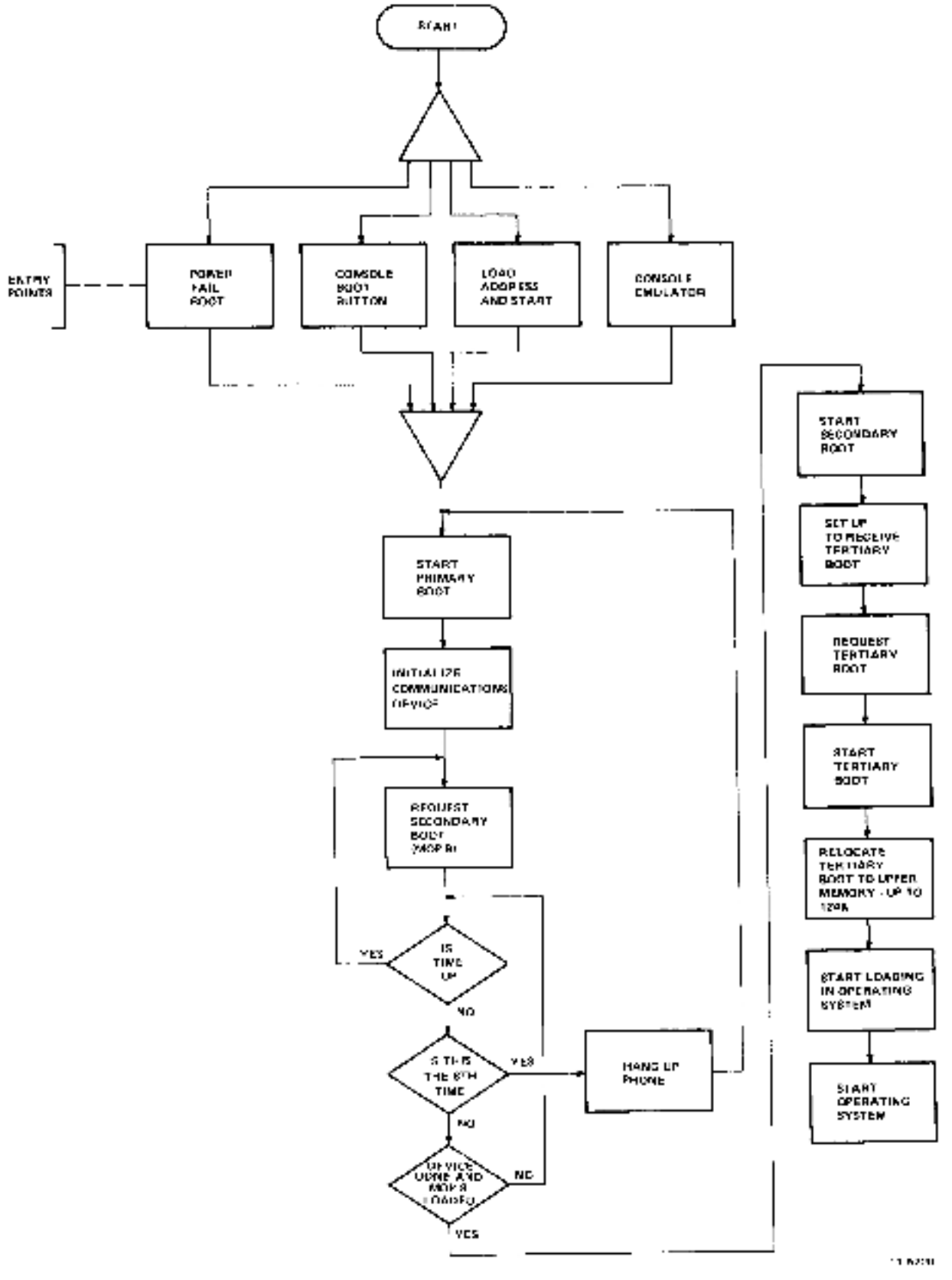

Figure 11-3 M9301-YE Down Line Load Flowchart

11.5.4 MOP Message Formats

The current MOP version (1.2, July 1976) operates within DDCMP maintenance envelopes. The current DDCMP version is 3.03-Dec 1975. All MOP messages are sent in a DDCMP maintenance mode envelope. This basic DDCMP message provides a CRC block check on the MOP data but does not provide any acknowledgement of receipt. It is similar to a numbered data message used for conformance with DDCMP formats and hence software/hardware drivers for DDCMP. Its form is as follows.

SYN, SYN, DLE, COUNT, Q, S, FILL, FILL, ADDR, CRC1, MOP MSG, CRC2

where:

SYN	=	The DDCMP sync characters: (8 bits) 226 synchronous – DU11, DUP11 337 asynchronous – DL11-E
DLE	=	The boot message header character: 220 (8 bits)
COUNT	=	The 14-bit count field: number of bytes in MOP message (14 bits)
Q	=	The async link control flag: (1 bit, always = 1)
S	=	The select link control flag: (1 bit, always = 1)
FILL	=	A fill character: 000 (8 bits)
ADDR	=	The tributary station address: (for pt-to-pt = 1) (8 bits)
CRC1	=	The header CRC on DLE through ADDR (16 bit CRC-16)
MOP MSG	=	The MOP message described below (COUNT 8 bit quantities)
CRC2	=	The data CRC on MOP message only (16 bit CRC-16)

Notice that the maintenance messages are preceded by 3 or more SYNC characters appropriate to the link used; synchronous – 226, asynchronous – 337.

The MOP message resides in the data field of the maintenance envelope. The form of the MOP message is:

CODE, INFO

where:

CODE	=	The MOP operation code (8 bits)
INFO	=	The information field (specific to each CODE)

The form of the MOP 8 program (request secondary boot) follows.

CODE (1 byte) = 8
MSG DATA = DEV TYPE, STA DDR, (1) PGM TYPE (0)

where:

DEV TYPE (1 byte)	=	The device type at the requesting system. DL11-E = 4, DU11 = 2, and DUP11 = 12_8.
STA DDR (1 byte)	=	The address of the requesting station. For point-to-point this is always 1. This is always 1 for the M9301-YE.
PGM TYPE (1 byte)	=	The type of secondary program requested. PGM type is always 0 for the M9301-YE.

The form of the MOP 0 program (memory load with transfer address) follows.

CODE (1 byte) = 0
MSG DATA = LOAD NUM, LOAD ADDR, IMAGE DATA, TRANSFER ADDR

where:

LOAD NUM (1 byte)	=	The load number for multiple load images. It may be preceded by loads without transfer address. This starts at 0 and is incremented for each load. This should always be 0 for the M9301-YE.
LOAD ADDR (4 bytes)	=	The memory load address (starting address) for the storage of the data image. It should be 0 in MOP 0. This should always be 0 for the M9301-YE.
IMAGE DATA	=	The image to be stored in the computer memory. The form sent will be machine dependent and may vary with the type and word length of the system [e.g., for PDP-11 systems, each byte represents one memory byte (8 bits)].
TRANSFER ADDR (4 bytes)	=	The starting address of the image just loaded. This is always 6 for the M9301-YE.

DECnet uses two CPU registers. R0 contains the device index code. R1 contains the CSR address of the device.

11.6 POWER-UP BOOT AND CONSOLE BOOT

When the operator performs a system power-up or presses the console boot switch, he causes the M9301 module to react by initiating a bootstrap routine. Table 11-1 shows the correct microswitch settings for the various options.

11.7 LOAD ADDRESS AND START PROCEDURE VIA CONSOLE SWITCH REGISTER

The user who wishes to initiate a function other than the one which he has specified in the microswitches can do so without resetting those microswitches. This involves a load address, placing an option code in the switch register, and start procedure. Thus it can be done only on machines with console switch registers.

The user must load address 173100 and then, before pressing the start switch, he must place a device code or option code in the switch register. These options and codes are shown in Table 11-2.

Table 11-1 Microswitch Settings

Function	S1-1	2**	3	4	5	6	7	8	9	10	Octal Code
Console Emulator with Diag.	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	000
Console Emulator without Diag.	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF	002
User Power Fail Routine through Vector 24 with Diag.	ON	ON	ON	ON	ON	OFF	OFF	ON	ON	OFF	062
User Power Fail Routine through Vector 24 without Diag.	ON	ON	ON	ON	ON	OFF	OFF	ON	OFF	ON	064
Boot Paper Tape Reader with Diag.	ON	ON	ON	ON	OFF	ON	ON	ON	OFF	OFF	106
DECnet Boot DL11E with Diag.	ON	ON	ON	ON	ON	ON	ON	OFF	ON	ON	010
DECnet Boot DL11E without Diag.	*	ON	ON	ON	ON	ON	ON	OFF	ON	OFF	012
DECnet Boot DU11 with Diag.	ON	ON	ON	ON	OFF	OFF	ON	OFF	OFF	ON	154
DECnet Boot DU11 without Diag.	*	ON	ON	ON	OFF	OFF	ON	OFF	OFF	OFF	156
DECnet Boot DUP11 with Diag.	ON	ON	ON	ON	OFF	OFF	OFF	ON	OFF	ON	164
DECnet Boot DUP11 without Diag.	*	ON	ON	ON	OFF	OFF	OFF	ON	OFF	OFF	166

*= switch setting makes no difference.
**= for pushbutton boot switch setting makes no difference.

Table 11-2 Load Address and Start Codes

Code	Function
165362	Enter console emulator after running diagnostics
165364	To enter console emulator without running diagnostics
173106	Boot the DL11 paper tape with diagnostics
173010	Boot the DECnet DL11-E with diagnostics
173012	Boot the DECnet DL11-E without diagnostics
173154	Boot the DECnet DU11 with diagnostics
173156	Boot the DECnet DU11 without diagnostics
173164	Boot the DECnet DUP11 with diagnostics
173166	Boot the DECnet DUP11 without diagnostics

Note that adding two to a code for an option with all diagnostics results in the entry for that option without any diagnostics.

11.8 OPERATION OF THE M9301-YE CONSOLE EMULATOR

The user can boot a program from any of the four types of peripheral devices by calling the bootstrap program for that peripheral through the console emulator.

The first routines to be executed (if 000 is the contents of the microswitches) are the primary CPU diagnostics.

The display routine is entered automatically. This routine will type the contents of R0, R4, R6, and R5 (note the sequence) on the Teletype or terminal in octal. Pressing the console boot switch causes the PDP-11/04 and PDP-11/34 systems to copy the PC into R5 before the power-up sequence starts. The console emulator (keyboard dispatch) program is then entered automatically. This routine types a carriage return, a line feed, and then the prompt symbol, @. The user can call bootstrap programs at this point by typing the appropriate device code. He then types an octal number following the device code specifying the drive number (default 0) and hits the carriage return key. Table 11-3 lists the peripheral bootstrap programs supported by the M9301-YE.

Note that 12 console fill characters are used between the carriage return and the line feed in the console emulator routine.

Table 11-3 Bootstrap Programs

Device	Description	Command*
DL11**	Low Speed Paper Tape Reader	TT
DL11E	Asynchronous Communications Interface	XL#
DU11	Synchronous Communications Interface	XU#
DUP11	Synchronous Communications Interface	XW#

*The # specifies the unit number (default 0). This may be from $0–17_8$ for XU and XW and $0–36_8$ for XL.
**This device is not bootable if S1 is OFF.

11.9 RESTARTING AT THE USER POWER FAIL ROUTINE

If the user wishes to restart his own software on a power-up he may do so by merely disabling the power fail restart switch in the microswitches (turn switch S1-2 OFF).

But the user can use the M9301-YE to run diagnostics (just the primary CPU tests described in Paragraph 11.10)) before running his power-up routine. This will in no way disturb the contents of memory and will in fact verify the machine's basic integrity after the power-down and -up sequence.

To use this option the operator should load the code 062 into the microswitches as described in Paragraph 11.7. This will result in the running of the primary CPU diagnostics upon power failure and then a simulated trap through 24 which will start the user's specified power-up routine.

If the code 064 is placed in the microswitches, then the simulated trap through 24 is executed without running diagnostics.

Figure 11-4 is an operator's flowchart showing the relation of the various options and codes available to the user.

11.10 DIAGNOSTICS

An explanation of the eight CPU and memory diagnostic tests follows. Three types of tests are included in the M9301-YE diagnostics:

1. Primary CPU tests (1–5)
2. Secondary CPU tests (6, 7)
3. Memory test (8)

11.10.1 CPU Tests

The primary CPU tests exercise all unary and double operand instructions with all source modes. These tests do not modify memory. If a failure is detected, a branch-self (BR.) will be executed. The run light will stay ON, because the processor will hang in a loop. If no failure is detected in tests 1–5, the processor will emerge from the last test and enter the register display routine (console emulator).

TEST 1 – SINGLE OPERAND TEST

This test executes all single operand instructions using destination mode 0. The basic objective is to verify that all single operand instructions operate; it also provides a cursory check on the operation of each instruction, while ensuring that the CPU decodes each instruction in the correct manner.

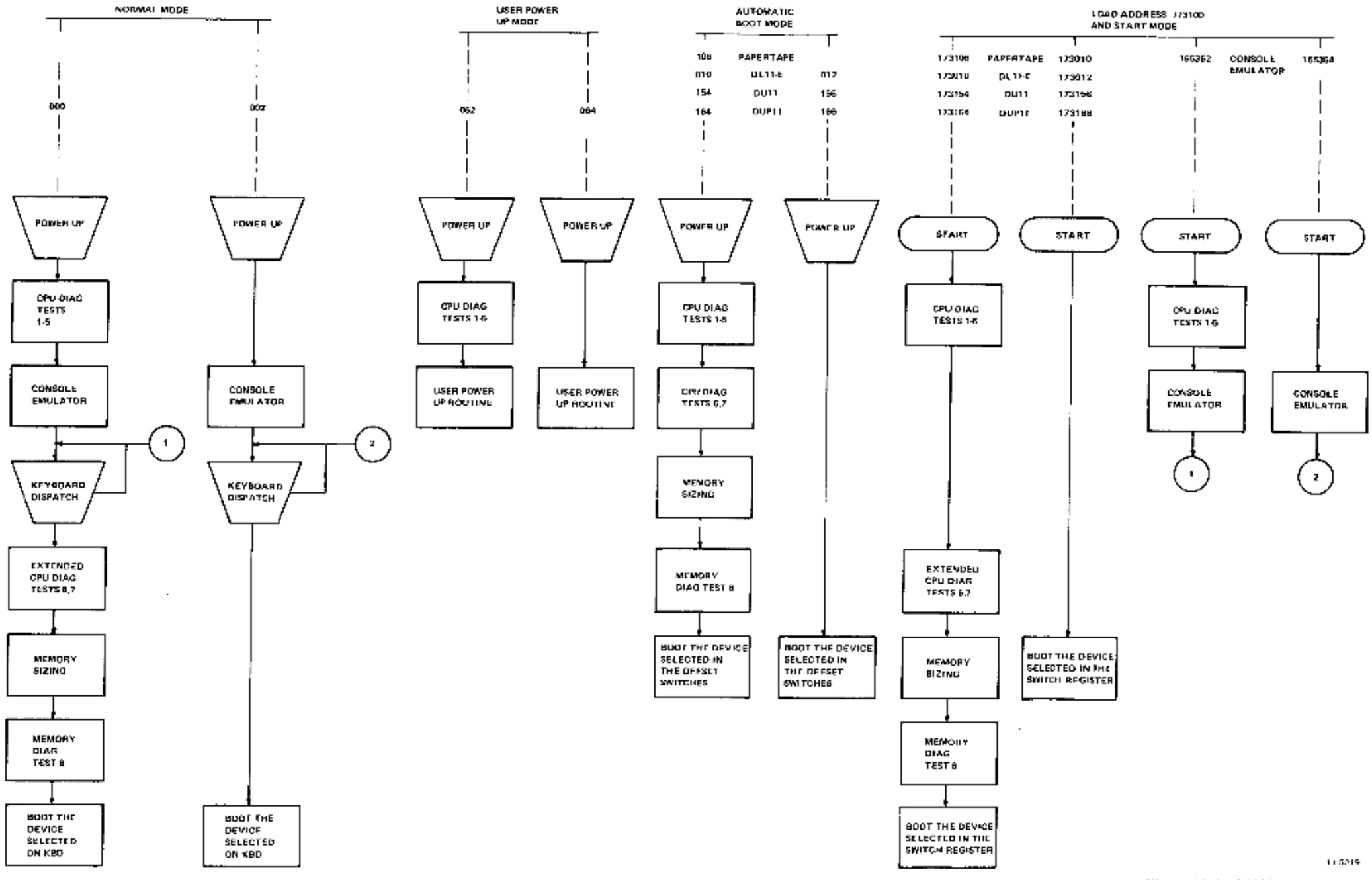

Figure 11-4 M9301-YE Operator's Flowchart

Test 1 tests the destination register in its three possible states: zero, negative, and positive. Each instruction operates on the register contents in one of four ways:

1. Data will be changed via a direct operation, i.e., increment, clear, decrement, etc.

2. Data will be changed via an indirect operation, i.e., arithmetic shifts, add carry, and subtract carry.

3. Data will be unchanged, but operated upon via a direct operation, i.e., clear a register already containing zeros.

4. Data will be unchanged but examined via a non-modifying instruction (TEST).

NOTE

When operating upon data in an indirect manner, the data is modified by the state of the appropriate condition code. Arithmetic shift will move the C bit into or out of the destination. This operation, when performed correctly, implies that the C bit was set correctly by the previous instruction. There are no checks on the data integrity prior to the end of the test. However, a check is made on the end result of the data manipulation. A correct result implies that all instructions manipulated the data in the correct way. If the data is incorrect, the program will hang in a program loop until the machine is halted.

TEST 2 – DOUBLE OPERAND, ALL SOURCE MODES

This test verifies all double operand, general, and logical instructions, each in one of the seven addressing modes (excludes mode 0). Thus, two operations are checked: the correct decoding of each double operand instruction, and the correct operation of each addressing mode for the source operand.

Each instruction in the test must operate correctly in order for the next instruction to operate. This interdependence is carried through to the last instruction (bit test) where, only through the correct execution of all previous instructions, a data field is examined for a specific bit configuration. Thus, each instruction prior to the last serves to set up the pointer to the test data.

Two checks on instruction operation are made in test 2. One check, a branch on condition, is made following the compare instruction, while the second is made as the last instruction in the test sequence.

Since the GO-NO GO tests reside in ROM memory, all data manipulation (modification) must be performed in destination mode 0 (register contains data). The data and addressing constants used by test 2 are contained within the ROM.

It is important to note that two different types of operations must execute correctly in order for this test to operate:

1. Those instructions that participate in computing the final address of the data mask for the final bit test instruction.

2. Those instructions that manipulate the test data within the register to generate the expected bit pattern.

Detection of an error within this test results in a program loop.

TEST 3 – JUMP TEST MODES 1, 2, 3
The purpose of this test is to ensure correct operation of the jump instruction. This test is constructed so that only a jump to the expected instruction will provide the correct pointer for the next instruction.

There are two possible failure modes that can occur in this test:

1. The jump addressing circuitry will malfunction, causing a transfer of execution to an incorrect instruction sequence or non-existent memory.

2. The jump addressing circuitry will malfunction in such a way as to cause the CPU to loop.

The latter case is a logical error indicator. The former, however, may manifest itself as an after-the-fact error. For example, if the jump causes control to be given to other routines within the M9301, the interdependent instruction sequences would probably cause a failure to eventually occur. In any case, the failing of the jump instruction will eventually cause an out of sequence or illogical event to occur. This in itself is a meaningful indicator of a malfunctioning CPU.

This test contains a jump mode 2 instruction which is not compatible across the PDP-11 line. However, it will operate on any PDP-11 within this test, due to the unique programming of the instruction within test 3. Before illustrating the operation, it is important to understand the differences of the jump mode 2 between machines.

On the PDP-11/20, 11/05, 11/15, and 11/10 processors for the jump mode 2 [JMP(R)+] the register (R) is incremented by 2 prior to execution of the jump. On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, (R) is used as the jump address and incremented by 2 after execution of the jump.

In order to avoid this incompatability, the JMP (R)+ is programmed with (R) pointing back on the jump itself. On 11/20, 11/05, 11/10, and 11/15 processors, execution of the instruction would cause (R) to be incremented to point to the following instruction, effectively continuing a normal execution sequence.

On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, the use of the initial value of (R) will cause the jump to loop back on itself. However, correct operation of the autoincrement will move (R) to point to the next instruction following the initial jump. The jump will then be executed again. However, the destination address will be the next instruction in sequence.

TEST 4 – SINGLE OPERAND, NON-MODIFYING BYTE TEST
This test focuses on one unique single operand instruction, the TST. TST is a special case in the CPU execution flow since it is a non-modifying operation. Test 4 also tests the byte operation of this instruction. The TSTB instruction will be executed in mode 1 (register deferred) and mode 2 (register deferred, autoincrement).

The TSTB is programmed to operate on data which has a negative value most significant byte and a zero (not negative) least significant byte.

In order for this test to operate properly, the TSTB on the low byte must first be able to access the even addressed byte and then set the proper condition codes. The TSTB is then reexecuted with the autoincrement facility. After the autoincrement, the addressing register should be pointing to the high byte of the test data. Another TSTB is executed on what should be the high byte. The N bit of the condition codes should be set by this operation.

Correct execution of the last TSTB implies that the autoincrement feature recognized that a byte operation was requested, thereby only incrementing the address in the register by one, rather than two. If the correct condition code has not been set by the associated TSTB instruction, the program will loop.

TEST 5 – DOUBLE OPERAND, NON-MODIFYING TEST
Two non-modifying, double-operand instructions – the compare (CMP) and bit test (BIT) – operate on test data in source modes 1 and 4, and destination modes 2 and 4.

The BIT and CMP instructions will operate on data consisting of all ones (177777). Two separate fields of ones are used in order to utilize the compare instructions, and to provide a field large enough to handle the autoincrementing of the addressing register.

Since the compare instruction is executed on two fields containing the same data, the expected result is true Z bit, indicating equality.

The BIT instruction will use a mask argument of all ones against another field of all ones. The expected result is a non-zero condition (Z).

Most failures will result in a one instruction loop.

At the end of test 5 the register display is enabled, provided the console emulator routine has been selected in the microswitches. The register display routine prints out the octal contents of the CPU registers R0, R4, SP and old PC on the console terminal. This sequence will be followed by a prompt character (5) on the next line.

An example of a typical printout follows,

	XXXXXX	XXXXXX	XXXXXX	XXXXXX
5				
Prompt Character	R0	R4	R6 (Stack Pointer)	R5 (Old PC)

NOTES:

1. Where X signifies an octal number (0–7).

2. Whenever there is a power-up routine or the boot switch is released on PDP-11/04 and PDP-11/34 machines, the PC at this time will be stored in R5. The contents of R5 are then printed as the old PC shown in the example.

3. The prompting character string indicates that primary diagnostics have been run and the processor is operating.

11.10.2 Secondary CPU and Memory Tests
The secondary CPU tests modify memory and involve the use of the stack pointer. The JMP and JSR instructions and all destination modes are tested. If a failure is detected, these tests, unlike the primary tests, will execute a halt.

Secondary CPU and memory diagnostics are run immediately after test 5 when they have been evoked by means other than the console emulator, provided that the correct microswitches have been set. If the console emulator has been entered at the completion of test 5, the secondary CPU and memory diagnostics will be run when the appropriate boot command is given.

Note that the user can execute the secondary CPU and memory diagnostics without running a bootstrap program. A false boot command (a non-existent device code or unit number followed by a carriage return) will cause execution of tests 6, 7, and 8 before the attempt to boot is made. If these tests are executed successfully, the device will be called but not found. The processor will trap to location 4, which has been set up by the M9301-YE. A new PC is obtained at location causing a halt to be executed at location 0.

TEST 6 – DOUBLE OPERAND, MODIFYING BYTE TEST
The objective of this test is to verify that the double-operand, modifying instructions will operate in the byte mode. Test 6 contains three subtests:

1. Test source mode 2, destination mode 1, odd and even bytes
2. Test source mode 3, destination mode 2
3. Test source mode 0, destination mode 3, even byte.

The move byte (MOVB), bit clear byte (BICB), and bit set byte (BISB) are used within test 6 to verify the operation of the modifying double-operand functions.

Since modifying instructions are under test, memory must be used as a destination for the test data. Test 6 uses location 500 as a destination address. Later, in test 7 and the memory test, location 500 is used as the first available storage for the stack.

Note that since test 6 is a byte test, location 500 implies that both 500 and 501 are used for the byte tests (even and odd, respectively). Thus, in the word of data at 500, odd and even bytes are caused to be all 0s and then all 1s alternately throughout the test. Each byte is modified independently of the other.

TEST 7 – JSR TEST
The JSR is the first test in the GO-NO GO sequence that utilizes the stack. The jump subroutine command (JSR) is executed in modes 4 and 6. After the JSR is executed, the subroutine which was given control will examine the stack to ensure that the correct data was placed in the correct stack location (500). The routine will also ensure that the line back register points to the correct address. Any errors detected in this test will result in a halt.

TEST 8 – DUAL ADDRESSING AND DATA CHECK
Finally the memory test performs both dual addressing and a data check of all the available memory on the system below 28K. This test will leave all of memory clear. Like the secondary tests the memory test will halt when an error is detected. At the time the memory error halt is executed, R0 will contain the address at which the failure was detected. R4 will contain the failing data pattern and R6 will contain the expected data pattern. Thus after a memory failure has occurred, the user can enter the console emulator and have this information printed out immediately by the display routine (see section on console emulator).

11.11 TROUBLESHOOTING
When a halt occurs, the user should reboot the system by pressing the BOOT/INIT switch. The registers R0, R4, R6, and R5 will be displayed on the terminal in that order.

R0	Failing address
R4	Failing data
R6	Expected data
R5	Old PC

The diagnostic program in the M9301-YE will cause the processor to halt at one of four addresses: 165604, 165620, 165634, or 165774 in the event of a failure. The user should consult the diagnostic program listing to find the failing test and begin troubleshooting. Possible causes of the failure include bus errors, a bad M9301 module, and a bad CPU.

11.12 FLOATING DEVICE PRIORITY

The M9301-YE assumes the following priority of devices in the floating address space.

1. DJ11
2. DH11
3. DQ11
4. DU11
5. DUP11

The M9301-YE will not function properly unless the above priority and correct Unibus address spacing is followed.

CHAPTER 12
M9301-YF

12.1 INTRODUCTION

The M9301-YF is designed to function in PDP-11 systems. It provides bootstrapping capabilities for all major PDP-11 devices and includes routines for console emulation and some basic CPU and memory GO-NO GO diagnostic tests.

This bootstrap module has been designed for maximum flexibility of operation. Its function may be initiated in any of four ways: automatically at power up (exception 11/45, 11/50); by depressing the console boot switch; by a LOAD ADDRESS and START sequence; and by using the console emulator through a terminal.

The M9301-YF is essentially a combination of the M9301-YA and the M9301-YB. The M9301-YF can boot from the console switch register, but only unit number 0. Figure 12-1 is a memory map for the M9301-YF. Note that the TA11 is not supported.

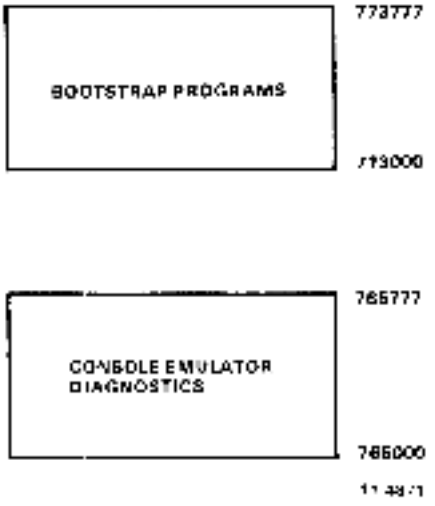

Figure 12-1 M9301-YF Memory Map

The lower addresses, 765000–765777, can be disabled by putting switch S1-1 in the OFF position. This would free addresses 765000–765777, allowing the user to occupy this address space.

Note that if switch S1-1 is OFF, the console emulator and diagnostics are no longer available. Also, the DL11 and PC11 cannot be booted if S1-1 is OFF.

12.2 OPERATION OF THE M9301-YF

The user can boot a program from any of 11 peripheral devices by calling the bootstrap program for that peripheral through the console emulator. The first routines to be executed (if 000 is the contents of the microswitches) are the primary CPU diagnostics.

The display routine is entered automatically. This routine will type the contents of R0, R4, R6, and R5 (note the sequence) on the Teletype or terminal in octal. Pressing the console boot switch causes the PDP-11/04 and PDP-11/34 systems to copy the PC into R5 before the power-up sequence starts. The console emulator (keyboard dispatch) program is then entered automatically. This routine types a carriage return, a line feed and then the prompt symbol, $. The user can call bootstrap programs at this point by typing the appropriate device code. He then types an octal number following the device code specifying the drive number (default 0) and hits the carriage return key. Table 12-1 lists the peripheral bootstrap programs supported by the M9301-YF.

Table 12-1 M9301-YF Bootstrap Programs

Device	Description	Command
DL11	Low-Speed Paper Tape	TT
PC11	High-Speed Paper Tape	PR
RK11	RK03/05 Disk Drive	DK
RK611	RK06 Disk Drive	DM
RP11	RP02/RP03 Disk Drive	DP
RX11	RX01 Floppy Disk Drive	DX
TC11	DECtape	DT
TM11	TU10 800 bpi Magtape	MT
Massbus Devices* (RH11, RH70)		
RS03/RS04	Fixed Head Disk	DS
RP04/RP05/RP06	Disk Pack Drive	DB
TU16	Magnetic Tape	MM

*Different devices on same Massbus controller are not supported.

An explanation of the functions performed by the various bootstrap programs follows.

RX11 Diskette Bootstrap – Loads the first 64 words (200_8 bytes) of data from track one, sector one into memory locations 0–176 beginning at location 0.

PC11 Paper Tape Reader Bootstrap – Loads an absolute loader formatted tape into the upper memory locations XXX746 to XXX777 (XXX is dependent on memory size). Once loading is completed, the boot transfers operation to a routine beginning at location XXX752.

Disk and DECtape Bootstraps (excluding RX11) – Load 1000_8 words (2000_8 bytes) of data from the device into memory locations $0\text{-}1776_8$.

Magtape Bootstraps

TM11 – Loads second record (2000_8 bytes maximum size) from the magtape into memory location $0–1776_8$.

TJU16 – Loads second record (2000_8 bytes maximum size) from magtape into memory locations $0–1776_8$. (Note that the first record contains the magtape directory.)

12.3 MICROSWITCH SETTINGS

A set of ten microswitches is located on the M9301 module. They determine which ROM routines are selected and give the user automatic access to any function.

The primary activating processes for the M9301-YF are the power-up sequence and the enabling of the console boot switch. Switch S1-1 must be in the ON position in order to enable activation of the M9301-YF console emulator and diagnostics. Switch S1-2 is the power-up reboot enable switch. It must be ON to enable to M9301-YF on power up. If switch S1-2 is OFF, then the processor will trap to location 24 (as normal) to execute the user power-up routine. When switch S1-2 is ON, the other switches, S1-3 through S1-10, determine what action the M9301-YF will take on power-up.

If the system includes a console boot switch, then any time that switch is pressed the M9301-YF will be activated. Note that some processors will have to be halted for this switch to have any effect. Enabling the console boot switch causes the processor to enter a ROM routine by creating a fake power-down/power-up sequence. The user should note that the position of switch S1-2 is irrelevant when the console boot switch is used.

Pushing the console boot switch thus results in a power-up sequence in the processor through an M9301-YF ROM routine. Prior to the power-up sequence, the M9301-YF asserts 773000 on the Unibus address lines. This causes the new PC to be taken from ROM location 773024 instead of location 000024. The new PC will be the logical OR of the contents of ROM location 773024 and the eight microswitches on the M9301-YF module. A switch in the ON position is read as a 0, while a switch in the OFF position is a 1. In this way all the M9301-YF options are accessible.

Each option is given a different address. Note that microswitch S1-10 is ORed with bit 1 of the data in ROM location 773024. S1-9 is ORed with bit 2, etc. No switch is provided for combination with bit 0, because an odd address could result when going through the trap sequence. Figure 12-2 shows the relationship of the switches to the bus address bits.

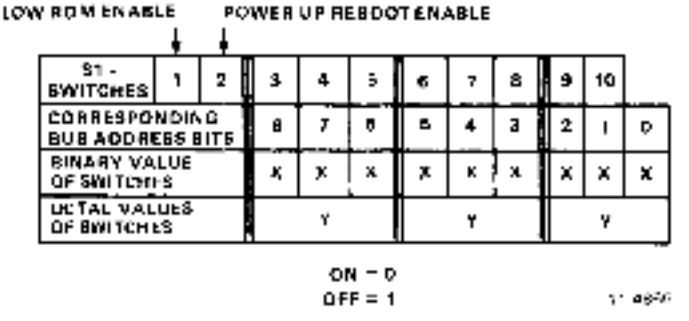

Figure 12-2 Program Flowchart for the M9301-YF

12.4 PROGRAM CONTROL THROUGH THE MICROSWITCHES

The microswitches on the M9301-YF enable the user either to start a bootstrap operation or to enter the console emulator, simply by pressing the boot switch. Note that a momentary power failure will have the same effect as pushing the boot switch. The user should also note that he can select any function without diagnostics by adding 2 to the appropriate octal code in the switches.

Figure 12-3 is a flowchart showing three of the four modes available to the M9301-YF user. In these three modes (normal mode, user power-up mode, and automatic boot mode), the choice and sequence of routines is entirely dependent on the offset switch settings.

Table 12-2 is a list of microswitch settings and corresponding octal codes which can be directly related to the flowchart in Figure 12-3.

Table 12-2 Options and Corresponding Switch Settings

Function	S3	S4	S5	S6	S7	S8	S9	S10	Octal Code
CPU Diagnostics with Console Emulator*	ON	ON	ON	ON	ON	ON	ON	ON	000
Console Emulator (without diag.)*	ON	ON	ON	ON	ON	ON	ON	OFF	002
CPU diag. Vector through location 24 User power fail routine*	OFF	OFF	ON	OFF	ON	ON	OFF	ON	644
Vector through location 24 (without diag.) User power fail routine*	OFF	OFF	ON	OFF	ON	ON	OFF	OFF	646
CPU Diag. Boot RP11*	ON	ON	ON	OFF	ON	ON	ON	ON	040
Boot RP11 (without diag.)	ON	ON	ON	OFF	ON	ON	ON	OFF	042
CPU Diag. Boot RJP04/05/06*	ON	OFF	ON	ON	OFF	ON	OFF	ON	224
Boot RJP04/05/06 (without diag.)	ON	OFF	ON	ON	OFF	ON	OFF	OFF	226
CPU Diag. Boot RJS03/04*	OFF	ON	ON	OFF	ON	ON	ON	ON	440
Boot RJS03/04 (without diag.)	OFF	ON	ON	OFF	ON	ON	ON	OFF	442

Table 12-2 Options and Corresponding Switch Settings (Cont)

Function	S3	S4	S5	S6	S7	S8	S9	S10	Octal Code
CPU Diag. Boot RK11*	ON	ON	OFF	OFF	ON	ON	OFF	ON	144
Boot RK11 (without diag.)	ON	ON	OFF	OFF	ON	ON	OFF	OFF	146
CPU Diag. Boot RK611*	OFF	OFF	ON	OFF	OFF	ON	ON	OFF	662
Boot RK611 (without diag.)	OFF	OFF	ON	OFF	OFF	ON	OFF	ON	664
CPU Diag. Boot RX11*	OFF	ON	OFF	OFF	ON	ON	OFF	ON	544
Boot RX11 (without diag.)	OFF	ON	OFF	OFF	ON	ON	OFF	OFF	546
CPU Diag. Boot TC11*	ON	ON	OFF	OFF	OFF	OFF	OFF	ON	174
Boot TC11 (without diag.)	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	176
CPU Diag. Boot TM11*	OFF	ON	ON	OFF	OFF	OFF	OFF	ON	474
Boot TM11 (without diag.)	OFF	ON	ON	OFF	OFF	OFF	OFF	OFF	476
CPU Diag. Boot TJU16*	ON	ON	ON	OFF	ON	OFF	OFF	ON	054
Boot TJU16 (without diag.)	ON	ON	ON	OFF	ON	OFF	OFF	OFF	056
CPU Diag. Boot DL11*	ON	OFF	OFF	OFF	OFF	ON	OFF	ON	364
Boot DL11 (without diag.)	ON	OFF	OFF	OFF	OFF	ON	OFF	OFF	366
CPU Diag. Boot PC11*	OFF	OFF	OFF	ON	ON	ON	OFF	ON	704
Boot PC11 (without diag.)	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	706

*S1-1 must be on.

1. To boot on power up, S1-2 must be on.
2. If the boot switch is used, the position of S1-2 is irrelevant.

Note: ON = 0, OFF = 1

12.5 LOAD ADDRESS AND START MODE

The user who wishes to initiate a function other than the one which he has specified in the microswitches can do so without resetting those microswitches. This involves a load address, placing an option code in the switch register, and a start procedure. It can be done only on machines with console switch registers. Figure 12-4 is a program flowchart. Table 12-3 lists the switch register codes in tabular format. The user must load address 173016, and then, before pressing the START switch, he must place a device code or option code in the switch register.

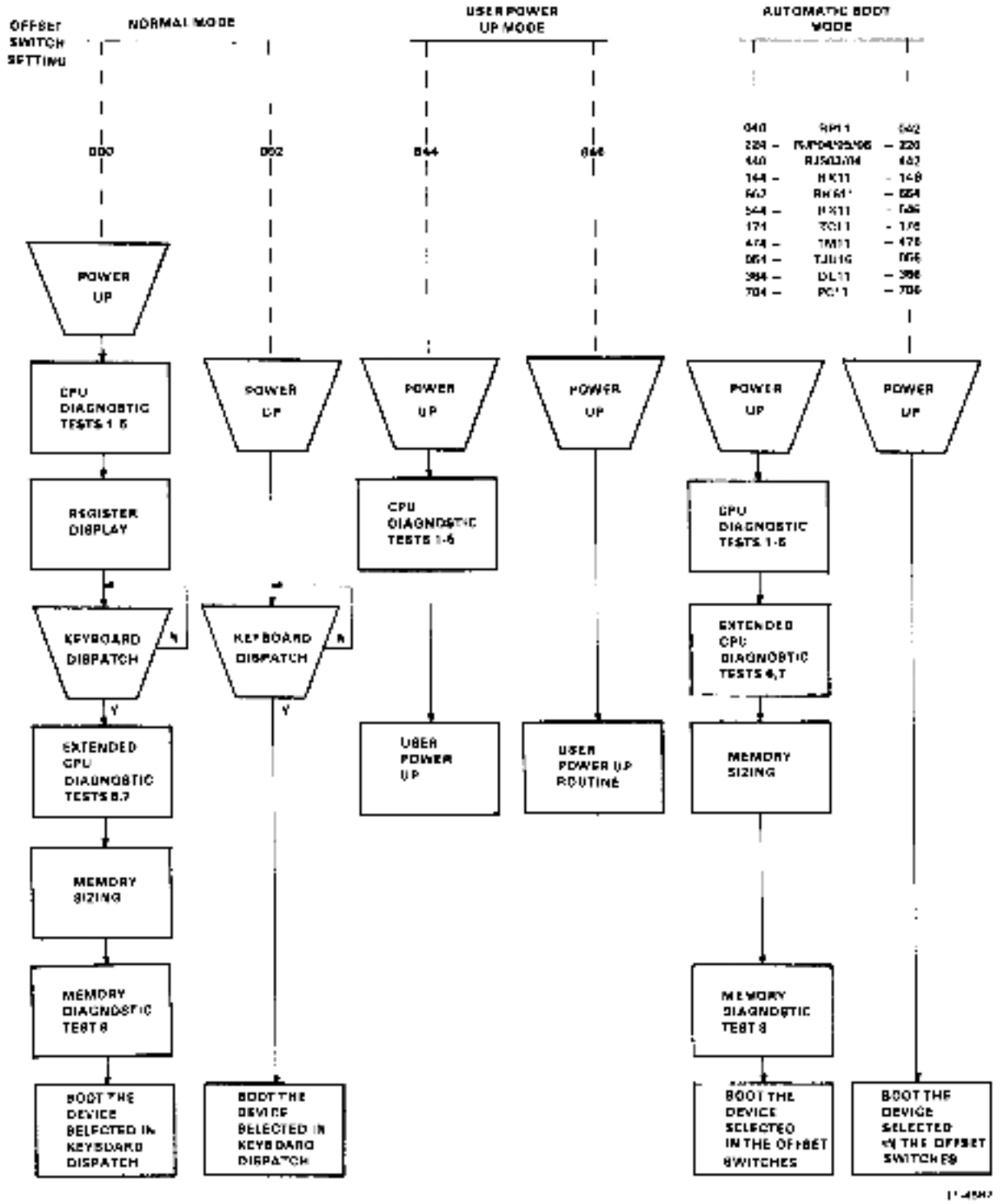

Figure 12-3 M9301-YF Flowchart

Table 12-3 Load Address and Start

Load Address	Result
165404**	To enter console emulator after running diagnostics
165406**	To enter console emulator without running diagnostics
173040*	To boot the RP11 with diagnostics
173042	To boot the RP11 without diagnostics
173224*	To boot the RJP04, RJP05, or RJP06 with diagnostics
173226	To boot the RJP04, RJP05, or RJP06 without diagnostics
173440*	To boot the RJS03 or RJS04 with diagnostics
173442	To boot the RJS03 or RJS04 without diagnostics
173144*	To boot the RK11 with diagnostics
173146	To boot the RK11 without diagnostics
173662*	To boot the RK611 with diagnostics
173664*	To boot the RK611 without diagnostics
173544*	To boot the RX11 with diagnostics
173546	To boot the RX11 without diagnostics
173174*	To boot the TC11 with diagnostics
173176	To boot the TC11 without diagnostics
173474*	To boot the TM11 with diagnostics
173476	To boot the TM11 without diagnostics
173054*	To boot the TJU16 with diagnostics
173056*	To boot the TJU16 without diagnostics
173364*	To boot the DL11 paper tape with diagnostics
173366*	To boot the DL11 without diagnostics
173704*	To boot the PC11 paper tape with diagnostics
173706*	To boot the PC11 without diagnostics

*S1-1 must be ON

**If the switches on the M9301-YF are set up to default boot the console emulator, the state of switch S1-10 will determine whether or not diagnostics are run. S1-10 must be on if diagnostics are to be run. S1-1 must also be ON.

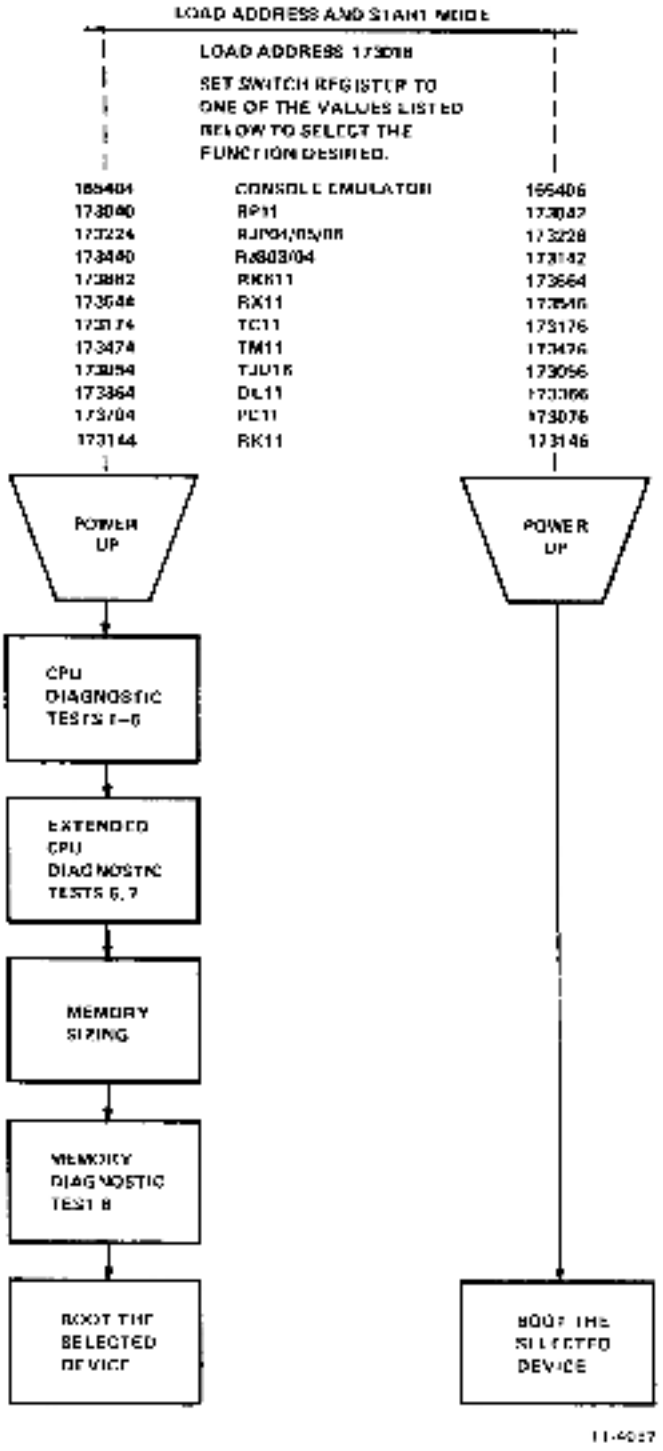

Figure 12-4 M9301-YF Load Address and Start Mode Flowchart

12.6 DIAGNOSTICS

An explanation of the eight CPU and memory diagnostic tests follows. Three types of tests are included in the M9301-YF diagnostics:

1. Primary CPU tests (1–5)
2. Secondary CPU tests (6, 7)
3. Memory test (8)

12.6.1 Primary CPU Tests

The primary CPU tests exercise all unary and double operand instructions with all source modes. These tests do not modify memory. If a failure is detected, a branch-self (BR.) will be executed. The run light will stay on, because the processor will hang in a loop, but there will be no register display. The user must use the halt switch to exit from the loop. If no failure is detected in tests 1–5, the processor will emerge from the last test and enter the register display routine (console emulator).

TEST 1 – SINGLE OPERAND TEST

This test executes all single operand instructions using destination mode 0. The basic objective is to verify that all single operand instructions operate; it also provides a cursory check on the operation of each instruction, while ensuring that the CPU decodes each instruction in the correct manner.

Test 1 tests the destination register in its three possible states: zero, negative, and positive. Each instruction operates on the register contents in one of four ways:

1. Data will be changed via a direct operation, i.e., increment, clear, decrement, etc.

2. Data will be changed via an indirect operation, i.e., arithmetic shifts, add carry, and subtract carry.

3. Data will be unchanged, but operated upon via a direct operation, i.e., clear a register already containing zeros.

4. Data will be unchanged but examined via a non-modifying instruction (TEST).

NOTE

When operating upon data in an indirect manner, the data is modified by the state of the appropriate condition code. Arithmetic shift will move the C bit into or out of the destination. This operation, when performed correctly, implies that the C bit was set correctly by the previous instruction. There are no checks on the data integrity prior to the end of the test. However, a check is made on the end result of the data manipulation. A correct result implies that all instructions manipulated the data in the correct way. If the data is incorrect, the program will hang in a program loop until the machine is halted.

TEST 2 – DOUBLE OPERAND, ALL SOURCE MODES

This test verifies all double operand, general, and logical instructions, each in one of the seven addressing modes (excludes mode 0). Thus, two operations are checked: the correct decoding of each double operand instruction, and the correct operation of each addressing mode for the source operand.

Each instruction in the test must operate correctly in order for the next instruction to operate. This interdependence is carried through to the last instruction (bit test) where, only through the correct execution of all previous instructions, a data field is examined for a specific bit configuration. Thus, each instruction prior to the last serves to set up the pointer to the test data.

Two checks on instruction operation are made in test 2. One check, a branch on condition, is made following the compare instruction, while the second is made as the last instruction in the test sequence.

Since the GO-NO GO tests reside in ROM memory, all data manipulation (modification) must be performed in destination mode 0 (register contains data). The data and addressing constants used by test 2 are contained within the ROM.

It is important to note that two different types of operations must execute correctly in order for this test to operate:

1. Those instructions that participate in computing the final address of the data mask for the final bit test instruction.

2. Those instructions that manipulate the test data within the register to generate the expected bit pattern.

Detection of an error within this test results in a program loop.

TEST 3 – JUMP TEST MODES 1, 2, AND 3
The purpose of this test is to ensure correct operation of the jump instruction. This test is constructed so that only a jump to the expected instruction will provide the correct pointer for the next instruction.

There are two possible failure modes that can occur in this test:

1. The jump addressing circuitry will malfunction causing a transfer of execution to an incorrect instruction sequence or non-existent memory.

2. The jump addressing circuitry will malfunction in such a way as to cause the CPU to loop.

The latter case is a logical error indicator. The former, however, may manifest itself as an after-the-fact error. For example, if the jump causes control to be given to other routines within the M9301-YF, the interdependent instruction sequences would probably cause a failure to eventually occur. In any case, the failing of the jump instruction will eventually cause an out of sequence or illogical event to occur. This in itself is a meaningful indicator of a malfunctioning CPU.

This test contains a jump mode 2 instruction which is not compatible across the PDP-11 line. However, it will operate on any PDP-11 within this test, due to the unique programming of the instruction within test 3. Before illustrating the operation, it is important to understand the differences of the jump mode 2 between machines.

On the PDP-11/20, 11/05, 11/15, and 11/10 processors, for the jump mode 2 [JMP(R)+], the register (R) is incremented by 2 prior to execution of the jump. On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, (R) is used as the jump address and incremented by 2 after execution of the jump.

In order to overcome this incompatability, the JMP (R)+ is programmed with (R) pointing back on the jump itself. On 11/05, 11/10, 11/15, and 11/20 processors, execution of the instruction would cause (R) to be incremented to point to the following instruction, effectively continuing a normal execution sequence.

On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, the use of the initial value of (R) will cause the jump to loop back on itself. However, correct operation of the autoincrement will move (R) to point to the next instruction following the initial jump. The jump will then be executed again. However, the destination address will be the next instruction in sequence.

TEST 4 – SINGLE OPERAND, NON-MODIFYING BYTE TEST
This test focuses on the one single operand instruction, the TST. TST is a special case in the CPU execution flow since it is a non-modifying operation. Test 4 also tests the byte operation of this instruction. The TSTB instruction will be executed in mode 1 (register deferred) and mode 2 (register deferred, autoincrement).

The TSTB is programmed to operate on data which has a negative value most significant byte and a zero (not negative) least significant byte.

In order for this test to operate properly, the TSTB on the low byte must first be able to access the even addressed byte and then set the proper condition codes. The TSTB is then reexecuted with the autoincrement facility. After the autoincrement, the addressing register should be pointing to the high byte of the test data. Another TSTB is executed on what should be the high byte. The N bit of the condition codes should be set by this operation.

Correct execution of the last TSTB implies that the autoincrement recognized that a byte operation was requested, thereby only incrementing the address in the register by one, rather than two. If the correct condition code has not been set by the associated TSTB instruction, the program will loop.

TEST 5 – DOUBLE-OPERAND, NON-MODIFYING TEST
Two non-modifying, double-operand instructions are used in this test – the compare (CMP) and bit test (BIT). These two instructions operate on test data in source modes 1 and 4, and destination modes 2 and 4.

The BIT and CMP instructions will operate on data consisting of all ones (177777). Two separate fields of ones are used in order to utilize the compare instructions, and to provide a field large enough to handle the autoincremententing of the addressing register.

Since the compare instruction is executed on two fields containing the same data, the expected result is a true Z bit, indicating equality.

The BIT instruction will use a mask argument of all ones against another field of all ones. The expected result is a non-zero condition (Z).

Most failures will result in a one instruction loop.

At the end of test 5, the register display routine is enabled, provided the console emulator has been selected in the microswitches, is enabled. The register display routine prints out the octal contents of the CPU registers R0, R4, SP, and old PC on the console terminal. This sequence will be followed by a prompt character ($) on the next line.

An example of a typical printout follows.

	XXXXXX	XXXXXX	XXXXXX	XXXXXX
$				
Prompt Character	R0	R4	R6 (Stack Pointer)	R5 (Old PC)

NOTES:

1. Where X signifies an octal number (0–7).

2. Whenever there is a power-up routine or the boot switch is released on PDP-11/04 and PDP-11/34 machines, the PC at this time will be stored in R5. The contents of R5 are then printed as the old PC shown in the example.

3. The prompting character string indicates that diagnostics have been run and the processor is operating.

12.6.2 Secondary CPU and Memory Tests

The secondary CPU tests modify memory and involve the use of the stack pointer. The JMP and JSR instructions and all destination modes are tested. If a failure is detected, these tests, unlike the primary tests, will execute a halt.

Secondary CPU and memory diagnostics are run immediately after test 5 when they have been evoked by means other than the console emulator, provided that the correct microswitches have been set. If the console emulator has been entered at the completion of test 5, the secondary CPU and memory diagnostics will be run when the appropriate boot command is given.

In contrast to the M9301-YA and M9301-YB, the M9301-YF reacts to a false boot command (an invalid address code) by trapping to location 4 with the end result being a halt if diagnostics are selected. This halt should not be interpreted as a diagnostic test failure.

TEST 6 DOUBLE OPERAND, MODIFYING, BYTE TEST

The objective of this test is to verify that the double-operand, modifying instructions will operate in the byte mode. Test 6 contains three subtests:

1. Test source mode 2, destination mode 1, odd and even bytes
2. Test source mode 3, destination mode 2
3. Test source mode 0, destination mode 3, even byte.

The move byte (MOVB), bit clear byte (BICB), and bit set byte (BISB) are used within test 6 to verify the operation of the modifying double-operand functions.

Since modifying instructions are under test, memory must be used as a destination for the test data. Test 6 uses location 500 as a destination address. Later, in test 7 and the memory test, location 500 is used as the first available storage for the stack.

Note that since test 6 is a byte test, location 500 implies that both 500 and 501 are used for the byte tests (even and odd, respectively). Thus, in the word of data at 500, odd and even bytes are caused to be all 0s and then all 1s alternately throughout the test. Each byte is modified independently of the other.

TEST 7 – JSR TEST

The JSR is the first test in the GO-NO GO sequence that utilizes the stack. The jump subroutine command (JSR) is executed in modes 1 and 6. After the JSR is executed, the subroutine which was given control will examine the stack to ensure that the correct data was placed in the correct stack location (500). The routine will also ensure that the line back register points to the correct address. Any errors detected in this test will result in a halt.

TEST 8 – DUAL ADDRESSING AND DATA CHECK

Finally the memory test performs both dual addressing and a data check of all the available memory on the system below 28K. This test will leave all of memory clear. Like the secondary tests the memory test will halt when an error is detected. At the time the memory error halt is executed, R4 will contain the address at which the failure was detected. R0 will contain the failing data pattern and R6 will contain the expected data pattern. Thus after a memory failure has occurred, the user can enter the console emulator and have this information printed out immediately by the display routine (see section on console emulator).

12.7 TROUBLESHOOTING

When a halt occurs, the user should reboot the system by pressing the BOOT/INIT switch. The registers R0, R4, R6, and R5 will be displayed on the terminal in that order.

R4	Expected data
R0	Received data
R6	Failing address (SP)
R5	Old PC

The diagnostic program in the M9301-YF will cause the processor to halt at one of four addresses: 165646, 165662, 165676, or 165776. The user should consult the diagnostic program listing to find the failing test and begin troubleshooting. Possible causes of the failure include bus errors, a bad M9301 module, and a bad CPU.

12.8 RESTARTING AT THE USER POWER FAIL ROUTINE

If the user wishes to restart software on a power-up, he may do so by merely disabling the power fail restart switch in the microswitches (turn switch S1-2 OFF). However, the user can use the M9301-YF to run diagnostics (the primary CPU tests just described) before running his power-up routine. This will in no way disturb the contents of memory and will in fact verify the machine's basic integrity after the power-down and -up sequence. In order to use this option, put the code 644 into the microswitches as described previously. Switch S1-9 must be OFF. This will result in the running of the primary CPU diagnostics and then a simulated trap through 24 which will start the user's specified power-up routine. If the code 646 is placed in the microswitches, then the simulated trap through 24 is executed without running any diagnostics.

CHAPTER 13
M9301-YH

13.1 INTRODUCTION

The M9301-YH has been created to provide bootstrapping capabilities for the PDP-11/60 and PDP-11/70 computers. It also provides basic diagnostic tests for the CPU, memory, and cache.

The M9301-YH provides flexibility of operation via user-settable microswitch options. Its functions can be initiated automatically on power-up, by pressing the console boot switch (PDP-11/60), or by a load-address start sequence.

NOTE

For the PDP-11/60, jumpers W1–W5 must be out. (W1–W5 provide pull-ups for bus grant lines.) For the PDP-11/70, jumpers W1–W5 must be in.

13.2 MEMORY MAP

The four tristate ROMs which contain the program information use two distinct address spaces:

1. XXX65000–XXX65776
2. XXX73000–XXX73776

All the diagnostic tests reside in the first address space. All the bootstrap loaders reside in the second address space (Figure 13-1).

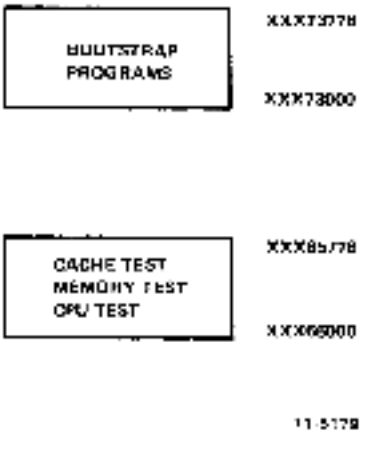

Figure 13-1 M9301-YH Program Memory Map

13.3 DIAGNOSTIC TEST EXPLANATION

The diagnostic portion of the program will test the basic CPU including the branches, the registers, all addressing modes, and many of the instructions in the PDP-11 repertoire. It will check memory from virtual address 1000 to the highest available address up to 28K. After main memory has been verified, with the cache off, the cache memory will be tested to verify that hits occur properly. Main memory will be scanned again to ensure that the cache is working properly throughout the 28K of memory to be used in the boot operation.

If one of the cache memory tests fails, the operator can attempt to boot the system anyway by pressing continue. This will cause the program to force misses in both groups of the cache before going to the bootstrap section of the program.

A list of the M9301-YH diagnostic tests follows.

TEST 1	This test verifies the unconditional branch
TEST 2	Test CLR, MODE 0, and BMI, BVS, BHI, BLOS
TEST 3	Test DEC, MODE 0, and BPL, BEQ, BGE, BGT, BLE
TEST 4	Test ROR, MODE 0, and BVC, BHIS, BHI, BNE
TEST 5	Test BHI, BLT, and BLOS
TEST 6	Test BLE and BGT
TEST 7	Test register data path and modes 2, 3, 6
TEST 10	Test ROL, BCC, BLT, and MODE 6
TEST 11	Test ADD, INC, COM, and BCS, BLE
TEST 12	Test ROR, BIS, ADD, and BLO, BGE
TEST 13	Test DEC and BLOS, BLT
TEST 14	Test COM, BIC, and BGT, BGE, BLE
TEST 15	Test ADC, CMP, BIT, and BNE, BGT, BEQ
TEST 16	Test MOVB, SOB, CLR, TST and BPL, BNE
TEST 17	Test ASR, ASL
TEST 20	Test ASH, and SWAB
TEST 21	Test JSR, RTS, RTI, and JMP
TEST 22	Test main memory from virtual 1000 to highest available address up to 28K
TEST 23	Test cache data memory
TEST 24	Test memory 28K with cache on

13.4 DIAGNOSTIC TEST DESCRIPTIONS

TEST 1 – This test verifies the unconditional branch.
The registers and condition codes are all undefined when this test is entered and they should remain that way upon completion of this test.

TEST 2 – Test CLR, MODE 0, and BMI, BVS, BHI, BLOS.
The registers and condition codes are all undefined when this test is entered. Upon completion of this test the SP (R6) should be zero and only the Z flip-flop will be set.

TEST 3 – Test DEC, MODE 0, and BPL, BGE, BGT, BLE.
Upon entering this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 000000.
Upon completion of this test the condition codes will be: N = 1, Z = 0, V = 0, and C = 0.
The registers affected by the test are: SP = 177777.

TEST 4 – Test ROR, MODE 0, and BVC, BHIS, BHI, BNE.
Upon entering this test the condition codes are: N = 1, Z = 0, V = 0, and C = 0.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 177777.
Upon completion of this test the condition codes will be: N = 0, Z = 0, V = 1, and C = 1.
The registers affected by the test are: SP = 077777.

TEST 5 – Test BHI, BLT, and BLOS.
Upon entering this test the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 077777.
Upon completion of this test the condition codes will be: N = 1, Z = 1, V = 1, and C = 1.
The registers are unaffected by the test.

TEST 6 – Test BLE and BGT.
Upon entering this test the condition codes are: N = 1, Z = 1, V = 1, and C = 1.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 077777.
Upon completion of this test the condition codes will be: N = 1, Z = 0, V = 1, and C = 1.
The registers are unaffected by the test.

TEST 7 – Test register data path and modes 2, 3, 6.
When this test is entered the condition codes are: N = 1, Z = 0, V = 1, and C = 1.
The registers are: R0 = ?, R1 = ?, R2 = ?, R3 = ?, R4 = ?, R5 = ?, and SP = 077777.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are left as follows: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 125252, R4 = 125252, R5 = 125252, SP = 125252, and MAPL00 = 125252.

TEST 10 – Test ROL, BCC, BLT, and MODE 6.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 125252, R4 = 125252, R5 = 125252, SP = 125252, and MAPL00 = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are left unchanged except for MAPL00 which should now equal 052524.

TEST 11 – Test ADD, INC, COM, and BCS, BLE.
When this test is entered the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 125252, R4 = 125252, R5 = 125252, SP = 125252, and MAPL00 = 052524.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are left unchanged except for R3 which now equals 000000, and R1 which is also 000000.

TEST 12 – Test ROR, BIS, ADD, and BLO, BGE.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 030000, R4 = 125252, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are left unchanged except for R3 which should be modified back to 000000, and R4 which should now equal 052525.

TEST 13 – Test DEC and BLOS, BLT.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 1, Z = 0, V = 0, and C = 0.
The registers are left unchanged except for R1 which should now equal 177777.

TEST 14 – Test COM, BIC, and BGT, BLE.
When this test is entered the condition codes are: N = 1, Z = 0, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 177777, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
Upon completion of this test the condititon codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are left unchanged except for R0 which should now equal 052525, and R1 which should now equal 052524.

TEST 15 – Test ADC, CMP, BIT, and BNE, BGT, BEQ.
When this test is entered the condition codes are: N = 0, Z = 0, V = 1, and C = 1.
The registers are: R0 = 052525, R1 = 052524, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 125252, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are now: R0 = 052525, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.

TEST 16 – Test MOVB, SOB, CLR, TST and BPL, BNE.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 052525, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
R0 is decremented by an SOB instruction to 000000; R1 is cleared and then incremented around to 000000.

TEST 17 – Test ASR, ASL.
When this test is entered the condition codes are: N = 0, Z = 1, V = 0, and C = 0.
The registers are: R0 = 125252, R1 = 000000, R2 = 125252, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 0, V = 0, and C = 0.
The registers are left unchanged except for R0 which is now equal to 000000, R1 which is now 000001, and R2 which is now 000000.

TEST 20 – Test ASH, and SWAB.
When this test is entered the condition codes are: N = 0, Z = 0, V = 0, and C = 0.
The registers are: R0 = 000000, R1 = 000001, R2 = 000000, R3 = 000000, R4 = 052525, R5 = 052525, and SP = 125252.
Upon completion of this test the condition codes are: N = 0, Z = 1, V = 0, and C = 1.
The registers are left unchanged except for R1 which should now equal 000000.

TEST 21 – Test JSR, RTS, RTI, JMP.
This test first sets the stack pointer to "KDPAR7" (172376), and then verifies that JSR, RTS, RTI, and JMP all work properly.

On entry to this test the stack pointer (SP) is initialized to 172376 and is left that way on exit.

TEST 22 – Test main memory from 1000 to highest available address up to 28K.
This test will test main memory with the cache disabled, from virtual address 001000 to the last address (up to 28K). The memory is sized before testing begins. If the data does not compare properly, the test will halt at either 165516 or 165536. If a parity error occurs, the test will halt at address 165750, with PC + 2 on the stack.

In this test the registers are initialized as follows: R0 = 001000, R1 = DATA READ, R2 = 001000, R3 = 177746 (cache control register), R5 = last memory address, SP = 000776.

The following two tests are cache memory tests. If either of them fails to run successfully it will come to a halt in the M9301 ROM. If you desire to try to boot your system, or diagnostic anyway, you can press continue and the program will force misses in both groups of the cache and go to the bootstrap that has been selected.

TEST 23 – Test cache data memory.
This test will check the data memory in the cache, first group 0 and then group 1. On the PDP-11/60, group 0 is the top 0.5K of cache and group 1 is the bottom 0.5K of cache. It loads 125252 into an address, complements it twice and reads the data. It then checks that address to ensure the data was a hit. The sequence is repeated on the same address with 052525 as the data. All cache memory data locations are tested in this way. If either group fails and the operator presses continue, the program will try to boot with the cache disabled.

The registers are initialized as follows for this test: R0 = 001000 (address), R1 = 000002 (count), R2 = 001000 (count), R3 = 177746 (CCR), R4 = 125252 (pattern), R5 = last memory address, SP = 000776 (flag of zero pushed on stack).

TEST 24 – Test up to 28K of memory with cache on.
This test checks memory from 001000 through 157776 to ensure that you can get hits all the way up through main memory. It starts with group 1 enabled, then tests group 0, and finally checks memory with both groups enabled. If any one of the three passes fails, the test will halt. Then if the operator presses continue, the program will try to boot with the cache disabled.

Upon entry the registers will be set up as follows: R0 = 001000 (address), R1 = 000003 (pass count), R2 = 001000 (first address), R3 = 177746 (CCR), R5 = last memory address, and SP = 000776.

Upon completion of this test, main memory from address 001000 through 157776 will contain its own address.

13.5 INSTALLATION

A flat-pack on the M9301 containing 10 microswitches determines what actions the M9301 ROM program will take. The various microswitch options are provided to give the user a flexibility of operation and an ability to use the module in a wide variety of system configurations. Following is a description of the functions provided by the microswitches. Table 13-1 lists the function of each switch, and more detailed explanations follow.

Table 13-1 Microswitch Functions

Switch	Function
S1-10	ON selects 11/60, OFF selects 11/70
S1-9	ON selects diagnostics, OFF selects no diagnostics
S1-8	OFF boot from device selected in microswitches ON boot from device selected in console switch register
S1-7	LSB]
S1-6	device code
S1-5	device code
S1-4	MSB]
S1-3	OFF selects all diagnostics ON selects all diagnostics except 21–24
S1-2	Power-up reboot enable for 11/70
S1-1	Low ROM enable

Microswitch S1-10
This switch is used to select the processor type. If the M9301 is used on the PDP-11/60 processor, the switch should be ON. If it is a PDP-11/70 processor, the switch should be OFF. OFF is indicated by a 1 (or light lit) when bit 01 at location 773024 (17773024 for 11/70) is read.

Microswitch S1-9
If this switch is OFF, then none of the diagnostic tests can be executed. If the switch is ON, then the ROM program will enter the diagnostic test section. The execution of the memory modifying tests (JSR, RTS, RTI, memory test, cache test) depends on the setting of microswitch S1-3 described later. OFF is a 1 in bit 02 at location 773024.

Microswitch S1-8
If this switch is OFF, the ROM program will not check the console switch register, but will boot from the device specified in the microswitches <7:4>. If this switch is ON, the ROM program will check the console switch register and boot accordingly (Paragraph 13.5). This microswitch option has been provided for installation environments which cannot assure the integrity of the console switch register contents (on power-up). OFF is a 1 in bit 03 at location 773024.

Microswitch <S1-7:S1-4>
These four microswitches should be set up to the device code, to select the desired device to boot from. This device is usually referred to as the default device. Drive 0 of the default device will be used. Paragraph 13.5 gives the various device codes. While setting up switches <7:4> for the device code, it should be remembered that a microswitch in the ON position gives a 0. In the OFF position it gives a 1. Switch 4 is the MSB, and switch 7 is the LSB.

Microswitch S1-3
If this switch is OFF, then the memory-modifying tests (tests 21, 22, 23, 24 described previously) will be executed before booting. If the switch is ON, these tests will not be executed, thus saving the previous core image. Microswitch S1-3 is sensed only if microswitch S1-9 is ON. OFF is a 1 for bit 08 at location 773024.

Microswitch S1-2
When used with a PDP-11/60 processor switch, S1-2 should be OFF. (On the PDP-11/60, if the boot on power-up option is desired, the slide-switch on the console should be set in the boot position.)

On the PDP-11/70, the boot on power-up option is available if the following conditions are met:

1. ECO 5 is installed
2. Jumper W6 is installed, in accordance with ECO 5.

If these conditions are satisfied, then the boot on power-up option can be selected by putting microswitch S1-2 ON. If the conditions are not met, switch S1-2 should be OFF.

Microswitch S1-1
If switch S1-1 is OFF, the low ROM (addresses XXX65000–XXX65776) is disabled. The normal position of switch 1 is ON, since the diagnostic portion of the program resides in the low ROM.

13.6 STARTING PROCEDURE
The bootstrap test program can be started in one of the following ways:

1. Automatically on power-up
2. Pressing the boot switch on the console (only for PDP-11/60)
3. Load address and start sequence from the console.

13.6.1 Power-Up Start

PDP-11/60
On the PDP-11/60, a 3-position slide-switch (BOOT/RUN/HALT) is provided on the console. If this switch is in the boot position and power-up occurs, automatic booting will take place from the device specified in the microswitches (Figure 13-2).

PDP-11/70
If S1-2 is ON, ECO 5 is installed, and jumper W-6 is installed, an automatic boot on power-up will occur from the device specified in the microswitches or console switch register (Figure 13-3).

PDP-11/60 and PDP-11/70
On power-up boot, the default device specified in microswitches <S1-7:S1-4> will be used if:

1. Microswitch S1-8 is OFF or
2. Microswitch S1-8 is ON and the low byte of the console switch register is a zero. (On the PDP-11/60, the console switch register is cleared on power-up.)

Other microswitch options are specified Paragraph 13.4.

Boot Switch Mode
This mode is available only on the PDP-11/60. Automatic booting will take place if a power up occurs when the slide-switch on the console is in the boot position.

13.6.2 Power-Up Boot Examples (11/60 only)

A. TU10 – Set the microswitches as follows:

S1-1	S1-2	S1-3	S1-4	S1-5	S1-6	S1-7	S1-8*	S1-9	S1-10
ON	OFF	OFF	ON	ON	ON	OFF	ON	ON	ON

On power-up, the system will execute all diagnostics and boot the TU10.

B. RK05 – Set the microswitches as follows:

S1-1	S1-2	S1-3	S1-4	S1-5	S1-6	S1-7	S1-8*	S1-9	S1-10
ON	OFF	OFF	ON	ON	OFF	OFF	OFF	OFF	ON

On power-up the system will boot drive 0 of the RK05 without executing diagnostics.

*Note that on the 11/60 the position of S1-8 is irrelevant, because the console switch register is cleared automatically. But if S1-8 is OFF, the program will never look at the console switch register, even if a LOAD ADDRESS and START is attempted. Also, the M9301-YH can default only to drive 0 of any device.

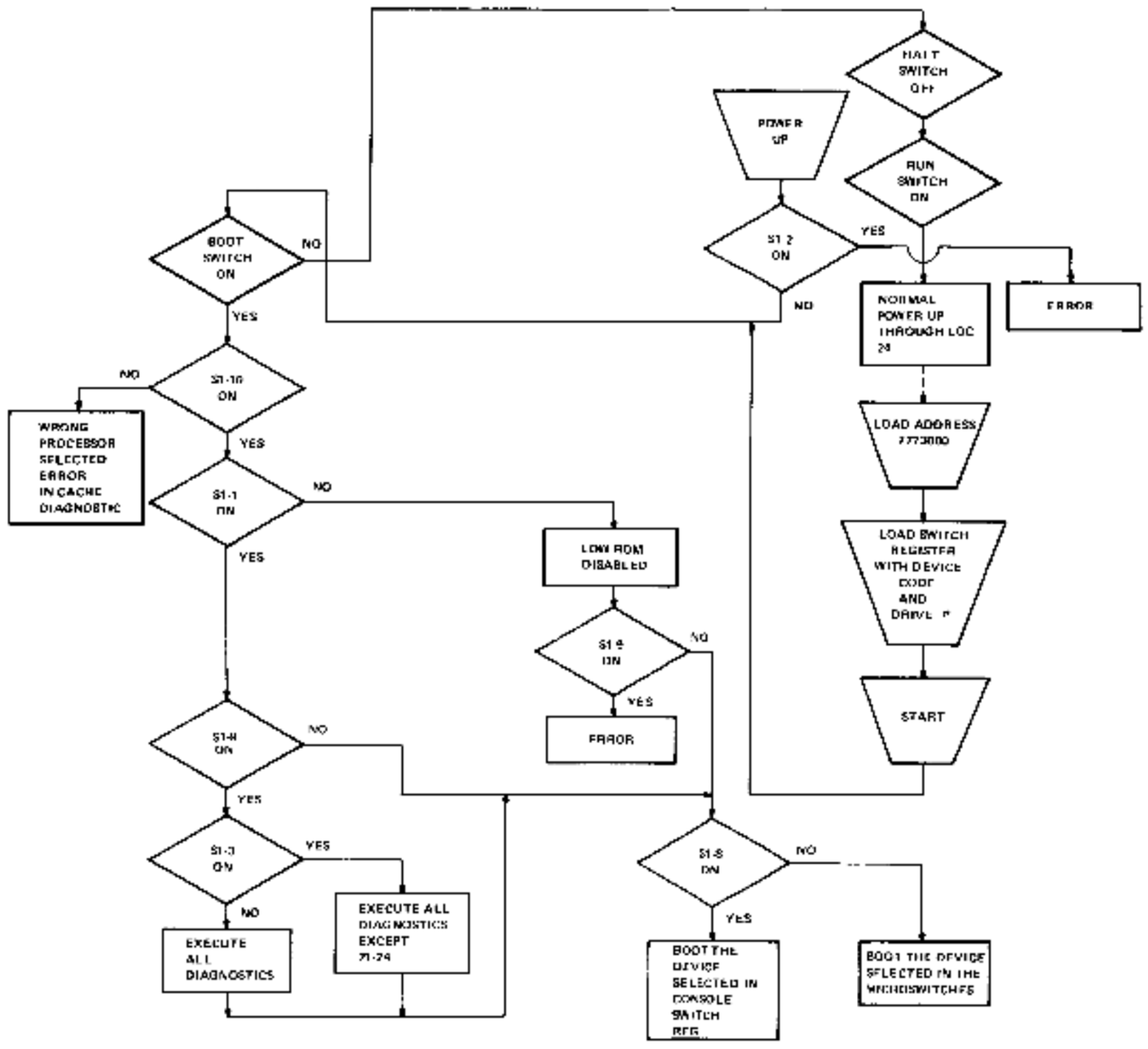

Figure 13-2 M9301-YH
Used With PDP-11/60

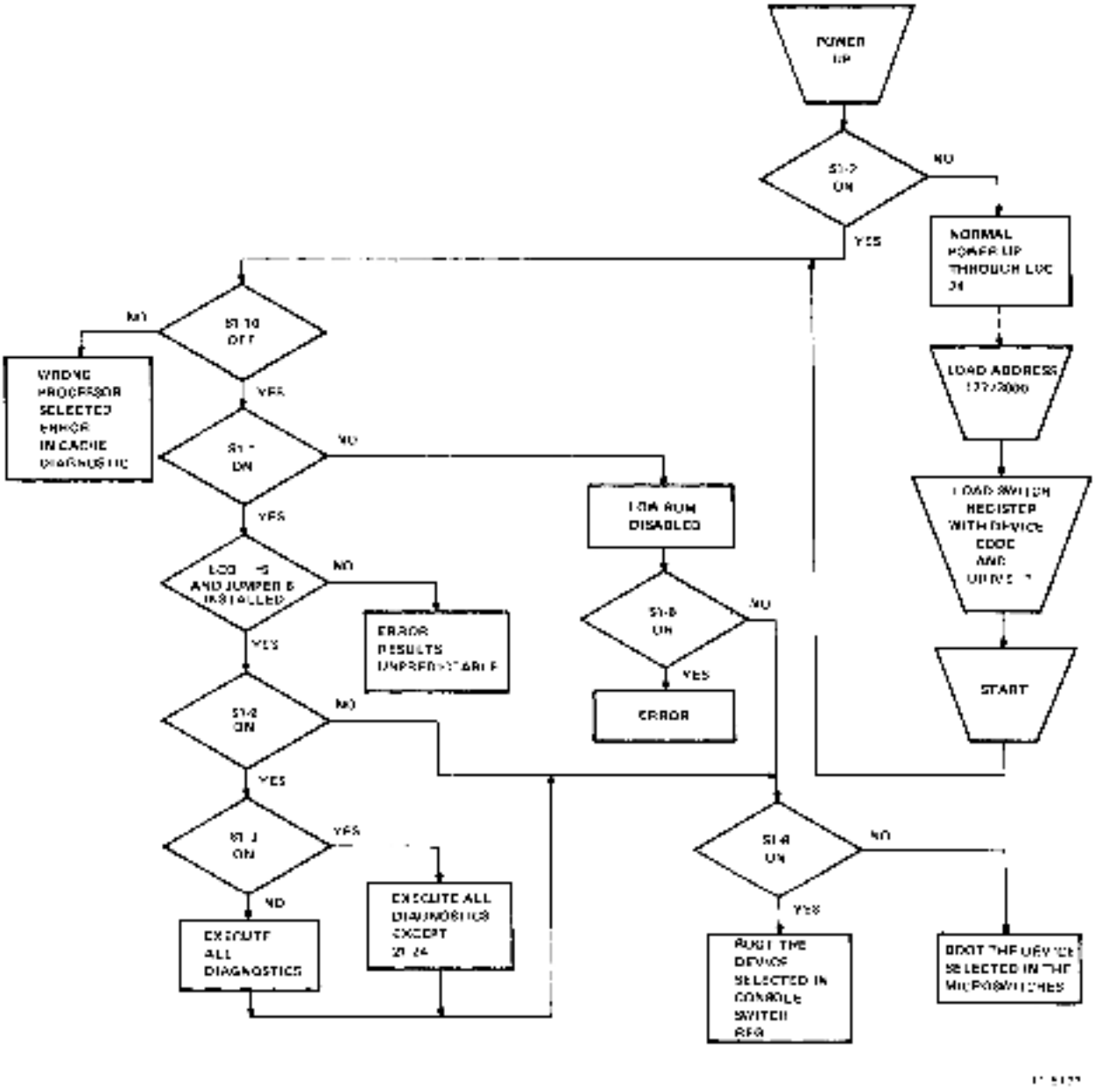

Figure 13-3 M9301-YH Used with PDP-11/70

13.6.3 Load Address Start Mode

This mode allows the user to boot from any device and unit number specified in the console switch register.

On the PDP-11/60, load address 773000. On the PDP-11/70, load address 17773000. Then the switch register should be loaded with the appropriate device code (Figures 13-2 and 13-3).

SW REG <6:3> contains the device code (0–12). SW REG <2:0> contains the drive unit number (0–7). The device codes are as follows:

0	Use the device specified in microswitches <S1-7:S1-4>.	
1	TM11/TU10	Magnetic Tape
2	TC11/TU56	DECtape
3	RK11/RK05	Disk
4	RP11/RP03	Disk
5	RK611/RK06	Disk
6	RH11-RH70/TU16	Magnetic Tape
7	RH11-RH70/RP04	Disk
10	RH11-RH70/RS04	Fixed Head Disk
11	RX11/RX01	Floppy Disk
12	PC11	Paper Tape Reader

In order to use the LOAD ADDRESS and START sequence, microswitch S1-8 must be in the ON position.

13.6.4 LOAD ADDRESS and START Examples (11/70 only)

C. RS04 – Set the microswitches as follows:

S1-1	S1-2	S1-3	S1-4	S1-5	S1-6	S1-7	S1-8	S1-9	S1-10
ON	ON	ON	–	–	–	–	ON	ON	OFF

Load address 17773000.
Load 103_8 in the console switch register.
Start.

The system will boot from drive number 3 of the RS04 disk drive.

13.7 ERRORS

A list of error halts indexed by the address displayed is shown in Table 13-2.

If the bootstrap operation fails as a result of a hardware error in the peripheral device the program will do a reset instruction and attempt to boot again.

13.8 OPERATOR ACTION AND ERROR RECOVERY

If the diagnostic portion of the ROM program detects an error, the processor will halt. The address displayed should be noted. Paragraph 13.6 contains a cross-reference of address displayed versus test number and subsystem under test. For further details, refer to the listing for the ROM program.

Most of the errors described in Table 13-2 are hard failures, and there may be no recovery from them.

If the processor halts in one of the two cache tests, error recovery is possible. When continue is pressed, the program will either attempt to finish the test (XXX65604 or XXX65720) or force misses in both groups of the cache and attempt to boot with the cache fully disabled (XXX6514, XXX65732, or XXX65752).

Table 13-2 Error Halts Indexed

Address Displayed	Test Number and Subsystem Under Test
XXX65004	Test 1, Branch Test
XXX65020	Test 2, CLR and Conditional Branch Test
XXX65036	Test 3, DEC and Conditional Branch Test
XXX65052	Test 4, ROR and Conditional Branch Test
XXX65066	Test 5, Conditional Branch Test
XXX65076	Test 6, Conditional Branch Test
XXX65126	Test 7, Register Data Path Test
XXX65136	Test 10, Conditional Branch Test
XXX65154	Test 11, CPU Instruction Test
XXX65172	Test 12, CPU Instruction Test
XXX65202	Test 13, CPU Instruction Test
XXX65210	Test 14, CPU Instruction Test
XXX65224	Test 14, CPU Instruction Test
XXX65246	Test 15, CPU Instruction Test
XXX65256	Test 16, Branch Test
XXX65300	Test 16, CPU Instruction Test
XXX65334	Test 17, CPU Instruction Test
XXX65352	Test 20, CPU Instruction Test
XXX65376	Test 21, JSR Test
XXX65406	Test 21, JSR Test
XXX65416	Test 21, RTS Test
XXX65430	Test 21, RTI Test
XXX65436	Test 21, JMP Test
XXX65520	Test 22, Main Memory Data Compare Error
XXX65540	Test 22, Main Memory Data Compare Error no recovery possible from this error
XXX65604	Test 23, Cache Memory Data Compare Error
XXX65614	Test 23, Cache Memory No Hit (pressing continue here will cause boot attempt, forcing cache misses)
XXX65720	Test 24, Cache Memory Data Compare Error
XXX65732	Test 24, Cache Memory No Hit (pressing continue here will cause boot attempt, forcing misses in the cache)
XXX65752	Test 22, 23, or 24, Cache Memory or Main Memory Parity Error. (Examine memory error register 777744 to find out more.) If cache parity error, pressing continue here will cause boot attempt forcing misses in cache.

XXX = 001 for PDP-11/60
XXX = 177 for PDP-11/70

If the program fails in an uncontrolled manner, it might be due to an unexpected trap to location 4 or 10. If this is suspected, then load the following:

LOC	CONTENTS
4	6
6	0
10	10
12	0

The above will cause all traps to vectors 4 and 10 to halt the processor at addresses 6 and 12 respectively (with addresses 10 and 14 in the display); the operator can examine the CPU error register at XXX77766 to get more error information. Bits in CPU error register are defined as follows:

Bit 02 = Red Zone Stack Limit
Bit 03 = Yellow Zone Stack Limit (11/70 only)
Bit 04 = Unibus Timeout
Bit 05 = Non-Existent Memory (Cache, 11/70 only)
Bit 06 = Odd Address Error
Bit 07 = Illegal Halt (11/70 only)

CHAPTER 14
M9301-YJ

14.1 INTRODUCTION

The M9301-YJ provides DECnet bootstrapping capabilities for the DMC11 on PDP-11 systems with or without the console switch register. In addition, the M9301-YJ includes a console emulator routine, bootstrap routines for floppy disk, magnetic tape, paper tape, and some basic CPU and memory GO-NO GO diagnostic tests.

This module has been designed for maximum flexibility of operation. Its function may be initiated in any of four ways; automatically at power-up, by pressing the console boot switch, by a LOAD ADDRESS and START sequence, or by using the console terminal while running the console emulator routine.

14.2 PROGRAM MEMORY MAP

Figure 14-1 is a program memory map for the M9301-YJ.

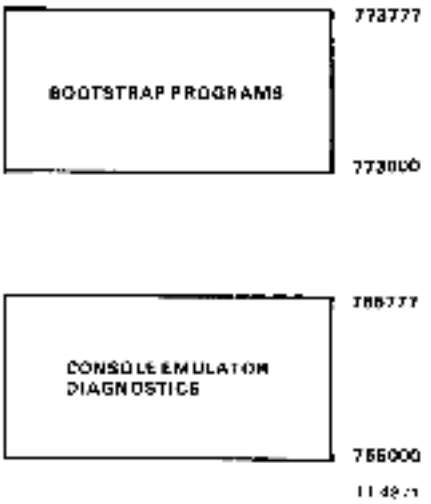

Figure 14-1 M9301-YJ Program Memory Map

The lower addresses (765000–765777) can be disabled by putting switch S1-1 in the OFF position. This would free addresses 765000–765777, allowing the user to occupy this address space. Note that if switch S1-1 is OFF, the console emulator, the low-speed paper tape boot, and diagnostics are no longer available.

Note also that the code in the M9301-YJ is written with the assumption that the processor will trap to Unibus location 24 on power-up if the default boot on power-up feature is to be used. It is also assumed that a power-up trap to Unibus location 24 will be invoked by the assertion and then negation of AC LO if the pushbutton boot feature is to be used.

Note further that the processor priority is left at 7 when the standard boots are exited.

14.3 MICROSWITCH FUNCTIONS

A set of ten microswitches is located on the M9301 module. They determine which ROM routines are selected and give the user automatic access to any function.

The primary activating processes for the M9301-YJ are the power-up sequence and the enabling of the console boot switch. Switch S1-1 must be in the ON position in order to enable activation of the M9301-YJ console emulator and diagnostics. Switch S1-2 is the power-up reboot enable switch. It must be ON to enable to M9301-YJ on power-up. If switch S1-2 is OFF, then the processor will trap to location 24 (as normal) to execute the user power-up routine. When switch S1-2 is ON, the other switches, S1-3 through S1-10, determine what action the M9301-YJ will take on power-up.

If the system supports a console boot switch, then any time that switch is pressed, the M9301-YJ will be activated. Enabling the console boot switch causes the processor to enter a ROM routine by creating a fake power-down/power-up sequence. The user should note that the position of switch S1-2 is irrelevant when the console boot switch is used. Pushing the console boot switch thus results in a power-up sequence in the processor through an M9301-YJ ROM routine.

Prior to the power-up sequence, the M9301-YJ asserts 773000 on the Unibus address lines. This causes the new PC to be taken from ROM location 773024 instead of location 000024. The new PC will be the logical OR of the contents of ROM location 773024 and the eight microswitches on the M9301-YJ module. A switch in the ON position is read as a 0 and a switch in the OFF position is a 1. In this way all the M9301-YJ options are accessible. Each option is given a different address. Note that microswitch S1-10 is ORed with bit 1 of the data in ROM location 773024; S1-9 is ORed with bit 2, etc. No switch is provided for combination with bit 0, because an odd address could result when going through the trap sequence.

The microswitches on the M9301-YJ enable the user either to start a bootstrap operation or to enter the console emulator, simply by pressing the boot switch. Note that a momentary power failure will have the same effect as pushing the boot switch.

The user should also note that he can select any function without diagnostics by adding 2 to the appropriate octal code in the switches. Figure 14-2 shows the relationship of the switches to the bus address bits.

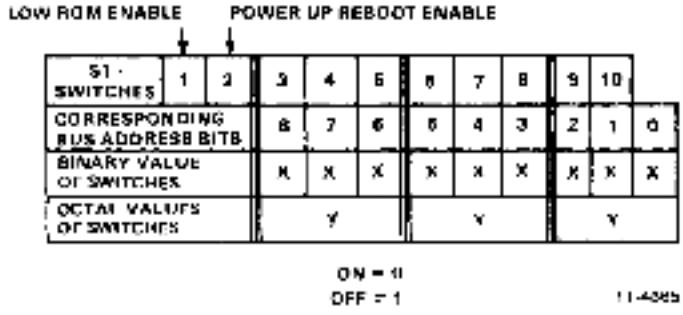

Figure 14-2 Offset Switches and Correspondence Bus Address Bits

14.4 BOOTSTRAPPING PAPER TAPE, FLOPPY DISK, AND MAGNETIC TAPE

The paper tape boot is unique in that it can do no error checking and the secondary bootstrap (e.g., the absolute loader) is read into the upper part of memory. The actual locations loaded by the paper tape boot are partially determined by the secondary bootstrap itself and by the size routine which determines the highest available memory address within the first 28K. The flexible disk (or floppy) reads sector 1 on track 1 into locations starting at zero. The magnetic tape boot reads the second block into locations starting at 0. If no errors are detected in the device, the bootstraps normally transfer control to location 0 in order to execute the secondary bootstrap just loaded. The only exception to this starting address is with the paper tape boot. It transfers control to location XXX374, where XXX is determined initially by the size routine to be at the top of memory; this is where the absolute loader will have just been loaded.

If a device error is detected, a reset will be executed, and the bootstrap will try again. The bootstrap will be retried indefinitely until it succeeds without error unless the operator intervenes. The advantage of retrying the boot is that if a particular device being booted is not on-line or loaded, because of a power failure and restart for example, the boot will give the device a chance to power-up. For disks this is essential. If a magnetic tape is to be booted, the user must reload the maglape and bring it back on-line and restart the boot.

Note that the only way to bootstrap a non-DECnet drive (unit or transport) other than drive 0 is by entering the console emulator to specify that drive number desired. Otherwise the bootstraps will default to drive 0. This means that only drive 0 of a device can be bootstrapped without operator intervention.

14.5 BOOTSTRAPPING THE DMC11-DECNET

The DMC11 bootstrap routine loads memtory in a satellite computer over a communication link, using the maintenance operation protocol (MOP) within the DECnet system. The entire bootstrapping process involves three separate bootstrap programs.

14.5.1 Primary Bootstrap

The primary bootstrap is a mode associated with the DMC11 within the M9301-YJ ROM. It requests a secondary boot from the host computer. The primary bootstrap routine will wait approximately 9 seconds for a reply and then make another request for the secondary boot. If, after eight tries, the host has not responded, the satellite will hang up the phone for approximately one half second. Data terminal ready (DTR) is turned OFF and then ON again. After interrupting the communication link, the primary bootstrap will initiate another series of requests for a secondary boot. The M9301-YJ supports two MOP codes, 8 and 0, as follows. The primary boot sends out a MOP 8 message. It expects a MOP 0 message back. The MOP 0 is the secondary boot which loads the tertiary boot with transfer address.

If the primary boot detects any soft errors, it will make the request again. A soft error results from the detection of either a non-fatal hardware error or a counterfeit message. Detection of a fatal error will cause the M9301-YJ to reinitialize, disable DTR for approximately 1 second, reassert DTR, and begin as it would on a power-up boot.

14.5.2 Secondary Bootstrap

The secondary bootstrap program, once loaded, requests a tertiary boot. These intermediate steps are important because the boot program which handles the actual loading of the operating system is too long and complex to be stored in the ROM. Detection of an error in the secondary boot will cause a reinitialization of the primary boot.

14.5.3 Tertiary Bootstrap

Once the tertiary boot has been requested and loaded, it relocates in upper memory and then begins the down-line loading of the operating system. Figure 14-3 is a flowchart showing the relation of the three bootstrap programs.

14.5.4 MOP Message Formats

The current MOP version (1.2, July 1976) operates within DDCMP maintenance envelopes. The current DDCMP version is 3.03-Dec 1975. All MOP messages are sent in a DDCMP maintenance mode envelope. This basic DDCMP message provides a CRC block check on the MOP data but does not provide any acknowledgement of receipt. It is similar to a numbered data message used for conformance with DDCMP formats and hence software/hardware drivers for DDCMP. Its form is as follows.

SYN, SYN, DLE, COUNT, Q, S, FILL, FILL, ADDR, CRC1, MOP MSG, CRC2

where:

SYN	=	The DDCMP sync characters: (8 bits) 226-synchronous
DLE	=	The boot message header character: 220 (8 bits)
COUNT	=	The 14-bit count field: number of bytes in MOP message (14 bits)
Q	=	The async link control flag: (1 bit, always = 1)
S	=	The select link control flag: (1 bit, always = 1)
FILL	=	A fill character: 000 (8 bits)
ADDR	=	The tributary station address: (for pt-to-pt = 1) (8 bits)
CRC1	=	The header CRC on DLE through ADDR (16-bit CRC-16)
MOP MSG	=	The MOP message described below (COUNT 8-bit quantities)
CRC2	=	The data CRC on MOP message only (16-bit CRC-16)

Notice that the maintenance messages are preceded by 3 or more SYNC characters appropriate to the link used; synchronous – 226.

The MOP message resides in the data field of the maintenance envelope. The form of the MOP message is:

CODE, INFO

where:

CODE	=	The MOP operation code (8 bits)
INFO	=	The information field (specific to each CODE)

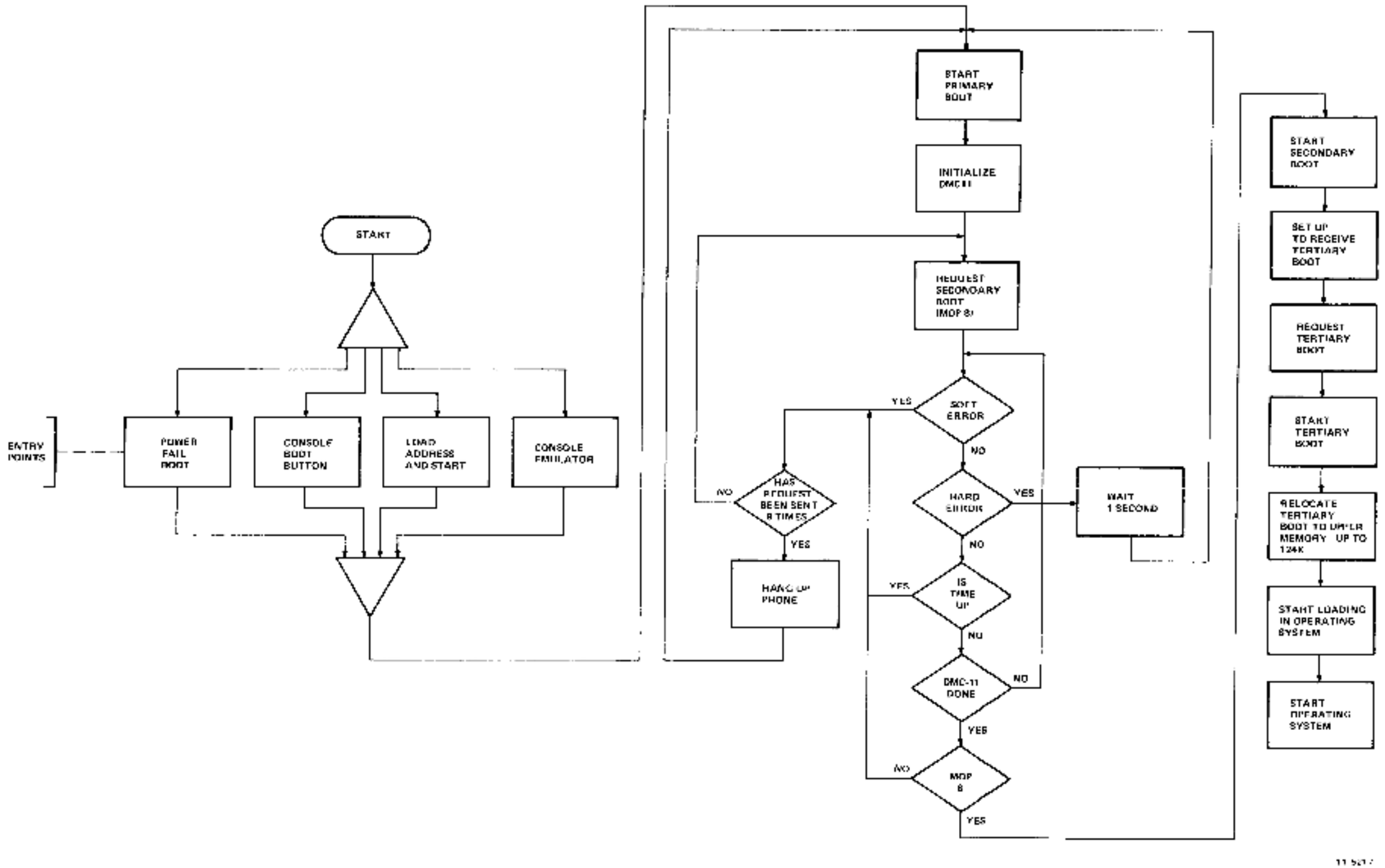

Figure 14-3 M9301-YJ Down Line Load Flowchart

The form of the MOP 8 program (request secondary boot) follows.

CODE (1 byte) = 8
MSG DATA = DEV TYPE 14_8, STA DDR (1), PGM TYPE (0)

where:

DEV TYPE (1 byte)	=	The device type at the requesting system. It is always 14_8 for the DMC11 and the M9301-YJ.
STA DDR (1 byte)	=	The address of the requesting station. For point-to-point this is always 1. This is always 1 for the M9301-YJ.
PGM TYPE (1 byte)	=	The type of secondary program requested. PGM type is always 0 for the M9301-YJ.

The form of the MOP 0 program (memory load with transfer address) follows.

CODE (1 byte) = 0
MSG DATA = LOAD NUM, LOAD ADDR, IMAGE DATA, TRANSFER ADDR

where:

LOAD NUM (1 byte)	=	The load number for multiple load images. It may be preceded by loads without transfer address. This starts at 0 and is incremented for each load. This should always be 0 for the M9301-YJ.
LOAD ADDR (4 bytes)	=	The memory load address (starting address) for the storage of the data image. It should be 0 in MOP 0. This should always be 0 for the M9301-YJ.
IMAGE DATA	=	The image to be stored in the computer memory. The form sent will be machine dependent and may vary with the type and word length of the system [e.g., for PDP-11 systems, each byte represents one memory byte (8 bits)].
TRANSFER ADDR (4 bytes)	=	The starting address of the image just loaded. This is always 6 for the M9301-YJ.

DECnet uses two CPU registers. R0 contains the device index code. R1 contains the CSR address of the device.

14.6 POWER-UP BOOT AND CONSOLE BOOT

When the operator performs a system power-up or presses the console boot switch, he causes the M9301 module to react by initiating a bootstrap routine. Table 14-1 shows the correct microswitch settings for the various options.

Table 14-1 Microswitch Settings

Function	S1-1	2	3	4	5	6	7	8	9	10	Octal Code
Vector 24	–	OFF	–	–	–	–	–	–	–	–	–
Console Emulator with Diag.	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	000
Console Emulator without Diag.	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF	002
Run Non-Memory Modifying Diag. then Vector 24	ON	ON	ON	ON	ON	OFF	ON	OFF	OFF	ON	054
Vector 24	ON	ON	ON	ON	ON	OFF	ON	OFF	OFF	OFF	056
Boot RX01 with Diag.	ON	ON	ON	ON	OFF	OFF	ON	OFF	ON	OFF	152
Boot RX01 without Diag.	–	ON	ON	ON	OFF	OFF	ON	OFF	OFF	ON	154
Boot TU10 with Diag.	ON	ON	ON	OFF	ON	OFF	ON	OFF	OFF	OFF	256
Boot TU10 without Diag.	–	ON	ON	OFF	ON	OFF	OFF	ON	ON	ON	260
Boot DL11 with Diag.	ON	ON	ON	ON	OFF	ON	ON	ON	ON	ON	100
Boot DMC11 Unit 0 with Diag.	ON	ON	ON	OFF	OFF	OFF	ON	OFF	OFF	ON	354
Boot DMC11 Unit 0 without Diag.	–	ON	ON	OFF	OFF	OFF	ON	OFF	OFF	OFF	356
Boot DMC11 Unit 1 with Diag.	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	ON	OFF	372
Boot DMC11 Unit 1 without Diag.	–	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	ON	374

Note that when using the console boot switch, the state of S1-2 has no relevance.

14.7 LOAD ADDRESS AND START PROCEDURE VIA CONSOLE SWITCH REGISTER

The user who wishes to initiate a function other than the one which he has specified in the microswitches can do so without presetting those microswitches. This involves a load address, placing an option code in the console switch register and pressing start; or simply by doing a load address and pressing start at one of the addresses specified below.

The user must load address 173072 and then, before pressing the start switch, he must place a device code or option code in the switch register. Or the user may place the device code or the option code in the console switch register and then press load address and start. These options and codes are shown in Table 14-2.

Table 14-2 Load Address and Start Codes

Code	Function
173000*	To enter console emulator after running primary diagnostics
173004*	To enter console emulator without diagnostics
173152*	To boot the RX01 with diagnostics
173154*	To boot the RX01 without diagnostics
173256*	To boot the TC10 with diagnostics
173260	To boot the TC10 without diagnostics
173100*	To boot the DL11 paper tape with diagnostics
173354*	To boot the DMC11 unit 0 with diagnostics
173356	To boot the DMC11 unit 0 without diagnostics
173372*	To boot the DMC11 unit 1 with diagnostics
173374	To boot the DMC11 unit 1 without diagnostics

*Switch S1-1 must be on.

14.8 OPERATION OF THE M9301-YJ CONSOLE EMULATOR

The user can boot a program from any of four peripheral devices by calling the bootstrap program for that peripheral through the console emulator.

The first routines to be executed (if 000 is the contents of the microswitches) are the primary CPU diagnostics.

The display routine is entered automatically. This routine will type the contents of R0, R4, R6, and R5 (note the sequence) on the Teletype or terminal in octal. Pressing the console boot switch causes the PDP-11/04 and PDP-11/34 systems to copy the PC into R5 before the power-up sequence starts. The console emulator (keyboard dispatch) program is then entered automatically. This routine types a carriage return, a line feed and then the prompt symbol, $. The user can call bootstrap programs at this point by typing the appropriate device code. He then types an octal number following the device code specifying the drive number (default 0) and hits the carriage return key. Table 14-3 lists the peripheral bootstrap programs supported by the M9301-YJ.

Note that 12 console fill characters are used between the carriage return and the line feed in the console emulator routine.

Table 14-3 Bootstrap Programs

Device	Description	Command*
DL11**	Low Speed Paper Tape Reader	TT
DMC11	Synchronous Line Interface	XM#
RX11	RX01 Floppy Disk Drive	DX#
TM11	TU10 800 BPI Magtape	MT#

*The # specifies unit number (default 0). It can be 0 or 1 for DX, 0–7 for MT, and 0–17 for XM.
**This device is not bootable if S1-1 is off.

14.9 RESTARTING AT THE USER POWER FAIL ROUTINE

If the user wishes to restart his own software on a power-up, he may do so by merely disabling the power fail restart switch in the microswitches (turn S1-2 OFF).

But the user can use the M9301-YJ to run diagnostics (just the primary CPU tests described in Paragraph 14.10.) before running his power-up routine. This will in no way disturb the contents of memory and will in fact verify the machine's basic integrity after the power-down and -up sequence.

To use this option the operator should load the code 054 into the microswitches as described in Paragraph 14.7. Be sure that switch S1-2 is ON. This will result in the running of the primary CPU diagnostics and then a simulated trap through 24 which will start the user's specified power-up routine.

If the code 056 is placed in the microswitches then the simulated trap through 24 is executed without running diagnostics.

Figure 14-4 is an operator's flowchart showing the relation of the various options and codes available to the user.

14.10 DIAGNOSTICS

An explanation of the eight CPU and memory diagnostic tests follows. Three types of tests are included in the M9301-YJ diagnostics:

1. Primary CPU tests (1–5)
2. Secondary CPU tests (6, 7)
3. Memory test (8)

14.10.1 CPU Tests

The primary CPU tests exercise all unary and double operand instructions with all source modes. These tests do not modify memory. If a failure is detected, a branch-self (BR.) will be executed. The run light will stay ON, because the processor will hang in a loop. If no failure is detected in tests 1–5, the processor will emerge from the last test and enter the register display routine (console emulator). Note that the CRC and LRC for diagnostic DZM9A are XXXXXX and XXXXXX.

TEST 1 – SINGLE OPERAND TEST

This test executes all single operand instructions using destination mode 0. The basic objective is to verify that all single operand instructions operate: it also provides a cursory check on the operation of each instruction, while ensuring that the CPU decodes each instruction in the correct manner.

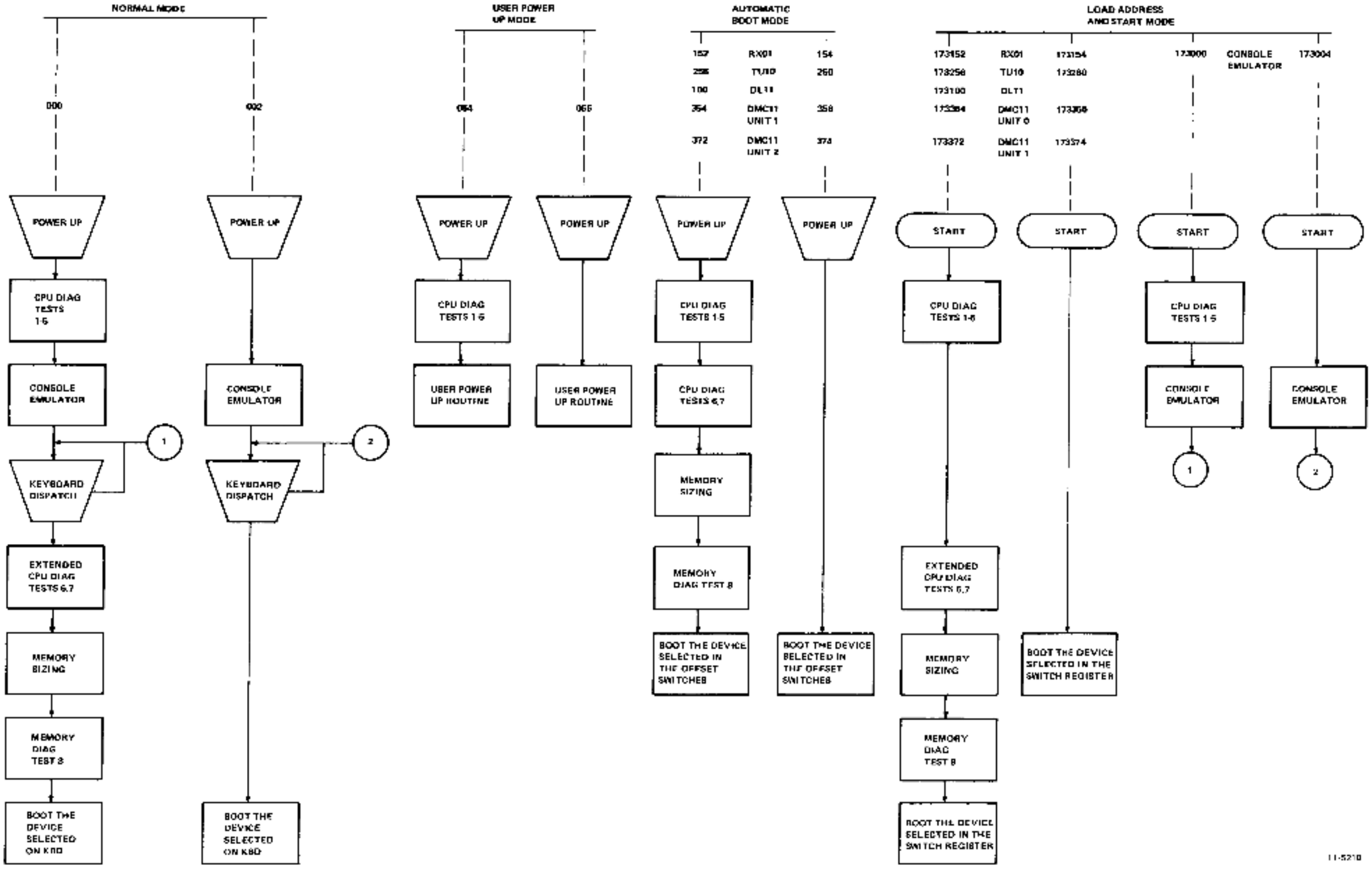

Figure 14-4 M9301-YJ Operator's Flowchart

r

Test 1 tests the destination register in its three possible states: zero, negative, and positive. Each instruction operates on the register contents in one of four ways:

1. Data will be changed via a direct operation, i.e., increment, clear, decrement, etc.

2. Data will be changed via an indirect operation, i.e., arithmetic shifts, add carry, and subtract carry.

3. Data will be unchanged, but operated upon via a direct operation, i.e., clear a register already containing zeros.

4. Data will be unchanged but examined via a non-modifying instruction (TEST).

NOTE

When operating upon data in an indirect manner, the data is modified by the state of the appropriate condition code. Arithmetic shift will move the C bit into or out of the destination. This operation, when performed correctly, implies that the C bit was set correctly by the previous instruction. There are no checks on the data integrity prior to the end of the test. However, a check is made on the end result of the data manipulation. A correct result implies that all instructions manipulated the data in the correct way. If the data is incorrect, the program will hang in a program loop until the machine is halted.

TEST 2 – DOUBLE OPERAND, ALL SOURCE MODES

This test verifies all double operand, general, and logical instructions, each in one of the seven addressing modes (excludes mode 0). Thus, two operations are checked: the correct decoding of each double operand instruction, and the correct operation of each addressing mode for the source operand.

Each instruction in the test must operate correctly in order for the next instruction to operate. This interdependence is carried through to the last instruction (bit test) where, only through the correct execution of all previous instructions, a data field is examined for a specific bit configuration. Thus, each instruction prior to the last serves to set up the pointer to the test data.

Two checks on instruction operation are made in test 2. One check, a branch on condition, is made following the compare instruction, while the second is made as the last instruction in the test sequence.

Since the GO-NO GO test resides in ROM memory, all data manipulation (modification) must be performed in destination mode 0 (register contains data). The data and addressing constants used by test 2 are contained within the ROM.

It is important to note that two different types of operations must execute correctly in order for this test to operate:

1. Those instructions that participate in computing the final address of the data mask for the final bit test instruction.

2. Those instructions that manipulate the test data within the register to generate the expected bit pattern.

Detection of an error within this test results in a program loop.

TEST 3 – JUMP TEST MODES 1, 2, 3
The purpose of this test is to ensure correct operation of the jump instruction. This test is constructed such that only a jump to the expected instruction will provide the correct pointer for the next instruction.

There are two possible failure modes that can occur in this test:

1. The jump addressing circuitry will malfunction, causing a transfer of execution to an incorrect instruction sequence or non-existent memory.

2. The jump addressing circuitry will malfunction in such a way as to cause the CPU to loop.

The latter case is a logical error indicator. The former, however, may manifest itself as an after-the-fact error. For example, if the jump causes control to be given to other routines within the M9301, the interdependent instruction sequences would probably cause a failure to eventually occur. In any case, the failing of the jump instruction will eventually cause an out of sequence or illogical event to occur. This in itself is a meaningful indicator of a malfunctioning CPU.

This test contains a jump mode 2 instruction which is not compatible across the PDP-11 line. However, it will operate on any PDP-11 within this test, due to the unique programming of the instruction within test 3. Before illustrating the operation, it is important to understand the differences of the jump mode 2 between machines.

On the PDP-11/20, 11/05, 11/15, and 11/10 processors, for the jump mode 2 [JMP(R)+] the register (R) is incremented by 2 prior to execution of the jump. On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, (R) is used as the jump address and incremented by 2 after execution of the jump.

In order to avoid this incompatability, the jump (R)+ is programmed with (R) pointing back on the jump itself. On 11/20, 11/05, 11/10, and 11/15 processors, execution of the instruction would cause (R) to be incremented to point to the following instruction, effectively continuing a normal execution sequence.

On the PDP-11/04, 11/34, 11/35, 11/40, 11/45, 11/50, 11/55, and 11/70 processors, the use of the initial value of (R) will cause the jump to loop back on itself. However, correct operation of the autoincrement will move (R) to point to the next instruction following the initial jump. The jump will then be executed again. However, the destination address will be the next instruction in sequence.

TEST 4 – SINGLE OPERAND, NON-MODIFYING BYTE TEST
This test focuses on one unique single operand instruction, the TST. TST is a special case in the CPU execution flow since it is a non-modifying operation. Test 4 also tests the byte operation of this instruction. The TSTB instruction will be executed in mode 1 (register deferred) and mode 2 (register deferred, autoincrement).

The TSTB is programmed to operate on data which has a negative value most significant byte and a zero (not negative) least significant byte.

In order for this test to operate properly, the TSTB on the low byte must first be able to access the even addressed byte and then set the proper condition codes. The TSTB is then reexecuted with the autoincrement facility. After the autoincrement, the addressing register should be pointing to the high byte of the test data. Another TSTB is executed on what should be the high byte. The N bit of the condition codes should be set by this operation.

Correct execution of the last TSTB implies that the autoincrement feature recognized that a byte operation was requested, thereby only incrementing the address in the register by one, rather than two. If the correct condition code has not been set by the associated TSTB instruction, the program will loop.

TEST 5 - DOUBLE OPERAND, NON-MODIFYING TEST
Two non-modifying, double-operand instructions – the compare (CMP) and bit test (BIT) – operate on test data in source modes 1 and 4, and destination modes 2 and 4.

The BIT and CMP instructions will operate on data consisting of all ones (177777). Two separate fields of ones are used in order to utilize the compare instructions, and to provide a field large enough to handle the autoincrementing of the addressing register.

Since the compare instruction is executed on two fields containing the same data, the expected result is true Z bit, indicating equality.

The BIT instruction will use a mask argument of all ones against another field of all ones. The expected result is a non-zero condition (Z).

Most failures will result in a one instruction loop.

At the end of test 5 the register display routine is enabled, provided the console emulator has been selected in the microswitches. The register display routine prints out the octal contents of the CPU registers R0, R4, SP, and old PC on the console terminal. This sequence will be followed by a prompt character (S) on the next line.

An example of a typical printout follows.

	XXXXXX	XXXXXX	XXXXXX	XXXXXX
S				
Prompt Character	R0	R4	R6 (Stack Pointer)	R5 (Old PC)

NOTES:

1. Where X signifies an octal number (0–7).
2. Whenever there is a power-up routine or the BOOT switch is released on PDP-11/04 and PDP-11/34 machines, the PC at this time will be stored in R5. The contents of R5 are then printed as the old PC shown in the example.
3. The prompting character string indicates that diagnostics have been run and the processor is operating.

14.10.2 Secondary CPU and Memory Tests
The secondary CPU tests modify memory and involve the use of the stack pointer. The JMP and JSR instructions and all destination modes are tested. If a failure is detected, these tests, unlike the primary tests, will execute a halt.

Secondary CPU and memory diagnostics are run immediately after test 5 when they have been evoked by means other than the console emulator, provided that the correct microswitches have been set. If the console emulator has been entered at the completion of test 5, the secondary CPU and memory diagnostics will be run when the appropriate boot command is given.

TEST 6 – DOUBLE OPERAND, MODIFYING BYTE TEST
The objective of this test is to verify that the double-operand, modifying instructions will operate in the byte mode. Test 6 contains three subtests:

1. Test source mode 2, destination mode 1, odd and even bytes
2. Test source mode 3, destination mode 2
3. Test source mode 0, destination mode 3, even byte.

The move byte (MOVB), bit clear byte (BICB), and bit set byte (BISB) are used within test 6 to verify the operation of the modifying double-operand functions.

Since modifying instructions are under test, memory must be used as a destination for the test data. Test 6 uses location 500 as a destination address. Later, in test 7 and the memory test, location 500 is used as the first available storage for the stack.

Note that since test 6 is a byte test, location 500 implies that both 500 and 501 are used for the byte tests (even and odd, respectively). Thus, in the word of data at 500, odd and even bytes are caused to be all 0s and then all 1s alternately throughout the test. Each byte is modified independently of the other.

TEST 7 – JSR TEST
The JSR is the first test in the GO-NO GO sequence that utilizes the stack. The jump subroutine command (JSR) is executed in modes 1 and 6. After the JSR is executed, the subroutine which was given control will examine the stack to ensure that the correct data was placed in the correct stack location (500). The routine will also ensure that the line back register points to the correct address. Any errors detected in this test will result in a halt.

TEST 8 – DUAL ADDRESSING AND DATA CHECK
Finally the memory test performs both dual addressing and a data check of all the available memory on the system below 28K. This test will leave all of memory clear. Like the secondary tests the memory test will halt when an error is detected. At the time the memory error halt is executed, R4 will contain the address at which the failure was detected plus two, R0 will contain the failing data pattern and R6 will contain the expected data pattern. Thus after a memory failure has occurred the user can enter the console emulator and have this information printed out immediately by the display routine (see section on console emulator).

14.11 TROUBLESHOOTING
When a halt occurs, the user should reboot the system by pressing the **BOOT/INIT** switch. The registers R0, R4, R6, and R5 will be displayed on the terminal in that order.

R0	Failing (received) data
R4	Failing address
R6	Expected data
R5	Old PC

The diagnostic program in the M9301-YJ will cause the processor to halt at one of four addresses: 165600, 165614, 165630, or 165776. The user should consult the diagnostic program listing to find the failing test and begin troubleshooting. Possible causes of the failure include bus errors, a bad M9301 module, and a bad CPU.

14.12 FLOATING DEVICE PRIORITY

The M9301-YJ assumes the following priority of devices in the floating address space.

1. DJ11
2. DH11
3. DQ11
4. DU11
5. DUP11
6. LK11-A
7. DMC11

The M9301-YJ will not function properly unless the above priority and correct Unibus address spacing is followed.

Reader's Comments

M9301 BOOTSTRAP/TERMINATOR MODULE
MAINTENANCE AND OPERATOR'S MANUAL
EK-M9301-TM-001

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

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

What features are most useful? _____

What faults do you find with the manual? _____

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

Does it satisfy your needs? _____ Why? _____

Would you please indicate any factual errors you have found. _____

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

— — — — — — — — — — — — Fold Here — — — — — — — — — — — —

— — — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — — —

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL.
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
Maynard, Massachusetts 01754

digital equipment corporation

Printed in U.S.A.