The Engineering Staff of
TEXAS INSTRUMENTS INCORPORATED
Semiconductor Group

TM 990/301 MICROTERMINAL

JANUARY 1980

TEXAS INSTRUMENTS
INCORPORATED

MANUAL HISTORY

This manual contains the following revisions:

Date	Revision Change (From - to)	ECN Number
12/18/79	A to B	454304

IMPORTANT NOTICES

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

TM 990/301 MICROTERMINAL

1. GENERAL

The Texas Instruments Microterminal offers all of the features of a minicomputer front panel at reduced cost The Microterminal, intended primarily to support the Texas Instruments TM 990/1XXM microcomputers, al lows the user to do the following:

- Read from ROM or read/write to RAM
- Enter/display Program Counter
- Execute user program in free running mode or in single instruction mode
- Halt user program execution
- Enter/display Status Register
- Enter/display Workspace Pointer (this term is unique to the Texas Instruments 9900 microprocessor)
- Enter/display CRU data (this term is unique to the Texas Instruments 9900 microprocessor)
- Convert hexadecimal quantity to signed decimal quantity
- Convert signed decimal quantity to hexadecimal quantity

2. SPECIFICATIONS

- Power Requirements
  +12V (±3%), 50 mA
  −12V (±3%), 50 mA
  +5V (±3%), 150 mA
- Operating Temperature: 0°C to 50°C (+32° to +122°F)
- Operating Humidity: 0 to 95 percent, non-condensing
- Shock: Withstand 2 foot vertical drop

3. INSTALLATION AND STARTUP

To install the Microterminal onto a TM 990/100M, TM 990/101M or TM 990/180M microcomputer, do the following:

- Attach jumpers to:
  - On TM 990/100M: J13, J14, and J15, and set J7 to EIA position
  - On TM 990/101M: E20-E21, E22-E23, and E24-E25
  - On TM 990/180M: J4, J5, and J6, and set J13 to EIA position.
- Attach the EIA cable from the Microterminal to connector P2. Signals between the Microterminal and the microcomputer are listed as in Table 1.
- To initialize the system, actuate the microcomputer RESET switch, then press the microterminal $\boxed{\text{CLR}}$ key.

NOTE

If the user has installed the *optional* filter capacitor on the $\overline{\text{RESTART}}$ input, this capacitor must be removed for proper operation (e.g., if C5 is installed on the TM 990/100M or TM 990/180M microcomputer, this capacitor must be removed).

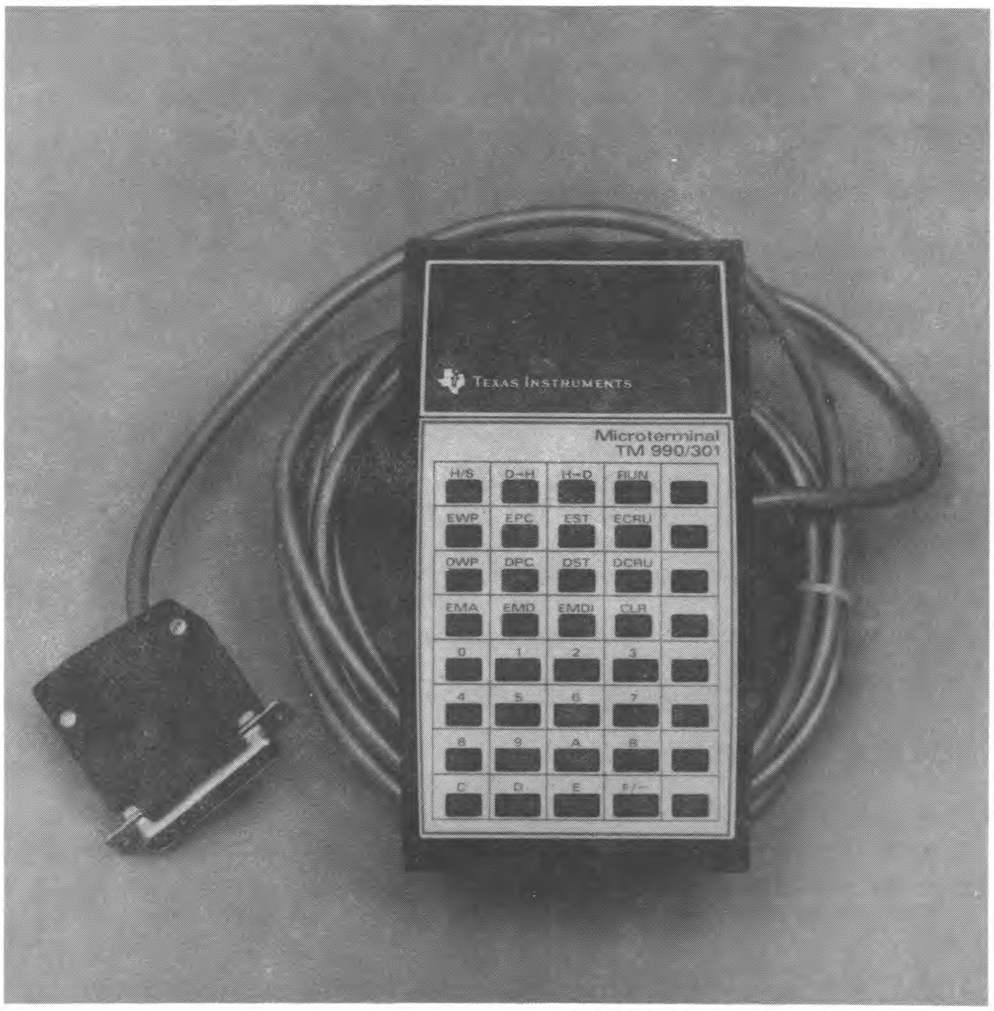


FIGURE 1. TM 990/301 MICROTERMINAL

TABLE 1. EIA CABLE SIGNALS

EIA Connector Pin	Interface Signal	At TM 990/100M/180M	
		P2 Pin	Signal
2	$\overline{\text{TERMINAL DATA OUT}}$	−2	RS232 RCV
3	$\overline{\text{TERMINAL DATA IN}}$	−3	RS232 XMT
7	GND	−7	GND
12	+12V	−12	+12V
13	−12V	−13	−12V
14	+ 5V	−14	+ 5V
16	$\overline{\text{HALT}}$	−16	$\overline{\text{RESTART}}$

CAUTION

Before attaching the Microterminal to a power source, verify voltage levels between ground and EIA connector pins 12, 13, and 14 at connector P2 on the board. Voltage should not exceed values in Table 1.

4. KEY DEFINITIONS

4.1 DATA KEYS

CLR Clear Key – Depressing this key blanks display, initializes and sends initialization message (ASCII code for A and ASCII code for Z) to host microcomputer.

0
1
.
.
F/– Hexadecimal Data Keys – Data is entered with the most significant digit (MSD) first. Depressing any one of these keys shifts that value into the right-hand display digit. All digits already in the data display are left shifted. For all operations other than decimal to hexadecimal conversion, the fourth digit from the right is shifted off the end of the right-hand display field when a data key is depressed. For a decimal to hexadecimal conversion, the fifth display digit from the right, rather than the fourth, is shifted off the end of the data field.

4.2 INSTRUCTION EXECUTION

H/S Pressing this key while a program is running (run displayed) will halt program execution. The address of the next instruction will be displayed in the four left-hand display digits, and the contents of that address will be displayed in the four right-hand digits. Pressing this key while the program is halted, will execute a single instruction using the values in the Workspace Pointer (WP), Program Counter (PC), and Status Register (ST), and the displays will be updated to the next memory address and contents at that address.

RUN Pressing this key initiates program execution at the current values in the WP, PC; run is displayed in the three right-hand display digits.

4.3 ARITHMETIC

H→D The signed hexadecimal data contained in the four right-hand display digits is converted to signed decimal data. Note that the most significant bit of the fourth display digit from the right is the sign bit (1 = negative). The conversion limits are minus $32{,}768_{10}$ (8000_{16}) to plus 32,767 ($7FFF_{16}$). Two H→D key depressions are required. The sequence is:

1. Depress H→D .
2. Enter data via four hex data key depressions.
3. Depress H→D . The results of the conversion are displayed in the five right-hand display digits.

D→H The decimal data contained in the five right-hand display digits is converted to hexadecimal. The conversion limits are the same as for hexadecimal to decimal conversion. If the decimal number is negative, press the F/– key first to begin with a negative sign. The sequence is:

1. Depress D→H .
2. Enter data via hex data key depressions.
3. Depress D→H . The results of the conversion are displayed in the four right-hand display digits.

4.4 REGISTER ENTER/DISPLAY

$\boxed{\text{EWP}}$ Pressing this key causes the value displayed in the four right-hand digits to be entered into the WP.

$\boxed{\text{DWP}}$ Pressing this key causes the WP contents to be displayed in the four right-hand display digits.

$\boxed{\text{EPC}}$ Pressing this key causes the value displayed in the four right-hand digits to be entered into the PC.

$\boxed{\text{DPC}}$ Pressing this key causes the PC contents to be displayed in the four right-hand display digits.

$\boxed{\text{EST}}$ Pressing this key causes the value displayed in the four right-hand digits to be entered into the ST.

$\boxed{\text{DST}}$ Pressing this key causes the ST contents to be displayed in the four right-hand display digits.

4.5 CRU DISPLAY/ENTER

$\boxed{\text{DCRU}}$ Pressing this key causes the data at the designated Communications Register Unit (CRU) addresses to be displayed. Designate from one to 16 CRU bits at a specified CRU address by using four hexadecimal digits. The first digit is the count of bits to be displayed. The next three digits are the CRU bit address (equal to bits 3 to 14 in register 12 for CRU addressing). When $\boxed{\text{DCRU}}$ is depressed, the bit count and address are shifted to the left-hand display, and the right-hand display will contain the values at the selected CRU output addresses. The output value will be zero-filled on the left, depending upon bit count entered. If less than nine bits, the value will be contained in the left two hexadecimal digits. If nine or more, the value will be right justified in all four hexadecimal digits

$\boxed{\text{ECRU}}$ Pressing this key enters a new data value at the CRU addresses and bit count shown in the left display after depressing $\boxed{\text{DCRU}}$. The new value is entered from the keyboard and displayed in the right-hand display. Pressing $\boxed{\text{ECRU}}$ enters this value onto the CRU at the address shown in the left display.

CAUTION

Avoid setting new values at the TMS 9902 on the TM 990/100M/180M through the CRU (TMS 9902 is at CRU address 0040_{16}), as this device controls I/O functions.

4.6 MEMORY ENTER, DISPLAY, INCREMENT

$\boxed{\text{EMA}}$ Pressing this key will cause (1) the memory address (MA) in the right-hand display to be shifted to the left-hand display and (2) the contents of that memory address to be displayed in the right-hand display.

$\boxed{\text{EMD}}$ Pressing this key causes the value in the right-hand display to be entered into the memory address contained in the left-hand display. The contents of that location will then be displayed in the four right-hand display digits (entered then read back).

$\boxed{\text{EMDI}}$ Pressing this key causes the same action as described for the $\boxed{\text{EMD}}$ key; it also increments the memory address by two and displays the contents at that new address. The memory address is displayed on the left and the contents at that address is displayed on the right.

5. EXAMPLES

5.1 EXAMPLE 1, ENTER PROGRAM INTO MEMORY

Enter the following program starting at RAM location $FE00_{16}$. Set the workspace pointer to $FF00_{16}$ and the status register to 2000_{16}. Single step through the program and verify execution. Then execute the program in free run mode and verify execution. Then halt program execution.

NOTE

In the following examples, XXXX indicates memory contents at current value in Memory Address Register.

OPCODE	INSTRUCTIONS		
04C0	CLR	R0	CLEAR WORKSPACE REGISTER 0
0580	INC	R0	INCREMENT WORKSPACE REGISTER 0
0280	CI	R0, >00FF	CHECK FOR COUNT 255
00FF			
16FC	JNE	$-6	JUMP TO INC RO IF NOT DONE
10FF	JMP	$-0	STAY HERE WHEN FINISHED

	KEY ENTRIES		DISPLAY	
Clear Display	Depress	CLR		
Enter PC Value	Depress	F/– E 0 0		FE00
Enter into PC	Depress	EPC		FE00
Display PC	Depress	DPC		FE00
Enter ST Value	Depress	2 0 0 0		2000
Enter into ST	Depress	EST		2000
Display ST	Depress	DST		2000
Enter WP Value	Depress	F/– F/– 0 0		FF00
Enter Into WP	Depress	EWP		FF00
Display WP	Depress	DWP		FF00
Enter MA Value	Depress	F/– E 0 0		FE00
Enter Into MA	Depress	EMA	FE00	xxxx
Enter CLR 0 Opcode	Depress	0 4 C 0	FE00	04C0
Enter data, increment MA	Depress	EMDI	FE02	xxxx
Enter INC 0 Opcode	Depress	0 5 8 0	FE02	0580
Enter Data, Increment MA	Depress	EMDI	FE04	xxxx
Enter CI Opcode	Depress	0 2 8 0	FE04	0280
Enter Data, Increment MA	Depress	EMDI	FE06	xxxx

		KEY ENTRIES	DISPLAY
ENTER CI Immediate Operand	Depress	$\boxed{0}\boxed{0}\boxed{F}\boxed{F}$	$\boxed{FE06}\boxed{00FF}$
Enter Data, Increment MA	Depress	$\boxed{EMDI}$	$\boxed{FE08}\boxed{xxxx}$
Enter JNE $-6 Opcode	Depress	$\boxed{1}\boxed{6}\boxed{F}\boxed{C}$	$\boxed{FE08}\boxed{16FC}$
Enter Data, Increment MA	Depress	$\boxed{EMDI}$	$\boxed{FE0A}\boxed{xxxx}$
Enter JMP $-0 Opcode	Depress	$\boxed{1}\boxed{0}\boxed{F}\boxed{F}$	$\boxed{FE0A}\boxed{10FF}$
Enter Data, Increment MA	Depress	$\boxed{EMDI}$	$\boxed{FE0C}\boxed{xxxx}$

The program has now been entered into RAM. Since the PC, ST and WP values have been previously set, the program can be executed in single step mode by depressing the H/S key.

		DISPLAY (AFTER)	EXECUTES INSTRUCTION
Depress	$\boxed{H/S}$	$\boxed{FE02}\boxed{0580}$	CLR RO
Depress	$\boxed{H/S}$	$\boxed{FE04}\boxed{0280}$	INC RO
Depress	$\boxed{H/S}$	$\boxed{FE08}\boxed{16FC}$	CI RO,>00FF
Depress	$\boxed{H/S}$	$\boxed{FE02}\boxed{0580}$	JNE $–6

This cycle will continue until R0 reaches the count of 255 at which point the program will continuously execute at location $FE0A_{16}$ because it is a jump to itself.

To verify this, depress:	DISPLAY
$\boxed{RUN}$	$\boxed{}\boxed{r\ u\ n}$

The program should now be "looping to self" at location $FE0A_{16}$. To verify this, depress:

$\boxed{H/S}$	$\boxed{FE0A}\boxed{10FF}$

Now examine the memory location corresponding to Register 0.

Depress	$\boxed{F}$ $\boxed{F}$ $\boxed{0}$ $\boxed{0}$	$\boxed{FE0A}\boxed{FF00}$
Depress	$\boxed{EMA}$	$\boxed{FF00}\boxed{00FF}$

This illustrates that FF_{16} did become the final contents of WP0. Note that, when the program was being entered into RAM, $\boxed{EMDI}$ was used rather than $\boxed{EMD}$ because of the rather desirable feature of automatic address incrementing. The advantage of using $\boxed{EMD}$ is that the actual contents of the addressed memory location are displayed after key depression (echoed back after being entered).

5.2 EXAMPLE 2, HEXADECIMAL TO DECIMAL CONVERSIONS

Convert 8000_{16} to a decimal number

Depress	$\boxed{\text{CLR}}$		
Depress	$\boxed{\text{H}\rightarrow\text{D}}$		
Depress	$\boxed{8}$ $\boxed{0}$ $\boxed{0}$ $\boxed{0}$		8000
Depress	$\boxed{\text{H}\rightarrow\text{D}}$	−3	2768

Convert 0020_{16} to a decimal number

Depress	$\boxed{\text{CLR}}$		
Depress	$\boxed{\text{H}\rightarrow\text{D}}$		
Depress	$\boxed{2}$ $\boxed{0}$		20
Depress	$\boxed{\text{H}\rightarrow\text{D}}$		32

5.3 EXAMPLE 3, DECIMAL TO HEXADECIMAL CONVERSIONS

Convert 45_{10} to hex

Depress	$\boxed{\text{CLR}}$		
Depress	$\boxed{\text{D}\rightarrow\text{H}}$		
Depress	$\boxed{4}$ $\boxed{5}$		45
Depress	$\boxed{\text{D}\rightarrow\text{H}}$		2D

Convert -1024_{10} to hex

Depress	$\boxed{\text{CLR}}$		
Depress	$\boxed{\text{D}\rightarrow\text{H}}$		
Depress	$\boxed{\text{F/}-}$ $\boxed{1}$ $\boxed{0}$ $\boxed{2}$ $\boxed{4}$	−	1024
Depress	$\boxed{\text{D}\rightarrow\text{H}}$		FC00

5.4 EXAMPLE 4, ENTER VALUE ON CRU

Send a bit pattern to the CRU at CRU address (bits 3 to 14 of R12) 090_{16} with a bit count of 9 containing a value of 5 (000000101_2).

Depress	$\boxed{\text{CLR}}$	$\boxed{}\boxed{}$
Depress	$\boxed{9}\ \boxed{0}\ \boxed{9}\ \boxed{0}$	$\boxed{}\boxed{9090}$
Depress	$\boxed{\text{DCRU}}$	$\boxed{9090}\boxed{\text{YYYY}}$
Depress	$\boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{5}$	$\boxed{9090}\boxed{0005}$
Depress	$\boxed{\text{ECRU}}$	

The data will be entered into the onboard TMS 9901 of the TM 990/100/180M. To verify the data on the TMS 990, do the following:

Depress	$\boxed{\text{CLR}}$	$\boxed{}\boxed{}$
Depress	$\boxed{9}\ \boxed{0}\ \boxed{9}\ \boxed{0}$	$\boxed{}\boxed{9090}$
Depress	$\boxed{\text{DCRU}}$	$\boxed{9090}\boxed{0005}$

YYYY indicates value at the current CRU address. Note that a $\boxed{\text{DCRU}}$ operation is always required to specify bit count/CRU address.

5.5 EXAMPLE 5. ENTER, VERIFY VALUE AT MEMORY ADDRESS

Enter 0040_{16} into location FE20 and verify that it got there.

Depress	$\boxed{\text{CLR}}$	
Depress	$\boxed{\text{F}}\ \boxed{\text{E}}\ \boxed{2}\ \boxed{0}$	$\boxed{}\boxed{\text{FE20}}$
Depress	$\boxed{\text{EMA}}$	$\boxed{\text{FE20}}\boxed{\text{xxxx}}$
Depress	$\boxed{0}\ \boxed{0}\ \boxed{4}\ \boxed{0}$	$\boxed{\text{FE20}}\boxed{0040}$
Depress	$\boxed{\text{EMD}}$	$\boxed{\text{FE20}}\boxed{0040}$

The contents of address FE20 are verified by an echo of data from memory to display following the pressing of $\boxed{\text{EMD}}$. If it is desired to view and enter data at address FE22, depress $\boxed{\text{EMDI}}$.

6. DEVICE SERVICE ROUTINE CODING

6.1 INTRODUCTION

When used with the Texas Instruments TM 990/100M, TM 990/101M, or TM 990/180M Microcomputers, the Microterminal requires no special user coding because the software device service routine required to accommodate the Microterminal is resident in *TIBUG*, the debug monitor used on these boards. If the user utilizes any other microcomputer in conjunction with the Microterminal, a device service routine must be coded by the user to accommodate the Microterminal.

6.2 HARDWARE OPERATION

The Microterminal interfaces to any microcomputer utilizing the signals indicated in Table 2. The block diagram is shown in Figure 2.

TABLE 2. MICROTERMINAL INTERFACE SIGNALS

Signal Name	Maximum Voltage	Level	Connector
+5V	6V	POWER	P1 – 14
+12V	+14V	POWER	P1 – 12
–12V	–14V	POWER	P1 – 13
GROUND	–	GROUND	P1 – 7
$\overline{\text{TERMINAL DATA IN}}$	+14V	RS-232-C	P1 – 3
$\overline{\text{TERMINAL DATA OUT}}$	+14V	RS-232-C	P1 – 2
$\overline{\text{HALT}}$	+6V	OPEN COLLECTOR NPN	P1 – 16

The Microterminal receives +12V, –12V, +5V and ground from the host microcomputer. Great care must be taken not to exceed the maximum rated voltages; otherwise, permanent damage to the Microterminal might occur. $\overline{\text{TERMINAL DATA OUT}}$ is an RS-232-C, 110 baud output from the Microterminal. A serial bit stream will be output from the Microterminal utilizing $\overline{\text{TERMINAL DATA OUT}}$ for commands and data from the Microterminal. Refer to Figure 3 for the format of commands and data. It should be noted that commands utilize one RS-232-C character (one start bit, a five bit command, two "don't care" bits, a parity bit and three stop bits), but a 16-bit data word utilizes four RS-232-C characters because only four data bits are included in an RS-232-C Microterminal data character. The software significance of commands and data is defined in paragraph 6.3. It should be noted that the Microterminal sends two unique ASCII characters (an A and Z) over $\overline{\text{TERMINAL DATA OUT}}$ when the $\boxed{\text{CLR}}$ key is depressed. $\overline{\text{TERMINAL DATA OUT}}$ utilizes standard RS-232-C voltage levels:

- 12.0 volts $\geqslant$ Logic 1 $>$ 6.0 volts
- –6.0 volts $>$ Logic 0 $\geqslant$ –12.0 volts

$\overline{\text{TERMINAL DATA IN}}$ is a RS-232-C, 110-baud input to the Microterminal. A serial bit stream of data is required from the host microcomputer to update LED displays in situations defined in paragraph 6.3. Figure 3 defines the format of a Microterminal input data character (one start bit, four data bits, three "don't care" bits, an even parity bit and two stop bits). As in the case of Microterminal output data, four data characters are required to form a 16-bit word. The voltage levels required on $\overline{\text{TERMINAL DATA IN}}$ are RS-232-C standard (the same levels defined for $\overline{\text{TERMINAL DATA OUT}}$). The frequency of $\overline{\text{TERMINAL DATA IN}}$, as with any terminal device must not vary more than 2 per cent from its proper rate (110 baud).

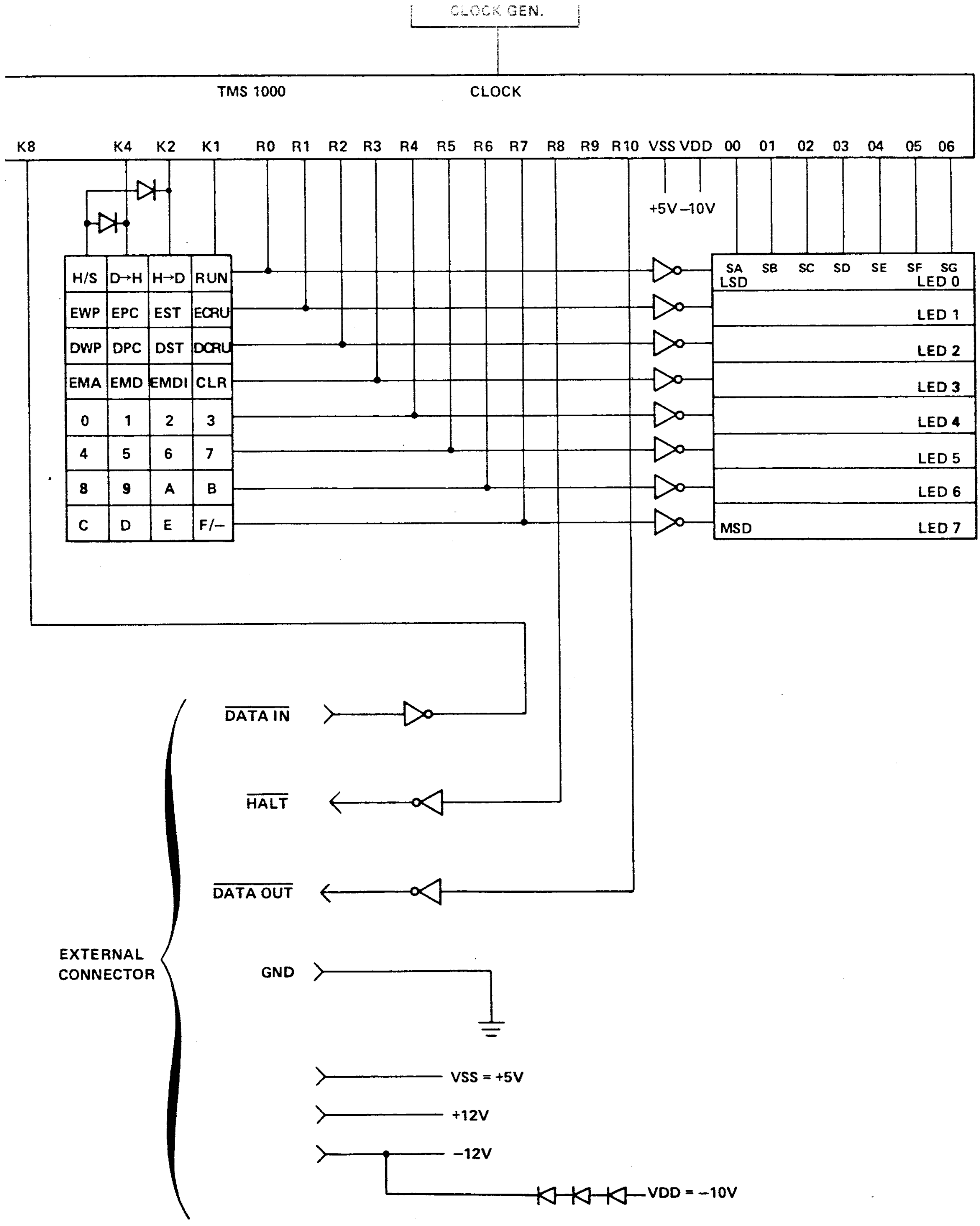
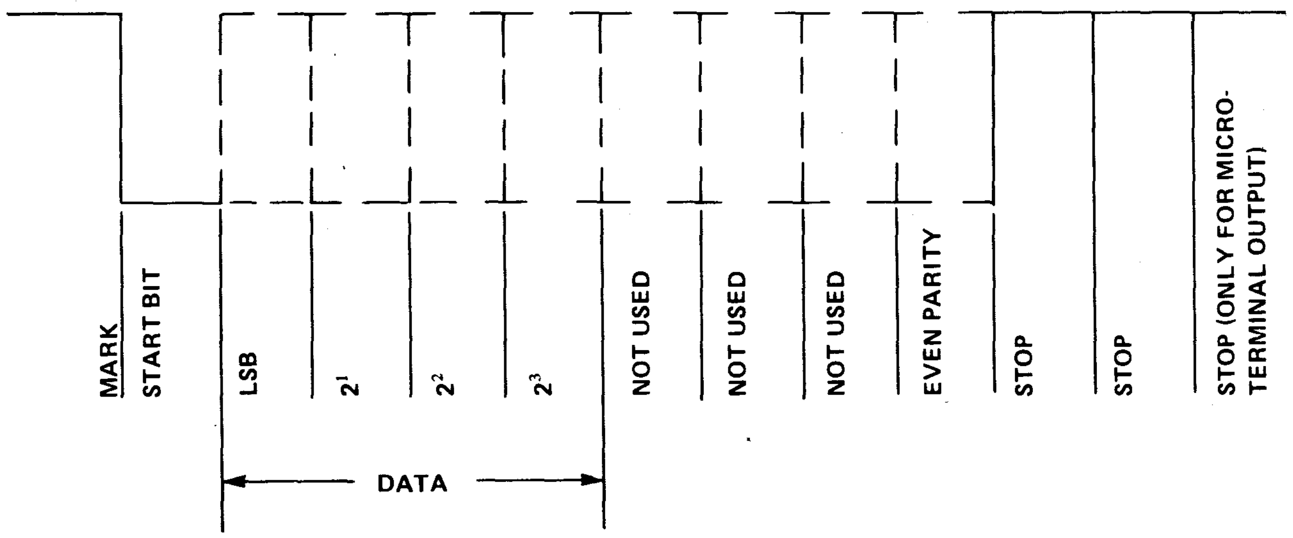
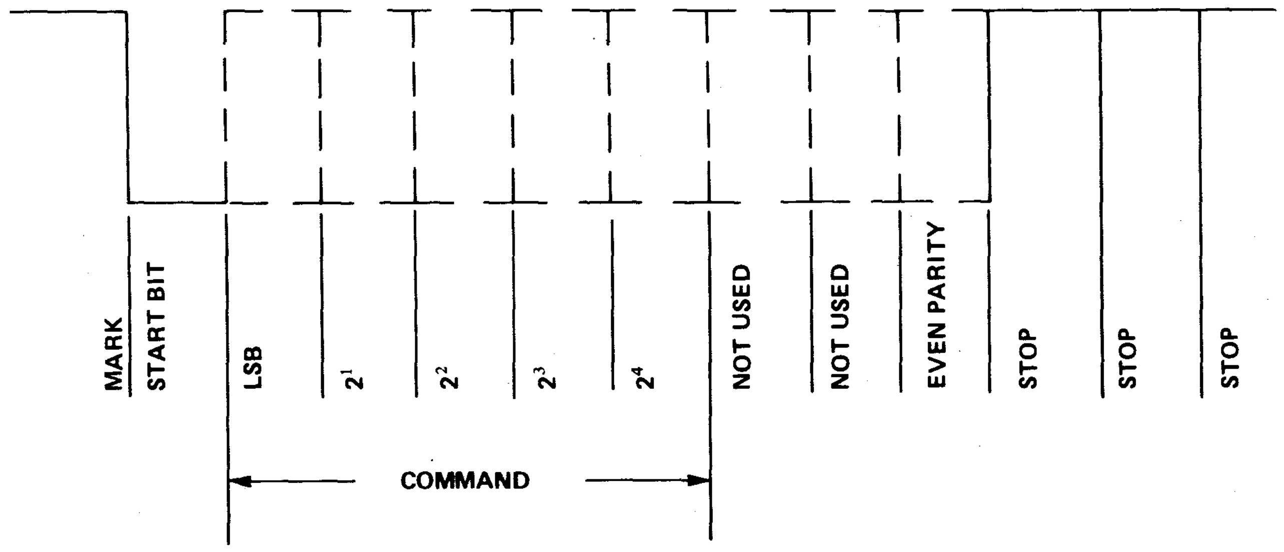


Figure 2. Microterminal Block Diagram



DATA CHARACTER

NOTE: DATA WORDS ARE SENT MSD FIRST, LSD LAST
MICROTERMINAL OUTPUT DATA WORDS CONTAIN 3 STOP BITS
990/100M OUTPUT DATA WORDS CONTAIN 2 STOP BITS



COMMAND CHARACTER

NOTE: THE INDICATED BIT POLARITY IS THAT OUTPUTTED OR SEEN BY THE SOFTWARE. THE INTERFACE BETWEEN THE MICROTERMINAL AND MICROCOMPUTER IS OF INVERSE POLARITY.

Figure 3. Character Format

$\overline{\text{HALT}}$ is the open collector output of a NPN transistor (pin 16 as shown in Table 1). This signal must be connected to a 1K resistor that is pulled up to 5 volts on the host microcomputer. $\overline{\text{HALT}}$ becomes active low for 30 μsec when the user depresses $\boxed{\text{H/S}}$ on the Microterminal while a user program is being executed. $\overline{\text{HALT}}$ provides one source for $\overline{\text{LOAD}}$, a nonmaskable interrupt, on the TM 990/100M Microcomputer. Since the purpose of the $\overline{\text{HALT}}$ signal is to halt program execution, the user will be required to provide an interrupt to the host microcomputer when $\overline{\text{HALT}}$ becomes active low. The logic levels for $\overline{\text{HALT}}$ are:

Logic 1 = +5V Supply Level

GROUND $\leqslant$ Logic 0 $<$ 0.8V

Since $\overline{\text{HALT}}$ is an NPN transistor output, great care must be taken not to short the signal to a voltage.

6.3 SOFTWARE OPERATION

The Microterminal is internally controlled by a TMS 1000 microprocessor which does the following:

- Scans the keyboard to detect and process key depressions
- Refreshes the 7-segment LED displays
- Outputs commands to the host microcomputer to specify the function that the microcomputer must perform
- Outputs any required data to the microcomputer
- Outputs two unique characters (ASCII A and ASCII Z) when $\boxed{\text{CLR}}$ is depressed
- Lowers $\overline{\text{LOAD}}$ for 30 μsec to halt program execution
- Receives input data from the host microcomputer for display

Figure 4 is the TMS 1000 software flowchart. The host microcomputer must receive and decode commands, receive any required data coming after the command, output any required data to the Microterminal and interrupt program execution when $\overline{\text{HALT}}$ = 0.

The required communication between the Microterminal and host microcomputer is shown in Table 3. As an example of the communication sequence, consider the following case of the user desiring to utilize the EMA function (display contents and address of designated memory address):

1. The user enters a memory address using four hex (0 to F) key depressions. The TMS 1000 detects and displays these four hexadecimal entries.

2. The user depresses $\boxed{\text{EMA}}$. The TMS 1000 sends a command character (00_{16}) over the interface via $\overline{\text{TERMINAL DATA OUT}}$ to specify the EMA operation.

3. The host microcomputer receives and decodes the command character. The microcomputer must now prepare to receive four data characters and assemble them into a 16-bit memory address register specifying the address of the desired memory location. The command contains three stop bits with the start bit of the first data character coming after the third stop bit of the command character. The start bit of each data character comes after the third stop bit of the preceding data character.

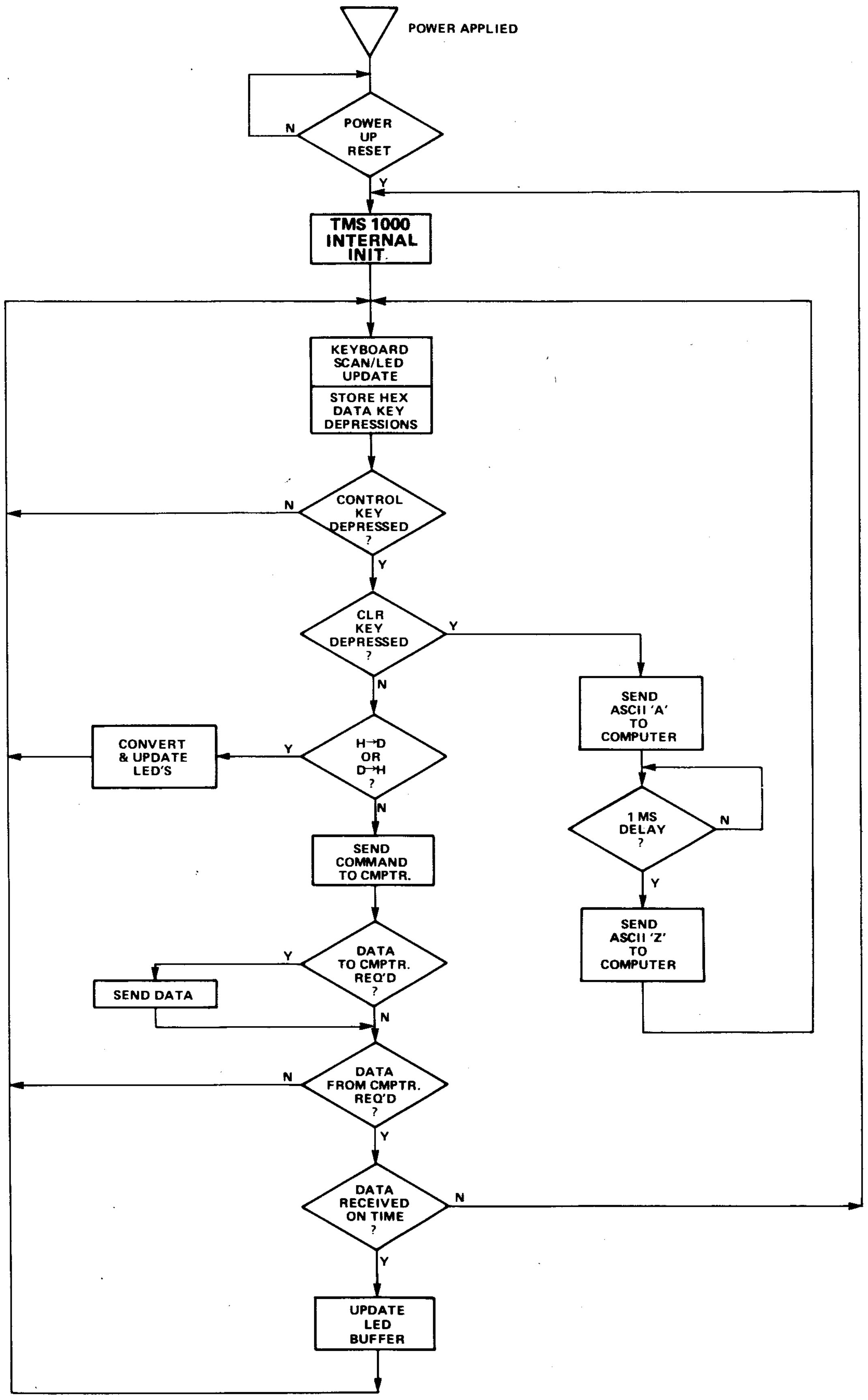


Figure 4. TMS 1000 Software Flowchart

TABLE 3. MICROTERMINAL CONTROL SEQUENCE

Key Depressed	Hexadecimal Code	Microterminal Sends	Microcomputer Sends In Response
EWP	06_{16}	Command (1 EIA Character)	–
		Data to Enter (4 EIA Characters)	–
EST	04_{16}	Command (1 EIA Character)	–
		Data to Enter (4 EIA Characters)	–
EPC	02_{16}	Command (1 EIA Character)	–
		Data to Enter (4 EIA Characters)	–
DWP	$0C_{16}$	Command (1 EIA Character)	*Data to be displayed (4 EIA Characters)
DST	$0A_{16}$	Command (1 EIA Character)	*Data to be displayed (4 EIA Characters)
DPC	08_{16}	Command (1 EIA Character)	*Data to be displayed (4 EIA Characters)
EMA	00_{16}	Command (1 EIA Character)	*Data to be displayed (4 EIA Characters)
		Address (4 EIA Characters)	
EMD	14_{16}	Command (1 EIA Character)	*Data to be displayed (4 EIA Characters)
		Data to Enter (4 EIA Characters)	
EMDI	12_{16}	Command (1 EIA Character)	*Data to be displayed (4 EIA Characters)
		Data to Enter (4 EIA Characters)	
DCRU	$0E_{16}$	Command (1 EIA Character)	*Data to be displayed (4 EIA Characters)
		Bit Count/Address (4 EIA Characters)	
ECRU	10_{16}	Command (1 EIA Character)	–
		Data (4 EIA Characters)	–
CLR		ASCII A (1 EIA Character)	–
		ASCII Z (1 EIA Character)	–
RUN	16_{16}	Command (1 EIA Character)	–
H/S	(Unit in Run Mode)	$\overline{\text{HALT}}$ = 0 For 30 μsec	**PC to be displayed (4 EIA Characters)
			Data to be displayed (4 EIA Characters)
H/S	18_{16} (Unit not in Run Mode)	Command (1 EIA Character)	*PC to be displayed (4 EIA Characters)
			Data to be displayed (4 EIA Characters)

*First character must be received by the Microterminal within 5 milliseconds after the end of the third stop bit of the command word is sent by the Microterminal. Each succeeding character must be received by the Microterminal within 5 milliseconds after the end of the second stop bit of the preceding data character is received from the microcomputer.

**First EIA character must be received by the Microterminal between 6 to 10 milliseconds of $\overline{\text{HALT}}$ becoming active low. Each succeeding character must be received by the Microterminal within 6 to 10 milliseconds of center point of second stop bit of the preceding character.

4. The host microcomputer fetches the 16-bit data word from the memory location specified by the contents of the memory address register.

5. This data is output from the microcomputer via four data characters on the $\overline{\text{TERMINAL DATA IN}}$ signal. The start bit for the first data character must occur before five ms has elapsed after receiving the third stop bit of the last memory address character from the Microterminal. Each of the remaining three data characters from the microcomputer must present a start bit within 5 milliseconds after the second stop bit of the preceding microcomputer data character. Longer time delays will cause the Microterminal to detect a data transmission error and blank the display.

6. The Microterminal displays the receive data.

The following points must be noted by the user who wishes to code a device service routine for the Microterminal:

- When data characters follow a Microterminal command character, each start bit follows the third stop bit of the previous character.

- All required input data to the Microterminal must occur within the required time frame; otherwise, the Microterminal will detect a data transmission error and blank the display.

- Command and data characters are of the format shown in Figure 3. When the Microterminal outputs a command followed by data, each character has three stop bits followed by the start bit of the next character.

- A 16-bit data word requires four data characters of the form shown in Figure 3.

- The Microterminal will always output two characters in succession (an ASCII A and ASCII Z) when $\boxed{\text{CLR}}$ is depressed. For the user that might have several different types of terminal devices, these two characters might serve as an ID for the Microterminal.

- $\overline{\text{HALT}}$ must be wired to an interrupt in order to halt program execution.

6.4 LISTING OF SAMPLE MICROTERMINAL DEVICE SERVICE ROUTINE

Enclosed is a listing of a stand-alone device service routine for the Microterminal utilizing the TM 990/100M Microcomputer as the host device. A stand-alone service routine rather than *TIBUG* is included to focus understanding on the Microterminal functions (i.e., *TIBUG* contains many functions besides the Microterminal routines). Refer to Figures 5 to10for detailed program flow of parts of the stand-alone device service routine. Table 4 explains DSR assembly language action in response to keys pressed on the Microterminal. Note the following points about the stand-alone device service routine.

- Since this program is of a stand-alone nature, it is not necessary to use the A and Z character output by the Microterminal to identify itself when $\boxed{\text{CLR}}$ is depressed. These two characters will be ignored because they will be recognized as invalid commands by the microcomputer command scanner.

- When the RESET pushbutton is depressed, the TM 990/100M will begin execution at the Program Counter location specified by the contents of memory location 0002_{16} with the workspace pointer **specified by the contents of memory location 0000_{16}.**

- LOAD is the entry point when the nonmaskable interrupt of the TMS 9900 is activated by the HALT signal (from the Microterminal) becoming low or the output from a TM 990/100M circuit activated two instructions after a LREX instruction (or two instructions are executed after the LREX instruction, then LOAD is entered). LOAD is used to halt program execution or for single instruction execution.

TABLE 4. DSR ACTION TO KEY COMMANDS

Microterminal Key Command	DSR Action And Comment			Listing Source Line
EPC	INPT	R14	XOP1, INPUT PC TO R14	079
	JMP	MTIN	TO COMMAND SCANNER	080
EST	INPT	R15	XOP1, INPUT ST TO R15	081
	JMP	MTIN	TO COMMAND SCANNER	082
EWP	INPT	R13	XOP1, INPUT UP TO R13	083
	JMP	MTIN	TO COMMAND SCANNER	084
DPC	OTPT	R14	XOP0, OUTPUT PC FROM R14	085
	JMP	MTIN	TO COMMAND SCANNER	086
DST	OTPT	R15	XOP0, OUTPUT ST FROM R15	087
	JMP	MTIN	TO COMMAND SCANNER	088
DWP	OTPT	R13	XOP0, OUTPUT WP FROM R13	089
	JMP	MTIN	TO COMMAND SCANNER	090
EMA	INPT	R8	XOP1, INPUT ADDR TO R8	091
	MOV	*R8, R9	DATA AT ADDR TO R9	098
	OTPT	R9	XOP0, OUTPUT (R9)	125
	JMP	MTIN	TO COMMAND SCANNER	126
EMD	INPT	R9	XOP1, INPUT DATA TO R9	093
	MOV	R9, *R8	TO MEMORY INDIRECT R8	094
	MOV	*R8, R9	TO R9 INDIRECT R8	098
	OTPT	R9	XOP0, OUTPUT (R9)	125
	JMP	MTIN	TO COMMAND SCANNER	126
EMDI	INPT	R9	XOP1, INPUT DATA TO R9	096
	MOV	R9, *R8+	TO MEMORY, INDIRECT R8, INCREMENT R8 TO NEXT ADDR.	097
	MOV	*R8, R9	(NEXT M.A.) TO R9	098
	OTPT	R9	XOP0, OUTPUT (R9)	125
	JMP	MTIN	TO COMMAND SCANNER	126
STEP (H/S)	SETO	@STEPFG	SET STEP FLAG	104
	CLR	@HALTFG	CLEAR HALT FLAG	105
	LREX		CAUSE LOAD INTERRUPT WHICH GOES TO STEP ROUTINE	106

TABLE 4. DSR ACTION TO KEY COMMANDS (Concluded)

Microterminal Key Command	DSR Action And Comment	Listing Source Line
DCRU	*DO LOAD ROUTINE	195-202
	INPT R10 XOP1, GET BIT COUNT AND CRU ADDR	108
	• • •	
	*MOVE CRU ADDR TO R12	110-112
	*EXECUTE MOVE (CRU) TO R9	113-116
	OTPT R9 XOP0, OUTPUT (R9)	125
	JMP MTIN TO COMMAND SCANNER	126
ECRU	INPT R9 XOP1, CRU DATA TO R9	118
	• • •	
	*EXECUTE MOVE (R9) TO CRU	119, 120
	JMP MTIN TO COMMAND SCANNER	121
RUN	RTWP BRANCH *R14 WITH (R13) = ADDR. OF WP AND (R15) = STATUS REG	077
HALT (H/S)	SETO R0 SET DELAY FLAG	122
	OTPT R14 XOP0, OUTPUT VALUE OF PC	123
	MOV *R14, R9 MOVE DATA AT PC TO R9	124
	OTPT R9 OUTPUT (R9)	125
	JMP MTIN TO COMMAND SCANNER	126

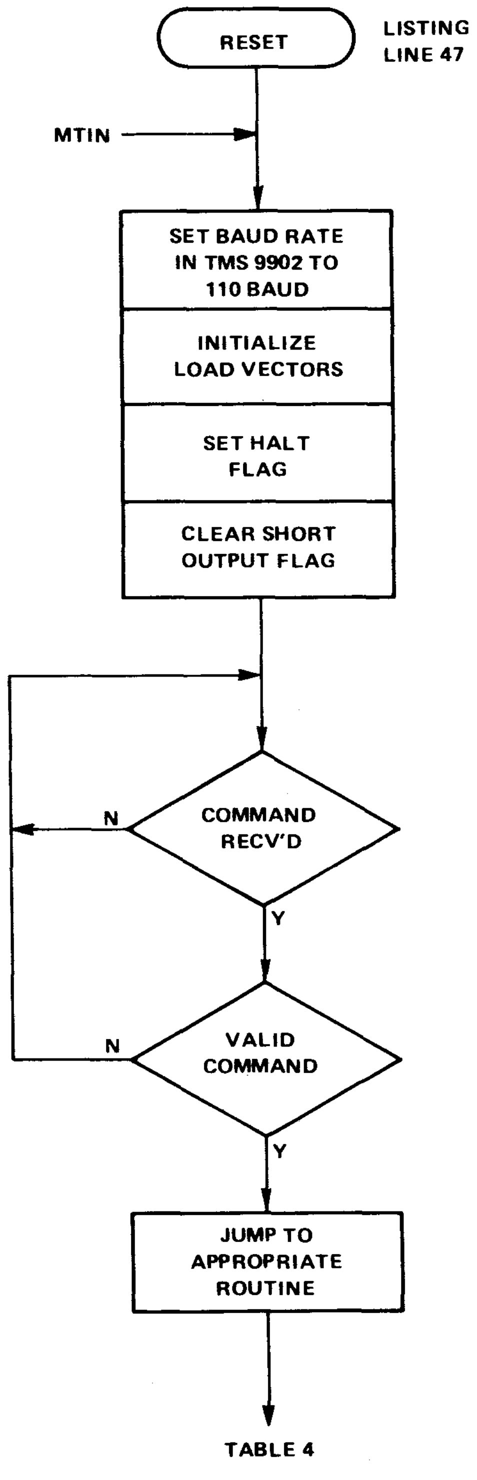


Figure 5. Microterminal Initialization and Command Scanner

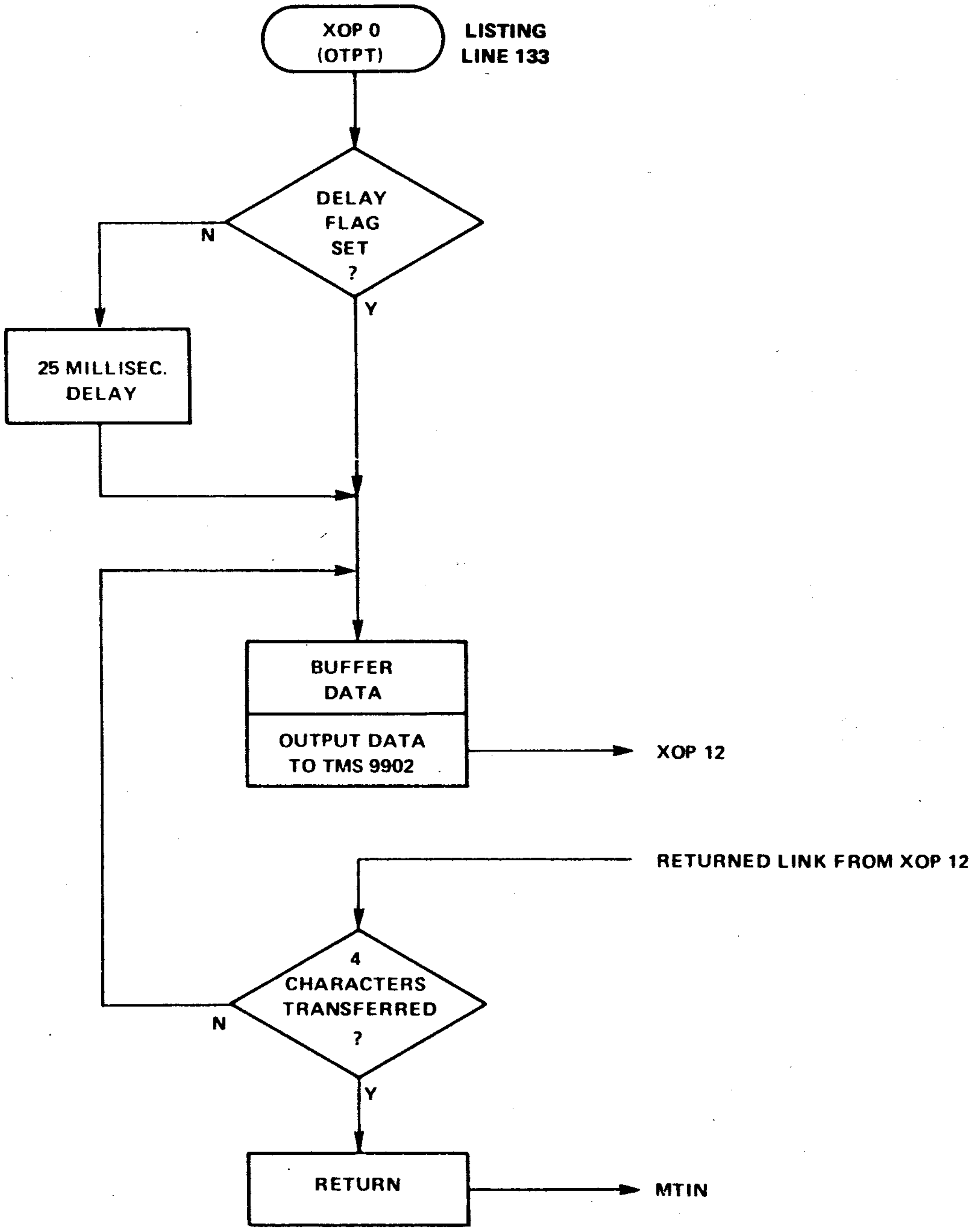


Figure 6. Output Data to Microterminal

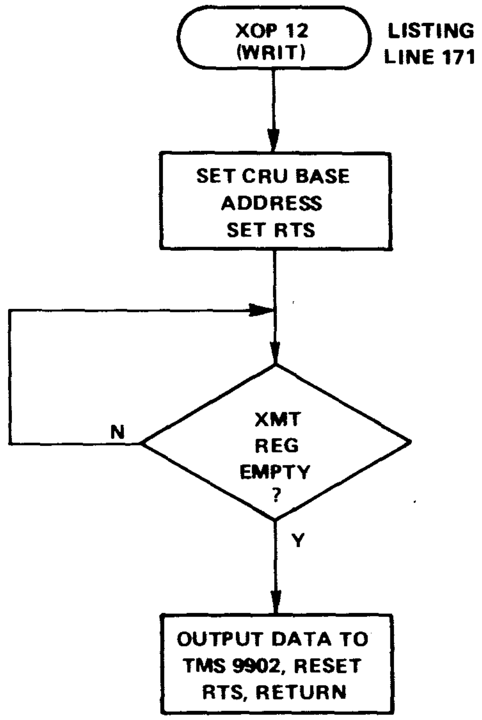

XOP 12
(WRIT)
LISTING LINE 171
SET CRU BASE ADDRESS SET RTS
XMT REG EMPTY ?
N
Y
OUTPUT DATA TO TMS 9902, RESET RTS, RETURN

Figure 7. Output Data to TMS 9902 (to Microterminal)

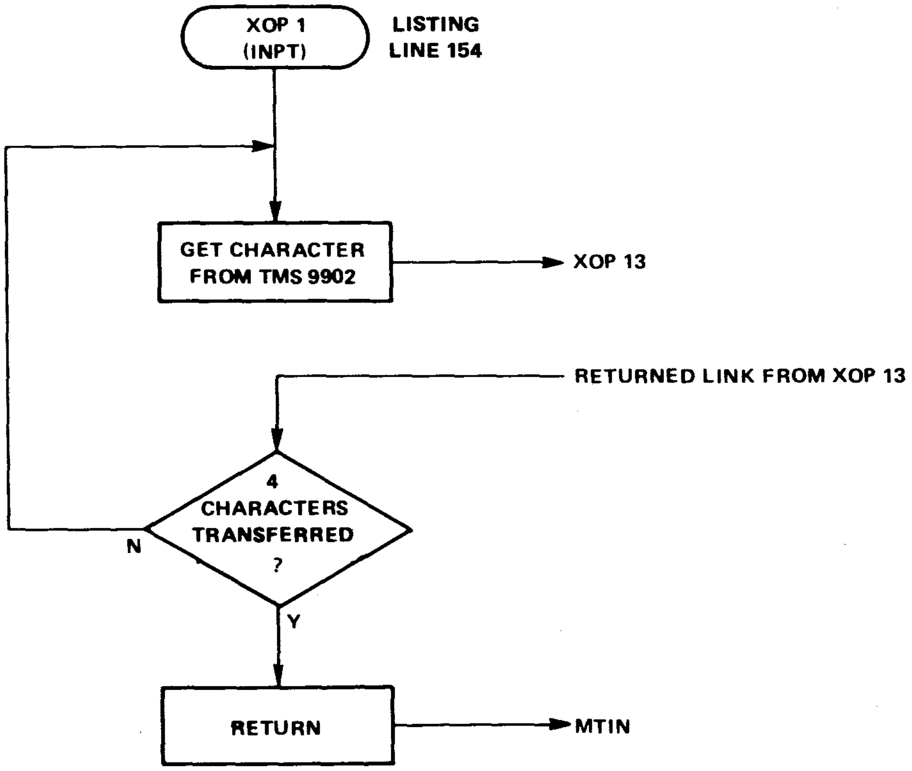

XOP 1
(INPT)
LISTING LINE 154
GET CHARACTER FROM TMS 9902
XOP 13
RETURNED LINK FROM XOP 13
4 CHARACTERS TRANSFERRED ?
N
Y
RETURN
MTIN

Figure 8. Input Data From Microterminal

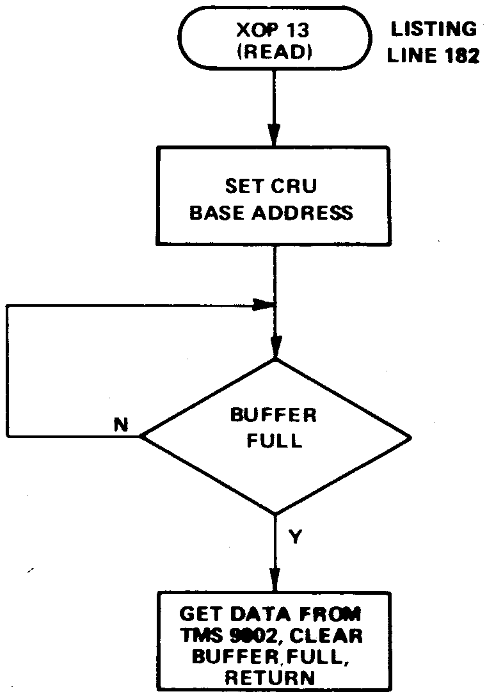


Figure 9. Get Data From TMS 9902 (from Microterminal)

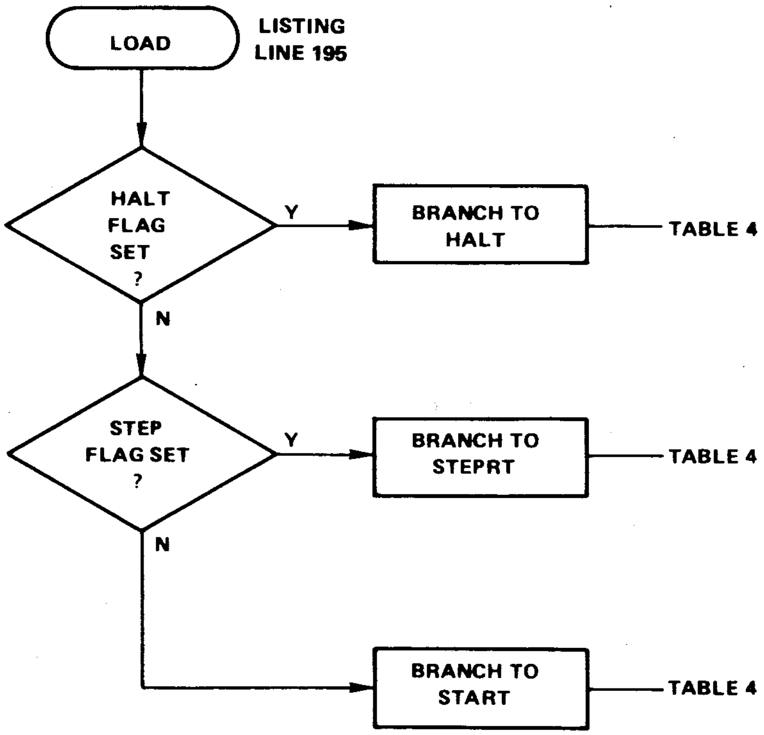


Figure 10. Load Signal Execution

```
                                                        PAGE 0001

                        *
                        ************************************************************
                        *  THIS IS A STAND ALONE DEVICE SERVICE ROUTINE FOR THE
                        *  TEXAS INSTRUMENTS TM990/301 MICROTERMINAL UTILIZING THE
                        *  TEXAS INSTRUMENTS TM990/100M MICROCOMPUTER AS THE HOST
                        *  DEVICE.THE 990/100M UTILIZES MEMORY LOCATIONS 00 THRU
                        *  80(ALL NUMBERS ARE HEX)FOR INTERRUPT VECTORS AND XOP
                        *  VECTORS,MEMORY LOCATIONS 80 THRU 7FF FOR EXECUTABLE
                        *  CODE AND MEMORY LOCATIONS FE00 THRU FFFF FOR RAM STORAGE
                        *  THE DATA INTERFACE BETWEEN THE MICROTERMINAL AND
                        *  TM990/100M IS RS232C SERIAL.A TMS9902 ACIA IS UTILIZED
                        *  ON THE TM990/100M AS THE UART DEVICE.
                        ************************************************************
                               IDT     'MCTER'
             000B  LINK    EQU     R11
             000C  CRUBAS  EQU     R12
             FFB0  MREGS   EQU     >FFB0
             FFD4  XREGS   EQU     >FFD4
             FFC6  IREGS   EQU     >FFC6
             FFF8  STEPFG  EQU     >FFF8
             FFFA  HALTFG  EQU     >FFFA
                               DXOP    OTPT,0
                               DXOP    INPT,1
                               DXOP    READ,13
                               DXOP    WRIT,12
       0000  FFB0              DATA    MREGS,INIT   RESET VECTORS
       0002  ----
       0004  FFFF              DATA    >FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF
       0006  FFFF
       0008  FFFF
       000A  FFFF
       000C  FFFF
       000E  FFFF
       0010  FFFF              DATA    >FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FF
       0012  FFFF
       0014  FFFF
       0016  FFFF
       0018  FFFF
       001A  FFFF
       001C  FFFF
       001E  0FFF
       0020  FFFF              DATA    >FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFF
       0022  FFFF
       0024  FFFF
       0026  FFFF
       0028  FFFF
       002A  FFFF
       002C  FFFF
       002E  0FFF
       0030  FFFF              DATA    >FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFF
       0032  FFFF
       0034  FFFF
       0036  FFFF
       0038  FFFF
       003A  FFFF
```

```
                                                      PAGE 0002
 003C  FFFF
 003E  0FFF
 0040  FFD4          DATA   XREGS,OTPTEN  XOP0 VECTORS
 0042  ----
 0044  FFD4          DATA   XREGS,INPTEN  XOP1 VECTORS
 0046  ----
 0048  FFFF          DATA   >FFFF,>FFFF,>FFFF,>FFFF
 004A  FFFF
 004C  FFFF
 004E  FFFF
 0050  FFFF          DATA   >FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFF
 0052  FFFF
 0054  FFFF
 0056  FFFF
 0058  FFFF
 005A  FFFF
 005C  FFFF
 005E  0FFF
 0060  FFFF          DATA   >FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFFF,>FFF
 0062  FFFF
 0064  FFFF
 0066  FFFF
 0068  FFFF
 006A  FFFF
 006C  FFFF
 006E  0FFF
 0070  FFC6          DATA   IREGS,WENTRY
 0072  ----
 0074  FFC6          DATA   IREGS,RENTRY
 0076  ----
 0078  FFFF          DATA   >FFFF,>FFFF,>FFFF,>FFFF
 007A  FFFF
 007C  FFFF
 007E  FFFF
                *                                                           *
                *************************************************************
                *  MICROTERMINAL INITIALIZATION AND COMMAND SCANNER-THIS    *
                *  ROUTINE IS ENTERED AS A RESULT OF A RESET PUSHBUTTON     *
                *  DEPRESSION.THE BAUD RATE IS SET TO 110 BAUD,THE LOAD     *
                *  VECTORS ARE INITIALIZED AND A WAIT LOOP IS ENTERED       *
                *  AWAITING COMMAND INPUT FROM THE MICROTERMINAL.           *
                *************************************************************
 0080  020C     INIT LI     CRUBAS,>80  LOAD CRU BASE REG
 0082  0080
 0002**0080'
 0084  1D1F          SBO    31          INITIALIZE UART
 0086  3220          LDCR   @CR,8
 0088  ----
 008A  1E0D          SBZ    13
 008C  3320          LDCR   @BR,12      SET BAUD RATE
 008E  ----
 0090  0202          LI     2,>FFFC     INITIALIZE LOAD VECTORS
 0092  FFFC
 0094  04C1          CLR    1
 0096  CCB1          MOV    *1+,*2+
0055  0098  0201          LI     1,LOAD
      009A  ----
0056  009C  C481          MOV    1,*2
0057  009E  2F42  MTIN    READ   2           GET MICROTERMINAL COMMAND
0058  00A0  04C0          CLR    0           CLEAR SHORT OUTPUT FLAG
0059  00A2  0720          SETO   @HALTFG     SET HALT FLAG
      00A4  FFFA
0060  00A6  06C2          SWPB   2           RIGHT JUSTIFY COMMAND
0061  00A8  0242          ANDI   2,>1E       STRIP OFF UNDEFINED BITS
      00AA  001E
0062  00AC  0282          CI     2,>18       CHECK FOR INVALID COMMAND
      00AE  0018
0063  00B0  15F6          JGT    MTIN        DISREGARD INVALID COMMAND
0064  00B2  0222          AI     2,JMTB      ADD COMMAND AND JUMP TABLE BIAS
      00B4  ----
0065  00B6  0452          B      *2          GO EXECUTE SPECIFIED FUNCTION
0066  00B8  10--  JMTB    JMP    EMA         JUMP TABLE
      00B4**00B8
0067  00BA  10--          JMP    EPC
0068  00BC  10--          JMP    EST
0069  00BE  10--          JMP    EWP
0070  00C0  10--          JMP    DPC
0071  00C2  10--          JMP    DST
0072  00C4  10--          JMP    DWP
0073  00C6  10--          JMP    DCRU
0074  00C8  10--          JMP    ECRU
0075  00CA  10--          JMP    EMDI
0076  00CC  10--          JMP    EMD
0077  00CE  0380          RTWP
0078  00D0  10--          JMP    STEP
0079  00D2  2C4E  EPC     INPT   14          GET PC
      00BA**100B
0080  00D4  10E4          JMP    MTIN        GO AWAIT NEXT COMMAND
0081  00D6  2C4F  EST     INPT   15          GET ST
      00BC**100C
0082  00D8  10E2          JMP    MTIN
0083  00DA  2C4D  EWP     INPT   13          GET WP
      00BE**100D
0084  00DC  10E0          JMP    MTIN
0085  00DE  2C0E  DPC     OTPT   14          OUTPUT PC
      00C0**100E
0086  00E0  10DE          JMP    MTIN
0087  00E2  2C0F  DST     OTPT   15          OUTPUT ST
      00C2**100F
0088  00E4  10DC          JMP    MTIN
0089  00E6  2C0D  DWP     OTPT   13          OUTPUT WP
      00C4**1010
0090  00E8  10DA          JMP    MTIN
0091  00EA  2C48  EMA     INPT   8           GET MEMORY ADDRESS REG
      00B8**1018
0092  00EC  10--          JMP    EMDI1       GO EXECUTE FUNCTION
0093  00EE  2C49  EMD     INPT   9           GET DATA
      00CC**1010
0094  00F0  C609          MOV    9,*8        STORE DATA IN MEMORY
0095  00F2  10--          JMP    EMDI1
```

```
00F4  2C49  EMDI    INPT   9             GET DATA
00CA**1014
00F6  CE09          MOV    9,*8+         STORE DATA IN MEMORY AND AUTO INC
00F8  C258  EMDI1   MOV    *8,9          GET DATA FROM MEMORY
00EC**1005
00F2**1002
00FA  10--          JMP    HALT1
00FC  2C0E  STEPRT  OTPT   14            OUTPUT PC
00FE  C25E          MOV    *14,9         GET PC MEMORY DATA
0100  0700          SETO   0             SET DELAY FLAG
0102  10--          JMP    HALT1
0104  0720  STEP    SETO   @STEPFG       SET STEP FLAG
0106  FFF8
00D0**1019
0108  04E0          CLR    @HALTFG       CLEAR HALT FLAG
010A  FFFA
010C  03E0          LREX                 FIRE LOAD INTERRUPT
010E  0380          RTWP                 EXECUTE USER CODE
0110  2C4A  DCRU    INPT   10            GET BIT COUNT AND CRU ADDRESS
00C6**1024
0112  04C9          CLR    9             CLEAR DATA REG
0114  C30A          MOV    10,12
0116  024C          ANDI   12,>0FFF      SAVE CRU ADDRESS
0118  0FFF
011A  0A1C          SLA    12,1          PUT IN PROPER WORD POSITION
011C  09CA          SRL    10,12         STRIP OUT ZEROES
011E  0A6A          SLA    10,6
0120  022A          AI     10,>3409      SET UP STCR OP CODE
0122  3409
0124  048A          X      10            EXECUTE STCR
0126  10--          JMP    HALT1
0128  2C49  ECRU    INPT   9             GET DATA
00C8**102F
012A  022A          AI     10,>FC00      SET UP LDCR OP CODE
012C  FC00
012E  048A          X      10            EXECUTE LDCR
0130  10B6          JMP    MTIN
0132  0700  HALT    SETO   0             SET DELAY FLAG
0134  2C0E          OTPT   14            OUTPUT PC
0136  C25E          MOV    *14,9         GET PC MEMORY DATA
0138  2C09  HALT1   OTPT   9             OUTPUT DATA
00FA**101E
0102**101A
0126**1008
013A  10B1          JMP    MTIN
            *
            *************************************************************
            *  MICROTERMINAL OUTPUT-XOP R,0                             *
            *  CONVERTS 1 MICROTERMINAL 16 BIT OUTPUT WORD TO 4 EIA     *
            *  OUTPUT WORDS.                                            *
            *************************************************************
013C  C01D  OTPTEN MOV     *13,0                 CHECK DELAY FLAG
0042**013C
013E  16--          JNE    BDLY                  BYPASS DELAY IF FLAG SET
0140  0200          LI     0,>0F00               25 MS DELAY
```

```
                                                   PAGE 0005

 0142  0F00
 0144  0600  DLY    DEC    0
 0146  16FE         JNE    DLY
 0148  0200  BDLY   LI     0,4               LOAD NUMBER OF TRANSFERS
 014A  0004
 013E◆◆1604
 014C  C25B         MOV    ◆11,9             MOVE DATA TO REG9
 014E  C049  CNOT   MOV    9,1               MOVE DATA TO REG1
 0150  09C1         SRL    1,12              SAVE MSD
 0152  0A81         SLA    1,8               LEFT JUSTIFY
 0154  0221         AI     1,>3000           ADD CODE BIAS
 0156  3000
 0158  2F01         WRIT   1                 CALL TIBUG OUTPUT ROUTINE
 015A  0A49         SLA    9,4               SHIFT IN NEW DIGIT
 015C  0600         DEC    0                 DECREMENT WORD COUNT
 015E  16F7         JNE    CNOT              GO BACK IF NOT FINISHED
 0160  0380         RTWP
             ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
             ◆  MICROTERMINAL INPUT-XOP R,1
             ◆  CONVERTS 1 MICROTERMINAL 16 BIT INPUT WORD TO 4 EIA INPUT
             ◆  WORDS.
             ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
 0162  0201  INPTEN LI     1,>0004           LOAD NUMBER OF TRANSFERS
 0164  0004
 0046◆◆0162'
 0166  04DB         CLR    ◆11               CLEAR DATA
 0168  C15B  CNIN   MOV    ◆11,5             MOVE DATA TO REG5
 016A  0A45         SLA    5,4               SHIFT TO LEFT
 016C  C6C5         MOV    5,◆11             PUT DATA IN MEMORY
 016E  2F43         READ   3                 GET EIA WORD
 0170  0A43         SLA    3,4               LEFT JUSTIFY
 0172  09C3         SRL    3,12
 0174  E6C3         SOC    3,◆11             SET ONES IN DATA WORD
 0176  0601         DEC    1                 DECREMENT WORD COUNT
 0178  16F7         JNE    CNIN              GO BACK IF NOT FINISHED
 017A  0380         RTWP
             ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
             ◆  WRITE CHARACTER-XOP R,12
             ◆  TRANSFER THE CHARACTER IN THE LEFT BYTE OF REGISTER R TO
             ◆  THE UART.
             ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
 017C  020C  WENTRY LI     CRUBAS,>0080      SET CRU BASE ADDRESS
 017E  0080
 0072◆◆017C'
 0180  1D10         SBO    16                SET RTS
 0182  1F16         TB     22                CHECK XMT REG EMPTY
 0184  16FB         JNE    WENTRY            GO BACK IF NOT EMPTY
 0186  321B         LDCR   ◆11,8             OUTPUT DATA TO UART
 0188  1E10         SBZ    16                RESET RTS
 018A  0380         RTWP
             ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
             ◆  READ CHARACTER-XOP R,13
             ◆  PUTS THE CHARACTER ASSEMBLED IN THE UART INTO REGISTER R.
             ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
 018C  020C  RENTRY LI     CRUBAS,>0080      SET CRU BASE ADDRESS
```

```
      018E  0030
      0076◆◆018C'
0183  0190  1F15         TB      21                  CHECK BUFFER FULL
0184  0192  16FC         JNE     RENTRY              GO BACK IF BUFFER NOT FULL
0185  0194  04DB         CLR     ◆11                 CLEAR DATA
0186  0196  361B         STCR    ◆11,8               GET DATA FROM UART
0187  0198  1E12         SBZ     18                  CLEAR BUFFER FULL
0188  019A  0380         RTWP
0189                ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
0190                ◆  LOAD ROUTINE-THIS ROUTINE IS ENTERED IF A LREX INSTRUCTIO
0191                ◆  IS EXECUTED(THERE IS A TWO INSTRUCTION DELAY AFTER THE LR
0192                ◆  BEFORE THIS ROUTINE IS ENTERED)OR IF THE LOAD SIGNAL FROM
0193                ◆  THE MICROTERMINAL BECOMES ACTIVE LOW.
0194                ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
0195  019C  0201  LOAD   LI      1,HALTFG            CHECK HALT FLAG
      019E  FFFA
      009A◆◆019C'
0196  01A0  C091         MOV     ◆1,2
0197  01A2  13--         JEQ     LOAD1               JUMP IF NOT SET
0198  01A4  0460         B       @HALT               GO EXECUTE HALT FUNCTION
      01A6  0132'
0199  01A8  0641  LOAD1  DECT    1                   CHECK STEP FLAG
      01A2◆◆1302
0200  01AA  C091         MOV     ◆1,2
0201  01AC  13--         JEQ     LOAD2               JUMP IF NOT SET
0202  01AE  0460         B       @STEPRT             GO EXECUTE STEP FUNCTION
      01B0  00FC'
0203  01B2  0460  LOAD2  B       @INIT               REINITIALIZE FOR INVALID LOA
      01B4  0080'
      01AC◆◆1302
0204  01B6  0638  BR     DATA    >0638             TMS9902 DATA RATE REG CONTENTS
      008E◆◆01B6'
0205  01B8    62  CR     BYTE    >62               TMS9902 CONTROL REG CONTENTS
0206                     END     INIT
      0088◆◆01B8'
```