


codex
A Subsidiary of  **MOTOROLA INC.**

**SYSTEM SELF-TEST/
CObug
System Diagnostic Facilities
USER'S GUIDE**

Document No. 72714



c 1980 Codex Corporation
Intelligent Terminals Group

First Edition
Through Version 5.0

Preface

This User's Guide describes the operation and use of the System Self-Test and CObug firmware packages. Designed for use with both diskette and disk-based Codex Intelligent Terminal Systems, System Self-Test and CObug are interactive programs which provides users with debugging and basic input/output functions. This User's Guide is intended for programmers and system development personnel familiar with Codex Intelligent Terminal Systems.

Other Codex literature that may prove useful includes the Codex Disk Operating System (CODOS) User's Guide, the CODOS Reference Manual, and the Operator's Guides and Hardware Reference Manuals appropriate to the user's system.

CHAPTER 1. INTRODUCTION	1-01
Hardware Support Required	1-03
Additional Hardware Supported	1-03
Software Support Required	1-03
Firmware Installation	1-03
CHAPTER 2. OPERATION (DISKETTE-BASED SYSTEMS)	2-01
Automatic Testing Procedures	2-01
Power-On Initialization and Self-test	2-01
Reset Switch Initialization and Self-Test	2-02
Abort System Self-Test Function	2-03
Enter CObug Function	2-03
Keyboard Test	2-04
Printer Test	2-04
Diskette All Sectors Read-Only Test	2-04
Diskette All Sectors Write/Read Test	2-05
System Self-Test Commands	2-06
System Self-Test Error Messages	2-07
CHAPTER 3. OPERATION (DISK-BASED SYSTEMS)	3-01
System Reset	3-01
Power-On Reset and Self-Test	3-01
Switch-Initiated Reset	3-02
System Self-Test	3-02
Abort the System Self-Test Function	3-03
Enter the CObug Function	3-03
Entering CObug without Initialization	3-04
Keyboard Test	3-04
Printer Test	3-04
Read-Only Disk Test	3-05
Read/Write Disk Test	3-05
System Self-Test Commands	3-06
System Self-Test Error Messages	3-06
CHAPTER 4. INTRODUCTION	4-01
CObug Installation	4-02
CHAPTER 5. CObug COMMANDS	5-01
Invoking CObug (diskette-based systems)	5-01
Invoking CObug (disk-based systems)	5-02
Uninitialized CObug Starting (disk-based systems)	5-02
Initialized CObug Starting (disk-based systems only)	5-03
Running CObug starting	5-03
CObug Commands	5-04
Print Command (PRNT)	5-05
Load Command (LOAD)	5-06
Restart Command (RSTR)	5-06
Memory Examine and Change Command	5-06
Insert, Display, and Remove Breakpoint Command	5-07

Table of Contents	Page
Target Program Execution Commands	5-08
Display Contents of MPU Command	5-09
Cobug Error Messages	5-09
CHAPTER 6. Cobug SUBROUTINES	6-01
BEGEND	6-04
CBCDHX	6-05
CHEXL	6-06
INADDR	6-07
INCH	6-08
INCHNP	6-09
OUTCH	6-10
OUT2HS	6-11
OUT4HS	6-12
PCRLF	6-13
PDATA	6-14
PDATA1	6-15
PSPACE	6-16
BYTE	6-17
LPINIT	6-18
LIST	6-19
LDATA	6-20
LDATA1	6-21
CONTRL	6-22
CKBRKK	6-23
NMI	6-24
REMSWI	6-25
INZMRO	6-26
SFE	6-27
OUT2HD	6-28
OUT4HD	6-29
PCR	6-30
MRC2	6-31
MRC4	6-32
SWAPHL	6-33
PCHDA	6-34
PREGS	6-35
INAD20	6-36
RBRS	6-37
INADD	6-38
MEMTST and MEMT1	6-39

CHAPTER 1. INTRODUCTION

System Self-Test performs a complete test of Codex Intelligent Terminal Systems and their links to system peripherals. When power is first applied to the system, System Self-Test initializes and tests the systems and its components, reinitializes the system, and indicates any errors or equipment malfunctions.

During the system evaluation phase of its operation, System Self-Test progressively checks each system component, starting with the most basic, and communicates with the operator by whatever means are available (for example, if an equipment malfunction in the CDX-68 Basic Display Terminal is detected, the appropriate error message is printed on the system printer). No system component is employed for testing purposes until it has been itself tested and found to be operating correctly.

System Self-Test requires minimal operator intervention. Under normal conditions, System Self-Test tests system operation and loads the Codex Disk Operating System (CODOS) into memory. The user can, however, exercise considerable control of System Self-Test by using its manual mode.

System Self-Test resides on a 2K-byte firmware module installed on the microcomputer module of the Codex Intelligent Terminal System.

System Self-Test features include:

- * Evaluates and verifies the basic integrity of CDX-68 Basic Display Terminal-based systems.
- * Displays any error conditions on the CDX-68 Basic Display Terminal display screen or system printer.
- * Minimal operator interface requirements.
- * Progressively checks the operation of the various parts of the system, starting with the most basic components. No portion of the system is used in the test until its integrity has been proven.
- * Advises operator of status and prompts operator with suggested action whenever testing is held or stopped.

In addition, System Self-Test performs the following tests and functions:

- * Initializes the interrupt jump table.
- * Resets the CDX-68 Basic Display Terminal.

- * Tests the CDX-68 Basic Terminal RS-232C communications link.
- * Provides option for testing keyboard.
- * Tests and initializes firmware scratchpad RAM memory.
- * Tests and initializes the bottom 32 bytes of RAM used for disk or diskette controller parameters.
- * Provides the option of testing the parallel printer interface link and printing of character set.
- * Tests the selected disk or diskette controller firmware PROM.
- * Sizes RAM memory to 32K, 48K, 56K, or 58K bytes and displays corresponding memory message.
- * Performs fast random memory test on the remainder of RAM memory (20 hex to the sized top-of-memory).
- * Performs read-only tests over the full head-movement range (half range for double-sided diskettes) of all system disk and diskette drives.
- * Provides optional all-sector read test on all system drives.
- * Provides optional read/write test on all sectors of ready diskette drives or on sectors 23 and 24 of ready disk drives.
- * Prevents operator from loading CODOS when required portion of system is inoperable.
- * Provides jump table entry to calculate CRC signature over desired memory range and to compare with expected values.
- * Provides jump table entry to generate standard disk error messages.
- * Provides 3 jump table entries to keyboard polling routines.
- * Provides jump table entry to the routine calculating which sector in a multiple disk access was in error.
- * Indicates end-of-test status.

Hardware Support Required

The minimum hardware configuration required to support System Self-Test consists of:

- CDX-68 Basic Display Terminal with the appropriate firmware modules.

SST also supports a wide variety of optional hardware.

Additional Hardware Supported

System Self-Test supports nearly all Codex Intelligent Terminal System hardware components, including disk and diskette data storage, up to 56K bytes of memory, and a variety of printers.

Software Support Required

System Self-Test requires no software support, although it is typically used in conjunction with the Codex Disk Operating System (CODOS) and the Codex system debugger (CObug).

Firmware Installation

System Self-Test resides on a 2K EPROM which plugs directly into the Microcomputer Module D of Codex Intelligent Terminal Systems.

To install the System Self-Test EPROM, first remove any chip already in socket U14 of the Microcomputer Module D circuit board. Place this chip immediately in conductive foam in order to prevent damage. Remove the System Self-Test EPROM from its conductive foam and plug it into socket U14.

As with any such discrete component, the System Self-Test EPROM is sensitive to damage from improper handling and static electricity. The chip should be stored in conductive foam when not in place on the circuit board.

CHAPTER 2. OPERATION (DISKETTE-BASED SYSTEMS)

In normal operation, System Self-Test requires very little attention from the operator; however, it can also provide a variety of operational tests and commands that enhance its testing capabilities. This chapter discusses first the automatic test procedures and then each of the optional tests. Table 1 describes SST commands; Table 2 lists SST error messages along with the most likely cause for each.

Automatic Testing Procedures

SST is initialized when power is first applied to the CDX-68 Basic Display Terminal and again each time the reset switch (on the Terminal's back cover) is actuated. The section Power-On Initialization and Self-Test describes the former and Reset Initialization and Self-Test discusses the latter.

Power-On Initialization and Self-test

CAUTION! BE SURE THAT THE DOORS ON THE CDX-FS SERIES DISKETTE DRIVES ARE OPEN AND THAT NO DISKETTES ARE IN PLACE. OTHERWISE, THE CONTENTS OF THE DISKETTE MAY BE DESTROYED WHEN POWER IS APPLIED.

- a. With the doors on the diskette drives open and no diskettes in the drives, apply power to the CDX-68 Basic Display Terminal, the CDX-FS Series Diskette Storage system, and the CDX-SP Series Printer (if attached). System Self-Test initializes the system, performs a portion of the selftest routine, and, if no errors are detected, displays:

```
SYSTEM SELFTEST X.X  
Copyright 1979 by CODEX Corp
```

```
Insert Disk 0
```

where X.X is the revision number.

Note that should SST detect an error, it displays the appropriate error message.

- b. Insert a CODOS system diskette into each diskette drive and close the drive doors. SST completes the system self-test, reinitializes the system, loads the CODOS Disk Operating System into memory, and displays:

```
56K memory operational
END SYSTEM SELFTEST
```

```
CODOS X.X
=
```

Note that the amount of memory indicated by the first statement may vary but does not affect SST operation; also the CODOS version number may be different without affecting SST operation.

In the above screen message, the equal sign (=) is an indication that the CODOS Disk Operating System has completed its previous task and awaits another. This is the only time when diskettes should be inserted or removed or power removed from the system.

Reset Switch Initialization and Self-Test

Press the reset switch on the back of the CDX-68 Basic Display Terminal (the switch is in the upper right-hand corner of the terminal when viewed from behind). System Self-Test initializes the system, performs a portion of the selftest routine and, upon detecting that drive 0 is ready, completes the system self-test, reinitializes the system, and loads the CODOS Disk Operating System into memory, resulting in the following display:

```
SYSTEM SELFTEST X.X
Copyright 1979 by CODEX Corp
```

```
Insert Disk 0
```

```
56K memory operational
```

```
END SYSTEM SELFTEST
```

```
CODOS X.X
```

```
=
```

Note that the amount of memory or the version of CODOS may vary without affecting System Self-Test operation.

Abort System Self-Test Function

The Abort System Self-Test function may be entered immediately after the message "Insert Disk 0" is displayed, while SST is waiting for the operator to respond. This function brings SST to an orderly conclusion and returns the system to CODOS.

- a. Enter a CONTROL-A (that is, press the A and CTRL keys simultaneously). The system aborts System Self-Test and displays:

```
SYSTEM SELF-TEST X.X  
Copyright 1979 by CODEX Corp
```

```
Insert Disk 0
```

```
END SYSTEM SELFTEST
```

```
Waiting Drive 0
```

- b. Insert a diskette into diskette drive 0 and close the drive door. The system initializes itself, loads the CODOS Disk Operating System into memory, and displays:

```
CODOS X.X
```

```
=
```

Again, the version number of CODOS may vary without affecting SST operation.

Enter CObug Function

Note that the CObug firmware may be entered any time that System Self-Test is waiting for an operator response.

- a. With the CDX-68 Basic Display Terminal in one of the System Self-Test wait states, enter a CONTROL-C (that is, press the C and CTRL keys simultaneously). System Self-Test initializes the system, goes to CObug, and displays:

```
COBUG X.X
```

```
*
```

Note that the version number of CObug may be different without affecting operation.

In the above message, the asterisk (*) is the CObug ready message.

Keyboard Test

The keyboard test function may be entered immediately after the message "Insert Disk " is displayed, while SST is waiting for the operator to respond. This function permits testing of all keyboard keys.

- a. Enter a CONTROL-K (that is, press the K and CTRL keys together). System Self-Test initiates the keyboard test routine and displays:

Keyboard Test

- b. Enter one at a time each of the keyboard characters except BREAK, ALL CAPS, AUTO LF, ON-LINE, PAGE MODE, and AUX PORT. The system displays each displayable character as it is entered and displays the hexadecimal equivalent of the non-displayable characters.
- c. Enter a BREAK character to exit from this test. The system displays:

End Keyboard Test

Printer Test

The printer test function may be entered immediately after the message "Insert Disk 0" is displayed, while SST is waiting for the operator to respond. This function permits testing of the system printer.

- a. Enter a CONTROL-P (press the CTRL and P keys simultaneously). The printer prints:

```
SYSTEM SELF-TEST X.X
Copyright 1979 CODEX Corp.
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNO
PQRSTUVWXYZ[\]_`abcdefghijklmnopqrstuvwxyz{|}`
```

Diskette All Sectors Read-Only Test

This function may be entered immediately after the message "Insert Disk " is displayed, while SST is waiting for the operator to respond. This function tests not only the diskette drives but the diskettes themselves.

- a. Enter a CONTROL-R (press the R and CTRL keys together). The system displays:

```
READ Disk Test
Depress RETURN to continue
```

- b. Insert a diskette into each drive to be tested, close the drive doors, and press the RETURN key. The diskette drive reads all sectors on the diskettes (2002 on a single-sided diskette and 4004 on a dual-sided diskette), verifies their CRSSs, and displays:

End Disk Test

Depress RETURN to continue
- c. Open the door to drive 0 if the "Insert Disk 0" prompt is required. Then press RETURN. If drive 0 is closed and ready, the System Self-Test will continue; if successful, it attempts to load CODOS from the diskette in drive 0.

Diskette All Sectors Write/Read Test

This function may be entered immediately after the message "Insert Disk 0" is displayed, while SST is waiting for the operator to respond. This function tests all functions of the diskette storage system, including the diskettes themselves.

CAUTION! THIS TEST WILL DESTROY ALL THE DATA STORED ON THE DISKETTE UNDER TEST!

- a. Enter a CONTROL-W (press the W and CTRL keys together). The system displays:

WRITE Disk Test
Depress RETURN to continue
- b. Insert a diskette into each drive to be tested, close the doors, and press the RETURN key. The diskette drive writes the test pattern A55A onto the diskette, reads and verifies the CRC characters, and displays:

End Disk Test

Depress RETURN to continue
- c. Remove the diskettes and press RETURN. The system displays:

Insert Disk 0

The CDX-68 Basic Display Terminal is now ready for another System Self-Test or the insertion of a CODOS diskette.

Table 1. System Self-Test Commands

<u>Keyboard Entry</u>	<u>Description</u>
Control-A	Aborts System Self-Test and bootloads CODOS immediately
Control-C (1)	Enters CObug system debugger
Control-K	Performs the keyboard test
Control-P	Performs the printer test
Control-R	Performs the all-sectors read-only diskette test
Control-W	Performs the all-sectors read/write diskette test

- (1) This command may be entered from any System Self-Test wait state. The other commands can be entered only in an "Insert Disk 0" System Self-Test wait state.

Table 2. System Self-Test Error Messages

<u>Message</u>	<u>Probable Cause</u>
Console Not Ready	The CDX-68 Basic Display Terminal and the Microcomputer Module D failed to communicate.
Printer Not Ready	The Microcomputer Module D and the Printer Interface Module have interface problems. This message is normal when a printer is not connected into the system or when it is selected without being powered on.
MEMORY ERROR	A memory error problem occurred.
nnnn MEMORY ERROR	The memory was sized to a non-standard top-of-memory address nnnn.
Disk Parameter MEMORY ERROR (Bell Sounds)	A memory problem below Memory location 0020 hex occurred. The RAM memory on the Microcomputer Module D was unresponsive to System Self-Test.
DISK ROMCRC ERRO nnnn instead of mmmmm	The Microcomputer Module D, with its System Self-Test, cannot communicate with the Diskette Controller Module, particularly its firmware PROM.
SST ROMCRC ERROR	This error indicates a problem with nnnn instead of mmmmm the SST PROM.
CObug ROMCRC ERROR	This error indicates a problem with nnnn instead of mmmmm the CObug PROM.
INOPERABLE - END SYSTEM SELF TEST	One of three identified failures was detected: 1) CObug ROMCRC ERROR; 2) Console not ready; or 3) DISK ROMCRC and CObug not entered.

MessageProbable Cause

DISK ERROR #, drive
n, sector nnnn

This error indicates an error with the diskette drive, the interconnect cable, or the diskette in question. Drive error numbers are:

- 1 - Data CRC error
- 2 - Diskette is write-protected
- 3 - Diskette not ready
- 4 - Read deleted data mark error
- 5 - Timeout error
- 6 - Invalid diskette address
- 7 - Seek error
- 8 - Data mark error
- 9 - Address mark CRC error
- D - System failed all-sectors write/read disk test

Waiting for Drive 0

Drive zero is not ready for bootloading. This message is generally not an indication of a defect; System Self-Test attempts to bootload CODOS when the operator readys drive zero.

INTERRUPT
disregarded

System disregards an unexpected interrupt (IRQ, NMI, or SWI) during System Self-Test.

CHAPTER 3. OPERATION (DISK-BASED SYSTEMS)

This chapter describes the operating procedures and error messages of System Self-Test for systems using 10Mb Disk Storage. In normal operation, System Self-Test requires virtually no operator effort; System Self-Test also includes a variety of tests and commands, used manually, that provide a wider range of tests. This chapter describes first the procedures required to test the system and operate it in conjunction with the CODOS Disk Operating System; each of the tests is then described separately. Table 3 describes each of the commands; Table 4 lists the error messages and their probable causes.

System Reset

System Self-Test is reset when power is first applied to the CDX-68 Basic Display Terminal and each time the Terminal reset switch (on the back of the Basic Display Terminal) is actuated. Both means of resetting the system are discussed in the following paragraphs.

Power-On Reset and Self-Test

- a. With a cartridge disk loaded (see the 10 Mb Disk Storage Operator's Guide), set the PROT FIXED switch to the off position and the PROT RMVBL switch to the on position.
- b. Apply power to the Basic Display Terminal, the disk drive, and the system printer. Move the DISC DRIVE switch to the on position and then apply power to the system printer. The power and drive lamps on the Basic Display Terminal, the disk drive, and the printer light. After the initial warm-up delay, the system displays:

```
SYSTEM SELF-TEST X.X  
Copyright 1979 by Codex Corp.  
Depress RETURN to continue
```

(the X.X represents the version number).

Note: Should the system detect an error, it displays the appropriate error message on the screen or printer (the error messages are listed at the end of this chapter).

- c. Depress the SELECT switch to enable the printer to receive data. The SELECT lamp lights.

The operator now has the option of:

Performing the System Self-Test

Aborting the Self-Test function and bootloading the
CODOS Disk Operating System

Entering and initializing CObug

Performing the keyboard test

Performing the printer test

Switch-Initiated Reset

Press the RESET switch on the back of the Basic Display Terminal. The system aborts the operation it is currently performing and displays:

```
SYSTEM SELF-TEST X.X  
Copyright 1979 by Codex Corp.  
Depress RETURN to continue
```

The user now has the option of:

Performing the System Self-Test

Aborting the Self-Test function and boot-loading
the CODOS Disk Operating System

Entering CObug without initialization

Entering and initializing CObug

Performing the keyboard test

Performing the printer test

System Self-Test

Depress the RETURN key. System Self-Test initializes and tests the system and then displays:

```
56K memory operation  
END SYSTEM SELF-TEST X.X  
CODOS X.X  
=
```

or:

```
56K memory operational  
Waiting for LUD 0
```

Depending on whether or not the disk drive is ready (as indicated by the READY light on the front panel of the

drive). Note that the amount of memory in the above statement may vary without affecting operation.

With the waiting LUD 0 output, the Terminal is in a wait state. The user now has the following options (in addition to enabling the drive):

Aborting the System Self-Test function and boot-loading the CODOS Disk Operating System

Entering and initializing CObug

Performing the read-only test on the disk

Performing the read/write tests on the disk

Abort the System Self-Test Function

This function may be entered directly after the system reset wait state or the "Waiting LUD 0" wait state.

Enter a CONTROL-A (that is, press the A and CTRL keys simultaneously). The system aborts the self-test function, loads the CODOS Disk Operating system into memory, and displays:

```
CODOS X.X
=
```

Enter the CObug Function

System Self-Test can enter and initialize CObug from either the system reset wait state or the "Waiting LUD 0" wait state.

Enter a CONTROL-C. The system jumps to CObug and displays:

```
CObug X.X
Copyright 1979 by Codex Corp.
*
```

(as shown in the above example, the asterisk (*) is the CObug prompt).

Entering CObug without Initialization

The user can abort a target program, and, using the CObug functions, examine the contents of memory, including the stack. This function may be entered only through the system reset wait state. Enter a CONTROL-B. The system jumps to CObug and displays:

```
CObug X.X
Copyright 1979 by Codex Corp.
*
```

Keyboard Test

- a. With the system in a system reset wait state, enter a CONTROL-K. System Self-Test jumps to the keyboard test routine and displays:

```
Keyboard Test
```

- b. Enter one at a time each of the keyboard characters except BREAK, ALL CAPS, AUTO LF, ON-LINE, PAGE MODE, and the AUX PORT keys. The system displays each displayable character as it is entered and displays the hexadecimal equivalent of the non-displayable characters.
- c. To exit from this test, depress the BREAK key and observe the message:

```
End Keyboard Test
```

Printer Test

With the system in a system reset wait state, enter a CONTROL-P from the keyboard. The printer prints:

```
SYSTEM SELF-TEST X.X
Copyright 1979 CODEX Corp
!"#$%&'()*-*,./0123456789:;<=>?ABCDEFGHIJKLMNO
PQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxy{\\}
```

Read-Only Disk Test

- a. With the system in a "Waiting LUD 0" state, enter a CONTROL-R. The system displays:

Read Disk Test
Depress RETURN to continue

- b. Ready those disk drives (LUDs) that are to be tested and then press RETURN. The system reads all sectors on all ready LUDs and displays:

End Disk Test
Depress RETURN to continue

Read/Write Disk Test

- a. Ready those disk drives (LUDs) which are to be tested and then press RETURN. The system displays:

Write Disk Test
Depress RETURN to continue

- b. Enter a RETURN character. The system performs a two-surface read/write test on sectors 23 and 24 of all ready LUDs and displays:

End Disk Test
Depress RETURN to continue

- c. If System Self-Test is to continue to completion, ready LUD 0 before depressing RETURN. Otherwise, depress RETURN to go to the "Waiting LUD 0" state.

Table 3. System Self-Test Commands
(for disk-based systems)

<u>Keyboard Entry</u>	<u>Description</u>
CONTROL-A (Note 2)	Aborts System Self-Test and boot-loads CODOS immediately
CONTROL-B (Note 2)	Enters CObug without initialization
CONTROL-C (Note 1)	Enters and initializes CObug
CONTROL-K (Note 2)	Performs the keyboard test
CONTROL-P (Note 2)	Performs the printer test
CONTROL-R (Note 3)	Performs the all-sector read-only disk test on all ready LUDs.
CONTROL-W (Note 3)	Performs the two-surface read/write disk test on all ready LUDs.

Note 1: May be entered in any wait state.

Note 2: May be entered only after power is applied or the RESET switch on the back of the Basic Display Terminal is actuated (the "Depress RETURN to continue" state).

Note 3: May be entered only after the RETURN key has been pressed and the Terminal Screen is displaying "Waiting LUD 0."

Table 4. System Self-Test Error Messages

<u>Message</u>	<u>Probable Cause</u>
SST ROMCRC ERROR	This indicates an error with the SST EPROM.
Console Not ready	The CDX-68 Basic Display Terminal and the Microcomputer Module D failed to communicate.
PRINTER NOT READY	The Microcomputer Module D and the CDX-SP series printer have interface problems. This message is normal when the system has no printer or if the printer is not powered on while selected (placed on-line).

<u>Message</u>	<u>Probable Cause</u>
MEMORY ERROR	A memory error problem occurred.
nnnn MEMORY ERROR	The memory was sized to a non-standard, top-of-memory address nnnn.
Shared MEMORY ERROR	The 10 Mb Disk Controller Shared Memory Module failed its memory test.
Disk Parameter MEMORY ERROR	A memory problem occurred below memory location 0020 hex
Bell Sounds Continuously	Microcomputer Module D RAM memory was unresponsive to System Self-Test.
DISK ROMCRC ERROR	The Microcomputer Module D, with its system Self-Test firmware, cannot communicate with the Disk Controller Interface Module, particularly its firmware PROM.
INOPERABLE- END SYSTEM SELF-TEST	One of three identified failures detected: 1) CObug ROMCRC ERROR, 2) Consoles not ready, 3) DISK ROMCRC and CObug not entered.

Message -----	Probable Cause -----
DISK ERROR #, LUD n, sector nnnn	This error indicates an error with the disk drive, the interconnect cable, or the disk in question. LUD error numbers are: 0 - No errors, normal return status 1 - Data CRC error 2 - Header write protect error 3 - LUD not ready 4 - Header bad sector indicator 5 - Disk operation not complete 6 - Invalid disk address 7 - Seek error 8 - Compare error 9 - Data or control FIFO error : - Data buffer location error ; - Read or write error at sector time
CObug ROMCRC ERROR	This error indicates a problem with the CObug PROM or, alternatively, that the PROM is not available.
Waiting for LUD 0	Drive zero is not ready. This message is not generally an indication of a defect (usually, the drive is not yet on-line). System Self-Test attempts to boot-load CODOS as soon as the operator readys drive 0.
INTERRUPT Disregarded	System disregards an unexpected interrupt (IRQ, NMI, or SWI).

CHAPTER 4. INTRODUCTION

CObug is a comprehensive, easy-to-use software development and debugging tool. Designed for use with Codex Intelligent Terminal Systems, CObug provides a full set of memory test and Input/Output subroutines. These subroutines may be executed in a stand-alone mode to test for specific conditions or to examine the operation of a specific program; they may also be incorporated into user programs (an entry jump table provides access to the different subroutines).

An interactive program, CObug permits users to select different options through the keyboard of the CDX-68 Basic Display Terminal; the results of the subroutine may be viewed on the Terminal's display screen or directed to a system printer for hard-copy output. CObug resides in a 2K-byte firmware module that plugs into the microcomputer module of the Codex Intelligent Terminal System.

CObug features include:

- Interactive access

- Provides debug and basic input/output capability to the CDX-68 Basic Display Terminal

- Provides easy user access to subroutines

- Provides 12 user commands

In addition, CObug permits the user to perform the following functions:

- load the CODOS Disk Operating System into system memory

- display, and, if required, change the contents of memory

- display, and, if required, change the contents of the MPU's registers

- set breakpoints

- display breakpoints

- remove breakpoints singly

- remove all breakpoints

- continue running the target program from the current location

go to specified location and begin running the target program

restart the System Self-Test

Print the contents of memory in both hexadecimal and ASCII form

CObug is interactive, allowing the user to communicate with it through the CDX-68 Basic Display Terminal. In addition, the user can easily incorporate CObug subroutines into his applications programs. The user gains access to these subroutines through entry points listed in a jump table; these subroutines are described in Chapter 7.

CObug Installation

CObug is shipped either installed (in socket U13 of the Basic Display Terminal's Microcomputer Module D) or separately (as a replacement part). In the latter case, CObug is shipped in conductive foam to protect the chip from static electricity. To install the replacement PROM, remove any PROM already in socket U13 on the terminal's Microcomputer Module D, insert this PROM in the conductive foam, and install the new PROM in the socket.

CAUTION: The CObug PROM is susceptible to damage from static electricity and improper handling. Store this PROM (and all others) in conductive foam when not in place on the circuit board.

CHAPTER 5. CObug COMMANDS

This chapter discusses the CObug commands. These commands are listed in Table 1 and discussed in the following paragraphs. Any of these commands may be entered when CObug is displaying its prompt. No CARRIAGE RETURN entry is required to initiate a command. All parameters entered are assumed to be hexadecimal values.

Invoking CObug (diskette-based systems)

The System Self-Test (SST) program for diskette-based systems provides the user with a command to enter CObug. For detailed information on SST and its operation, see the first part of this Guide.

CObug may be entered from SST by entering the command CTRL-C (that is, pressing the CONTROL and C keys simultaneously) at any SST wait state. For example:

- a. After applying power to all terminal system components, open the door of diskette drive number zero (the left-most drive) and press the reset switch on the back panel of the Terminal. Upon detecting no errors, the system displays:

```
SYSTEM SELF-TEST X.X
Copyright 1979 by Codex Corporation
Insert Disk 0
```

- b. Instead of readying drive 0, enter a CONTROL-C (press the CTRL and C keys simultaneously) to enter and initialize CObug. CObug displays:

```
CObug X.X
*
```

where the asterisk (*) is the CObug prompt (that is, CObug has finished its previous task and awaits another).

- c. In addition, any time System Self-Test is awaiting an operator response (when any "Depress RETURN to continue" message is displayed), entering a CONTROL-C causes the system to enter CObug.
- d. CObug may also be engaged by entering the command "COBUG" in response to the equal sign (=) prompt displayed by the CODOS Disk Operating System (CObug is part of the System Test and Diagnostic package).

- e. Some special CDX-68 systems have an ABORT button on the front panel or an NMI switch located on the rear panel. Depressing either of these switches causes the system to enter CObug (providing the interrupt table is intact).
- f. CObug may be entered under software control using the START entry (equivalent to the CTRL-C command) or the NMI entry (equivalent to the CTRL-B command).

Invoking CObug (disk-based systems)

The System Self-Test (SST) program for disk-based systems provides the user with two commands to enter CObug. Each is described below.

As an aid to distinguishing the two means of entering CObug, remember that CONTROL-C restarts CObug and executes all CObug initialization code before performing the debug function. CONTROL-B (CObug without the initialization of the complete CObug function) assumes that CObug has been previously initialized and goes to the delay function. If CObug has not been initialized since the last time power was turned on (SST does this if allowed to run interrupted), then depressing CONTROL-B will produce undefined results.

For detailed information on the CDX-SST/D System Self-Test Firmware and its operation, refer to the first part of this Guide.

Uninitialized CObug Starting (disk-based systems)

To start CObug when it has not been previously initialized:

- a. Reset the CDX-68 Basic Display Terminal (the Terminal is reset when power is first applied or when the reset button on the rear panel is actuated). Upon detecting no errors, the system displays:

```
SYSTEM SELF TEST X.X
Copyright 1979 by Codex Corporation
Depress RETURN to continue
```

- b. Instead of depressing the RETURN key to continue SST, the operator may enter a CONTROL-C (the CTRL and C keys depressed simultaneously) to enter and initialized CObug. CObug then displays:

```
CObug 5.0
*
```

where the asterisk (*) is the CObug prompt (that is, CObug has finished its previous task and awaits another).

- c. In addition, any time System Self-Test awaits an operator response (any time the "Depress RETURN to continue" message is displayed), CONTROL-C may be entered to engage CObug.
- d. CObug may be engaged by entering the command "GOCOBUG" in response to the CODOS equal sign (=) prompt (CObug is part of the System Test and Diagnostic package).

Initialized CObug Starting (disk-based systems only)

Running CObug starting

In the running start of CObug, CObug does not initialize itself before performing its debug function. This permits a user to abort a program, and using CObug, to examine memory locations which would otherwise have been changed by the CObug initialization.

- a. Actuate the reset switch located on the rear panel of the CDX-68 Basic Display Terminal. The system displays:

```
System Self-Test X.X
Copyright 1979 by Codex Corporation
Depress RETURN to continue
```

- b. Enter a CONTROL-B (that is, the CTRL and B keys depressed simultaneously). The system displays:

```
CObug X.X
Copyright 1979 by Codex Corporation
*
```

On specially-equipped CDX-68 systems, use of the ABORT or NMI buttons can cause control to immediately enter CObug (equivalent in effect to the CTRL-B command). When debugging a target program, this circumvents the need to enter CObug by way of System Self-Test.

Table 5. CObug Commands

Command	Execution
PRNT	Print memory in both hexadecimal and ASCII format.
LOAD	Load CODOS.
RSTR	Restart the system self test.
n/	Display contents only.
n/m c	Display contents of memory (CObug displays memory "m") and then replace contents with "c."
LF	Open the next sequential location to n.
^	Open the previous location to n.
CR	Close the open location.
X	Return to the CObug scanning loop.
n;V	Set a breakpoint at location n.
;U	Remove all breakpoints.
n;U	Remove the breakpoint at location n.
n;G	Execute the target program at n.
;P	Continue executing from encountered breakpoint.
\$V	Display all breakpoints.
\$R	Display/change registers.

Print Command (PRNT)

This command prints the specified portion of the current memory map in both hexadecimal and ASCII forms.

- a. Enter the word PRNT. The system displays:

BEG ADR nnnn on the next line.

Note: The nnnn is the beginning address used in the previous PRNT function; if it does not need to be changed, enter a CR.

- b. Enter in hexadecimal the selected beginning address and a CR. (Leading zeroes need not be entered.) CObug stores the entered address and displays:

END ADR nnnn

Note: The nnnn is the ending address used in the previous PRNT function; if it does not need to be changed, enter a CR.

- c. Enter in hexadecimal the selected ending address and a CR. The system now tests whether the ending address is larger than the beginning address.
 1. If the ending address is smaller, the system displays the following message. Proceed to step a.

BEG ADR nnnn

2. If the starting address is smaller, the system displays:

ON PRT

- d. Enter a "Y" to send the display to the printer, an "X" to abort the function and return to the CObug * (note that this is the only way to exit this function), and enter anything else to send the display to the display screen. The system prints or displays the memory contents as appropriate and then displays:

BEG ADR nnnn

- e. Proceed to step b.

Note: Entering a CONTROL-W while the memory contents are being displayed causes the display to wait at the end of the current line until some other character is entered.

Load Command (LOAD)

Enter the word LOAD. The system loads the CODOS Disk Operating System into system memory and displays the following message:

```
CODOS X.X  
=
```

where the equal sign (=) is the CODOS prompt.

Restart Command (RSTR)

With all disk drives off-line, enter the command RSTR. This command returns the system to System Self-Test. At this point, it is similar to when the equipment was just powered up or as if the reset switch on the back of the Basic Display Terminal had been actuated. Refer to "Invoking CObug," above.

Memory Examine and Change Command

CObug performs this command in three steps: examining the contents of a selected memory location (opening the memory location), changing the contents of this memory location if required, and closing the memory location.

- a. Enter the address of the memory location to be examined and a / (slash) character. CObug reads the memory location and displays the contents in hexadecimal.

Note: If the memory contents are not to be changed, proceed to step c.

- b. If the contents of the memory location are to be changed, enter (in hexadecimal) the new data to be stored at this location.

The operator now must decide in closing this memory location whether to return to CObug control program, to examine the previous sequential memory location, or to examine the following sequential memory location.

- c. Perform one of the following substeps and close this memory location.
 1. Enter a CR character. CObug closes the present memory location, returns to its control program, and displays an asterisk on the next line.
 2. Enter an ^ (Up Arrow) character. CObug closes this memory location and examines the previous sequential memory location. The system now displays the previous memory address, a slash, and the contents of this memory location. Return to step b.
 3. Enter a LF (Line Feed) character. CObug closes this memory location and examines the next sequential memory location. The system now displays the new memory address, a slash, and the contents of this memory location. Return to step b.

Insert, Display, and Remove Breakpoint Command

These commands enable CObug to insert up to eight breakpoints in a target program, to display the breakpoints on command, and to remove the breakpoints. The breakpoint sequence is:

User designates the breakpoint(s') address(es)

User initiates the execute function

CObug replaces the program instruction(s) with breakpoint(s) and executes the target program

On detecting a breakpoint, the system returns to CObug.

CObug inserts the program instruction and the system displays the contents of the MPU registers.

NOTE: If the user should initiate the execution from an address where a breakpoint is set; that breakpoint will not be established in memory, but all other breakpoints will be in memory. This is done to prevent looping on a breakpoint.

Entering a Breakpoint (n;v)

- a. Enter the address (n) of the selected breakpoint, a semicolon, and a v. CObug enters the breakpoint in the target program, returns to its control program, and displays an asterisk.

Displaying all Breakpoints (\$V)

- b. Enter the character \$V. CObug displays all the breakpoints, returns to its control program, and displays an asterisk.

Removing Breakpoints (n;U or ;U)

The user has the option of removing just one breakpoint (step c) or of removing all the breakpoints at one time (step d).

- c. To remove a single breakpoint, enter the address (n) and the character ;U (semicolon U). CObug removes this breakpoint and displays an asterisk.
- d. To remove all the breakpoints, enter the characters ;U (semicolon U). CObug removes all the breakpoints and displays an asterisk.

Target Program Execution Commands

The user has the option of selecting whether he wants to start running the target program from an address given in the command or from the present target program address (program counter address). Refer to Table 6 and enter the appropriate command.

Table 6. Target Program Commands

Command	Function
n;G	Set the program counter to address n and start executing the target program.
;P	Start from the present program counter address and continue executing the target program.
n;P	Start from the present program counter address, and if a breakpoint is set at this address, establish a bypass of n-1.

The system executes the target program until one of the following conditions has been reached:

An SWI MPU instruction is detected.

A WAI MPU instruction is detected.

The system RESET switch on the back of the Basic Display Terminal is actuated.

Display Contents of MPU Command

This function enables CObug to display the contents of the system M6800 MPU registers for examination and change.

- a. Enter the characters \$R. The system displays the contents of the MPU registers in the following sequence: program counters, index register, A accumulator, B accumulator, condition code register, and stack pointer.
- b. If the contents of an MPU register are to be changed, enter the new hexadecimal value after the current value is displayed.
- c. Enter RETURN to exit this sequence and go immediately to the next asterisk (*) prompt. Enter RETURN only to leave the current value unchanged and exit. Enter anything else to continue.

CObug Error Messages

NO CHANGE

Memory change is attempted on memory out of range or on read-only memory.

BEEP (bell sound)

General error response that appears when incorrect forms of commands are entered, commands are entered at inappropriate times, or commands are unknown.

NO BKPTS SET

May appear in response to the \$V command. All breakpoints have been removed by use of the ;U command or an attempt has been made to set a breakpoint at an undefined location.

RDY DSK

The disk was not ready when a LOAD command was attempted. CObug returns to its asterisk prompt.

CHAPTER 6. CObug SUBROUTINES

This chapter discusses the CObug subroutines. Table 7 identifies each subroutine while the following paragraphs describe the subroutines in detail, including call, required subroutine inputs, and subroutine outputs.

Table 7. CObug Subroutines

Entry Addr -----	Mnemonic -----	Function -----
F000	START	Restart CObug monitor
F003	BEGEND	Get begin and end addresses
F006	CBCDHX	Convert ASCII hex to binary
F009	CHEXL	Convert 4MSB to ASCII hex
F00C	CHEXR	Convert 4LSB To ASCII hex
F00F	INADDR	Get hex address indirect (X)
F012	INCH	Receive one character
F015	INCHNP	Receive one character no parity
F018	OUTCH	Send ASCII character
F01B	OUT2HS	Display 2 hex characters followed by a space (X)
F01E	OUT4HS	Display 4 hex characters followed by a space (X)
F021	PCRLF	Display a CR and LF
F024	PDATA	Display a CR, LF, and a string (X)
F027	PDATA1	Display a data string (X)
F02A	PSPACE	Display a space
F02D	BYTE	Receive a hex byte

F030	LPINIT	Initialize the printer PIA
F033	LIST	Print character in (A)
F036	LDATA	Print data string (X)
F039	LDATA1	Print CR, LF, data (X)
F03C	CONTRL	Return to CObug
F03F	CKBRKK	Check for BREAK key
F042	NMI	NMI entry point (ABORT)
F045	REMSWI	Return to CObug monitor & switch SWI vector to point to CObug monitor SWI handler
F048	INZMRO	Jump vector to initialize CObug monitor RAM only
F04B	SFE	Jump vector for the CObug monitor SWI handler
F04E	OUT2HD	Display two hexadecimal characters and a /
F051	OUT4HD	Send four hexadecimal characters and a /
F054	PCR	Display a carriage return
F057	MRC2	Change one byte
F05A	MRC4	Change double byte
F05D	SWAPHL	Swap high and low bytes
F060	PCHDA	Display a character from A and dash
F063	PREGS	Display registers
F066	INAD20	Continue input with second character
F069	RBRS	Remove breaks
F06C	SETB	Set break points

F06F	INADD	Input hexadecimal characters
F072	MEMTST	Random noise memory test (operator-supplied addresses)
----	MEMT1	Random noise memory test; addresses supplied in locations: EF80 and EF81 (beginning) EF82 and EF83 (ending)

BEGEND

(get beginning and ending addresses)

Function:

Requests the input of the beginning and ending address as defined in the random noise MEMORY TEST functions. Verifies that inputs are hexadecimal characters; also verifies that the beginning address is smaller than the ending address.

Call:

JSR BEGEND

Subroutine Input:

None.

Subroutine Output:

EF80 BEGA 16-Bit
Beginning address

EF82 ENDA 16-Bit
Ending Address

Comments:

Accumulator A and B, along with the index register, are used by this subroutine. If their contents are meaningful, they must be saved prior to calling this subroutine.

CBCDHX

(convert an ASCII hexadecimal character to a binary number)

Function:

Verify the input character is a hexadecimal digit character. Converts the character to a 4-bit binary number with the four high-order bits equal to zero. Sets N (negative) condition for non-hexadecimal characters.

Call:

JSR CBCDHX

Subroutine Input:

Character to convert must be in Accumulator A.

Subroutine Output:

Accumulator A contains the 4-bit binary number represented by the input character and the N (negative) condition code is cleared. If input is not a non-hexadecimal character, accumulator A contains the character input and the N (negative) condition code is set. The B and X registers are preserved.

CHEXL

(convert four most significant bits to ASCII hexadecimal)

Function:

Converts the four most significant bits of accumulator A to an ASCII-coded hexadecimal digit character.

Call:

JSR CHEXL

Subroutine Input:

Contents of Accumulator A.

Subroutine Output:

An ASCII-coded hexadecimal digit character in accumulator A. The B and X registers are preserved.

INADDR

(get hexadecimal address indirectly)

Function:

Converts up to four input hexadecimal characters to a 16-bit binary address.

Call:

JSR INADDR

Subroutine Input:

The index register contains the address of where to store the result.

Subroutine Output:

Eight most significant bits of 16-bit address are stored into the memory location specified by the index register. The eight least significant bits are stored into the next higher memory location. Accumulator A contains last character received. Accumulator B contains the number of hexadecimal characters received. The index register is unchanged. The subroutine returns to the calling program when an invalid character, or the fifth hexadecimal digit, is entered.

INCH

(put one character into Accumulator A)

Function:

Wait for and accept the input of one character from the CDX-68 Basic Display Terminal; if enabled echo one character back.

Call:

JSR INCH

Subroutine Input:

There is a no-echo flag (AECHO) at EF99. It must be set to other than zero before each call to INCH (for each character that is not to be echoed to the CDX-68).

Subroutine Output:

Accumulator A contains an 8-bit input character as received from the CDX-68. The B and X registers are preserved.

INCHNP

(receive one character with no parity)

Function:

Wait for and accept the input of one character from the CDX-68 and, if echo is enabled, echo the character back to terminal. Clear the high-order bit (parity) of the input character.

Call:

JSR INCHNP

Subroutine Input:

There is a no-echo flag (AECHO) at EF99. It must be set to other than zero before each call to INCHNP for each character that is not to be echoed to the CDX-68.

Subroutine Output:

Accumulator A contains the input character as received from the selected input with the high-order bit cleared. INCHNP appears as AECHO if it was not zero. The B and X registers are preserved.

OUTCH

(send ASCII character from Accumulator A)

Function:

Send one character to the CDX-68 Basic Display Terminal.

Call:

JSR OUTCH

Subroutine Input:

Accumulator A contains the character to be sent to the CDX-68 terminal.

Subroutine Output:

Accumulator A contains the character sent. The B and X registers are preserved.

OUT2HS

(print two hexadecimal characters and a space)

Function:

Converts the contents of an 8-bit binary byte to two hexadecimal characters and sends these converted characters, followed by a space character, to the CDX-68 Basic Display Terminal.

Call:

JSR OUT2HS

Subroutine Input:

The index register contains address of the byte to be converted and sent.

Subroutine Output:

Accumulator A contains the last character sent. The B accumulator is preserved and the X register is incremented by one.

OUT4HS

(print four hexadecimal characters and a space)

Function:

Convert the contents of two consecutive 8-bit bytes to four hexadecimal characters and sends the converted characters, followed by a space, to the CDX-68 Basic Display Terminal.

Call:

JSR OUT4HS

Subroutine Input:

The index register contains the address of the first byte to be converted and sent.

Subroutine Output:

Accumulator A contains the last character sent. The index register contains the input address plus two. The B accumulator is preserved.

PCRLF

(print a carriage return and line feed character)

Function:

Sends a Carriage Return (CR), a Line Feed (LF), and a Null character to the CDX-68 Basic Display Terminal.

Call:

JSR PCRLF

Subroutine Input:

None.

Subroutine Output:

Accumulator A contains a Null character (0). The B and X registers are preserved.

PDATA

(print a carriage return and line feed string)

Function:

Sends a Carriage Return (CR), a Line Feed (LF), and a user-specified string of data characters to the CDX-68 Basic Display Terminal.

Call:

JSR PDATA

Subroutine Input:

The index register contains the starting address of the user data string to send. The output string is terminated by an EOT (04) character.

Subroutine Output:

The index register contains the address of the EOT character. Accumulator A contains the EOT character. The B accumulator is preserved.

PDATA1

(print string of characters)

Function:

Sends a user-specified string of data characters to the CDX-68 Basic Display Terminal.

Call:

JSR PDATA1

Subroutine Input:

The index register contains the starting address of the user data string to send. The output string is terminated by an EOT (04) character.

Subroutine Output:

The index register contains the address of the EOT character. Accumulator A contains the EOT character. Accumulator B is preserved.

PSPACE

(print a space)

Function:

Send a space character to the CDX-68 Basic Display Terminal.

Call:

JSR PSPACE

Subroutine Input:

None.

Subroutine Output:

Accumulator A contains a space character. The B and X registers are preserved.

BYTE

(enter hexadecimal byte)

Function:

Obtains two hexadecimal characters and converts them to binary, checking each character for non-hexadecimal composition.

Call:

JSR BYTE

Subroutine Input:

None.

Subroutine Output:

Accumulator A is added to Accumulator B, with the results residing in Accumulator A.

LPINIT

(initialize the line printer)

Function:

Prepares the line printer for reception of data.

Call:

JSR LPINIT

Subroutine Input:

The index register contains the contents of the address holding the data.

Subroutine Output:

The contents of Accumulator A, indicating status input, are stored in the PIA.

LIST

(transfer character in Accumulator A to the line printer)

Function:

Prints the character from Accumulator A and checks for printer error.

Call:

JSR LIST

Subroutine Input:

Stores the contents of Accumulator A, containing data, into PIA.

Subroutine Output:

The contents of Accumulator A are sent to PIA.

LDATA

(send data string to line printer)

Function:

Prints a data string, carriage return, and line feed.

Call:

JSR LDATA

Subroutine Input:

Data string, carriage return, and line feed are sent to the printer in sequence from Accumulator A.

Subroutine Output:

The carry bit is cleared. Accumulator A is checked to determine if all data has been sent.

LDATAL

(print data string to EOT)

Function:

Print the data string in its entirety.

Call:

JSR LDATAL

Subroutine Input:

The index register contains a pointer to the data string.

Subroutine Output:

The carry bit in the condition code register is cleared.

CONTRL

(return to CObug)

Function:

Returns to CObug upon system start-up or interrupt.

Call:

JSR CONTRL

Subroutine Input:

The index register LSB is zero; the non-SWI flag is set.

Subroutine Output:

Jump is made to CObug.

CKBRKK

(check for BREAK or CONTROL key)

Function:

Check the system to determine if the BREAK or CTL-W keys were pressed.

Call:

JSR CKBRKK

Subroutine Input:

None.

Subroutine Output:

The condition code carry indicator is set if the BREAK key is pressed. The A, B, and X registers are preserved.

NMI

(non-maskable interrupt handler)

Function:

Processes non-maskable interrupts, saves user stack upon entry, sets up the CObug stack, removes all breakpoints, and prints the contents of all registers.

Call:

JSR NMI

Subroutine Input:

The stack contains the program counter values.

Subroutine Output:

Breaks are removed and the registers are printed by subroutines.

REMSWI

(return to monitor and SWI vector to monitor SWI handler)

Function:

Returns the system to CObug control by printing the SWI vector to CObug's SWI handler.

Call:

JSR REMSWI

Subroutine Input:

The index register is loaded with the SWI vector.

Subroutine Output:

The pointer from the SWI vector to CObug's SWI handler is stored and system control is returned to CObug.

INZMRO

(jump vector to initialize CObug monitor RAM only)

Function:

Creates a jump vector to initialize the CObug the RAM and initializes the SWI and NMI vectors.

Call:

JSR INZMRO

Subroutine Input:

None.

Subroutine Output:

If INZMRO was entered directly (using a JSR), a return to the caller is made. If INZMRO was entered using the start routine, START execution continues after INZMRO is complete.

SFE

(jump vector for the CObug monitor SWI handler)

Function:

Creates a jump vector for the CObug SWI handler and sets the SWI flag.

Call:

JSR SFE

Subroutine Input:

None.

Subroutine Output:

A branch is made to CObug processing.

OUT2HD

(send two hexadecimal characters plus a diagonal)

Function:

Converts the contents of a binary byte to two hexadecimal characters and sends them (followed by a diagonal).

Call:

JSR OUT2HD

Subroutine Input:

The index register contains the addresses of the bytes to be converted.

Subroutine Output:

The index register is incremented by one. Accumulator A contains the diagonal.

OUT4HD

(send four hexadecimal characters plus a diagonal)

Function:

Converts the contents of two consecutive binary bytes to four hexadecimal characters and sends them (followed by a diagonal).

Call:

JSR OUT4HD

Subroutine Input:

The index register contains the addresses of the bytes to be converted.

Subroutine Output:

The index register is incremented by two; Accumulator A contains the diagonal.

PCR

(send a carriage return)

Function:

Prints a carriage return.

Call:

JSR PCR

Subroutine Input:

Accumulator A is loaded with the carriage return.

Subroutine Output:

Accumulator A is cleared.

MRC2

(change one byte)

Function:

Changes a single byte at a designated address.

Call:

JSR MRC2

Subroutine Input:

The designated address OPENCL loaded from index register.

Subroutine Output:

Accumulator B is cleared, and the index register is again loaded from the designated address.

MRC4

(change double byte)

Function:

Changes a double byte at a designated address.

Call:

JSR MRC4

Subroutine Input:

The designated addresses OPENCL and OPENCL + 1 are loaded from the index register.

Subroutine Output:

Accumulator B is cleared and the index register is again loaded from the designated address.

SWAPHL

(swap high and low bytes)

Function:

Exchanges the contents of the memory location pointed to by the index register.

Call:

JSR SWAPHL

Subroutine Input:

The index register is loaded with PRPCH.

Subroutine Output:

The index register plus one contains the contents of Accumulator A. The index register contains contents of Accumulator B.

PCHDA

(send character from Accumulator A and a dash)

Function:

Prints the character that is stored in Accumulator A and the dash character.

Call:

JSR PCHDA

Subroutine Input:

Accumulator A is loaded with the dash character.

Subroutine Output:

A jump to the subroutine-to output-character is made.

PREGS

(display registers)

Function:

Prints the contents of all registers plus a carriage return and line feed.

Call:

JSR PREGS

Subroutine Input:

The index register loaded with the pointer-to-program registers (PRPCH; memory).

Subroutine Output:

Branches are made to send characters from Accumulator A.

INAD20

(continue input with the second character)

Function:

The input address is returned with the last character received and the number of hexadecimal characters received.

Call:

JSR INAD20

Subroutine Input:

Accumulator B contains the number of hexadecimal characters to be received.

Subroutine Output:

Accumulator A contains the last character received.
Accumulator B contains the number of hexadecimal character received.

RBRB

(remove breakpoints)

Function:

Removes breakpoints, restores instructions, and resets the breakpoint flag.

Call:

JSR RBRB

Subroutine Input:

None.

Subroutine Output:

None.

INADD

(put hexadecimal address into memory; 16-bit address)

Function:

Receives a sequence of hexadecimal characters to make an address.

Call:

JSR INADD

Subroutine Input:

The index register loaded with 16-bit address in memory.

Subroutine Output:

Accumulator B contains the hexadecimal character count.

MEMTST and MEMT1
(random noise memory test)

Function:

Tests the system memory bounded by the beginning address BEGA and ending address ENDA and produces error messages as required. If entered at MEMTST, the operator is requested to enter these addresses (see the description of the PRNT command). If entered at MEMT1, it is assumed that these are already present in BEGA and ENDA.

Call (MEMTST):

JSR MEMTST

Call (MEMT1):

LDX \$F073
TSR 3,X

Subroutine Input:

If entered at MEMT1, it is assumed that these are already present in BEGA and ENDA.

Subroutine Output:

The memory location BEGA (EF80) contains the start-of-memory address. The memory location ENDA (EF82) contains the end-of-memory address.



CODEX CORPORATION

20 Cabot Boulevard
Mansfield, Massachusetts 02048

CODEX PHOENIX

INTELLIGENT TERMINAL SYSTEMS
2002 West 10th Place
Tempe, Arizona 85281
(602) 994-6580