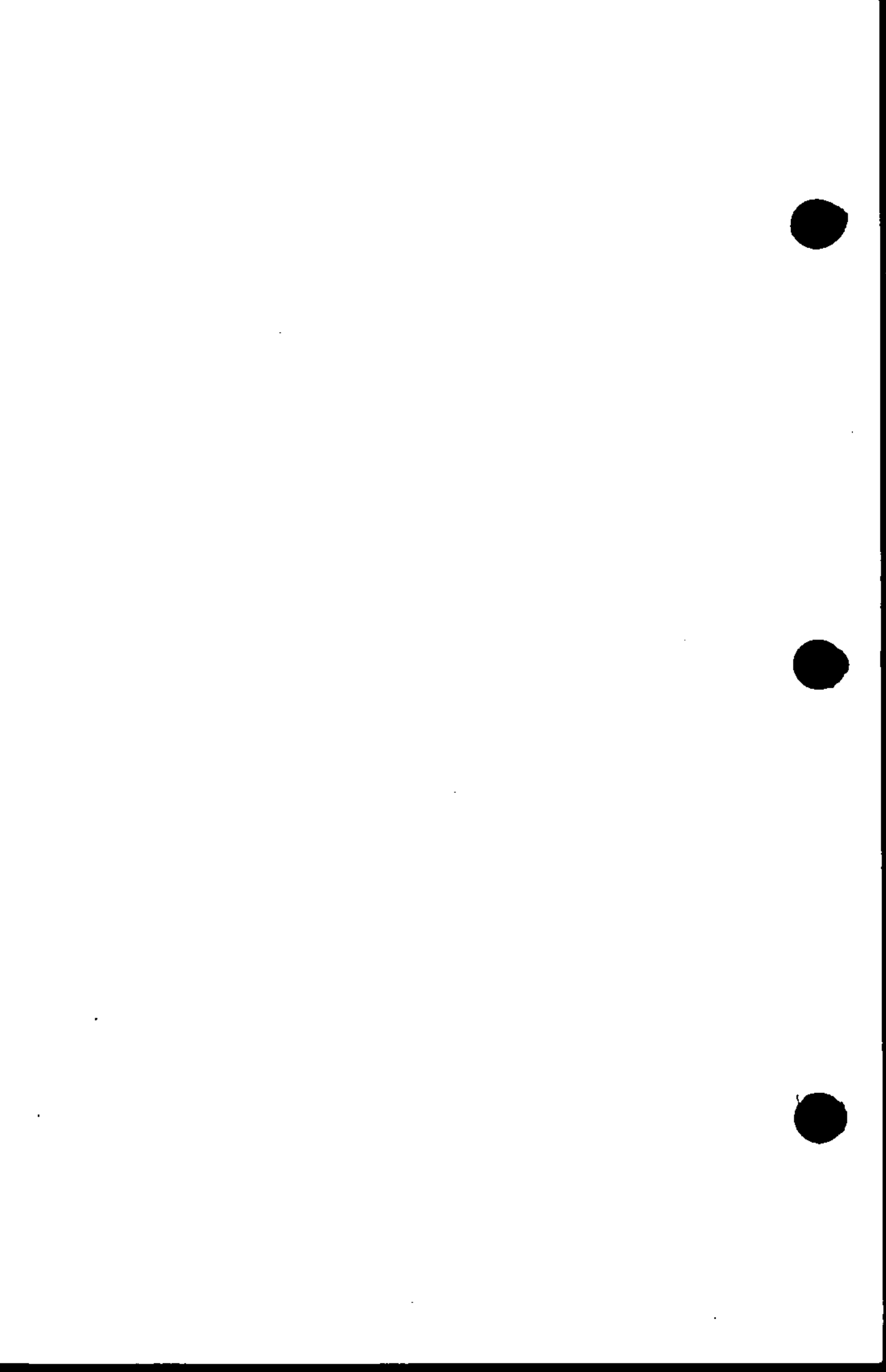
FLUKE 900

DYNAMIC TROUBLESHOOTER

OPERATOR MANUAL

REV. 4.00



FLUKE 900

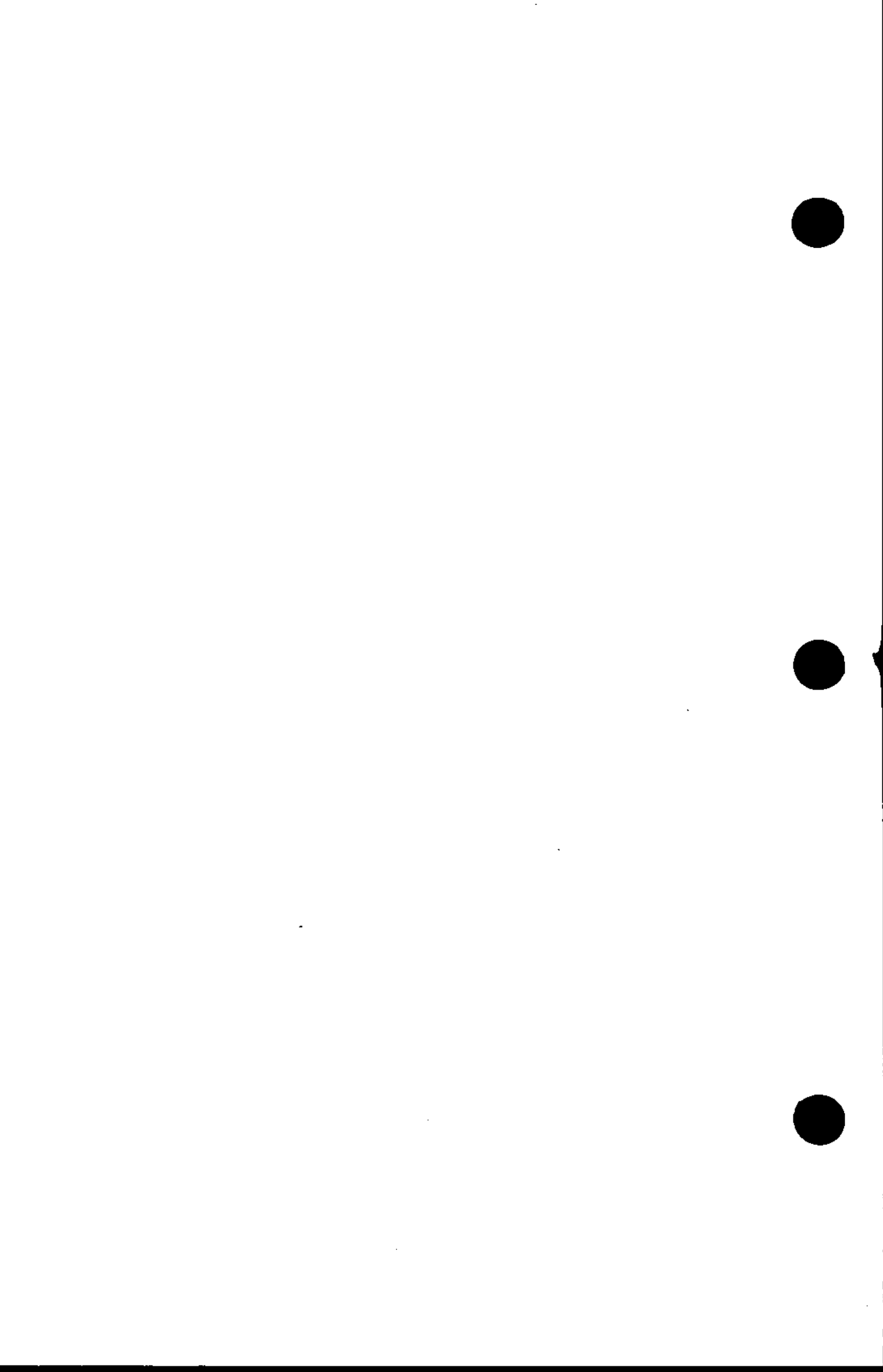
TABLE OF CONTENTS

SECTION DESCRIPTION PAGE

1 FLUKE 900 SYSTEM

1.1 SYSTEM OVERVIEW 1
1.2 SPECIFICATIONS 4
1.2.1 DEVICES SUPPORTED 4
1.2.2 SIGNAL CHARACTERISTICS 5
1.2.3 FAULT CAPTURE 5
1.2.4 TIMING PARAMETERS 6
1.2.5 FREQUENCY MEASUREMENT 6
1.2.6 USER MEMORY 7
1.2.7 INTERFACES 7
1.2.8 TEST CLIPS 9
1.2.9 PHYSICAL AND ELECTRICAL 10

1.3 SYSTEM COMPONENTS AND OPTIONS 11
1.4 INSTALLATION
1.4.1 SHIPPING INFORMATION 13
1.4.2 UNPACKING 13
1.4.3 CONNECTING INTERFACE BUFFER AND POWER 14
1.4.4 LINE FUSE REPLACEMENT 14
1.4.5 CONNECTING TEST CLIPS AND PATCH LEADS 15
1.4.6 POWER UP AND SELFTEST 15
1.5 OPERATOR INTERFACES
1.5.1 SCREEN 18
1.5.2 KEYBOARD 18
1.5.3 MONITOR 20
1.5.4 CARTRIDGE 21
1.5.5 FUNCTION KEY MENU 21

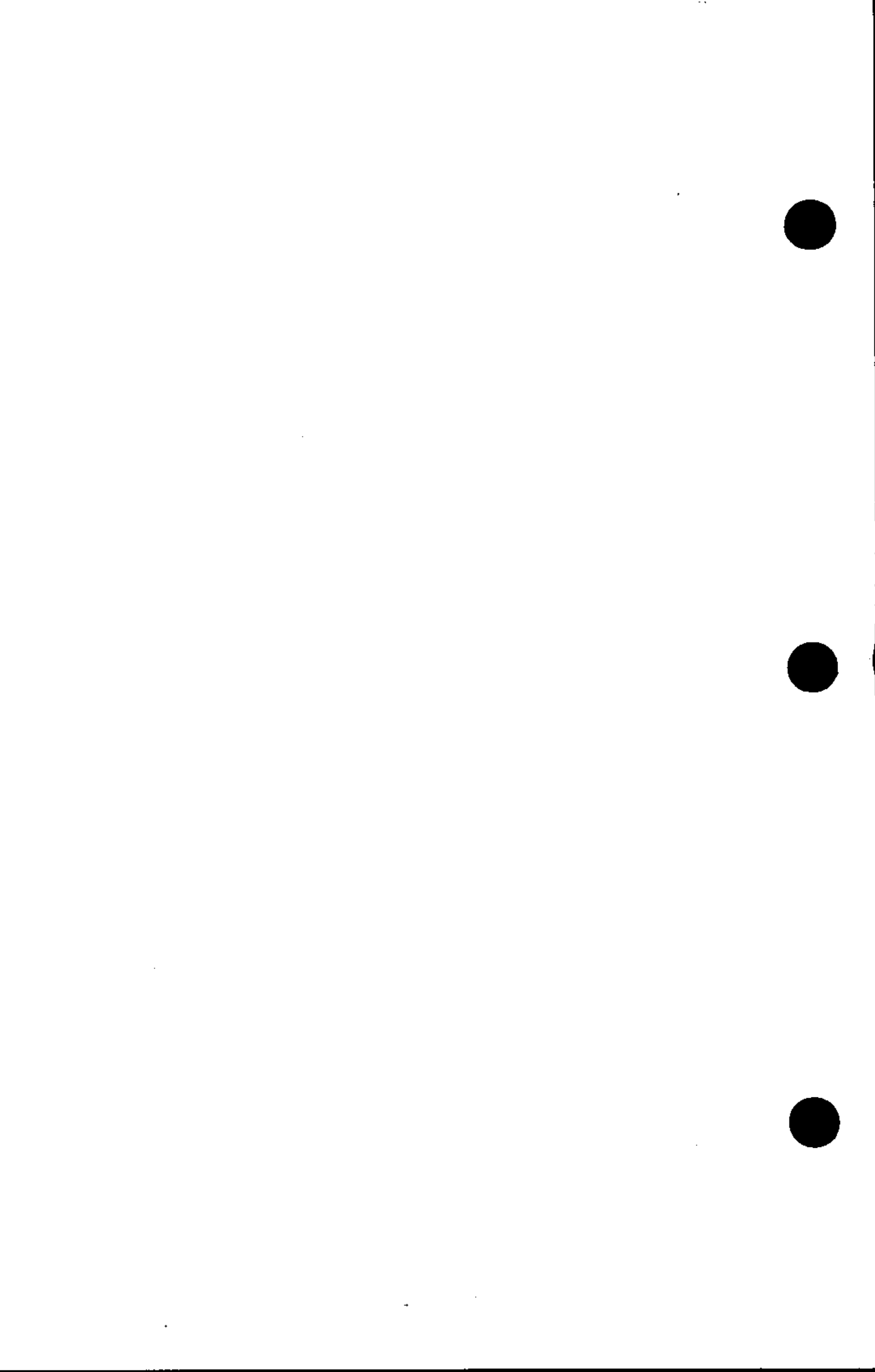


SECTION	DESCRIPTION	PAGE

1.6 OPTION SETTING IN SYSTEM MODE
- 1.6.1 SETTING TIME & DATE 26
- 1.6.2 CONFIGURING THE SERIAL COMMUNICATION PORT 27
- 1.6.3 SETTING THE VOLUME OF THE AUDIBLE TONE 29
- 1.6.4 FORMATTING PRINTER LISTINGS 30
- 1.6.5 DEFINING LOG FORMATS FOR TEST RESULTS 31

2 TECHNICAL PRINCIPLES

- 2.1 TEST METHODOLOGY 1
- 2.2 DYNAMIC REFERENCE COMPARISON 3
  - 2.2.1 DRC - DEVICE LEVEL CONCEPTS 4
  - 2.2.2 DRC - BOARD LEVEL CONCEPTS 7
- 2.3 CONDITION TESTS/OPERATOR CHECKS 8
- 2.4 PROGRAMMING 9
- 2.5 DATA LOGGING 10
- 2.6 TESTING THEORY 11
  - 2.6.1 COMBINATORIAL DEVICES 13
  - 2.6.2 SYNCHRONOUS DEVICES 15
  - 2.6.3 PROGRAMMABLE DEVICES 16
- 2.7 TESTING APPLICATIONS
  - 2.7.1 BOARD LEVEL 21
  - 2.7.2 DEVICE LEVEL 22
  - 2.7.3 APPLICATION ADVICE 26



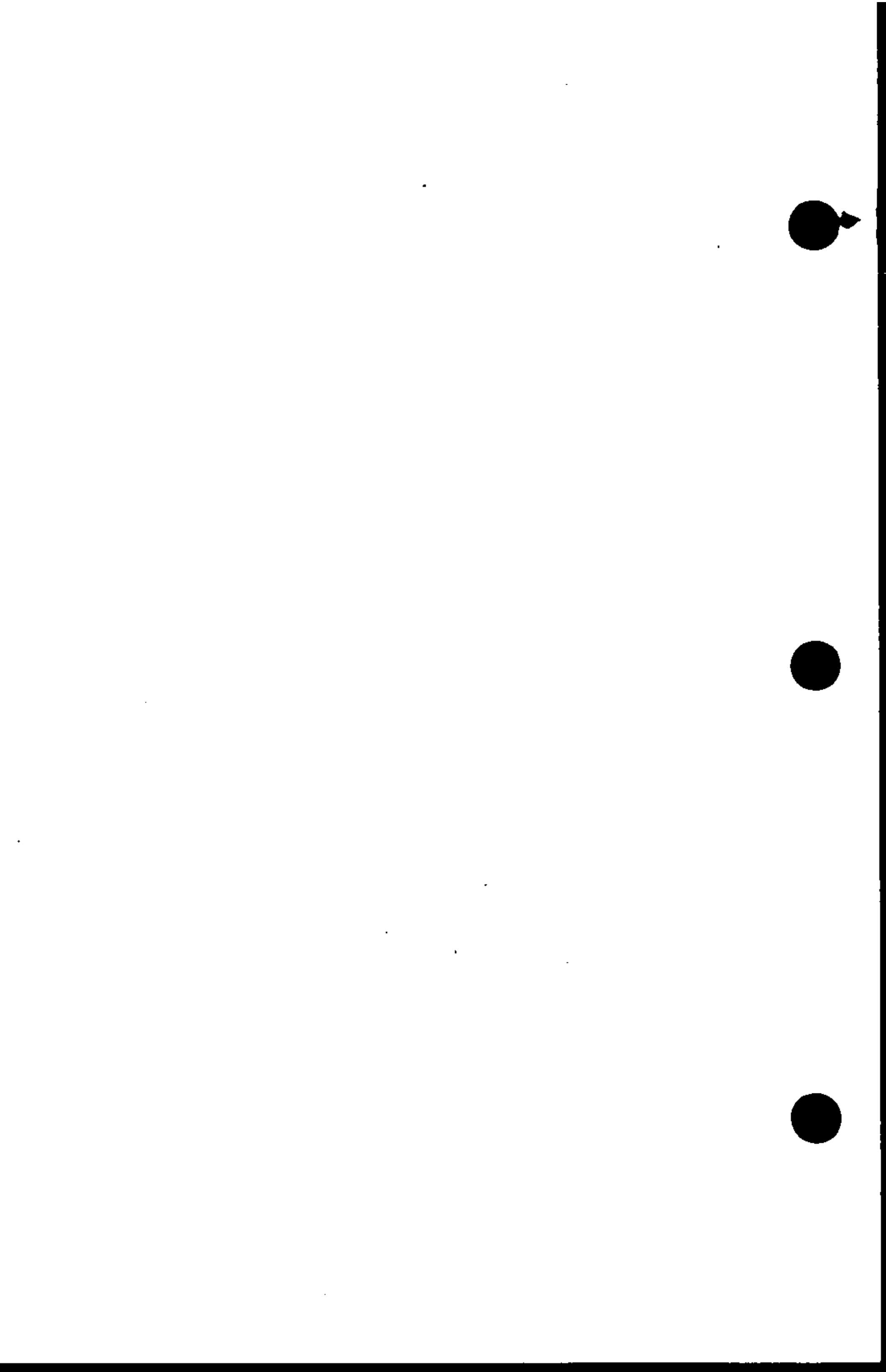
SECTION	DESCRIPTION	PAGE

3 MANUAL MODE
Functionally Testing Devices in their Circuit

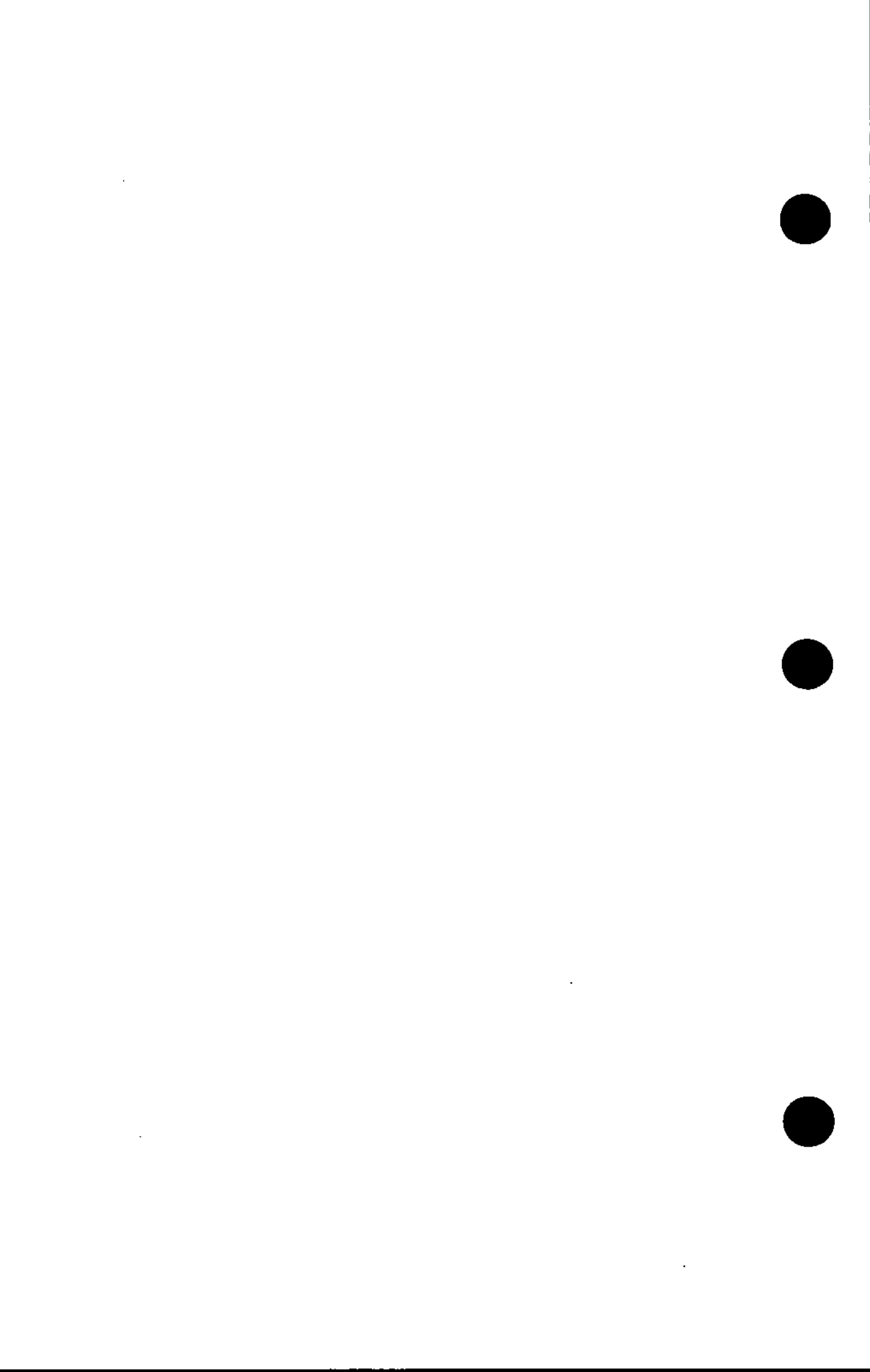
3.1 MAIN MANUAL SCREEN 1
3.2 SETTING PARAMETERS 4
3.2.1 DEFINING SIZE AND POWER PINS ... 5
3.2.2 SPECIFYING LOW RD DRIVE 7
3.2.3 COMPENSATING DUT LOADING 7
3.2.4 SETTING THE DURATION OF TEST ... 8
3.2.5 SETTING SYNCHRONIZATION TIME ... 9
3.2.6 GATING OUT CONDITIONS 9
3.2.7 SETTING LOGIC 1 10
3.2.8 DEFINING OR IGNORING PINS 11
3.2.9 RESETTING THE BOARD 13
3.2.10 TRIGGERING START OF TEST 16
3.3 TEST EXECUTION 17
3.3.1 RD TEST 19
3.3.2 CLIP CHECK 21
3.3.3 TEST RESULTS 22
3.4 FREQUENCY COUNTER 30
3.5 DATA LOGGING 32

4 SEQUENCE MODE
Functional Board Diagnosis

4.1 OVERVIEW 1
4.2 DIRECTORY OF SEQUENCES 3
4.3 RUNNING A SEQUENCE 4
4.4 TEMPORARY PARAMETER CHANGES 6
4.5 RUN SEQUENCE MENU LABELS 7
4.6 SITUATIONS FOR USING LOG FILES 10
4.7 SAMPLE LOG FILE 11
4.8 TEST RESULTS 12



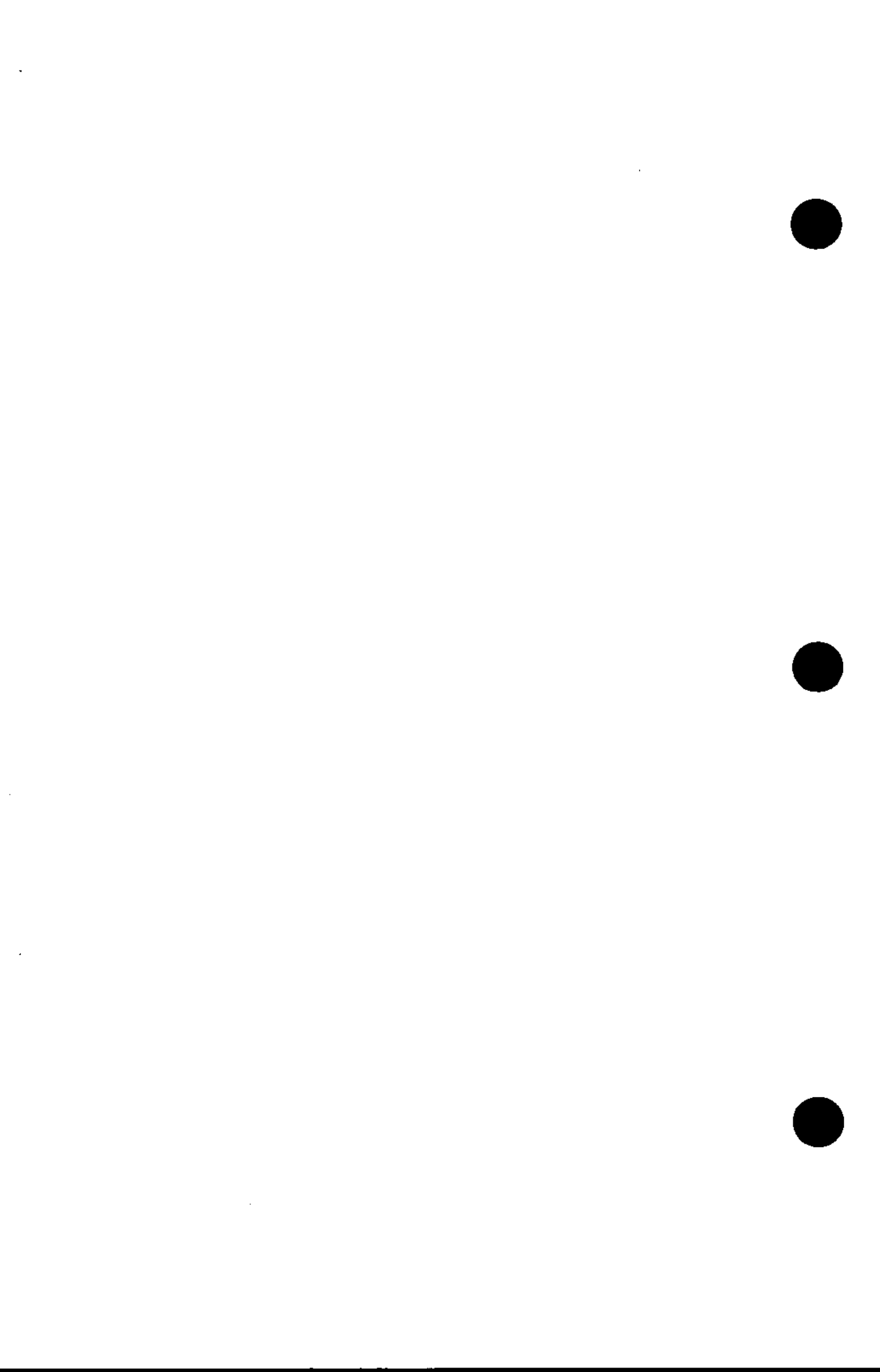
SECTION	DESCRIPTION	PAGE
5	DEVELOP MODE Creating Sequences	
5.1	DEVELOP SUBLEVEL	1
5.2	SEQUENCE DEVELOPMENT OVERVIEW	3
5.3	FILE INTERELATION	4
5.4	NEW SEQUENCE MODE	6
5.5	SEQUENCE ENHANCEMENT	10
5.5.1	SOURCE FILE SYNTAX	10
5.5.2	DISPLAY	12
5.5.3	FUNCTION/END_FUNCTION	12
5.5.4	JUMP	13
5.5.5	IF {condition} THEN {command}	14
5.5.6	SEQUENCE FILE LISTING	15
5.5.7	LOCATION FILE LISTING	16
5.5.8	USER LIBRARY FILE LISTING	17
5.6	SEQUENCE CREATION GUIDELINES	
5.6.1	SEQUENCING PREPARATION	18
5.6.2	DIAGNOSTIC STIMULUS	19
5.6.3	VERIFYING THAT DEVICES PASS	19
5.6.4	ADDING SIGNAL CONDITION TESTS	20
5.6.5	STRUCTURING THE SEQUENCE	21
5.6.6	ADDING OPERATOR MESSAGES	22
5.6.7	COMPILING THE FILES	22
5.6.8	FINAL VERIFICATION AND DOCUMENTATION	22
6	FILE UTILITIES	
6.1	CARTRIDGE <<cartridge>>	2
6.2	COMPILE <<compile>>	4
6.3	COPY <<copy>>	6
6.4	DELETE <<delete>>	8
6.5	DIRECTORY <<dir>>	10
6.6	DOWNLOAD <<dnload>>	13
6.7	EDIT <<edit>>	15
6.8	PRINT <<print>>	19
6.9	RECOVER <<recover>>	21
6.10	UPLOAD <<upload>>	23



SECTION	DESCRIPTION	PAGE

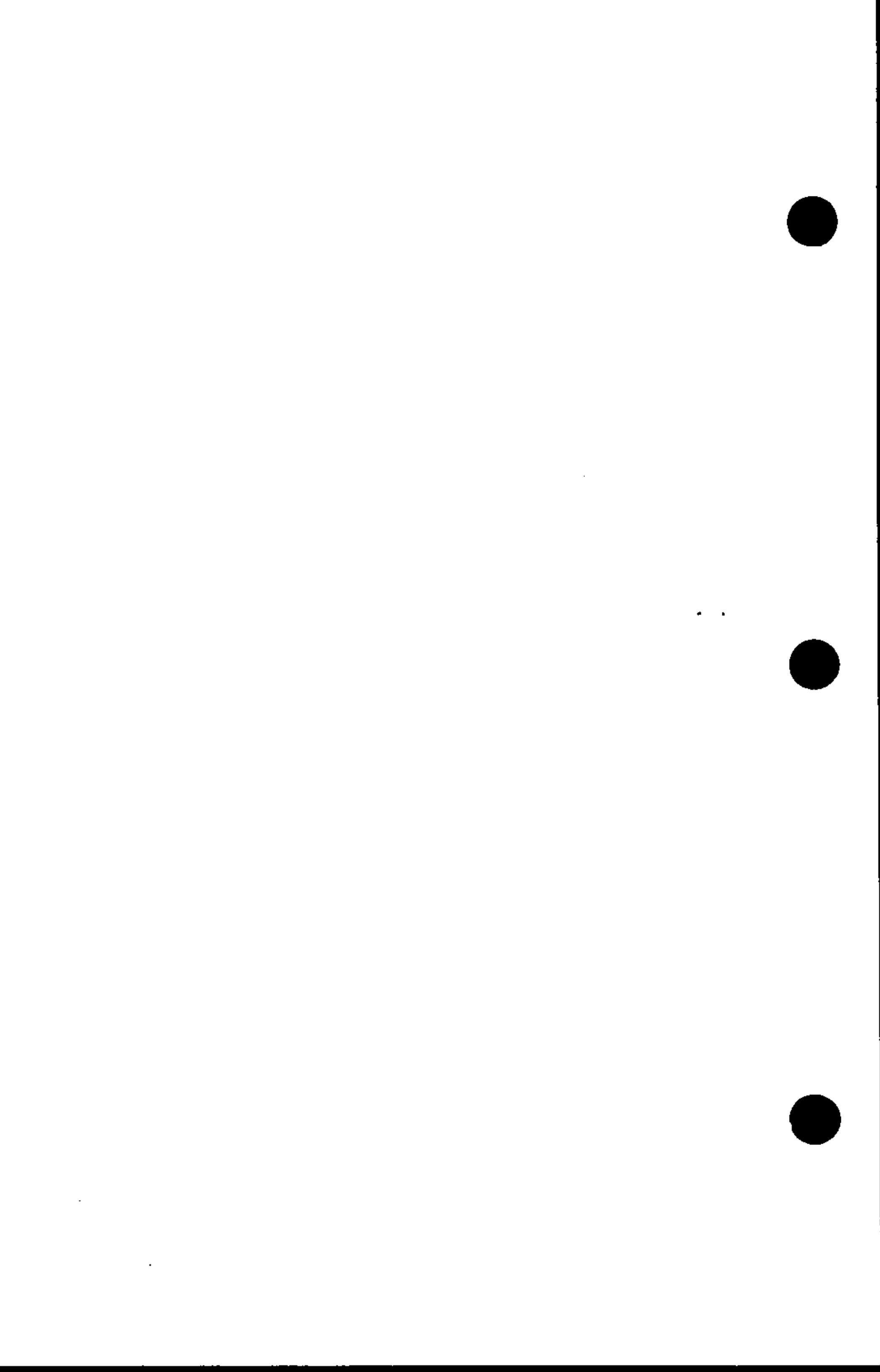
7 DEVICE LIBRARIES

7.1 LIBRARY FORMAT 2
7.2 SPECIFYING RD/DUT SYNCHRONIZATION 4
7.3 RD TEST 11
7.3.1 IDENTIFY 12
7.3.2 C_SUM 13
7.3.3 SPECIFYING RD TEST VECTORS 14
7.3.4 VERIFYING AND PRINTING RD TEST VECTORS 17
7.4 OUT OF CIRCUIT DEVICE TESTING 18

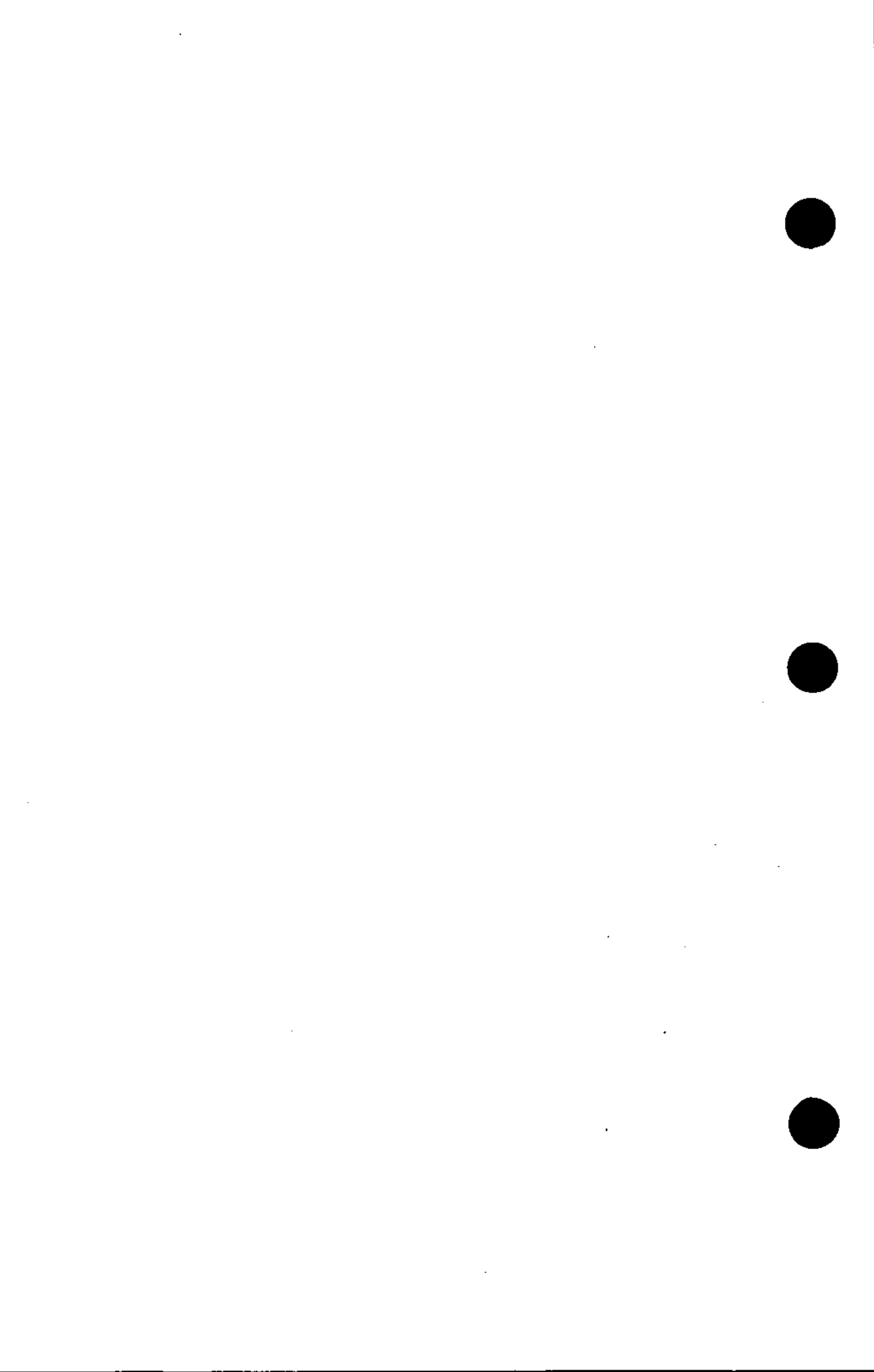


SECTION	DESCRIPTION	PAGE
APPENDIX I	FUNCTION KEY INDEX	
APPENDIX II	COMMAND LANGUAGE	

ACTIVITY 5
CLIP_CHK 7
COMMENT 9
COMPARE 11
C_SUM 13
DISPLAY 15
END 17
END_FUNCTION 19
F_MASK 21
FUNCTION 23
GATE 25
IF...THEN... 27
IGNORE 29
JUMP 31
LOAD 33
LOC_FILE 35
NAME 37
RD_DRV 39
RDT_ENABLE 41
RDTEST 43
RESET 45
SIZE 47
SOUND 49
S_TIME 51
SYNC_COND 53
SYNC_IGNORE 55
SYNC_GATE 57
SYNC_GR_END 59
SYNC_PAT 61
SYNC_PINS 63
SYNC_RESET_OFF 65
SYNC_RND 67
SYNC_VECT 69
TEST 71
THRSLD 73
TRIGGER 75
T_TIME 77



SECTION	DESCRIPTION
APPENDIX III	APPLICATION NOTES
APPENDIX IV	REMOTE CONTROL PROTOCOL
APPENDIX V	LIBRARY LISTING



HOW TO USE THIS MANUAL

This manual may be read in part, for the user who wishes to quickly gain enough proficiency to begin testing, or completely for the user who wishes to create automatic test sequences.

Section 1 contains detailed specifications and ordering information, installation instructions and a description of components and settable options.

Section 2 describes the test approach and provides the technical foundation for understanding all the operating modes.

Section 3 covers Manual Mode and describes all testing features as they apply to any single device, in or out of circuit.

Section 4 describes Sequence Mode and how to test boards from a stored sequence.

Section 5 describes Develop Mode and how to create sequences.

Section 6 covers file utilities for the manipulation of test sequence and data files.

Section 7 describes how to add new devices to the Out-of-Circuit and Reference Comparison test databases.

HOW TO USE THIS MANUAL

Appendices include an index of Function Keys, a specification of the Sequence Command Language, Testing Application Notes, Remote Control Protocol and a listing of the Standard Reference Device Library.

All users should read Section 1 for machine overview and installation procedures. Those who are merely testing with preprogrammed sequences and those developing sequences should read Section 2 on technical principles. The test sequence user will find Section 4 on Sequence Mode adequate for his needs. Advanced users should read Section 3 on Manual Mode for testing without a sequence. Sections 5 and 6 are appropriate for those developing sequences. Section 7 on Device Libraries is also for the advanced programming user.

When describing keys, the convention used is single brackets < > for labelled keys and double brackets << >> for Function Keys. Example: <ESC>, <<manual>>.

WARRANTY

Notwithstanding any provision of any agreement the following warranty is exclusive:

The JOHN FLUKE MFG. CO., INC. warrants each instrument it manufactures to be free from defects in material and workmanship under normal use and service for the period of 1 year from date of purchase. This warranty extends only to the original purchaser. This warranty shall not apply to fuses, test clips (clips are warranted for 90 days), or any product or parts which have been subject to misuse, neglect, accident, or abnormal conditions of operations.

In the event of failure of a product covered by this warranty, John Fluke Mfg. Co., Inc. will repair and calibrate an instrument returned to an authorized Service Facility within 1 year of the original purchase; provided the warrantor's examination discloses to its satisfaction that the product was defective. The warrantor may, at its option, replace the product in lieu of repair. With regard to any instrument returned within 1 year of the original purchase, said repairs or replacement will be made without charge. If the failure has been caused by misuse, neglect, accident, or abnormal conditions of operations, repairs will be billed at a nominal cost. In such case, an estimate will be submitted before work is started if requested.

THE FOREGOING WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS, OR ADEQUACY FOR ANY PARTICULAR PURPOSE OR USE. JOHN FLUKE MFG. CO., INC. SHALL NOT BE LIABLE FOR ANY SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN CONTRACT, TORT OR OTHERWISE.

If any failure occurs, the following steps should be taken:

1. Notify the JOHN FLUKE MFG. CO., INC. or nearest Service Facility, giving full details of the difficulty, and include the model number, type number, and serial number. On receipt of this information, service data, or shipping instructions will be forwarded to you.

2. On receipt of the shipping instructions, forward the instrument, transportation prepaid. Repairs will be made at the Service Facility and the instrument returned, transportation prepaid.

SHIPPING TO MANUFACTURER FOR REPAIR OR ADJUSTMENT

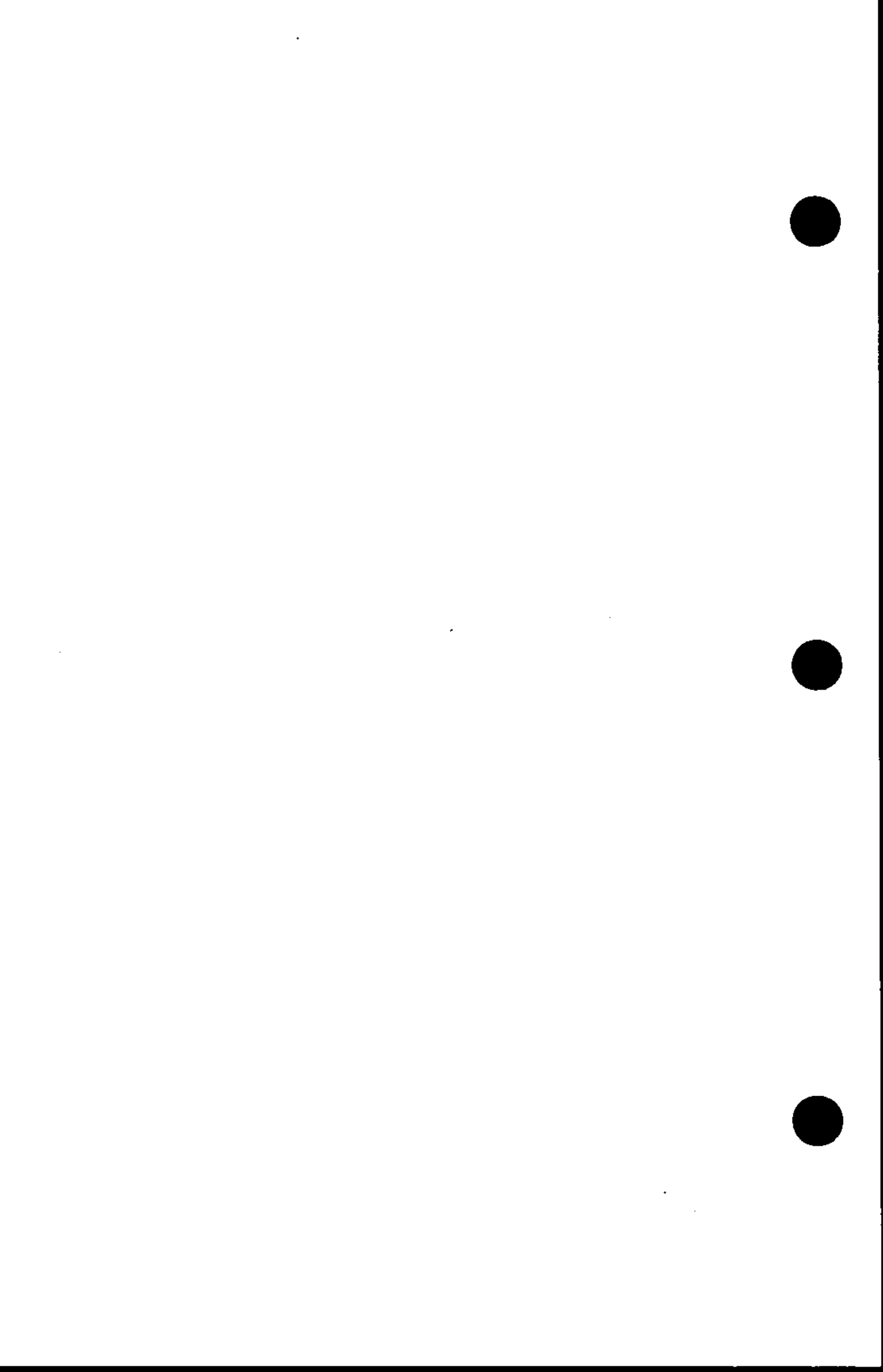
All shipments of JOHN FLUKE MFG. CO., INC. instruments should be made via United Parcel Service or "Best Way" prepaid. The instrument should be shipped in the original packing carton; or if it is not available, use any suitable container that is rigid and of adequate size. If a substitute container is used, the instrument should be wrapped in paper and surrounded with at least four inches of excelsior or similar shock-absorbing material.

CLAIM FOR DAMAGE IN SHIPMENT TO ORIGINAL PURCHASER

The instrument should be thoroughly inspected immediately upon original delivery to purchaser. All material in the container should be checked against the enclosed packing list. The manufacturer will not be responsible for shortages against the packing sheet unless notified immediately. If the instrument is damaged in any way, a claim should be filed with the carrier immediately. (To obtain a quotation to repair shipment damage, contact the nearest Fluke Technical Center.) Final claim and negotiations with the carrier must be completed by the customer.

The JOHN FLUKE MFG. CO., INC. will be happy to answer all applications or use questions, which will enhance your use of this instrument. Please address your requests or correspondence to: JOHN FLUKE MFG. CO., INC., P.O. BOX C9090, EVERETT, WASHINGTON 98206.

John Fluke Mfg. Co., Inc., P.O. Box C9090,
Everett, Washington 98206



1 FLUKE 900 SYSTEM

1.1 SYSTEM OVERVIEW

The FLUKE 900 dynamic troubleshooter uses Dynamic Reference Comparison (DRC) to compare the Real Time functional performance of digital devices as they operation In-Circuit, with known good Reference Devices running in parallel and synchronism.

The real-time performance of the Device Under Test (DUT) is compared to that of a known good Reference Device (RD), inserted into a Zero Insertion Force (ZIF) socket on the face of the tester. The RD is fed with the same signals extracted from the DUT, via a test clip which plugs into the FLUKE 900 Interface Buffer (IB).

The IB has a programmable voltage threshold which can be set to see logic 1 at any level between 0 to 5 volts, with a resolution of 100 mV. The IB translates the incoming signals into ECL differential levels which are transformed back to TTL signals in the main unit. This provides a high level of accuracy in signal reproduction so that the RD in the ZIF socket sees similar inputs to the DUT.

The proprietary DIFTEST circuits used in the FLUKE 900 enable it to test standard and custom devices without the need to know the internal structure or pinouts of a DUT. The system ensures that the RD and the DUT are synchronized prior to comparing them. The FLUKE 900 monitors and analyzes the behaviour of the Device Under Test within a Performance Envelope (PE). It is user

(PE). It is user definable in the time domain with 10 ns resolution and in voltage within 100 mV. The PE parameters can be compared to the time base and volts/division settings of an oscilloscope.

The PE is used to compensate for the difference in loading between the DUT and RD. If the DUT response time exceeds the PE setting of Fault Mask, comparison testing is halted and failure information is displayed on the LCD display.

Other features which make the FLUKE 900 a powerful troubleshooting tool are:

1. Built in frequency counter that can measure signal activity on the pins of the DUT plus an external node. This is useful for checking out clock frequencies and other repeatable signals on the board.

2. LED logic monitor indicating signal activity on DUT pins. Each pin can be predefined for an expected high, low or active signal.

3. A two word user-definable trigger function, which can be a combination of the signals on the DUT pins plus an external node. This aids the user in isolating the fault area, similar to what a logic analyzer might do.

The user stores information about each device on a removable data cartridge including location on the board, changes in default PE, triggers, frequency and activity checks. This "SEQUENCE" file can contain other information like

operator prompts which appear as comments on the LCD. A sequence provides a high quality repeatable test procedure which will compliment the knowledge of the operator.

Board test results can be stored in a log file which stamps it with time and date information, providing data on repair stations and for statistical analysis of common failure modes. This information can also be communicated to other systems via the RS232C serial port.

Out-of-circuit device verification may be done from a library pattern stored in ROM for most TTL devices. This can be expanded by the user and stored on cartridge. In addition, the system will identify an unknown or unmarked device if it matches one in the library.

1.2 SPECIFICATIONS

1.2.1 DEVICES SUPPORTED

(a) Reference Devices must be digital IC's with TTL compatible inputs and outputs, up to 28 DIP pins.

(b) SSI to VLSI devices with one LS load drive capability. For weaker devices, with as little as one third this drive capability, the keyboard selectable "RD Drive Low" option can be chosen to enable comparison testing. It should be noted that this reduces the maximum signal specification by 40 percent to 12 MHz.

(c) Testable devices include the following technologies:

N, LS, S, ALS, AS, F, HC, HCT, ACT, FACT

Other technologies that do not meet the RD specification defined in (a) and (b) (e.g. HC, 74Cxx series), are testable if they have TTL functionally compatible equivalents and if device timing variation falls within the FMASK range.

(d) Custom Components: PAL, HAL, PLA, FPLA and Gate Arrays.

(e) Memories: Static and Dynamic RAM, ROM, PROM, EPROM, FIFO and LIFO.

(f) RD is supplied with up to 400 mA at 5 volts. RD's with standard power pins and a majority of non-standard configurations are supported.

(g) S0 version surface mounted devices may be compared via the appropriate clip to an equivalent DIP version RD.

1.2.2 SIGNAL CHARACTERISTICS

(a) Maximum data rate on any input, output or I/O pin is 20 MHz with a minimum pulse width of 20 ns.

(b) The input circuit in the Interface Buffer is fully protected in the -25 V to +25 V range. Signals exceeding those levels may cause damage to the Interface Buffer.

(c) The input circuit has a programmable threshold which can be set in 100 mV increments in the range of 0 to 5 V. Setting accuracy is +/- 100 mV.

(d) The threshold is a single level above which signals are seen as "Logic 1", and below it as "Logic 0".

1.2.3 FAULT CAPTURE

(a) FLUKE 900 has a 10 ns fault capture resolution on full frequency range (DC to 20 Mhz).

(b) Fault capture window is programmable in 10 ns increments in the range of 20 to 200 ns.

(c) Non-multiplexed fault circuit ensures the capture of static, dynamic (timing-related) and intermittent faults.

1.2.4 TIMING PARAMETERS

(a) Test duration is programmable from 1 msec to 9999 ms in 1 ms steps, or continuous.

(b) Synchronization time is programmable from 10 to 9990 ms in 10 ms steps or it can be turned OFF.

(c) Reset Duration is programmable from 10 to 32767 ms in 1 ms steps. Offset is programmable from 0 to +/- 32767 in 1 ms steps. Offset is the time between the trailing edge of the Reset pulse and the comparison interval.

1.2.5 FREQUENCY MEASUREMENT

(a) Guaranteed frequency range of DC to 20 MHz with an accuracy better than 0.5%. Minimum pulse width is 20 ns.

(b) Autoranging measurement is performed on any and all pins plus an external lead.

(c) Repeated averaged measurements of period, time high and time low provides timing accuracy in the nanosecond range.

1.2.6 USER MEMORY

(a) Volatile system RAM for user files is 48K in size.

(b) Non-volatile RAM for user files resides on cartridges of 32K size.

1.2.7 INTERFACES

(a) Plug-in cartridge with 32K non-volatile RAM can be used to store data such as sequences, log files and RD test patterns. The Write Protect feature prevents accidental data erasure.

(b) Twenty-eight pin LED monitor displays the activity on the DUT during the test and the failed pins at the end of test.

(c) Eight lines by forty characters graphic LCD display is used to present system information.

(d) Operator input into the system is done through a membrane keyboard featuring soft keys for easy menu selection and ASCII keyboard for data entry.

(e) The serial communication port (RS232C) is a standard feature of the system. It is programmable to operate in the following modes:

Pin Assignments	DTE or DCE (Data Terminal or Data Communication Equipment)
Baud Rate	300,600,1200,2400,4800,9600 or 19200
Character Length	5, 6, 7 or 8 bits
Stop Bits	1, 1.5 or 2 bits
Parity	Even, odd or none.
Timeout	1, 2, 5, 10, 30 seconds 1, 10 minutes

(f) The patch leads on the Interface Buffer have the following characteristics:

GND - two ground reference connections for the board under test

EXT - a 29th input channel in addition to the 28 from the test clip with an impedance of 10 Kohm.

RST - a normally tristated reset signal. For a negative polarity pulse the lead is driven high for 10 ms, low for the pulse duration and back high for 10 ms. The reverse occurs for a positive pulse. Internal drive will sink/source 50 mA at 0.8/4.2 V. External drive will sink/source at 0.8/VCC.

VCC - a voltage from 2V to 15V used to supply external drive for RST.

1.2.8 TEST CLIPS

(a) The standard test clips provided are universal and come in 3 different sizes:

16 Pin - 0.3" spacing DIP Clip
24 Pin - 0.3" spacing DIP Clip
28 Pin - 0.6" spacing DIP Clip

The 16 pin clip can be used on any 0.3" DIP package with 16 pins or less. The 24 pin clip can be used on any 0.3" DIP package with 24 pins or less, and the 28 pin clip can be used on any 0.6" DIP package with 28 pins or less.

(b) Optional DIP test clips with 8, 14, 18, 20, 22 (0.4"), 24 (0.6") pins are available if, due to PCB density, clipping with the larger standard clips proves to be a problem.

(c) Optional SMT test clips are available in 0.15" width for 8, 14 and 16 pin devices, and in 0.3" width for 16, 18, 20, 24 and 28 pin devices.

(d) Test clip loading is 10 Kohm @ 30 pf per pin.

(e) Automatic clip check verifies proper clip size and orientation at the start of each test cycle. Improper clipping or failure to detect the correct power on the DUT will result in an error and the termination of the test.

Clip check also verifies proper clipping during test.

1.2.9 PHYSICAL AND ELECTRICAL

Power 115/230 VAC, 47-63 HZ @ 100 watts

Weight 12 lbs. (approx. 5.44 Kg)

Dimensions

Main Unit	12 x 15 x 3.5 inches 30.5 x 38.1 x 8.9 cm
Interface Buffer	8 x 8 x 0.75 inches 20.3 x 20.3 x 1.9 cm
Cable Length	48 inches 122 cm

1.3 SYSTEM COMPONENTS AND OPTIONS

PART NUMBER	DESCRIPTION
900	FLUKE 900 Main Unit including: -Interface Buffer (IB) -Test Clips: 16 pin 0.3" (Y900-16D) 24 pin 0.3" (Y900-24D) 28 pin 0.6" (Y900-28D) -One Data Cartridge (Y900-001) -One Set of Patch Cords (Y900-005) -RD Tray and Cartridge Holder (Y900-003) -Operator Manual -Training Certificate -Power Cord
Y900-001	Data Cartridge
Y900-002	Set of Three Data Cartridges (Y900-001)including Cartridge Tray (Y900-004)
Y900-003	Reference Device Tray & Data Cartridge Holder
Y900-004	Cartridge Holder (for 3 cartridges)
Y900-005	Set of Five (5) Patch Leads and Clips
Y900-006	Model 900 Carrying/Shipping Case
Y900-007	Model 900 Operator Manual
Y900-008	Model 900 Training Manual

PART NUMBER	DESCRIPTION
DIP CLIPS	
Y900-08D	8 Pin Dip Test Clip (0.3")
Y900-14D	14 Pin Dip Test Clip (0.3")
Y900-16D	16 Pin Dip Test Clip (0.3")
Y900-18D	18 Pin Dip Test Clip (0.3")
Y900-20D	20 Pin Dip Test Clip (0.3")
Y900-22D	22 Pin Dip Test Clip (0.4")
Y900-24D	24 Pin Dip Test Clip (0.3")
Y900-24DW	24 Pin Dip Test Clip (0.6")
Y900-28D	28 Pin Dip Test Clip (0.6")

S.M.D. CLIPS

Y900-08S	8 Pin Surface Mount T.C.(0.15")
Y900-14S	14 Pin Surface Mount T.C.(0.15")
Y900-16S	16 Pin Surface Mount T.C.(0.15")
Y900-16SW	16 Pin Surface Mount T.C.(0.3")
Y900-18S	18 Pin Surface Mount T.C.(0.3")
Y900-20S	20 Pin Surface Mount T.C.(0.3")
Y900-24S	24 Pin Surface Mount T.C.(0.3")
Y900-28S	28 Pin Surface Mount T.C.(0.3")

1.4 INSTALLATION

This Section contains information on unpacking, connecting and turning on the FLUKE 900. It describes the main features for user interaction and details the setting of options in System mode.

1.4.1 SHIPPING INFORMATION

The FLUKE 900 is shipped in a rugged foam-packed carton. When you receive the unit, inspect it thoroughly for possible shipping damage. When reshipping the FLUKE 900, it is strongly advised to use the original packing carton. Replacement cartons are available from the factory or through your local sales office.

1.4.2 UNPACKING

Inside the carton are several packages held in place by foam retainers:

- cardboard box containing the Interface Buffer
- linen covered box containing User's Manual and Reference Device Tray with one Data Cartridge
- FLUKE 900
- small accessories including power cord, test clips and patch leads.

The User's Manual Box and cardboard box are first removed from the top of the foam retainer. The entire retainer may then be lifted out of the carton and the interlocking foam pieces pulled apart. It is advisable to keep the packing materials to use again for any future shipping requirements.

1.4.3 CONNECTING INTERFACE BUFFER AND POWER

The Interface Buffer plugs into the right side of the main unit through the two ribbon cable connectors labelled J1 and J2. Care should be taken to follow three rules:

1. IB must be installed with power off.
2. Both J1 and J2 must be plugged in.
3. J1 and J2 must be installed correctly as labelled and not reversed.

The power cable plugs into a receptacle at the rear left of the main unit.

1.4.4 LINE FUSE REPLACEMENT

The line fuse is located in a recessed compartment next to the power cord receptacle. To replace it, remove the power cord and slide the clear plastic panel over, exposing the fuse receptacle. Pull the black plastic lever to pop out the fuse. The proper rating for replacement fuses is 3A, 250 V.

1.4.5 CONNECTING TEST CLIPS AND PATCH LEADS

Clips and leads may be inserted into the IB under power since it is fully protected to + or - 25 V. Patch leads are best connected to an unpowered system under test to avoid accidental damage to the board under test. Clipping on IC devices with test clips is normally done with power to the board under test. The clips are inserted into the IB with their logo facing up.

1.4.6 POWER UP AND SELFTEST

To turn the unit on after connecting the IB and AC power cord, press the toggle switch located on the rear left side. The front panel LED will illuminate and the screen will display a series of selftest messages beginning with a memory test. An extensive selftest and calibration is performed for approximately a minute. A timing verification is made in the signal path from the IB to the reference device ZIF socket on the front of the unit. In normal operation the four most common causes of apparent selftest failure are:

1. RD inserted in ZIF socket.
2. Test clip attached to board under test.
3. EXT lead attached to board under test.
4. Key depressed on IB.

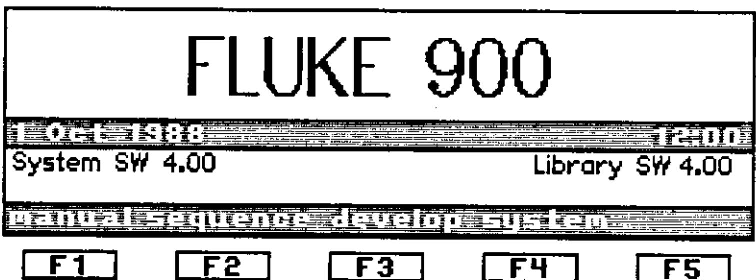
If selftest fails, first check the possible causes listed above. At this point, <F1> may be pressed to re-execute the selftest or <F2> to ignore a failed result and continue with file handling operations. Actual testing will not be allowed unless selftest passes.

If a true hardware fault is suspected, press <F3> to enter debug mode. A printer may now be connected to the serial port (see Section 1.6.2 for port option setting). Press <<prt_res>> to print out a diagnostic listing similar to the following:

```
FLUKE 900 - Selftest results

Software version:  4.00
Library version:   4.00

HSB rev:  1

MB rev:   0

IB rev:   0

General results:
          X0000 00000 00000 00000 00000  01000
          00000 00000 00000 01X0X X1XXX XXXXX
Flags:
          00000000 00000000 00000000 00000000
          00000000 00000000 00000000 00000000

Individual results:
   26     00000000 10000000 00000000 00000000
   46     00000000 01000000 00000000 00000000
   51     00001000 00000000 00000000 00000000
```

This will assist the determination of the exact problem when communicating with authorized service personnel.

The main screen after power up appears as:



Power-up Screen

The FLUKE 900 software revision levels are indicated on this screen. "System" refers to the ROM-based operating system that provides the user features. "Library" refers to the ROM-based device library resident for recall by generic device number. Note that this is a base library, to which the user may easily add further or even unknown library devices.

Contact your nearest sales office to obtain the latest revision in ROM software.

1.5 OPERATOR INTERFACES

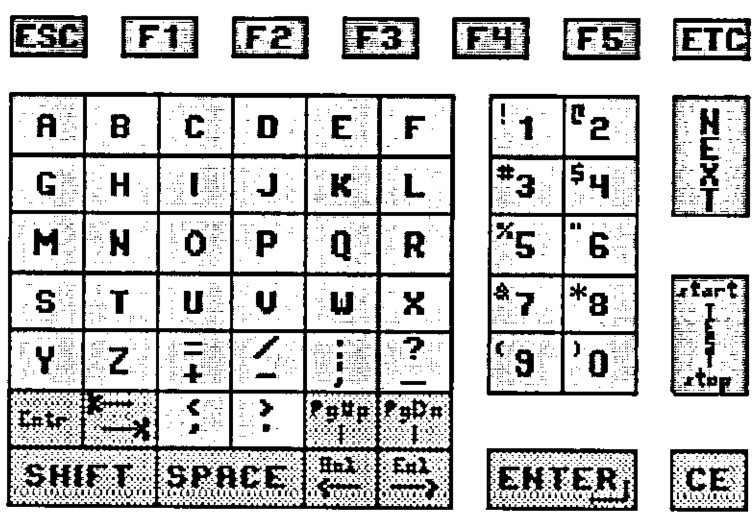
1.5.1 SCREEN

The high resolution supertwist LCD screen displays 8 lines of 40 characters which are divided into several groups:

- Information Area for test results and parameters.
- Status Line showing the current mode of the tester and providing prompts for user entries. Additionally, the time and date are indicated on the right side.
- Command Lines for data entry. In most modes, information is typed in this area and perhaps changed before being transferred to the Information Area upon pressing <ENTER>.
- Function Key labels which apply to keys F1 through F5.

1.5.2 KEYBOARD

The membrane keyboard shown below provides audible tone feedback when pressed. When describing keys, the convention used is single brackets <> for labelled keys and double brackets <<>> for soft Function Keys.



Keyboard Layout

Alphanumeric Keys

Used for data entry, they also print lower case using the Shift key. After pressing for two seconds, the keys repeat.

<CE> (Clear Entry) performs a backspace function to delete the last character. Shift <CE> deletes the last word. Cntr <CE> deletes the complete entry. <ENTER> terminates a line of data. As you will see, it typically puts user inputs into the fields of the main display.

Cursor Control Keys

Movement by space, tab field, page and line
(BOL = beginning of line, EOL = End of line).
(SHIFT up arrow = Page Up, Cntr up arrow = top of file)
(Down arrow, SHIFT and Cntr = Page down, bottom of file)

Test, Next Keys

These are used in manual and sequence modes to test and step from device to device. Duplicate keys are located on the Interface Buffer. The TEST key is alternate action start and stop.

Function Keys

Labels for F1 through F5 are found on the bottom line of the display. <ESC> returns control to a previous key level. <ETC> displays additional labels that are present on a current key level. When there is a second level, "-etc-" appears on the right of the command line.

1.5.3 MONITOR

Two rows of LED's beside the ZIF (zero insertion force) socket act as a logic monitor for signal activity on the test device. In addition, when a fault is captured, flashing LED's will indicate the offending pin(s). The pin numbers for the right side of the socket are arranged in columns according to the various device sizes. The appropriate column is indicated by an illuminated LED.

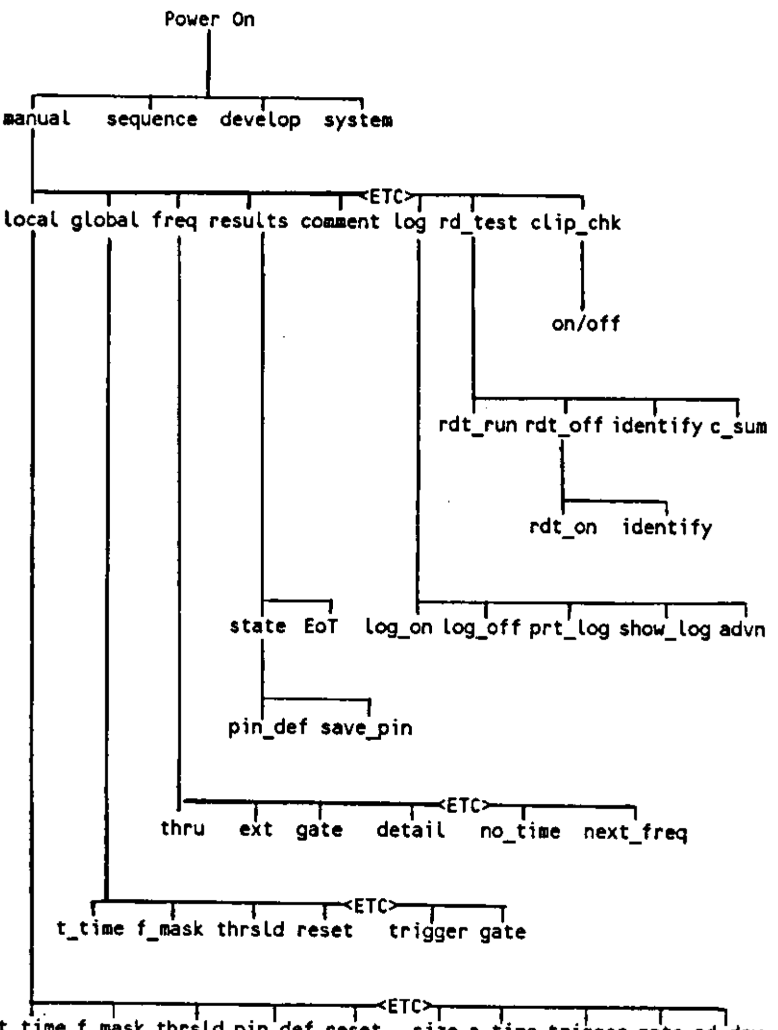
1.5.4 CARTRIDGE

A data cartridge that typically contains preprogrammed test information inserts into the slot on the rear right side of the unit. Insertion and removal with the label facing upward may be done safely with the FLUKE 900 powered up. Note that file creation and writing to the cartridge should be complete before removal to avoid garbled data. Cartridges contain 32K of nonvolatile RAM storage guaranteed for 10 years of data retention. When the write protect switch is put on, cartridge data is safe from accidental erasure.

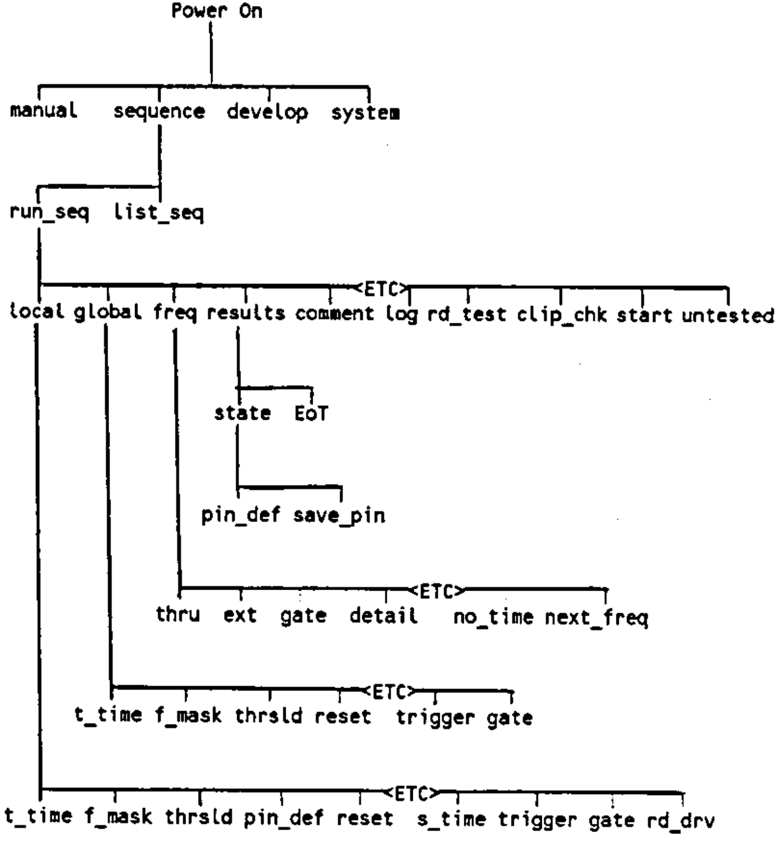
1.5.5 FUNCTION KEY MENU

The labels of the five function keys appear on the bottom line of the display. They change according to a tree structure of menu levels summarized below. Note that <ESC> is used to step back up the menu tree towards the main power up screen and <ETC> selects between two sets of labels on the same menu level.

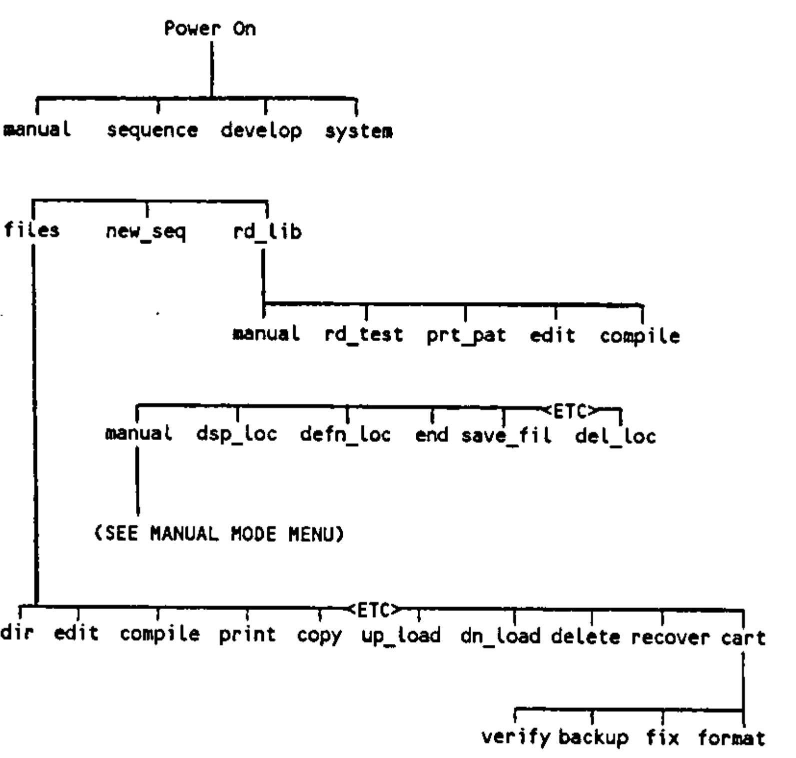
MANUAL MODE MENU



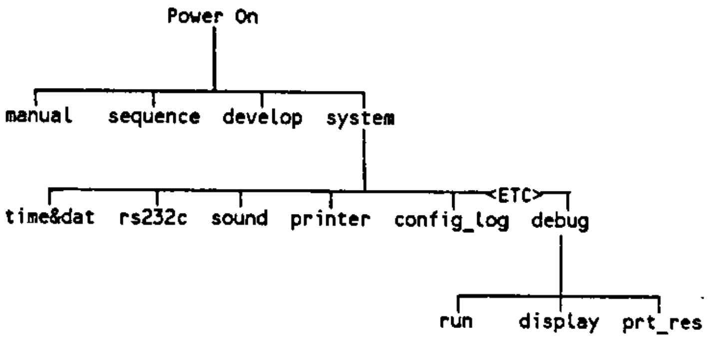
SEQUENCE MODE MENU



DEVELOP MODE MENU

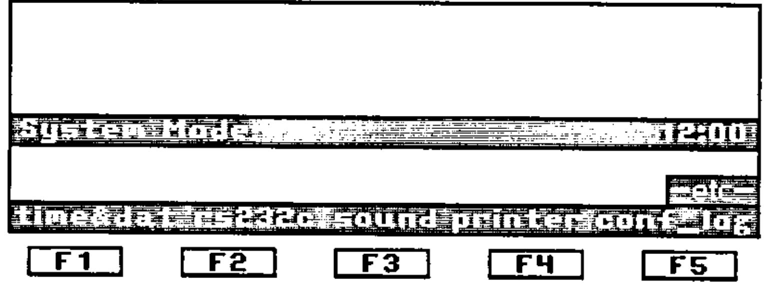


SYSTEM MODE MENU



1.6 OPTION SETTING IN SYSTEM MODE

System mode permits a user to configure a variety of tester options from the keyboard. These settings are retained in a battery-backed storage even while powered down. The System screen appears as follows:



System Screen

1.6.1 SETTING TIME & DATE

Press <<time&dat>>. Follow the command line prompt and enter a desired new value. The Syntax for the time using a 24 hour day is HH:MM and/or the date using the day, month, year form is DD/MM/YY. Time and date must be separated by a space. The entry for 3:05 PM, August 17, 1990 would be:

15:05 17/8/90

This information will appear as headers for test result logging or any file printouts.

1.6.2 CONFIGURING THE SERIAL COMMUNICATION PORT

Press <<rs232c>> and observe the following screen:

```
BAUD RATE:   9600        PARITY:  NONE
STOP BITS:    2          MODE:    DCE CL
BITS/CHAR:    8          TIMEOUT: NONE
Changing rs232c setup                12:00

      roll   advance                  end
  F1     F2     F3     F4     F5
```

RS232C Setup Screen

<<advance>> is used to highlight a specific parameter before changing it. <<roll>> is used to cycle through the available settings of a highlighted parameter.

OPTION SETTING IN SYSTEM MODE

The mode parameter defines the handshaking control between tester and data source or destination.

PIN	DTE	DCE
2	XMIT DATA	RCV DATA
3	RCV DATA	XMIT DATA
4	RTS (SOURCE)	RTS (SENSE)
5	CTS (SENSE)	CTS (SOURCE)
6	DSR (SENSE)	DSR (SOURCE)
7	GROUND	GROUND
20	DTR (SOURCE)	DTR (SENSE)

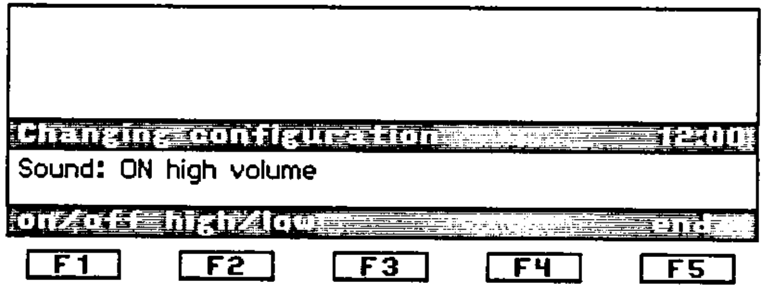
The designation OL means open loop and the FLUKE 900 will ignore the control line status. CL means closed loop. FLUKE 900 will stop sending data when it senses a control line going low. In the closed loop setting it also responds to received XON, XOFF protocol characters.

The timeout parameter is the length of time the Fluke 900 will wait for a positive control line signal before it assumes a failure by the receiving equipment and aborts its data transmission.

Once the settings are established, pressing <<end>> or <ENTER> will update the new values into the nonvolatile RAM.

1.6.3 SETTING THE VOLUME OF THE AUDIBLE TONE

Press <<sound>> and observe the current setting:

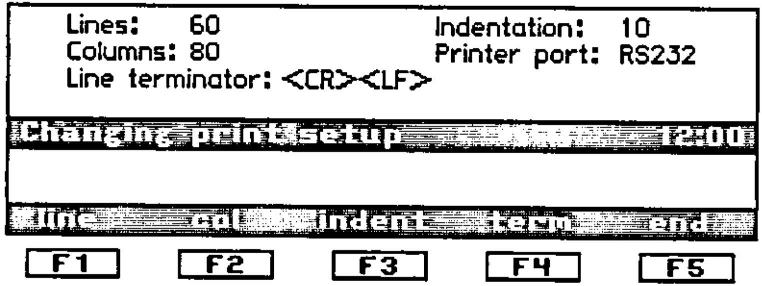


Sound Screen

<F1> and <F2> change the displayed setting and <ENTER> updates it into nonvolatile RAM.

1.6.4 FORMATTING PRINTER LISTINGS

Press <<printer>> and observe the definition of a page in lines and columns. A standard indentation from column 1 and selection of a line terminator (CR, LF or CR LF) may also be made.



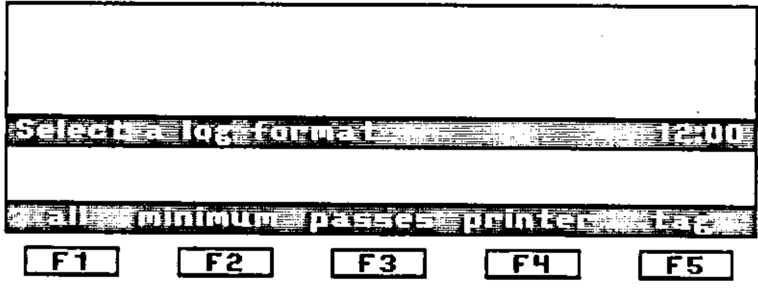
Printer Setup Screen

Press <<line>>, <<col>> or <<indent>> for a command line prompt that requires a numeric entry. Press <<end>> to update the nonvolatile RAM values.

Note: The printer port is permanently assigned to the RS232 port.

1.6.5 DEFINING LOG FORMATS FOR TEST RESULTS

A log format is a data filter that selects only specified meaningful parts of the test result data. Press <<config_log>> to show the names of five log formats that can be renamed and redefined by the user.



Log Format Screen

For example, pressing <<all>> displays eight settings:

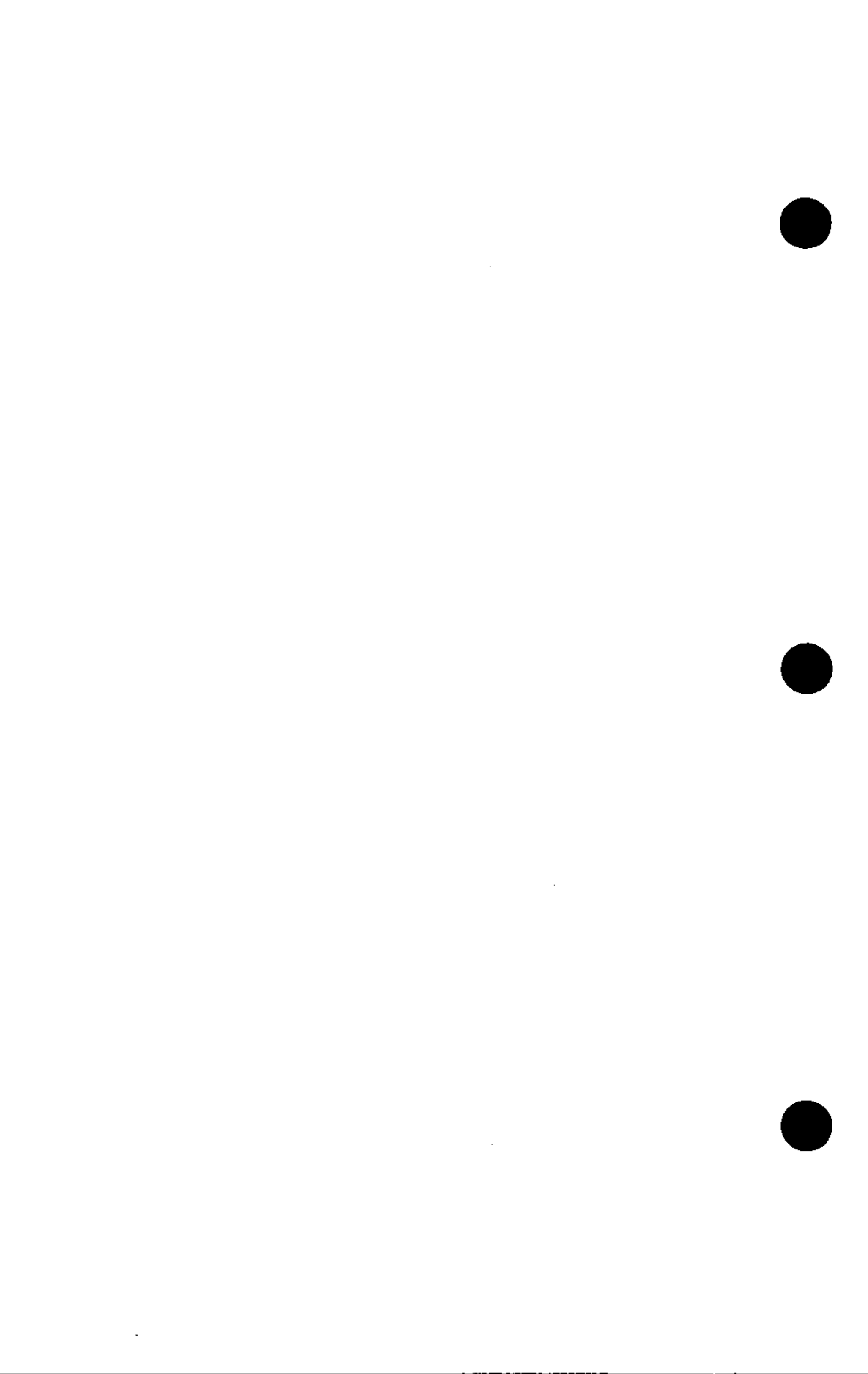
```
log data to a file              sequence flow    on
  parm changes     on           user comments    on
  failures         on           system changes   on
  passes           on           untested loc's   on
Log format name: all                           12:00
New_setup_name:
roll   advance                                  end
 F1      F2       F3       F4       F5
```

Log Settings Screen

The name of any log format can be changed with a command line entry. As before, <<advance>> moves the highlighted cursor and <<roll>> scrolls through various settings. Data logging is usually done to keep track of test results while using a sequence. The explanation of the "all" log format shown above is:

- data logged to a file. This file may be stored on a cartridge or in the system RAM for subsequent upload to a computer or printer. If this setting is changed to "printer", the data goes to the communication port as it is generated.
- local parameter changes by the operator are logged.
- device failures are logged. In addition, if a parameter is changed and a pass result occurs, both the failure and subsequent pass are logged. For the case where a failure occurs, the clip is adjusted and a pass occurs, nothing is logged.

- device passes are logged.
- anytime a user deviates from the clipping order of the sequence, it is recorded.
- user comments such as "board repaired" are logged.
- times other log files are opened and closed are logged.
- all device locations not tested with the sequence are logged.



2 TECHNICAL PRINCIPLES

2.1 TEST METHODOLOGY

Circuit board testing always involves the application of a controlled signal stimulus and the evaluation of a measured response against expected parameters. When this is employed for the board as a whole, it is termed a "go/nogo test". When used to isolate the source of a failure down to the component or internal node of a board, it is termed "diagnostics".

The FLUKE 900 is a fault isolation tool designed to add diagnostic capability easily to an existing go/nogo test. A straightforward application would be to measure the response of devices in their circuit to a board's resident selftest. Another example is diagnosis in system test where a board is exercised during normal product operation. Finally, as a complement to another tester, such as a microprocessor emulation tester, the FLUKE 900 can use the test stimulus routines to isolate a failing area to the faulty node or component.

The FLUKE 900 employs a principle called Dynamic Reference Comparison (DRC) to passively monitor and analyze internal board signal activity. Because there is no backdriving that would create artificial conditions, the method is ideal for detecting timing and static problems that occur in the natural board environment. The main requirement is that the stimulus be controllable so it can be run from the same point repeatably. The FLUKE 900's reset and trigger features make this easy to do.

In addition to identifying faulty components with high accuracy, good devices are differentiated from failing ones in spite of feedback loops and data dependant faults that confuse other methods. FLUKE 900 complements a range of user skills and other test equipment to be a complete diagnostic solution or an effective diagnostic enhancement.

Various levels of automation may be programmed into the tester according to the knowledge of board operation and troubleshooting experience of the test technician. A test sequence is a guided clip troubleshooting procedure designed to prompt the user step by step through a board. If a user has little insight into the failure mode, he is prompted through the <NEXT> key to clip in a preprogrammed order, typically by signal flow. An experienced user may choose to jump to a specific device to verify a hunch, shortcutting the predetermined order. The sequence relieves the operator from remembering test setup, parameter and device number information since all references are stored by board location (e.g. U48).

The information provided as test results may be used in a manually interpreted or automatic fault tracing algorithm. The functionality of each device is evaluated to give a PASS or FAIL determination. The tolerances of this test are preprogrammable and operator variable for flexibility of analysis. Additional signal checks include logic status, static or active check, and frequency measurement. A built-in logic monitor reports these pin-by-pin results at the end of a test in an easy-to-read pictoral format. Thus, even with a PASS result, missing signals will assist in backtracing.

2.2 DYNAMIC REFERENCE COMPARISON (DRC)

DRC evaluates a suspect digital device that is operating in its circuit by comparing its output signals to those of a known good device operating in synchronism under the same input stimuli. This technique is sometimes referred to as hardware or golden chip comparison. In principle, any device for which you have a known good sample may be verified in its circuit.

In order to make this practical over a wide range of digital devices, adjustable parameters are required in two areas of device performance:

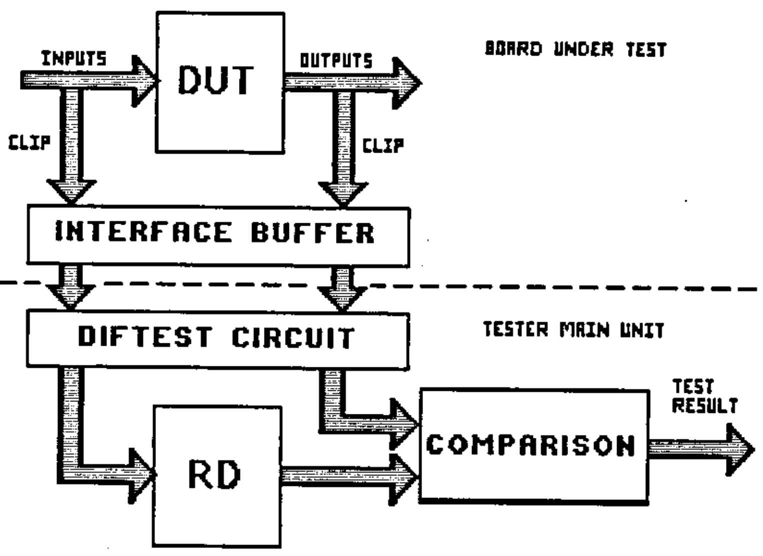
1. Initialization of suspect and known good devices.

2. Allowance for in-circuit effects relative to an unloaded reference device.

Initialization usually involves ignoring the comparison result until the reference device (RD) and device under test (DUT) are in the same state. A variety of techniques are used relying on either stimulus activity received by the DUT or additional stimulus applied to the RD to make one "catch up" to the other. In circuit effects that must be compensated for include fanout loading, noise, bus contention, indeterminate states and apparent timing race problems. The initialization and compensation parameters may be explained by examining DRC first at the device level, then at the board level.

2.2.1 DRC - DEVICE LEVEL CONCEPTS

The diagram below shows the basic signal routing when a test clip is applied to a DUT.

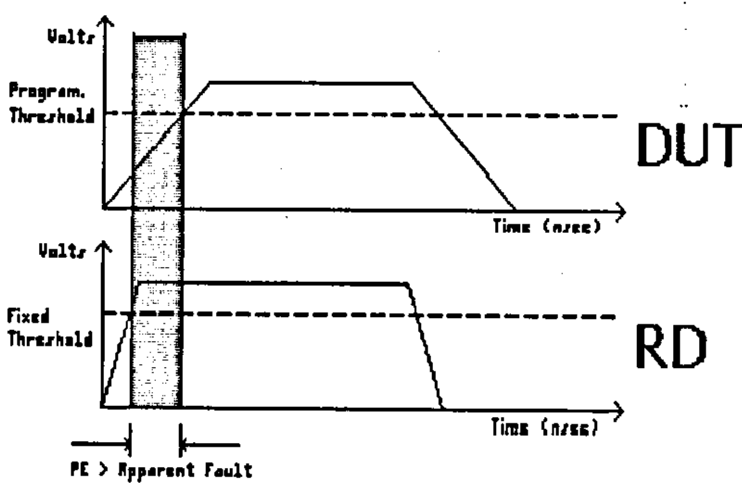


The input signals to the DUT are extracted and applied to the RD. Output signals are extracted and routed to a comparison circuit where timing tolerances and masking windows are applied. Only conditions that exceed the defined parameters are latched as a FAIL test result. The FLUKE 900 has a "Diftest" sensing circuit on the RD to determine in real time whether a pin can drive, and thus, whether it (and the corresponding DUT pin) is input or

output. Signals from DUT input, output and bidirectional pins are routed appropriately to RD inputs or the comparison circuit.

In considering how a test result is produced from the comparison circuit, it is helpful to think of what two corresponding RD and DUT output signals would look like. They will differ because of in-circuit conditions and, in fact, there may be a range of differences seen over many devices that we term the Performance Envelope (PE) of a good device.

Performance Envelope (PE)



Performance Envelope

Note that the apparent fault is affected by the variable threshold sensed on the DUT through the Interface Buffer. In addition, a fault mask parameter (FMASK) can be defined as the maximum allowable discrepancy between RD and DUT on each signal transition. Differences exceeding FMASK will be considered a true fault. FMASK is programmable in 10 nsec increments, from 20 nsec to 200 nsec. Threshold is programmable from 0 to 5 volts in 100 millivolt increments. Note that the commonly observed difference between a loaded DUT and unloaded RD is 30 nsec. With FMASK set at 30, a fault will be indicated with a 40 nsec discrepancy or, in other words, there is a 10 nsec fault capture ability.

Other relevant parameters at the device level are Synchronization Time (STIME), Trigger (TRIG), Ignore Pin (PIN_DEF) and Gate. Synch Time is the time before comparison when RD and DUT are synchronized. There are a number of techniques, specific to each device, which are tried to synchronize RD and DUT. If unsuccessful, the test will not commence. Trigger is a user-definable two word event on the device pins that indicates RD and DUT are initialized. The Pin Definition parameter permits the ignoring of specified pins. Gate is a selective ignore of all pins depending on the state of other signals and is useful to mask out indeterminate conditions.

2.2.2 DRC - BOARD LEVEL CONCEPTS

The task of synchronizing RD and DUT is assisted by whatever means are available to initialize the board under test. For microprocessor controlled circuits, this normally involves the FLUKE 900's Reset parameter. The polarity and timing of this Interface Buffer pulse signal is definable for restarting the board repeatedly every time the <TEST> key is pressed. For the setup where another system or tester must be in control of the start of test, the External Trigger/ External Gate lead on the Interface Buffer is used. A diagnostic verification normally uses a Test Time (TTIME) parameter set to the duration of the go/nogo test.

2.3 SIGNAL CONDITION TESTS AND OPERATOR CHECKS

These features provide troubleshooting assistance in addition to the basic DRC device functionality tests. Signals to the board under test and DUT can be assigned attributes and flagged to the user when they deviate. Typically, clock signals are assigned a specific frequency, chip selects and strobes are designated "active", inputs that are pulled up or down by resistors are designated H and L, and key off-board input signals are checked for their presence. Such condition testing may be done independently or together with DRC to assist in backtracing a fault.

Several features ensure that the user has properly set up the test. A clip check verification makes sure that the DUT clip is the proper size and orientation. It also checks that the DUT is receiving proper power. Since DRC relies on the presence of a known good RD, a useful feature is the RD_Test. It not only ensures that the RD is correctly inserted into the ZIF socket on the FLUKE 900's front panel, but also applies a test pattern to the RD to make sure it is good. This RD_Test feature can also be used on its own to perform simple device pre-screening and to identify an unknown device with its generic numbered equivalent.

2.4 PROGRAMMING

As with most ATE, the creation of a Sequence program is done by learning a good board. The learning, however, does not involve stimulus programming or recording of actual data streams. The general charateristics of the signals to and from a device are established - a relatively straight forward task. The Performance Envelope (PE) is set to an acceptable margin for each device on a good board. Normally, a few good boards are verified to ensure that the range of good boards all pass. Once these test parameters are established, along with a nominal order of clipping the various devices, it can be recalled from storage to diagnose a faulty board.

A Sequence is a "guided clip troubleshooting procedure" and can be set up with varying degrees of automation. As a general guide, the more a programmer knows about a board's operation, the more he can preprogram for automatic use by less experienced operators. Enhancements to a basic Sequence can include:

- displayed operator prompts and troubleshooting hints
- condition tests to assist signal tracing
- clipping order based on test results.

The basic sequence programming is done through keystroke entry, more like a programmable calculator than software generation. Advanced programming builds on this through a file editor using a simple test command language (see Appendix II). This is accomplished using FLUKE 900's screen editor and keyboard or via a serial communication link to a PC screen editor.

2.5 DATA LOGGING

The FLUKE 900 will automatically record, for each board tested, such things as:

- test results, especially failed devices
- operator clipping and keyboard actions
- devices not yet tested
- start and finish times
- rework instructions.

This record can be listed on a printer, stored on a cartridge or sent over the communication port to a PC. It is useful to print out rework tags or for a user to reanalyze his troubleshooting. The information could be used to compile statistics on methods and failure trends.

2.6 TESTING THEORY

For the purpose of comparison testing, digital devices are classified into three groups as follows:

COMBINATORIAL

- those devices whose output pins reflect immediately the conditions on the input pins. They have no internal memory, but may have tristate output pins (e.g. NAND gate, Bus Driver).

SYNCHRONOUS

- devices with internal memory whose state is brought to an output pin or can be definitely inferred. They always have a clock input line (e.g. up/down counter, J-K Flip Flop).

PROGRAMMABLE

- devices which require data to be written into them before outputs are valid (e.g. UART, RAM).

Depending upon which category a certain device falls into, it will have parameters that must be set properly for the three step DRC test: define, initialize, compare.

Step 1 <u>Defining a Device</u>

- IC number (i.e. 7400)

or size (number of pins)

- rd_drv (reference device ability to drive 1 LS Load).

Most device numbers with their associated size and rd_drv are already predefined.

Step 2 Initializing RD and DUT to the Same State

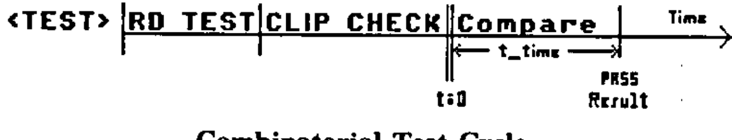
- Reset (a pulse issued on the R patch lead to reset a board)
- Offset (a delay to inhibit comparison after reset)
- S_Time (Synch Time enables DUT activity and extra RD stimuli to synchronize certain device types.)
- Trigger (starts comparison after a user-defined state appears on DUT)
- Gate (enables and disables comparison from a user-defined valid state).

Step 3 Comparing Output Pins within a Performance Envelope

- T_Time (Test Time defines the duration of comparison)
- F_Mask (Fault Mask defines how close to compare RD and DUT)
- Thrsld (Threshold defines logic 1 level)
- Pin_Def (Pin Definition permits specified DUT pins to be ignored).

2.6.1 COMBINATORIAL DEVICES

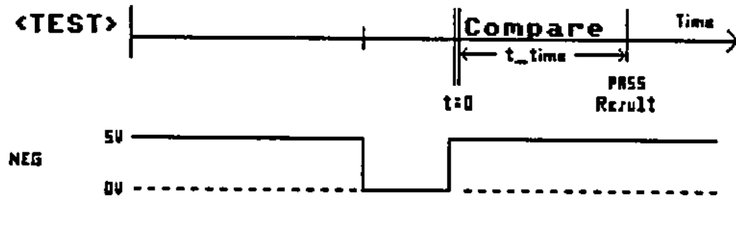
Let us consider a simple combinatorial device, such as a 7400, and draw a timing diagram of what happens after you clip onto a DUT and press <TEST>. It is known as a test cycle diagram.



Combinatorial Test Cycle

Notice that RD Test and Clip Check are performed first. This always occurs (unless rd_test or clip_check are off) and so we will omit showing it in further test cycle diagrams. For the purposes of the Time-to-Fault value in the test result, it is measured from the start of the comparison (t=0).

We did not yet use any of the initialization parameters listed previously in group 2. They are for more complex devices and circuit board conditions. Even on a simple 7400 we probably want to supply a reset pulse to force activity on the board under test. This appears as follows:

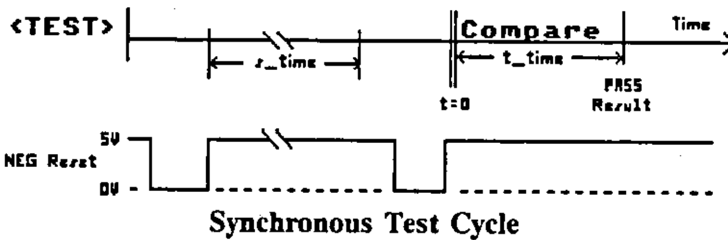


Combinatorial Test Cycle

The Reset pulse has a definable duration (100 ms default) before comparison begins and has a polarity which can be defined positive or negative.

2.6.2 SYNCHRONOUS DEVICES

Now consider a synchronous device such as a 74161 counter or 74165 shift register. This requires a Sync Time which lasts only as long as it takes to put RD and DUT into the same state (to a maximum defined by the S_Time value).



Synchronous Test Cycle

During S_Time, the RD and DUT are monitored to see if on-board activity puts them in the same state. We know it is in the same state by monitoring all outputs since there are no hidden states. For the '165 paralled load/serial out shift register, we must also observe a "parallel load" pulse or eight "serial shift" pulses to be certain that the hidden states are the same for DUT and RD. The Sync Conditions for each device form part of the ROM based device library and may also be created by users for new devices (See Section 7).

If no activity is seen on the DUT, synchronism may be attempted by stimulating the RD. For the '161 counter this would be a series of pulses to the clock line. The patterns applied come from the device library and may be (as in this case) a specific vector, the RD_Test stimulus vectors, or a random vector stimulus. If tristate outputs are present, a further parameter, Sync_Gate, is required to define validity of outputs.

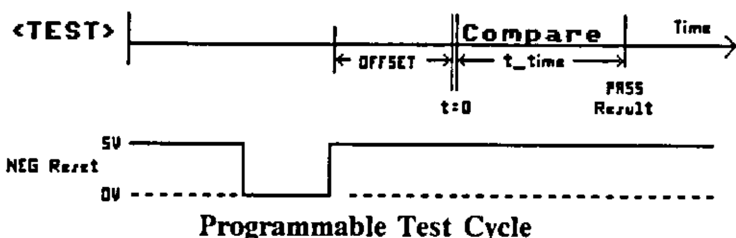
In summary, Sync Time is a period before reset/comparison during which every attempt is made to synchronize RD and DUT. Associated parameters include:

- Sync_Time (maximum duration of synchronization attempts)
- Sync_Cond (specified conditions, which must occur to synchronize RD and DUT)
- SYNC_Vect (RD stimulus during last part of S_Time)
- SYNC_RDT (RD stimulus during last part of S_Time)
- SYNC_PAT (RD stimulus during last part of S_Time)

2.6.3 PROGRAMMABLE DEVICES

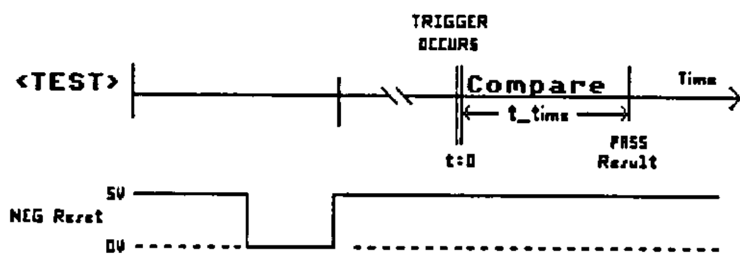
Consider a programmable device, such as an 8259 Interrupt Controller, which must receive mode control data to define its mode of operation. Prior to this, the pins are indeterminate and unable to be compared. If we wait after a Reset pulse for a sufficient time, the DUT will be initialized and valid comparison can begin.

The test cycle diagram with Reset Offset set to a negative value and s_time off, is:



Programmable Test Cycle

Note that comparison does not begin until offset has elapsed after the Reset pulse. This technique assumes that board activity did actually initialize the device. A preferred method is to actively monitor the 8259 inputs for receipt of mode control data and trigger the start of comparison from this.



Programmable Test Cycle

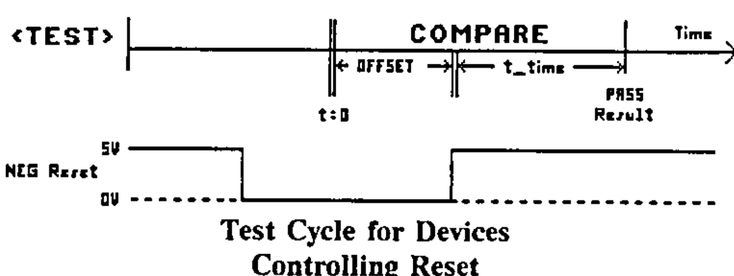
Note that comparison is held off only as long as it takes for the mode control word to appear on the DUT. If a failure is detected, the Time-to-Failure is measured from the Trigger point.

Trigger is a defined state on the DUT pins (plus the EXT patch lead) that enables comparison to begin. Gate has a similar function in defining a window of comparison.

Think of Trigger as: "start comparing when State X exists"

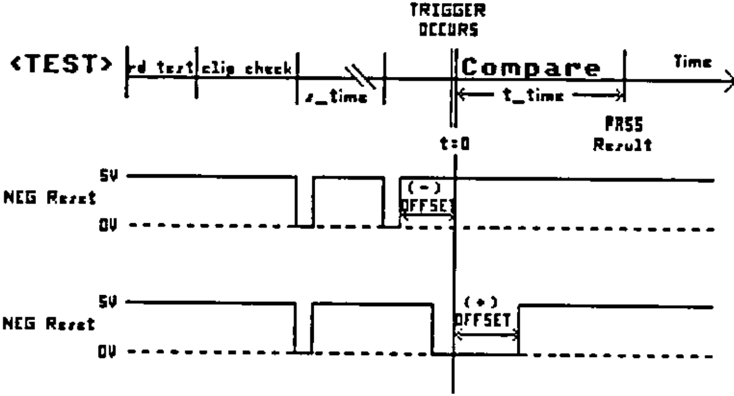
Think of Gate as: "compare only when State X exists".

Offset may be given a positive value, which advances comparison into the duration of the Reset pulse. This is mainly used to test the devices that actually control a board's reset function.



Test Cycle for Devices Controlling Reset

The following is a master test cycle diagram which incorporates all of the initialization parameters we have looked at:



2.7 TESTING APPLICATIONS

2.7.1 BOARD LEVEL

The general rule for making a troubleshooting sequence is to automate the common sense techniques used in manual troubleshooting. This usually involves testing for critical signals and devices first and following the flow of signals through a board. On a microprocessor-based board, critical signals include CLK, RESET, HLT and bus control signals. One approach to such a board is to check first for the presence of these signals with condition tests (e.g. freq, H, L, Active states) and then proceed with DRC testing. Alternatively, the devices responsible for clocking and control signals can be given a first priority in the sequence. On a multiprocessor board, the area of the master CPU is normally tested first.

Bus structured board testing proceeds from the bus outwards. Often, by merely testing 20 pin devices first, this is achieved. Twenty pins is the size of drivers, transceivers and latches. For nonprocessor cards, testing proceeds from the card edge through the board. Note that any sequence is only a nominal troubleshooting order that the user can override to shorten the process. He may want to focus initially on failure-prone areas or verify diagnostic messages from a functional test.

The diagnostic stimulus should be chosen to provide maximum activity. Ideally it can be repeatedly induced by resetting the board, or in the case of a multicard system, resetting the main board. Very occasionally, a reset will not

effect a restart or multiple reset pulses cannot be handled by a board. Testing can still proceed under partial reset conditions except for certain programmable devices or RAM that may not be initialized. In some cases, it may be better to use the External Trigger instead of Reset to start DRC testing from a signal indicating start of diagnostics.

2.7.2 DEVICE LEVEL

When a failure is detected,there are a number of things that may be done to give more insight about the device:

- reclip and retest to confirm the failure
- press <<state>> to see if the failing pin is active or stuck
- press <<EoT>> to see if elapsed time to fault is consistent
- increase FMASK to check the hardness of the fault
- for a bus device, test a few others on the bus before concluding which one is bad.

Loading Adjustments

When a device drives a high fanout or capacitive load, the signal rise time will be extended. Increasing the FMASK setting will compensate for this. Loading is normally within 50 ns of the response time for a device. If a greater FMASK is necessary, the user should investigate other causes.

Floating Inputs

Inputs floating or connected to a tristate source will cause an apparent fault when the signals are floating at an indeterminate level between 1 and 0. The DUT and Interface Buffer may see logic 1 at different levels. The best solution is to enable an external gate on a control signal that is active when device inputs are valid.

Note: Adjustments to threshold that may appear to solve the problem are probably not universal. Problems reappear on other boards because different devices have different characteristic thresholds.

Noise

To ignore noise and ringing on the inputs of a device, Threshold may have to be increased. Moving GND lead of IB close to DUT may also assist. Power supply spikes from the AC line input are a particularly difficult form of noise to ignore.

Race

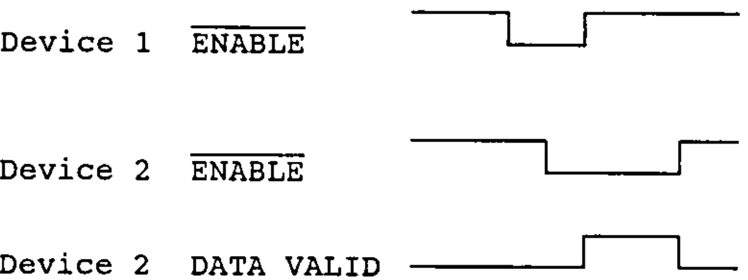
When the inputs to a sequential device (e.g. clock and data) are within a few nanoseconds, the slight skew of the System may cause the RD to clock in wrong data. If this is the cause of problems, you will generally observe failures on all output pins (e.g. Q, $\overline{Q}$ of a flipflop). Try to move Threshold up and down to separate signal edges.

Asynchronous Inputs

An external interrupt or any asynchronous signal acquired by latching with a synchronous board clock may exhibit a race problem if the edges are very close. The latching device in this case is not testable with DRC.

Bus Contention

This occurs when more than one device is enabled on a tristate bus at the same time. Bus drivers such as 74244, 74245 may exhibit this problem which is illustrated in the following timing diagram:



Output data for device 2 is indeterminate before the last part of its output enable signal. The solution is to gate comparison testing from a signal which indicates valid data. Most microprocessor designs provide this signal on the bus controller chip.

High Speed Signals

Recall that FMASK will ignore faults that are less than its duration. This implies that an FMASK setting of 60 ns cannot test signals faster than about 8 MHz at 50 percent duty cycle). To test 20 MHz signals, FMASK should be set to 20 ns. The general rule is that a signal's pulse width must be larger than FMASK.

Initializing Programmable Devices

Programmable devices are indeterminate and will fail if they are compared before on-board activity initializes them. Synchronization data built into the library normally handles this. For new devices or RAM, it is left to the user to mask out indeterminate states using Trigger. Some systems read from RAM that is uninitialized and the contents of the RD will not be the same. The solution is to trigger the start of comparison from a point after the read-before-write occurs. For example, set a trigger on the writing to the highest memory address since this could be the last step of a system's initialization routine.

2.7.3 APPLICATION ADVICE

The testing features are listed alphabetically below and peculiarities of their application are described.

EoT

- The end of test screen shows the states of all pins at the moment of failure. This is useful to determine the failing RAM address when solving read-before-write initialization difficulties of sequence creation.

- The time-to-fault reading will only be consistent if an External Trigger is employed on initial microprocessor activity. This is because CPU devices act on a Reset pulse with an inconsistent response time.

- Time-to-fault readings in the nanosecond range are useful for first fail conclusions. Readings in us and ms ranges are too coarse to be useful.

FMASK

- A lower setting provides a closer comparison and therefore higher quality of test.

- The setting should be high enough to pass the typical range of good boards.

- An abnormally high setting (i.e. 50 ns more than propagation/access time) should be investigated to see if a Gate is more appropriate.

FREQUENCY

- Duty cycle information is only available through <<freq>> and it is affected by the Threshold setting for narrow pulses.

GATE

- Use as an alternative to large FMASK settings.

- Bus control devices usually provide suitable signals for gating bus contention.

- The frequency and duty cycle of the gate setting may be observed to monitor complex timing relationships (e.g. a gate set on RAS and CAS of a RAM to observe access time).

PIN_DEF (Active Status)

- Activity check will reflect pin status even while a gate is not true. (i.e. an unselected bus device will still show active pins).

- Use sparingly since a bad board will have many missing signals and may be confusing.

- Use mostly on input pins since outputs are checked with DRC. Exceptions include one shots, line drivers and any other device not testable with DRC.

PIN_DEF (H, L Status)

- H and L status checks are performed with 10K pullup/pulldown resistors in the Interface Buffer. Pullups are present by default to bring unused input pins to a definite state. For an L status check, the pullup is present; for an H status check, a pulldown resistor is switched in.

PIN_DEF (F Status)

- Use to monitor ungated periodic clock signals.

- A Reset is issued with each pin frequency check.

PIN_DEF (IGN/COMP)

- When a single pin on a device requires a large F_MASK, that pin should be ignored while the device is tested a second time with a lower F_MASK on the rest of the device.

RESET

- Occasionally, a simple Reset fails to duplicate power down reset. Testing can still proceed, except for the few devices with uninitialized registers.

- Very rarely, multiple Reset pulses cause such problems as blowing a fuse. The extra pulses used for synchronization may be disabled with the command

SYNC_RESET_OFF in the .LOC file of the Sequence. Alternatively, Trigger may be used instead of Reset.

- Use of a negative Offset to wait for initialization on a device can be useful in trying to understand the operation of a good board. On a bad board, however, it is susceptible to false failures. Trigger is a more effective initialization parameter in this case.

- Use of a positive Offset is recommended for those devices that generate a board's reset so that DRC testing is done during the Reset pulse.

STATE

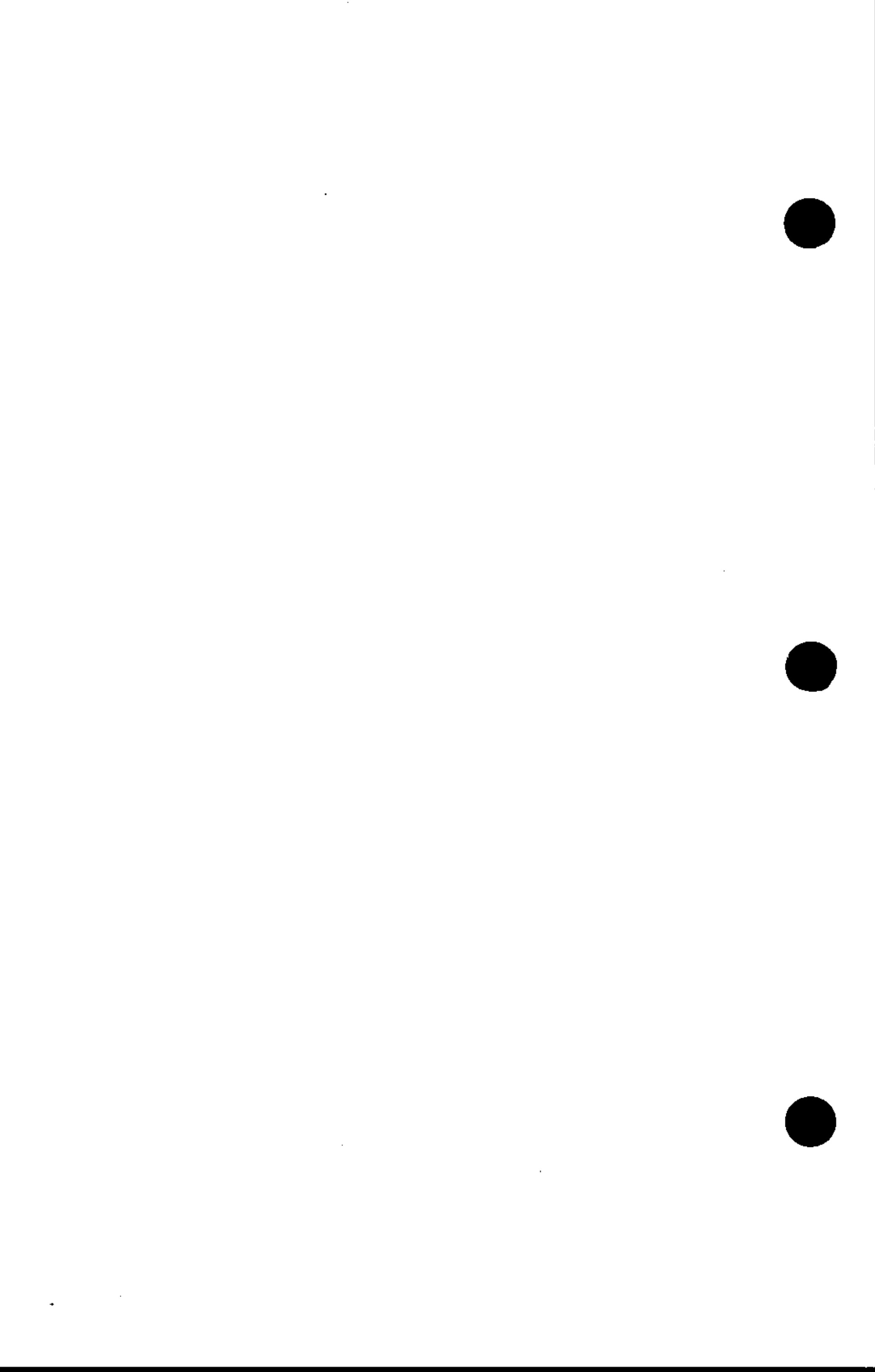
- The state screen shows the pin activity during test and can be a good qualitative indicator of how active a device was.

THRESHOLD

- It is recommended to keep Threshold constant and vary FMASK and Gate to make devices pass during Sequence Creation.

- Threshold sensitive circuits are subject to board-to-board variations.

- On a signal with a long rise time, lowering Threshold can permit a lower FMASK setting and thus a closer comparison on other pins of a device that do not require a large FMASK.



3 MANUAL MODE

Functionally Testing Devices in their Circuit

Manual Mode is used to verify a device suspected as faulty when no sequence exists. It may be employed by knowledgeable users where board volume is too low to warrant creating a sequence.

3.1 MAIN MANUAL SCREEN

Press <<manual>> to enter Manual Mode. Note that, as with all menu levels, you may exit Manual by pressing <ESC> at the left of the row of function keys. To protect against inadvertently exiting and losing any set parameters, the screen will prompt for confirmation. Press <<yes>> to exit or <<no>> to resume in Manual Mode with the following screen:

```
                    FMASK    30ns       THRSLD   2000mV
                    TTIME  1000ms       IGNORE   0 pins
SIZE   20 STD       STIME     OFF       RESET    NEG
RD_DRV HIGH         GATE      OFF       TRIG     OFF
Manual mode                                       12:00
Enter selection or chip name
                                                   etc
local     global      freq   results   comment
 F1         F2         F3       F4        F5
```

Manual Mode Screen

There are three columns of parameters which are set with default system values and the upper left corner of the screen is reserved for a DUT part number. Parameters may be changed by the user through <<local>> or <<global>>. A local parameter change affects only the device currently displayed on the screen, while a global change affects the current and subsequent devices. When a device is selected by entering its generic part number, the parameter values are updated from library memory. This ROM-based library is resident in the system and contains data for most common devices. Users may add other devices to their own library. Press "7400 <ENTER>" to bring up a typical standard part.

```
7400              FMASK    30ns      THRSLD   2000mV
                  TTIME  1000ms      IGNORE   0 pins
SIZE  14 STD      STIME     OFF      RESET    NEG
RD_DRV HIGH       GATE      OFF      TRIG     OFF
Manual mode                                    12:00
Enter selection or chip name
                                                 etc
local     global     freq   results   comment
 F1         F2        F3       F4        F5
```

Manual Testing

The first column of parameters specifies the DUT. RD_DRV (reference device drive) is set high for all parts that can drive 1 LS load and low for weaker devices. SIZE refers to the number of pins. STD (standard) means that power pins are: VCC=SIZE, GND=SIZE/2. That is, for a 7400, VCC is pin 14 and GND is pin 7. NSTD means any other power pin assignment. The other two columns are parameters for initializing and comparison testing:

- FMASK (fault mask), THRSLD (threshold) form the performance envelope
- T_TIME (test time) is the duration of comparison test
- S_TIME (sychronization time) is for library-defined synchronization
- RESET, TRIG (trigger) are for user-defined synchronization
- GATE is an active window to mask indeterminate signals
- IGNORE shows how many pins have been completely masked.

3.2 SETTING PARAMETERS

The function keys for the parameters are revealed by pressing <<local>>.

The order of the function key labels puts the most used parameters in the first screen for convenience. This includes <<pin_def>>, the pin definition parameter which is used to ignore pins and to specify signal condition tests. The second screen is accessed by pressing <ETC>. Pressing it again brings back the first screen.

```
7400              FMASK    30ns      THRSLD   2000mV
                  TTIME  1000ms      IGNORE   0 pins
SIZE  14 STD      STIME     OFF      RESET    NEG
RD_DRV HIGH       GATE      OFF      TRIG     OFF
Local parameters                              12:00
Enter selection or chip name
                                                etc
t_time  f_mask  thrsld  pin_def  reset
  F1      F2      F3      F4       F5
```

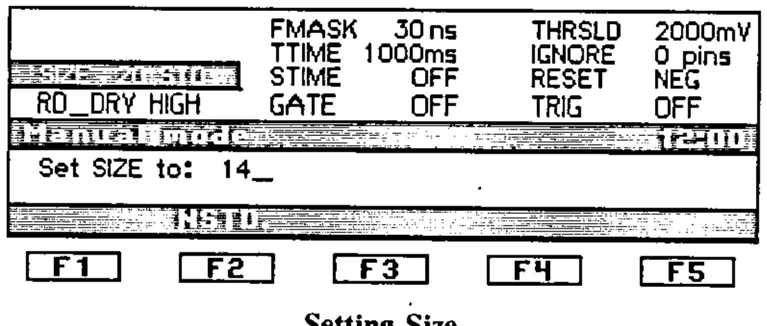
First Local Parameter Screen

```
7400                FMASK    30ns      THRSLD  2000mV
                    TTIME  1000ms      IGNORE  0 pins
SIZE  14 STD        STIME     OFF      RESET   NEG
RD_DRV HIGH         GATE      OFF      TRIG    OFF
Local parameters                               12:00
Enter selection or chip name
                                               etc
size         time     trig     create    rd_drv
 F1      F2      F3      F4      F5
```

Second Local Parameter Screen

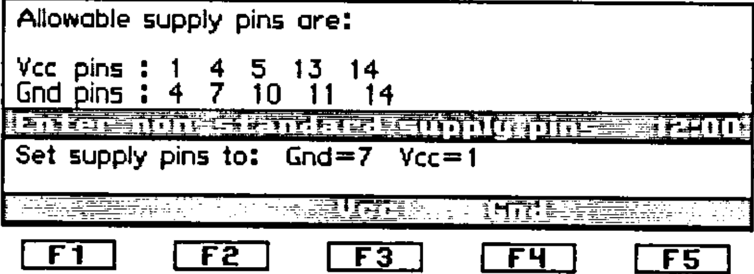
3.2.1 DEFINING SIZE AND POWER PINS

When a device is unknown to the library, the minimum DUT specification is to set SIZE. Pressing <size> permits entry of the number of pins and STD (standard) or NSTD (nonstandard) power configuration. Entering a standard 14 pin device appears as:



Setting Size

When entering a 14 pin non-standard device with Vcc and Gnd pins different, the screen appears as:



Size NSTD. Screen

The syntax shown above must be followed, using function keys to specify VCC and GND pins from the allowable list indicated on the screen. Allowable power pin assignments are:

SIZE	VCC	GND
8	1,4,7,8	4,5,8
14	1,4,5,13,14	4,7,10,11,14
16	1,4,5,8,15,16	4,7,8,12,13,16
18	1,4,5,8,9,17,18	4,7,8,9,14,15,18
20	1,4,5,8,9,19,20	4,7,8,9,10,16,17,20
22	1,4,5,8,9,21,22	4,7,8,9,10,11,18,19,22
24	1,4,5,8,9,23,24	4,7,8,9,10,11,12,20,21,24
28	1,4,5,8,9,27,28	4,7,8,9,10,11,12,14,24,25,28

3.2.2 SPECIFYING LOW RD DRIVE

The function key RD_DRV is used to specify a weak reference device that is not capable of driving 1 LS TTL load. For most RD's this will be left HIGH. Note that very weak devices (and an empty ZIF socket) always produce a PASS test result since they cannot drive the comparison circuit. (The RD_TEST feature is a check against inadvertently testing with an empty ZIF socket.)

3.2.3 COMPENSATING FOR DUT LOADING

This parameter compensates for in-circuit loading of the DUT relative to the RD. Other reasons for timing compensation include vendor or technology differences between RD and DUT, access time variations with memory devices, and wide variations in acceptable timing performance for certain devices. Note that small differences in propagation delay for unloaded devices may appear greater when they are under load in a circuit. The 10 nsec resolution of the FMASK setting is therefore not as coarse as it may seem.

To change FMASK, press <<f_mask>> and enter a value between 20 and 200 nsec on the command line.

```
7400                FMASK [illegible]     THRSLD   2000mV
                    TTIME   1000ms        IGNORE   0 pins
SIZE   14 STD       STIME      OFF        RESET    NEG
RD_DRV HIGH         GATE       OFF        TRIG     OFF
Enter fault mask (20-200ms)                        12:00
Set FMASK to: _


  F1        F2        F3        F4        F5
```

Fault Mask Screen

When <ENTER> is pressed, the new value will be transferred to the reverse field of the information screen. Testing and verification of the new value may proceed directly from this sublevel ("Local parameters" on the Status Line) or from the preceding Manual Mode level.

3.2.4 SETTING THE DURATION OF TEST

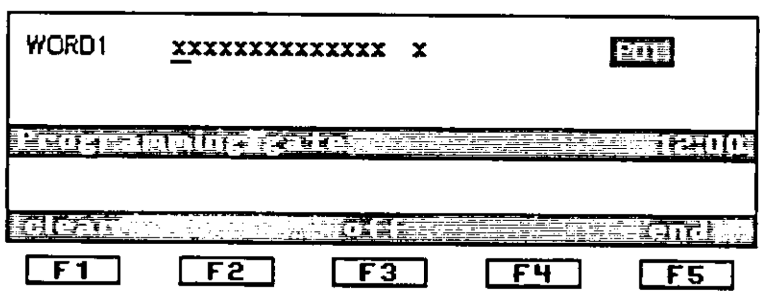
Test time or T_TIME is normally set as a global value to the approximate time of the board go/nogo test, since it will likely be the same for every device on the board. A local change to set Test Time as continuous can be done to check for an intermittent problem. Press <<t_time >> and proceed as with FMASK to set a new value.

3.2.5 SETTING S_TIME FOR SYNCHRONIZATION

Sync Time is enabled to initialize devices whose internal sequential states can be seen or inferred. The S_TIME requirements form part of the standard device library and include both synchronization methods and time. The default S_TIME will appear as 3000 msec for a synchronous device. This means that the DUT will be monitored and/or the RD stimulated for up to 3 seconds before producing a FAIL TO SYNCHRONIZE test result. The user may want to increase the S_TIME for devices that are running with slow signal rates (eg. a counter with a 10 Hz clock). Press <<s_time>> and proceed as with FMASK to set a new value.

3.2.6 GATING OUT INDETERMINATE CONDITIONS

Many devices, such as those with tristate outputs, have a natural gate automatically used by DRC. When such a device is not selected its outputs are not compared. The GATE parameter is used to further mask out indeterminate conditions like bus contention by monitoring a control signal that may not be a DUT input. Additionally, a bidirectional device may be tested in only one direction by gating on the DUT direction control input pin. Press <<gate>> to bring up the screen:



Gate Screen

As with Trigger, cursor control keys choose DUT pins or the EXT patch lead and 0,1 or X selects the pin states. <<end>> confirms the GATE changes and <<off>> temporarily disables the GATE.

3.2.7 SETTING LOGIC 1

Logic 1 can be changed from default (2000 mv) to screen out noise and ringing which can occur as downward spikes from VCC or upward spikes from ground. Signals that do not have sharp edges will be acquired by the Interface Buffer with slightly different timing depending on threshold level. THRSLD can therefore also be thought of as a parameter to adjust relative signal skew and overcome timing race problems. Press <<thrsld>> and proceed as with FMASK to set a new value.

3.2.8 DEFINING PIN CONDITIONS OR IGNORING PINS

Signal conditions on a pin-by-pin basis may be monitored and reported as a secondary test result. Note that this feature works independently of comparison testing, but can provide valuable information for backtracing a fault, interpreting DRC results and raising test/fault coverage.

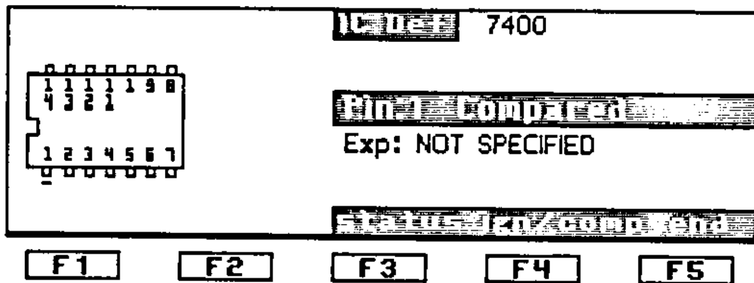
The following attributes for each pin are possible: H,L,A,F. H (high) and L (low) levels are useful to verify input pins that are tied up or down.

A (active) is an attribute that may be put on key input signals such as Output Enable, Strobe or necessary signals from off the board. Highlighting their absence will assist fault tracing. Activity Check can also help verify the outputs of one shots and open collector devices.

F (frequency) is recommended as an attribute on ungated clock inputs and for signals that don't apply to DRC.

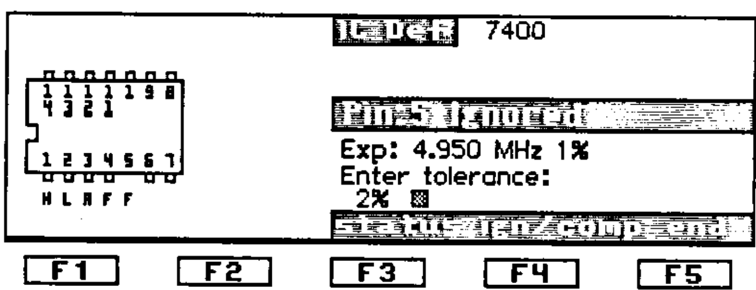
Output pin comparison results can be ignored through PIN_DEF to mask out indeterminate unused outputs or to observe beyond the first FAIL indication that freezes the comparison test.

Pressing <<pin_def>> for a 7400 brings up the following graphic screen:



Pin Definition Screen

Point to the pin of interest using the cursor control arrows and press <<ign/comp>> or <<status>>. Repeatedly pressing <<status>> brings up H,L,A and F attributes. As an example, the screen below is waiting for <ENTER> to confirm a frequency and tolerance attribute on pin 5.



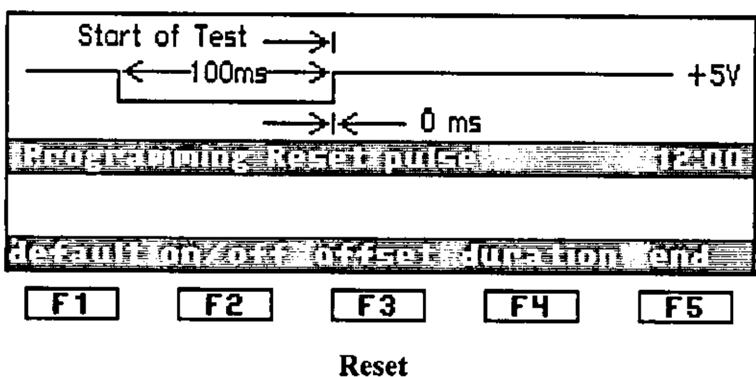
Defining Frequency on a Pin

Note that after all pin attributes have been assigned, <<end>> must be pressed to enable the PIN_DEF changes. Merely pressing <ESC> will not do this.

3.2.9 RESETTING THE BOARD

Reset is used whenever possible as the only control signal the tester can apply to a board under test. It repeatedly initiates the same diagnostic stimulus from system or selftest.

The RESET lead on the Interface Buffer is typically attached to a power-on reset signal on the board and assigned various characteristics through <<reset>>.



Reset

<<default>>	Reset is assigned default values, namely +5 V, negative polarity, 100 ms duration with 0 ms offset.
<<on/off>>	Alternate action key that disables or enables Reset lead.
<<offset>>	Positions trailing edge of Reset prior to start of test (negative value) or after the start of start of test (positive value).
<<duration>>	The duration of the Reset pulse.
<<end>>	Updates the Reset definition with the attributes established on the screen.

Pressing <ETC> brings up two more soft keys.

<<supply>> — Internal means a 5 V pulse sourced from the IB. External means a pulse sourced from a voltage applied to the V patch lead.

<<polarity>> — The normally tristate signal will go high-low-high for negative and low-high-low for positive.

Reset Offset is given a positive or negative value for two very different circuit conditions:

1. Positive OFFSET permits comparison to happen during the reset pulse itself. This is useful to test the devices that generate and control the on-board reset.

2. Negative OFFSET permits a period of DUT activity before comparison testing is enabled. It is useful for programmable devices that must be initialized by board activity before device outputs are valid. However, it is better to use a Trigger setting to actively monitor for the arrival of DUT initialization signals. OFFSET is an alternative method when little is known about board operation.

3.2.10 TRIGGERING START OF TEST

The detection of signal events on all DUT pins and the EXT patch lead can provide an indicator of initialization and general troubleshooting results. Press <<trig>> and observe two words that are the device size in length plus an extra "x".

```
WORD1     xxxxxxxxxxxxxx  x
WORD2     xxxxxxxxxxxxxx  x
Programming Trigger                 12:00
Set trigger word

clear              off              end
  F1      F2       F3       F4       F5
```

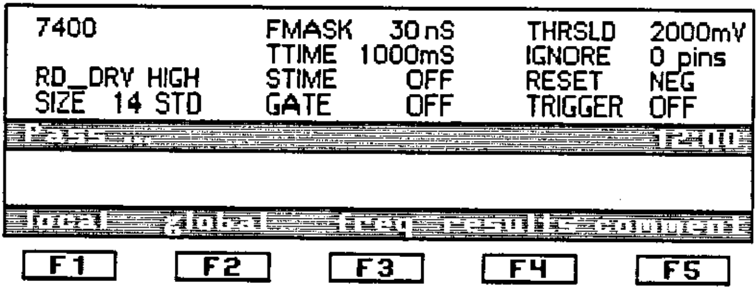
Trigger Screen

Cursor control keys will point to each pin as indicated and a 1, 0 or x/X (don't care) may be pressed. After the complete definition of the two possible events, press <<end>> to update the Trigger definition. Note that <<off>> disables the defined Trigger while retaining the specified trigger words. It may be re-enabled by entering Trigger and pressing <<end>>.

3.3 TEST EXECUTION

The procedure for testing a device in Manual Mode starts with entering the device number or size and setting any parameters that differ from default. The test clip is positioned over the suspect device with pin 1 on the clip lined up to pin 1 of the DUT. Larger overhanging clips may be used as long as pin 1 is properly positioned. The correct reference device is clamped into the ZIF socket and <TEST> is pressed. The Status Line indicates "Testing" while the following steps occur:

- RD Check
- Clip Check
- Test Cycle (refer to Section 2.6)
- Display Test Results



Manual Mode "PASS"

<F3>, <F4>, <F5> are:

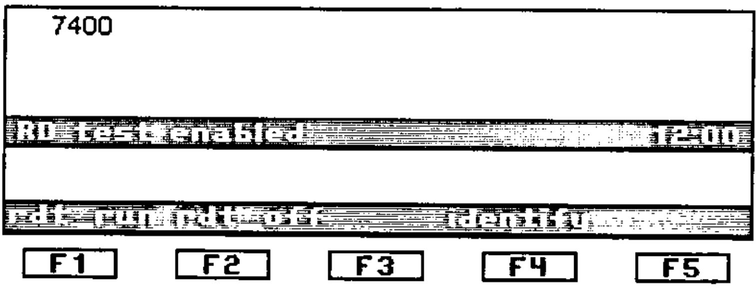
<<freq>>	a frequency measurement mode unrelated to the Test Cycle
<<results>>	a detailed graphic view of test results, user-selected for PASS, but automatically displayed for FAIL
<<comment>>	permits keyboard entry of text to a test result log file

We will return in more detail to these features later.

The second function key menu level in Manual Mode, accessed by <ETC>, permits control of RD Test, Clip Check and data logging.

3.3.1 RD TEST

RD Test is done automatically when testing devices that are in the library. It is a verification that the user has inserted the proper RD into the ZIF socket and, as much as is feasible, that the RD is good. This is done in varying degrees depending upon device type. The most complete is a truth table stored as RD TEST Vectors. Next is a checksum calculated for PROM and PAL type devices from reading about 100 address locations or applying a special PAL pattern. Finally, a simple presence check can verify that a device with drive capability is inserted in the socket.



RD TEST Screen

RD Test can be run from the above menu to do a prescreening device functionality test. Pattern rates are in the order of 10 kHz. Press <<rdt_off>> and <ESC> to disable RD Test if, for example, you wish to do signal condition testing without an RD.

In the RD Test sublevel, <<identify>> will play back the library patterns to the socket in an effort to match an unknown part with a generic device number. The number of pins must first be confirmed and the system assumes currently specified power pin configuration for the unknown RD.

If the RD is a PAL or PROM as identified in the Manual Mode device entry (eg. 2732, PAL20RA10), then pressing <<rdt_run>> will generate a checksum that will appear on the screen as a multidigit number. Press <<c_sum>> and enter this number to update RD Test with the expected checksum.

```
PAL20RA10
 Checksum error
 Checksum expected:  not loaded
          actual:    54940
RD test failed
rdt_run rdt_off c_sum

 F1    F2    F3    F4    F5
```

Checksum Verification

3.3.2 CLIP CHECK

Clip Check verifies that the clip is properly oriented by checking for logic 1 on Vcc pin(s) and logic 0 on Gnd pin(s).

Press <<clip_chk>> to enter the mode, <<on/off>> and <ENTER> to disable or enable it and <ESC> to exit the mode. A possible reason for disabling clip check would be to clip over part of a 40 pin device to detect trigger events and signal conditions.

3.3.3 TEST RESULTS

Test results fall into the following categories:

Operator Checks (1, 2, 3)
Performance Envelope Comparison (4, 5)
Signal Conditions (6, 5 plus 6), and
Unable to Test (7 a, b, c, d)

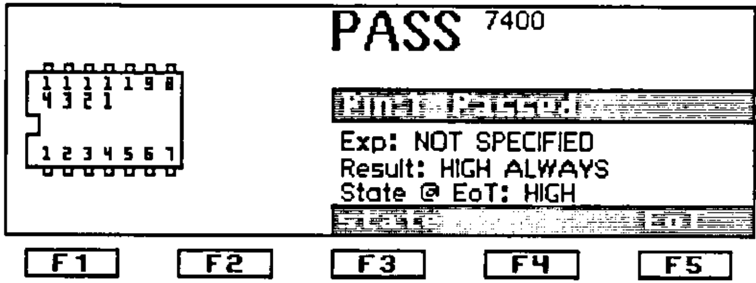
1. "RD test failed" appears on the status line when no RD, the wrong RD or a faulty RD is inserted in the socket. Failing to lower the contact clamping lever will also produce this message. (Severely shorted RD's produce the message "Excessive RD current").

2. "No clip inserted" and "Clip size incorrect" are operator alerts that cannot be overidden by turning Clip Check off. "Vcc-Gnd check failed" and "No signals from clip" indicate a wrongly oriented clip or lack of power to the DUT.

3. "Test aborted" appears when <TEST> is pressed again to stop the test cycle before comparison begins.

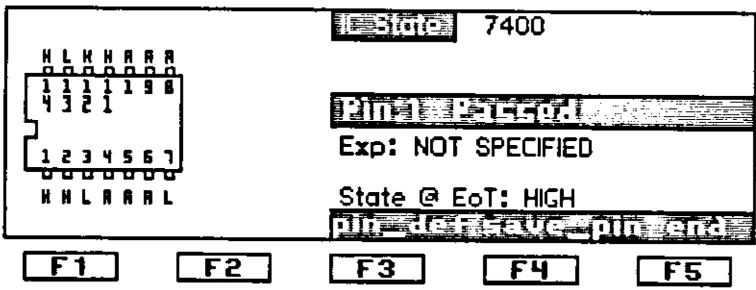
4. PASS - DUT within Performance Envelope

"Pass" on the status line indicates that the DUT did not exhibit a fault and any specified signal conditions were satisfied. Press <<results>> to show a detailed view.



DUT PASS Screen

The cursor control arrow keys are used to point to a pin of interest as highlighted in the status line. Expected preprogrammed conditions are listed as "Exp", actual results are listed as "Results", and the state of the pin at the end of test is listed as "State @ EoT". A complete snapshot of all pin signal conditions during the test is viewed by pressing <<state>>.

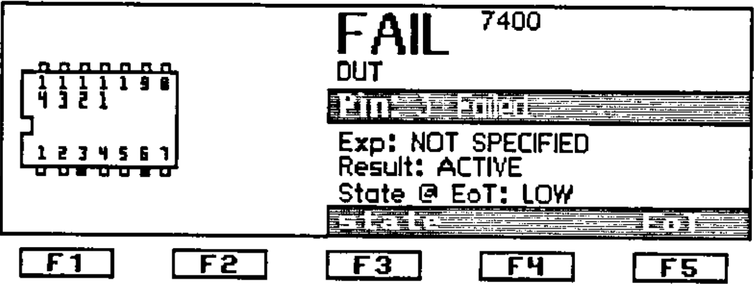


State Test Result Screen

From the State submode you can change expected signal conditions using <<pin_def>>, the same parameter accessible from Local Mode. Alternatively, you may assign actually observed H, L or A conditions to each pin using <<save_pin>>. A complete snapshot of all pin states at the end of test is also available by pressing <<EoT>>. Note that the use of the save_pin feature overrides all pins defined previously using pin_def.

5. FAIL - DUT Exceeded Performance Envelope

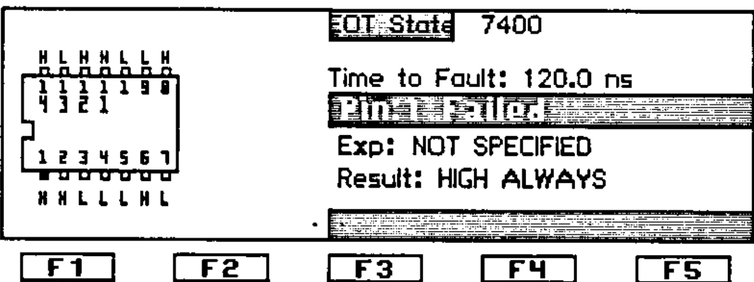
The graphic results screen appears along with "FAIL" in large letters when the DUT failed comparison as defined by the PE parameters. The failed pins are shown in reverse highlight and are flashing on the signal monitor.



DUT FAIL Screen

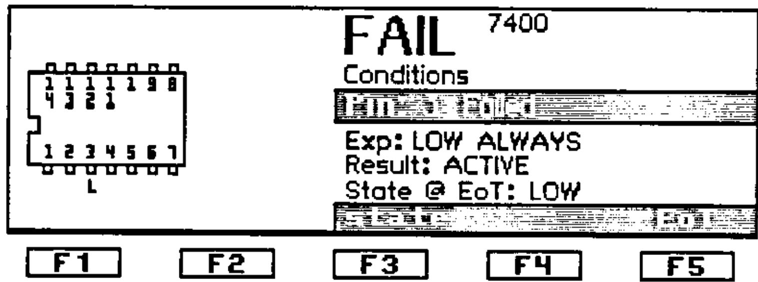
As with the PASS screen, snapshots may be viewed of all pin activity and state at end of test. Also shown by pressing <<EoT>> is the "Time to Fault" as measured from the start of comparison testing, with an accuracy of 40 ns or within the last digit shown (for ms, s). Note that most processor-based boards respond slightly differently each time to a reset pulse.

Therefore, time-to-fault readings will only be consistent if an external trigger is connected to the processor Reset Out or Memory Read signal (i.e. first op code fetch).



6. FAIL SIGNAL CONDITIONS
 - Operator Interpretation

The graphic results screen appears along with "Fail Conditions" when the DUT passed comparison but specified signal conditions were missing. These are indicated by letters (H,L,A,F) adjacent to the pins concerned.



Signal Condition Fail Screen

Depending on whether the pins are input or output and how they were programmed, it may indicate a faulty DUT or be a backtracing indicator. Snapshots may be viewed of pin activity with <<state>> and <<EoT>>.

7. UNABLE TO TEST - Operator Interpretation

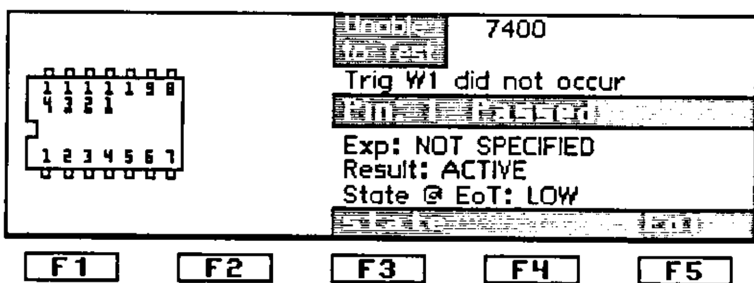
There are four conditions that will cause this message without ever doing a comparison test of the DUT.

a) "Failed to synchronize"
Board activity is insufficient to initialize DUT. Look elsewhere for the source of the problem.

b) "Synchronization timeout"
S_Time setting is too small for the specified sync vectors (Refer to Section 7.2)

c) "Gate did not occur"
Specified gate did not occur.

d) "Trigger did not occur", "Trig W1 did not occur", or "Trig W2 did not occur".
Specified trigger is not present.

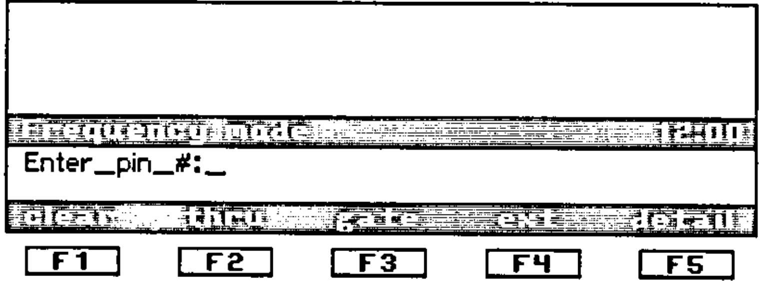


Unable to Test Screen

This typically indicates faulty activity from a circuit feeding this device.

3.4 FREQUENCY COUNTER

The measurement of frequency and pulse width timing on any pins and the EXT patch lead is possible under operator control. This is in addition to frequency condition testing that gives a pass/fail result. In Manual Mode, press <<freq>>.

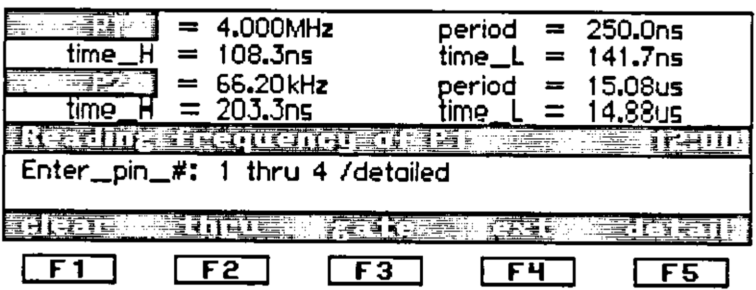


Frequency Mode Screen

Follow the command line prompt to enter the pins of interest. Note that pressing "1 <<thru>> 4" will choose pins 1,2,3,4. <<ext>> selects the external patch lead and <<gate>> will display the frequency of a complex gate function.

<<detail>> specifies, in addition to frequency, the period, time high and time low. The following screen shows a multiple detailed reading. The display can be scrolled with Page Up and Page Down cursor control keys.

Note below that the second set of function key labels has been brought up with <<ETC>>.



Detailed Frequency Screen

The multiplexed frequency reading normally cycles through the selected pins as shown on the changing Status Line. For a slow or inactive signal, the frequency measurement window may not be large enough to capture the signal. Therefore, to examine signals below 100 Hz or single pulse widths, press <<no_time>>, which disables the frequency window timeout. The measurement will then remain on a selected signal and may be manually stepped to the next one with <<next_frq>>. Note that every time the command line instruction is changed <ENTER> must be pressed to execute the new measurement command.

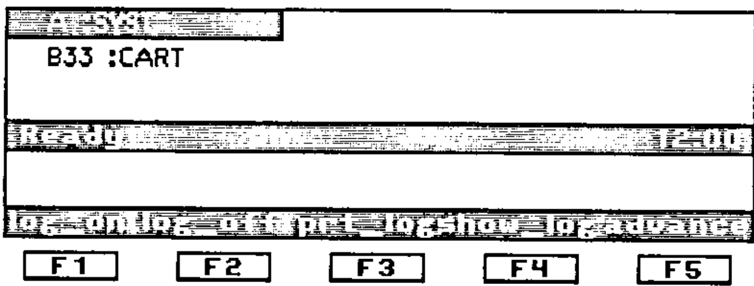
3.5 DATA LOGGING

A Log is used to record test results and user actions in a file or to a printer. Its application to automatic board repair is described further in Section 4.6. Here we look at how to activate it.

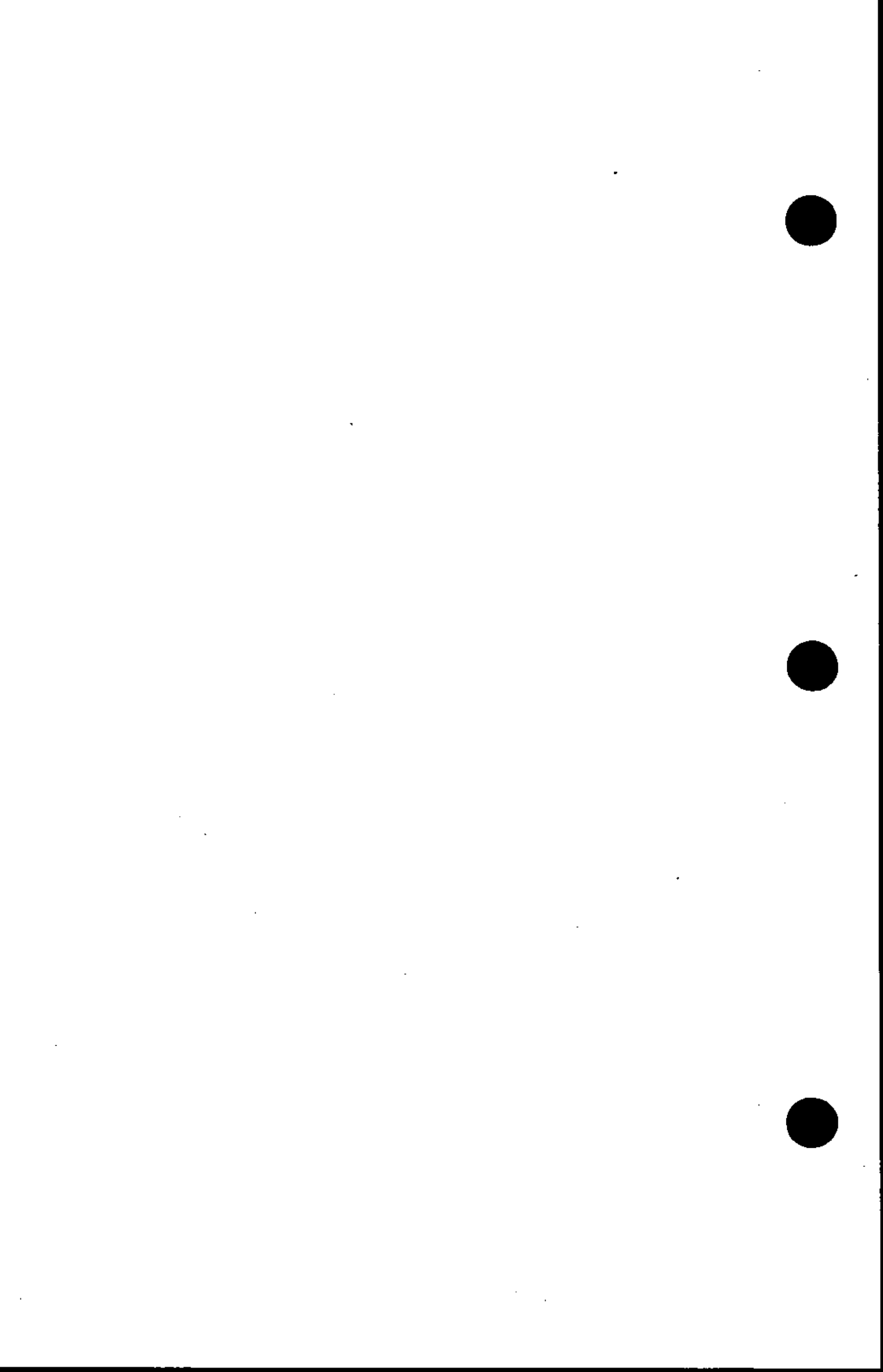
Recall from Section 1.4 that five "log formats" may be defined in System Mode. They are data filters that screen various types of data for routing to a file or printer. They may be assigned a convenient name such as "all" to route all types of data to a file. Press <<log>> then <<log_on>> to open a log file.

Choose a log format, for example <<all>> and enter a filename and destination. Note that messages entered with <<comment>> during Manual Mode testing are logged under this format.

There can be up to eight open log files with a pointer indicated by a reverse field on the screen. Use <<advance>> to point to the file to be shown/printed/logged off. They remain open for logging after <ESC> is pressed to return to Manual Mode.



Data Logging Screen



4 SEQUENCE MODE

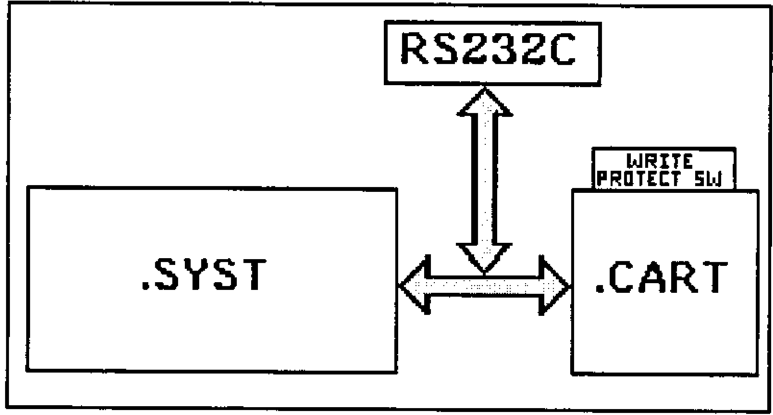
Functional Board Diagnosis

4.1 OVERVIEW

A Sequence is a guided clip troubleshooting procedure. Its simplest use involves loading a sequence file from the cartridge or from the serial communications port and "running" the sequence. The user follows instructions on the screen to clip and test devices in a predefined order. The clipping order by board location and each device's parameter settings come from a previously verified Sequence.

The test proceeds using two keys: <NEXT> and <TEST>. More typically, a user will jump out of the predefined order to verify devices according to his experience or the observed failure mode of the board. An advanced sequence application could involve grouping devices into smaller sequences pertaining to functional areas or test diagnostics. Automatic redirection is even possible in response to specified test results.

Sequences are interrelated files that may be resident in system RAM or on a nonvolatile RAM cartridge. Sequences may be loaded via the RS232 serial communication port into system RAM or cartridge.

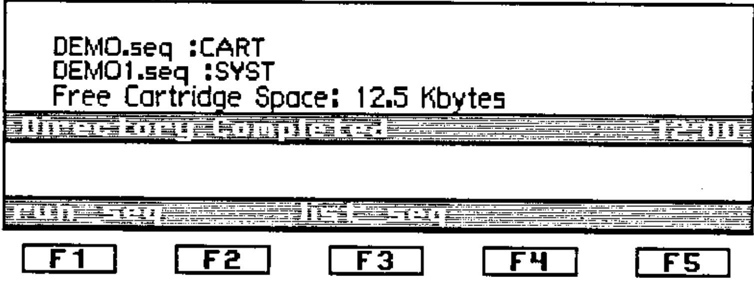


Sequence Transfer Diagram

Sequence names are a maximum of 15 characters with a source extension (eg. TESTBOARD1:CART). Testing with a Sequence is done in Sequence Mode, one of the four selections of the main menu level. Although Sequence Mode is run-only, local changes of parameters may be made as in Manual Mode. The Sequence remains uncorrupted and the set parameters are restored when a device is reselected.

4.2 DIRECTORY OF SEQUENCES

Press <<sequence>> in the main menu to bring up two new labels: <<run_seq>> and <<list_seq>>. The latter lists all sequences on cartridge and in the system.



Sequence Directory Screen

While the names are scrolling past, <<pause>> alternately freezes and resumes the listing. Free cartridge space is shown on the final line.

4.3 RUNNING A SEQUENCE

Press <<run_seq>> and enter the Sequence name on the command line. Cartridge is the assumed source unless <<:SYST>> is pressed. The screen displayed will show either the first device to clip or a message concerning test setup. It is important to duplicate the test setup under which a Sequence was created including Reset and GND lead connections, diagnostic stimulus, and board option settings. Arranging clips and RDs in an organized fashion is good practice as well.

```
DEMO.seq                       DEMO.loc
U8   8288                                      20p
MAKE SURE DIAGNOSTIC EPROM IS INSTALLED

Ready

local      global     freq   results   comment
  F1         F2         F3       F4        F5
```

Typical Sequence Screen

The first line on the screen references the files that make up a Sequence. The ".seq" extension is the file containing clipping order. The ".loc" extension is the file containing the device parameter settings.

The second line shows the device location, the device number and the recommended clip size. Note that the clip may be this size or larger since overhanging clip pins are ignored.

The third and fourth lines show optional user prompt comments.

Press <NEXT> to step from device to device in the sequence. Alternatively, a location may be specified explicitly to overide the predefined order. Example: "U48 <ENTER>". At the last device, "END OF SEQUENCE" is displayed and <NEXT> restarts the Sequence.

Note that the cartridge must remain inserted while running a sequence.

4.4 TEMPORARY PARAMETER CHANGES

The Local Parameter Submode functions the same in Sequence as in Manual Mode. Press <<local>> and proceed with changes as outlined in Section 3.2. There are various reasons for changing parameters during testing:

FMASK: Increasing the value will verify the "hardness" of fault

T_TIME: Intermittent faults are best examined with a continuous Test Time

THRSLD: Not recommended for general use. Exceptions are when measuring high frequencies and checking for fast active pulses (condition test "A"). Raising and lowering Threshold can assist interpretation.

PIN_DEF: Used to ignore a failing pin and check for other faults.

When a device location is reselected, the original Sequence parameters are restored.

4.5 RUN SEQUENCE MENU LABELS

Other function keys on the two menu screens available when running a Sequence are:

<<freq>> This functions as a separate frequency counter to provide further insight (see Section 3 for operation).

<<comment>> Used to add rework instructions to a file. With an open log file, press <<comment>> to bring up the following screen:

Add a comment to the log

Enter_comment: U48 replaced board not repaired

clear board repaired not replaced

F1	F2	F3	F4	F5

Note that the function key labels provide convenient words to speed message entry.

<<results>> This displays detailed information on the test result.

<<untested>> A list is displayed on the screen of all untested Sequence devices. This is a

useful reminder when a user is interrupted in testing. <ESC> returns to Sequence screen.

<<start>> — The execution is returned to the start of the Sequence. Confirmation is requested on re-initializing the list of untested devices.

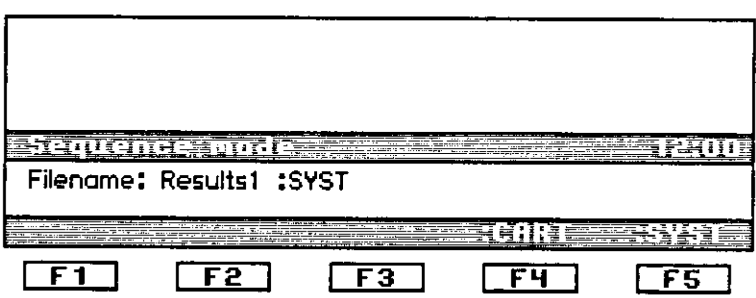
<<rd_test>>
<<clip_chk>> — These keys were explained in Section 3 on Manual Mode and are seldom used here.

<<log>> — The data logging function is particularly useful in Sequence Mode. To open a log file at the start of a Sequence (or any time), follow these steps:

Press <<log>>, <<log_on>>

F1 F2 F3 F4 F5

Select a log format, for example <<all>>



Enter filename and destination.

Press <ESC> to return to running the Sequence.

Note that, according to the log format (data filter) selected, test and operator information will be routed to a system RAM file, cartridge file or the serial port.

4.6 SUGGESTED SITUATIONS FOR USING LOG FILES

Service Depot with same technician doing test and rework:

> Track failures, parameter changes and untested devices to help user organize himself.

Production with separate rework area:

> Track failures, passes, user comments and untested locations to assist interdepartmental communication and recording of IC vendor failure statistics.

Less experienced test technician with knowledgeable supervisor:

> Track clipping order, failures and untested locations to help with troubleshooting interpretation.

All log files record the exact time they were opened and closed. Typically, each board has a separate file named after its serial number. Log files may be stored on cartridge, but since they consume a lot of space, it is better that they reside in system RAM. Files may be dumped to the serial port using <<prt_log>> in the log screen menu.

<<show_log>> directs the file to the screen.

<<advance>> highlights one of several possible log files present for printing, showing or logging off.

4.7 SAMPLE LOG FILE

```
1       ---------------------------------
2       Log_on:                              22:26 14 Jan 1988
3       Sequence:  XT:CART
4
5       Start logging to:  SN44063:SYST
6
7       U99  7400   ----   passed
8
9
10      U8  8288   ----  passed
11      > Jump to location U4
12
13      U4  PAL20RA10   ----   unable to test
14        RD test failed
15      > Local change: C_SUM 54940
16
17      U4  PAL20RA10   ----   failed 40.0 ns
18        Pin(s): 15 16 19 20
19      > Jump to location U48
20
21      U48  8259   ----   failed 983.0 ms
22        Incorrect frequency on pin(s): 5
23      Stop logging to: SY44063:SYST
24
25      Untested locations:
26      U1                      U5
27      U7                      U15
28      U6                      U9
29      U10                     U11
30      U16                     U2
31      U13                     U14
32      U22                     U85
33      U96                     U24
34      U97                     U18
```

4.8 TEST RESULTS

Test results fall into the following categories:

Operator Checks (1, 2, 3)
Performance Envelope Ccomparison (4, 5)
Signal Conditions (6, 5 plus 6), and
Unable to Test (7 a, b, c, d)

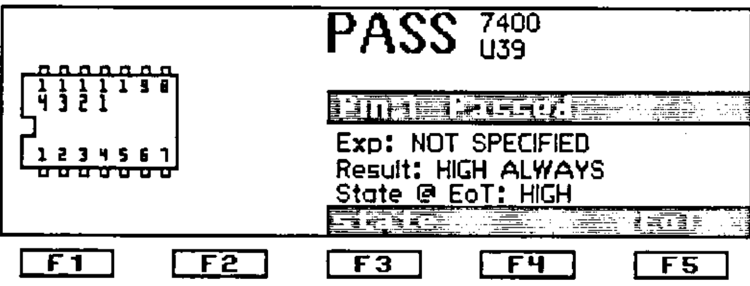
1. "RD test failed" appears on the status line when no RD, the wrong RD or a faulty RD is inserted in the socket. Failing to lower the contact clamping lever will also produce this message. (Severely shorted RD's produce the message "Excessive RD current").

2. "No clip inserted" and "Clip size incorrect" are operator alerts that cannot be overidden by turning Clip Check off. "Vcc-Gnd check failed" and "No signals from clip" indicate a wrongly oriented clip or lack of power to the DUT.

3. "Test aborted" appears when <TEST> is pressed again to stop the test cycle before comparison begins.

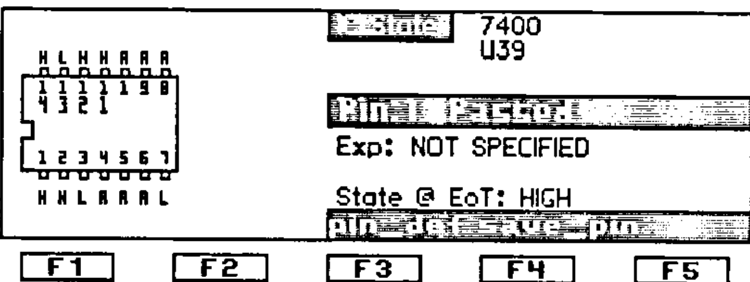
4. PASS - DUT within Performance Envelope

"Pass" on the status line indicates that the DUT did not exhibit a fault and any specified signal conditions were satisfied. Press <<results>> to show a detailed view.



DUT PASS Screen

The cursor control arrow keys are used to point to a pin of interest as highlighted in the status line. Expected preprogrammed conditions are listed as "Exp", actual results are listed as "Results", and the state of the pin at the end of test is listed as "State @ EoT". A complete snapshot of all pin signal conditions during the test is viewed by pressing <<state>>.

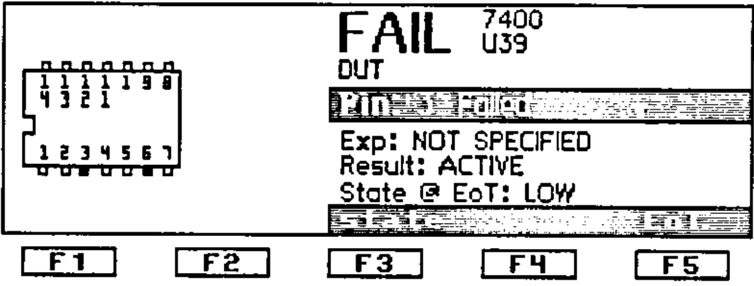


State Test Result Screen

From the State submode you can change expected signal conditions using <<pin_def>>, the same parameter accessible from Local Mode. Alternatively, you may assign actually observed H, L or A conditions to each pin using <<save_pin>>. A complete snapshot of all pin states at the end of test is also available by pressing <<EoT>>. Note that the use of the save_pin feature overrides all pins defined previously using pin_def.

5. FAIL - DUT Exceeded Performance Envelope

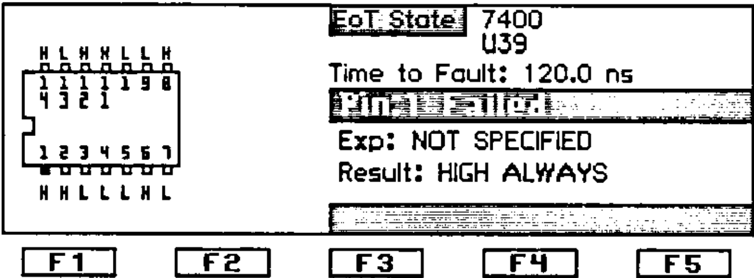
The graphic results screen appears along with "FAIL" in large letters when the DUT failed comparison as defined by the PE parameters. The failed pins are shown in reverse highlight and are flashing on the signal monitor.



DUT FAIL Screen

As with the PASS screen, snapshots may be viewed of all pin activity and state at end of test. Also shown by pressing <<EoT>> is the "Time to Fault" as measured from the start of comparison testing, with an accuracy of 40 ns or within the last digit shown (for ms, s). Note that most processor-based boards respond slightly differently each time to a reset pulse.

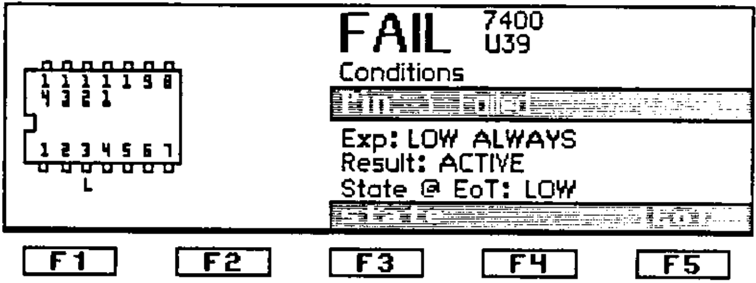
Therefore, time-to-fault readings will only be consistent if an external trigger is connected to the processor Reset Out or Memory Read signal (i.e. first op code fetch).



End of Test Screen

6. FAIL SIGNAL CONDITIONS
- Operator Interpretation

The graphic results screen appears along with "Fail Conditions" when the DUT passed comparison but specified signal conditions were missing. These are indicated by letters (H,L,A,F) adjacent to the pins concerned.



Signal Condition Fail Screen

Depending on whether the pins are input or output and how they were programmed, it may indicate a faulty DUT or be a backtracing indicator. Snapshots may be viewed of pin activity with <<state>> and <<EoT>>.

7. UNABLE TO TEST - Operator Interpretation

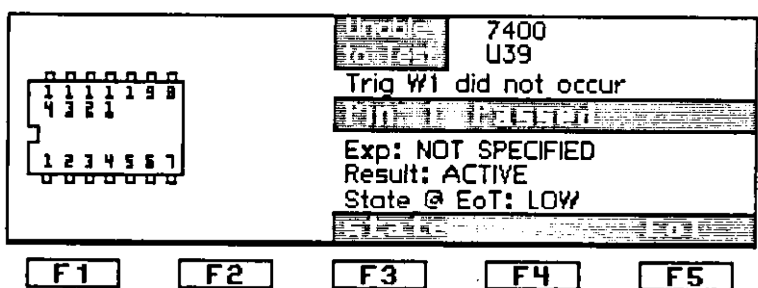
There are four conditions that will cause this message without ever doing a comparison test of the DUT.

a) "Failed to synchronize"
Board activity is insufficient to initialize DUT. Look elsewhere for the source of the problem.

b) "Synchronization timeout"
S_Time setting is too small for the specified sync vectors (Refer to Section 7.2)

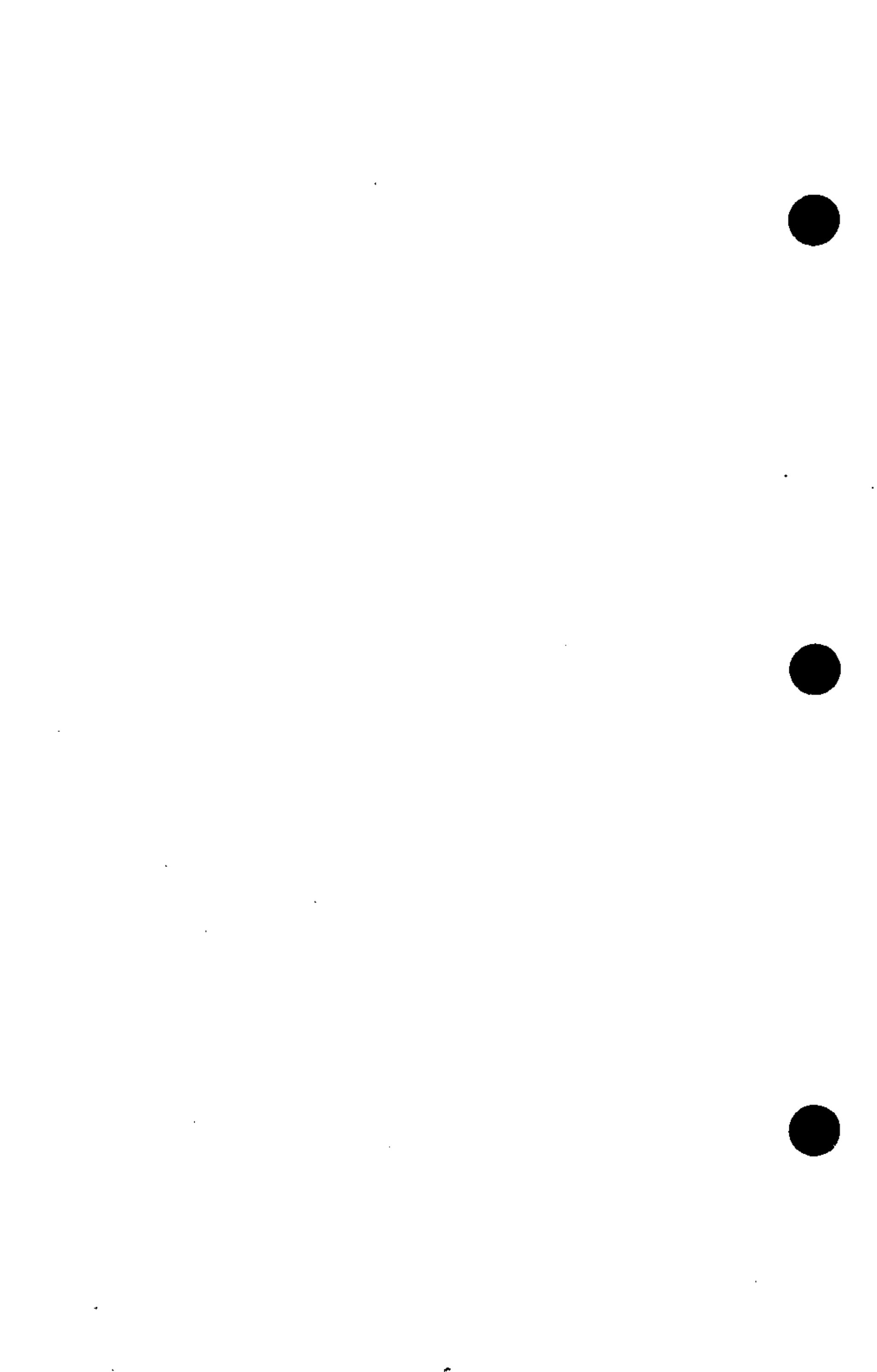
c) "Gate did not occur"
Specified gate did not occur.

d) "Trigger did not occur", "Trig W1 did not occur", or "Trig W2 did not occur".
Specified trigger is not present.



Unable to Test Screen

This typically indicates faulty activity from a circuit feeding this device.

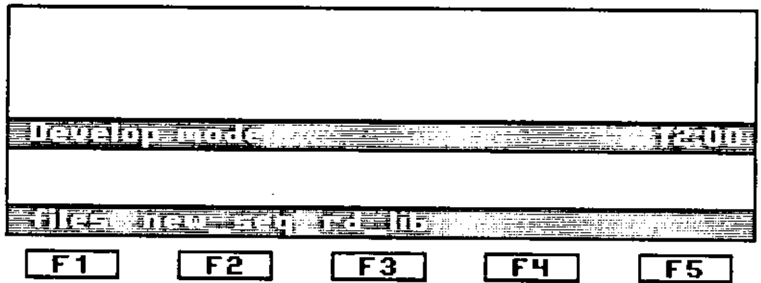


5 DEVELOP MODE

Creating Sequences

5.1 DEVELOP SUBLEVEL

The FLUKE 900 is its own complete programming station. All steps can be done from the keyboard in an interactive verification of a good board that is similar to Manual Mode testing. It is also possible, and sometimes easier, to perform Sequence file editing on a PC and download back to the tester. There are three sublevels accessed by pressing <<develop>>.



Develop Sublevel Screen

<<new_seq>> is used to initially create a new sequence. Files and their structure are transparent in this mode and keystroke entry is similar to Manual Mode. The resulting file has a .nsq extension on its name.

<<files>> is used to edit, enhance, complile and transfer sequences and data. During sequence creation, files that may be edited will have .SEQ, .LOC or .LIB extensions to their name.

<<rd_lib>> is used to create custom libraries for device testing in the ZIF socket. This process is covered in detail in Section 7.

5.2 OVERVIEW OF SEQUENCE DEVELOPMENT METHOD

To develop a sequence, the following steps should be taken:

- Set up repeatable comprehensive activity on the board to be tested. Use Reset whenever possible, either on the board under test or on another controlling board. Alternatively, arrange to manually cycle the diagnostics and use the External Trigger line to define the start of test.

- Enter New_Seq Mode and verify that each device passes comparison testing. Make adjustments in the local and global parameters to achieve this.

- Reorder the device locations to structure the trouble-shooting, for example by signal flow. This can be done with keystrokes in New_Seq Mode or the file listing can be rearranged with the screen editor. A PC editor is another possibility.

- Edit operator prompt messages into the file.

- Verify that all devices pass on other good boards.

- Document the Sequence, reasons for parameter changes and the test setup details. Retain a master copy of the files on cartridge or disk.

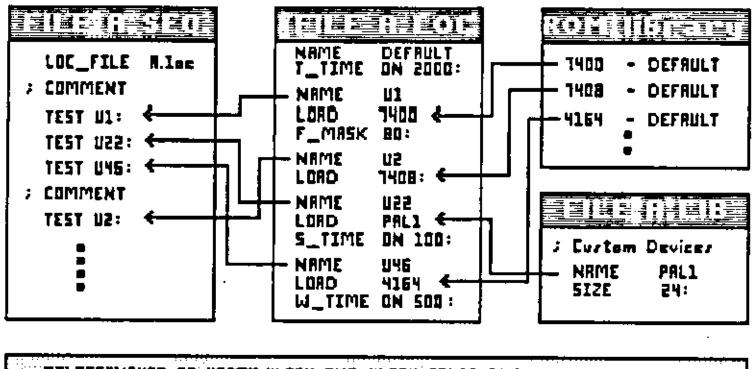
5.3 FILE INTERELATION

The interim file created as the first step of a Sequence is denoted ".nsq" (new sequence) and may not be directly edited or run as a troubleshooting sequence. When new_seq mode is ended and/or files are created through the editor, a set of interlocking files constitute the completed sequence. They are ".seq" (sequence), ".loc" (location) and ".lib" (library) files. There are two versions of these files:

- Source, which may be viewed and edited. They have upper case extensions (SEQ, LOC, LIB).

- Compiled, which are executable and are not editable. They have lower case extensions (seq, loc, lib).

The diagram on the following page shows how the file types are related.



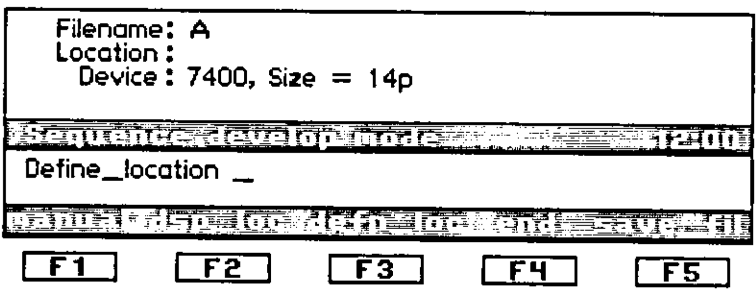
File Interelation

The ".SEQ" file establishes a troubleshooting order, the ".LOC" file is a database of parameter changes and ".LIB" files are a set of standard device parameters. Additional file extensions which are unrelated to Sequences are ".LOG" for test result data and ".LST" for printable directory listings.

5.4 NEW SEQUENCE MODE

Press <<new_seq>> and follow the prompt to name the Sequence. For an existing Sequence an alert is shown that you are appending or replacing a Sequence. On the New_Seq menu, <<manual>> has the same function as described in Section 3.

Proceed by pressing <<manual>>, entering a device by its number or size and changing any necessary local or global parameters. After ensuring that it passes, assign its board location. Press <ESC> to come back to the New_Seq menu, then <<defn_loc>> and enter the location designator.

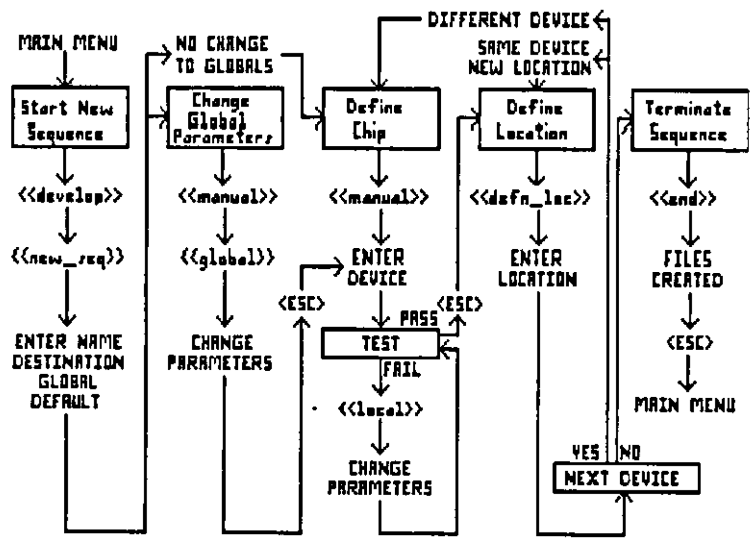


New Sequence Development

Note that you may assign more locations to the same device and parameter set. They must, of course, each pass a verification. An example of this could be a bank of RAM's. You may display a listing at any time of the device locations done so far by pressing <<dsp_loc>>.

It is not possible to step through the locations with <NEXT> as in Sequence Mode, but you can directly bring up a location's device and parameters by entering it (eg. U48 <ENTER>).

The complete new sequence procedure is shown on the following page:



Flow Diagram of New-Seq Procedure

Each device is defined along with any required parameter changes before its location on the board is defined. As this is done device by device, a new_seq file (.nsq extension) is created in system RAM. <<save_fil>> saves the ".nsq" file to the destination specified when opening the new_seq file (typically cartridge). It is good practice to do a "save file" frequently in case of a power failure.

New_seq mode may be exitted with <ESC> and re-entered to resume working on a ".nsq" file provided it was saved. Note that the device listing shown by pressing <<dsp_loc>> contains the word default, which is the "dummy" location for global parameters. Those devices listed above default were entered before the last "save file" was executed.

When new_seq verification of the board is complete, pressing <<end>> will automatically create ".SEQ" and ".LOC" files and their compiled versions, ".seq" and ".loc" of the ".nsq" file. These are true Sequence files that may be listed and edited. If you re-open a new_seq ".nsq" file and <<end>> again, you will be prompted to confirm that the previous versions will be overwritten. If you wish to retain the original files, you must interchange cartridges at this point (or rename a copy of the file you wish to retain before re-opening the new_seq file for changes).

The Sequence created may be run in Sequence Mode, but the device locations are in the order in which they were entered. A way to reorder it without invoking the editor is to do it in New_Seq Mode. This involves first opening the original New_Seq by name, entering the first desired location and redefining it at that same location. Once you have reselected every device, press <<end>> and the Sequence will be in the new order.

5.5 SEQUENCE ENHANCEMENT

After developing an initial sequence with correct devices, locations and test parameters, further enhancement is done by editing the files. The .nsq file will not reflect any edited changes so it is no longer the working master. Note that re-opening a .nsq file and ending it will regenerate .SEQ and .LOC files. Care should be taken that this does not overwrite the edited master versions.

Sequence enhancement uses the utilities found under the files level of Develop Mode (Section 6) and the commands detailed in Appendix II. The following subsections preview file syntax and some advanced commands that are not accessable in New_Seq mode.

5.5.1 SOURCE FILE SYNTAX

- Each line is a command. Blank lines have no effect, but make a listing more readable. Commands are grouped together by device or location to form a command group.

- Command groups end with a colon (:)

- Command fields are separated by at least one space. The Tab key is often used to improve readability.

- Maximum line length is 254 characters.

- Labels start with a letter and are alphanumeric (including underline character).

- A semicolon (;) denotes a comment in the file listing.

To view a file on the screen, press <<edit>> and enter the file name, type and source on the command line (e.g. A.SEQ:CART).

Example:

```
            LOC_FILE A;this comment has no effect
            TEST U1:
            TEST U22:
LABEL       TEST U46:
```

Scrolling through a listing on the screen is done with the up and down arrow keys. "Shift down arrow" scrolls a full page and "Cntr down arrow" scrolls to the end of the file.

5.5.2 DISPLAY

The DISPLAY command puts a message, typically an operator prompt, on the LCD screen.

Syntax: DISPLAY [line #] {"message"}

Parameters: line# may be 1, 2, 3, 4 or nothing.

5.5.3 FUNCTION/END_FUNCTION

When a series of commands are needed in several places in a sequence they can be grouped together into a FUNCTION and called as a sub-routine from another comand group. The TEST command and the ":" cannot be part of the function. Functions are ideal for parameter settings that are used with more than one DUT, such as an External Trigger or an operator prompt that needs to be repeated in more than one place. After a FUNCTION has been called and executed, control is returned to the command following the call.

Syntax: FUNCTION {name}

```
name command_1
     :
     command_n
     END_FUNCTION
```

A FUNCTION is defined by a label (name) and a command group ending with an END_FUNCTION statement. It is called by using the command FUNCTION followed by its label name.

```
Example:   TEST        U5:
           FUNCTION    MSG-1
           TEST        U7:


     MSG-1 DISPLAY     "Run selftest 2"
           END_FUNCTION:
```

5.5.4 JUMP

This will cause the sequence to continue execution at the specified label. A valid label starts in column 1 of the source file.

```
Syntax:    JUMP        {label}
```

5.5.5 IF {condition} THEN {command}

This command is used to test if the specified set of conditions are "true" and, if so, the system will execute the command defined by THEN. Otherwise , the program will continue executing at the next command.

Syntax: IF {condition} [operator {condition}]...THEN {command}

Parameters:

condition	PASSED	- DUT passed
	FAILED	- DUT failed
	RD_SHORTED	- Short in ZIF
	RD_TEST_FAILED	- Bad RD
	CLIP_TEST_FAILED	- Clip Check fail
	FAILED_TO_SYNC	- Unable to sync RD/DUT
	NO_GATE	- Gate word didn't occur
	NO_TRIGGER	- Trigger word(s) missing
	NO_TRIGGER_WORD1	
	NO_TRIGGER_WORD2	
operator	+ ... OR	
	* ... AND	
	< ... freq less than value	
	> ... freq greater than value	
	= ... freq equals value	
	<>... freq not equal to value	

Example 1

IF PASSED THEN DISPLAY "BOARD IS GOOD"

Example 2

IF FAILED + P4=A * P1< 5.0MHz THEN JUMP LABEL

Note that units for frequency must be spelled Hz, kHz, MHz.

5.5.6 SEQUENCE FILE LISTING

```
        LOC_FILE    BOARD1
        DISPLAY 1   "CONNECT RESET TO J25"
        TEST        U2:
        TEST        U6:
        TEST        U7:
        FUNCTION    MSG
        TEST        U4:
        IF P4 <22.95 MHz THEN JUMP LOC5:
        END:
   LOC5 FUNCTION    MSG
        TEST        U5:
        END:

    MSG DISPLAY     "DO NOT USE RD"
        END_FUNCTION
```

NOTE: This sequence directs the operator first to U2, U4, U6, U7, and if it fails, to U5. SEQ files created under NEW_SEQ do not show an "END" statement. The first statement specifies the LOC file.

5.5.7 LOCATION FILE LISTING

```
NAME          U2
LOAD          7400
F_MASK        80:

NAME          U4
SIZE          16
RDT_ENABLE    OFF
ACTIVITY      P4=19.95MHz 1%:

NAME          U5
SIZE          20
RDT_ENABLE    OFF
DISPLAY       "OSCILLATOR CHIP"
ACTIVITY      P1=A P2=H P14=19.95MHz 1%:

NAME          U6
LOAD          S17146B
GATE          ON P1=L
DISPLAY       "TESTING IN ONE DIRECTION":

NAME          U7
LOAD          DEMO_PAL:

NAME          DEFAULT
F_MASK        40
T_TIME        3000
RESET         ON NEG INT DUR=200 OFS=0:
```

NOTE: Global parameters are in a command group named "DEFAULT".

5.5.8 USER LIBRARY FILE LISTING

```
NAME        DEMO_PAL
LOAD        PAL20RA10
S_TIME      ON 9000
C_SUM       16170:

NAME        S17146B
LOAD        74245:
```

NOTE: The example illustrates defining a custom device and a simple part number cross reference.

5.6 SEQUENCE CREATION GUIDELINES

The following eight steps provide a detailed procedure that is recommended for the creation of effective board test Sequences.

5.6.1 SEQUENCING PREPARATION

Decide the quality of diagnostic testing desired, the degree of automation to be included and how much time is available for sequence creation. A 100 IC board will generally take from two to four days to complete the entire sequencing process. Assess the device coverage, percentage of analog components on the board and any areas that exceed the tester speed specifications.

Collect Reference Devices, schematics and any required fixtures including loopback plugs, I/O connectors and extender cards. Arrange RD's in ascending numerical order in an RD retainer tray. One possibility is to have a unique tray for each board type. Another is to have one or two trays that have all devices for a range of boards.

Ensure that clip contacts are clean. Check oxidation on device pins and clean with alcohol if necessary to avoid intermittent contact problems.

Check that +5 volt power supply is good on the board and not at a marginal level that may cause inconsistent results.

5.6.2 DIAGNOSTIC STIMULUS

Set up a reliable repeatable stimulus, ensuring that the activity is as extensive as possible. Use Reset wherever possible and verify that it really does initialize the board. Look for a signal that comes from the power on function, such as a capacitor to Vcc connected to the microprocessor reset line. On a single board system this is usually straightforward; on a multiple card system, the reset may even be located on another board.

5.6.3 VERIFYING THAT ALL DEVICES PASS DRC

Verify in NEW_SEQ Mode that each device on the board passes and, if not, make the required parameter changes. Refer to Section 2.7 and Appendix III for advice on solving problems. Note that since FMASK is set per device, not per pin, it is advisable to test twice when only one pin requires a large setting (eg. 150 ns). Test once at 150 and a second time at 30 ns, with the offending pin ignored.

On devices that pass, pressing <<results>>, <<state>>, to highlight active pins will give an indication of how well a device is stimulated. Two convenient ways of performing the NEW_SEQ procedure are:

1. Verify each device chip by chip, row by row.
2. Verify each device in numerical order (eg. 7400,7402...) placing a sticker on each IC to keep track of those done.

5.6.4 ADDING SIGNAL CONDITION TESTS

In general, the more you know about a board, the more conditions you can test for. For example, activity checks and triggers are most useful in a sequence structured by signal flow. Suitable conditions in order of increasing knowledge of the board operation are:

- Fixed frequencies (ungated, periodic)
- System clocks
- Activity presence
- Constant levels
- Triggers and Gates

Conditions should, in general, be assigned only to input pins since outputs are already checked through DRC.

Enable H and L status checks for pins that are pulled up or down. Frequencies are measured on fixed, ungated periodic signals. Active signals coming from off the board are checked.

5.6.5 STRUCTURING THE SEQUENCE

The next major step in Sequence creation is to reorder the devices for troubleshooting. This may be done by redefining all locations in NEW_SEQ mode as described in Section 5.4, or creating a new .SEQ file with the editor or via the download utility from a PC generated file. The order could be by RD type, fault occurence based on past experience or signal flow. It is advised that the order be by signal flow, since occasionally, failure modes induce apparent faults further down the chain. As an example, failing memory addressing logic can cause all memory devices to appear bad.

A typical micro-based board would have the following recommended Sequence order:

- reset circuit
- clock, timing generation circuit
- bus control, address decoding
- address bus drivers and logic
- databus drivers and logic
- memory devices
- I/O, remaining circuitry

To test boards controlled by other boards, start with the card edge interface.

If the board diagnostics are effective in first isolating a failure to a part of the board, several small Sequences appropriate to each part could be a good implementation. These smaller .SEQ files will share the same .LOC file for the whole board.

5.6.6 ADDING OPERATOR SCREEN MESSAGES

Employ the resident editor to add operator prompt messages to the .SEQ file. For extensive messages, this can be done also by uploading to a PC, using a full screen editor, and downloading back to the system. The added messages could be advice on connecting the patch leads, the type of RD or troubleshooting hints. Function calls are a convenient way to add duplicate messages (see Section 5.5.3).

5.6.7 COMPILING THE FILES

All edited files must be recompiled into executable versions. Press <<compile>> and enter the filenames with extensions. Do this for all edited .SEQ, .LOC and .LIB files.

5.6.8 FINAL VERIFICATION AND DOCUMENTATION

Verify the completed sequence by running it on other good boards. Edit any parameter changes (usually FMASK in the .LOC file) and recompile.

One way to allow for a range in performance among several good boards is to create a NEW_SEQ temporary (.nsq) file with the default FMASK of 30ns. After creating a .LOC file, go into it and edit a global FMASK of 40 ns to give an extra 10 ns margin in comparison resolution.

Document fully the newly created sequence including:

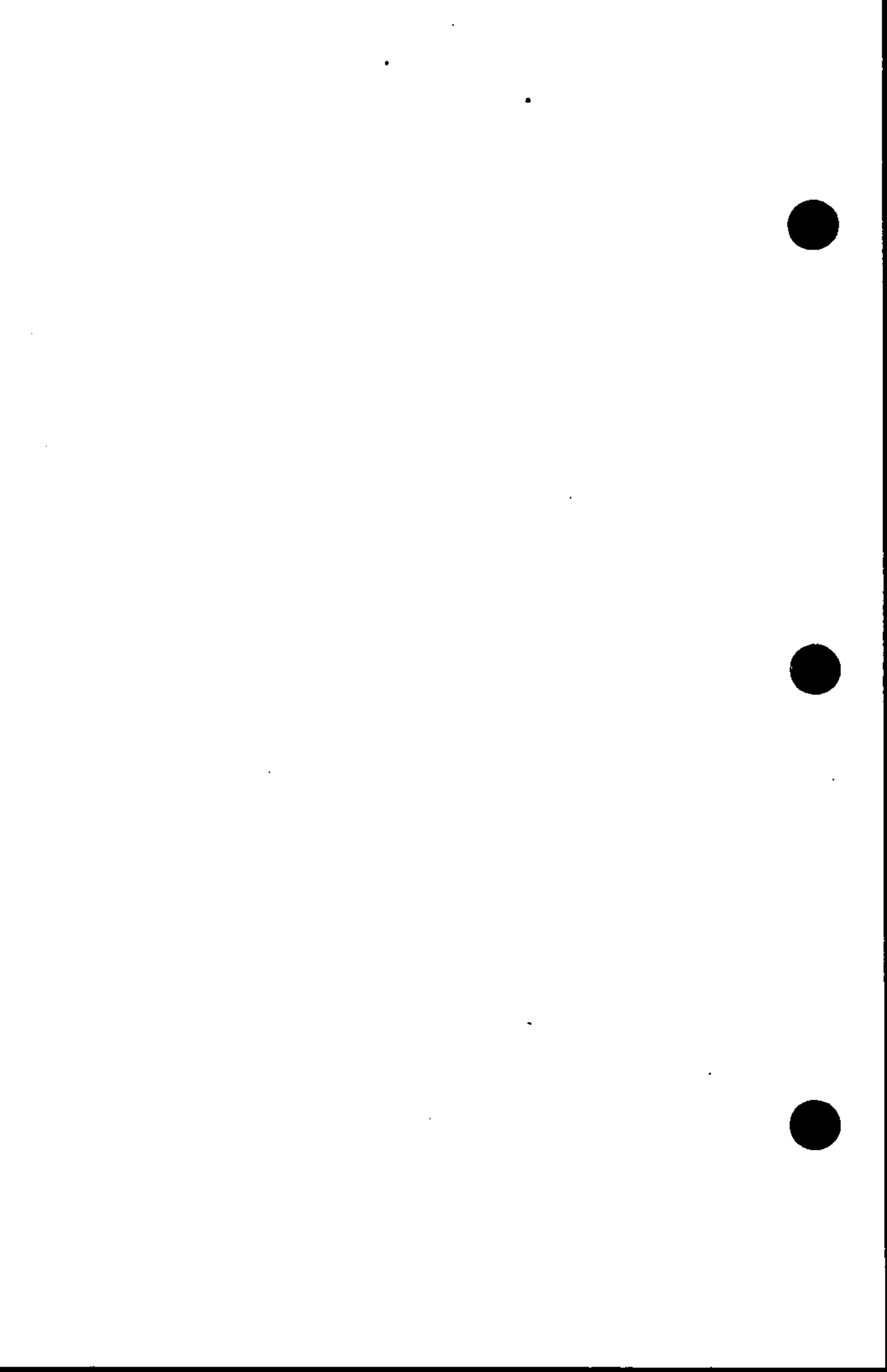
- printout listings of the files
- comment sheet on why parameters were changed
- date, revision level, description of diagnostics
- board statistics: number of ICs, number untested, percentage covered.

It is good practice to retain a master copy of all source files on a diskette or cartridge and file it away for safekeeping.

A good rule for file management is to keep a master cartridge (or floppy disk) containing source and temporary (.nsq) files so they may be modified. Run-only compiled files covering a number of board types may be put on another cartridge. They occupy less space, cannot be readily modified and may be distributed safely to the test operators.

Typical file sizes for a 100 device board that has operator prompts in the .SEQ file are:

FILE.LOC	5000
FILE.SEQ	5000
FILE.loc	1500
FILE.seq	1500
FILE.nsq	3000
Total	16000 bytes



6 FILE UTILITIES

The file utilities manipulate file data stored in either of two sources: the data cartridge (:CART) and system RAM (:SYST). Available utilities in the order they appear on the menu are: DIRECTORY, EDIT, COMPILE, PRINT, COPY, UPLOAD, DOWNLOAD, RECOVER, CARTRIDGE.

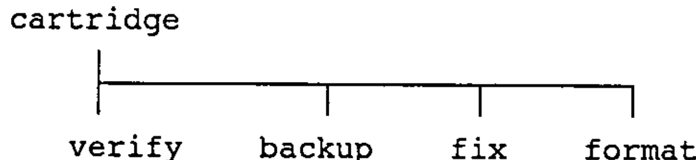
File names may be up to 15 characters in length, must start with an alpha character and may contain only alphanumeric characters including underscore (_). A complete file name specification includes the name, a type extension and a source/destination designator (e.g. FILE_1 .SEQ :CART). In many cases, it is not necessary to specify the complete name since the assumed default is appropriate. Defaults are listed with each utility.

Valid file types are: .SEQ, .LOC, .LIB, .LOG, .LST, .seq, .loc, .lib and .nsq. The lower case types are generally absolute files created by the machine and may be transferred but not edited or printed by the user. The key <<.type>> permits single key specification of upper case file type with the function keys. Note that pressing <<.SEQ>> while pressing <SHIFT> generates a lower case .seq.

In the syntax descriptions that follow, parameters enclosed in { } brackets are mandatory, those in [ ] brackets are optional.

6.1 CARTRIDGE <<cartridge>>

Menu



Description

The cartridge utilities do not require specification of filenames as they are executed on the entire cartridge contents.

Format must be executed on a new cartridge before file operations are possible. This function can also be used to completely erase the contents of a cartridge. A user verification prompt appears on the screen before proceeding with a format operation.

Backup is a quick duplication of an entire cartridge. The user is prompted to insert a source cartridge, the contents are loaded into system RAM and a blank cartridge is inserted and written into.

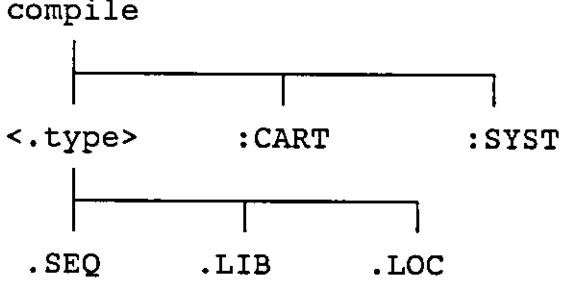
Verify performs a check that all files on a cartridge are uncorrupted and it sets the required index flags to permit further file handling. The most likely cause of a failed verification is powering down the machine without properly ending a file creation procedure. The write-

must be off to perform a verification.
Note that corrupted files are identified during the directory function, and verify would then be executed.

Fix will close any open file found on cartridge, normally caused by powering down while a file remains open.

6.2 COMPILE <<compile>>

Menu



Syntax

Compile {filename}[upper case type][:source/destination]

Default

{filename}.SEQ:CART

Examples

Compile FILE1.SEQ
Compile FILE1.LOC
Compile FILE1.LIB

Description

The compile function first prompts for entry of a source file name and type (SEQ, LOC, LIB) on the command line along with source :CART or :SYST. After <ENTER> is pressed, a compressed executable version is created (seq, loc, lib) and deposited in the same :CART or :SYST medium as the original source version. Existing compiled files with the same name are overwritten. If a syntax error is encountered during the compile, it aborts and displays the line number of the error within the source file. Syntax requirements for the compiler are described in Appendix II.

Compiler error messages include:

"Duplicate symbol XXXX" : the label XXXX is found on more than one line

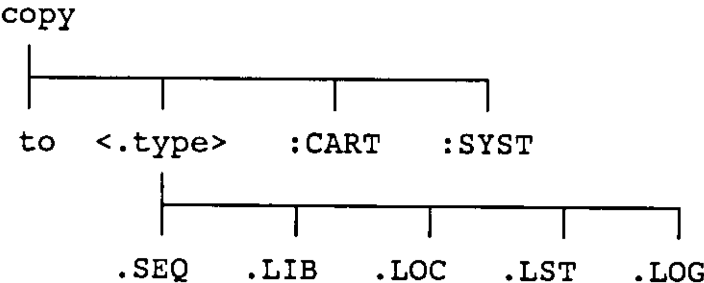
"Undefined symbol XXXX" : the label XXXX was referenced by does not exist

"No defaults in file" : no default chip to define globals

".loc file not designated" : the first line in a .seq file specifying the .loc file is improper

6.3 COPY <<copy>>

Menu



Syntax

Copy {filename}[.type][:source]
to {filename}[.type][:destination]

Default

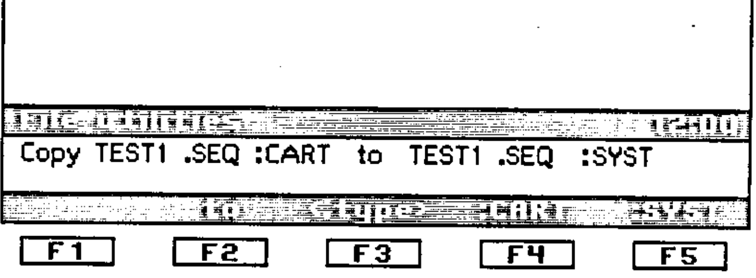
{filename}.SEQ:CART to {filename}.SAMETYPE:CART

Examples

Copy FILE1 to FILE1:SYST
Note: this copies all types of FILE1
Copy FILE1.SEQ:CART to FILE2
Note: this duplicates a file on cartridge
Copy :CART to :SYST
Note: this transfers cartridge contents to system RAM

Description

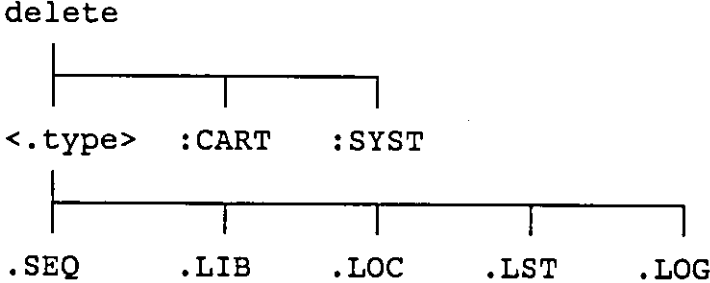
The copy function is used to transfer files between :CART and :SYST or copy to a renamed duplicate file within a single storage medium. This may be done for all file types. In general, copying is usually done between like file types. In particular, a lower case file extension musts be copied to its same type. The following entry on the command line will transfer a cartridge-resident SEQ file to system RAM.



The destination file may be renamed during this process (i.e. TEST1 to TEST2). If a type extension is not specified, all file types with the primary name are transferred. When copying a Sequence to a second cartridge, the files must first be transferred to system RAM from the original cartridge, the new cartridge inserted and a file transfer made from :SYST back out to :CART. Refer to Section 6.1 for copying the entire cartridge contents using the backup utility.

6.4 DELETE <<delete>>

Menu



Syntax

Delete {filenames}[.type][:source]

Default

Delete {filename}.SEQ:CART

Examples

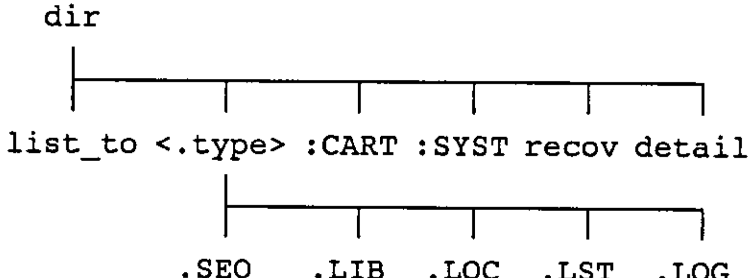
Delete FILE1.SEQ:CART
Delete FILE1
Note: this deletes all types of specified files on cartridge
Delete FILE1:SYST
Note: this does not affect FILE1 on cartridge

Description

The Delete function removes a filename from the cartridge directory, effectively freeing up the memory space for other uses. Deletions may also be made from system RAM. If no extensions are specified, all file types of the filename are deleted.

6.5 DIRECTORY <<dir>>

Menu



Syntax

Directory [:source][filename][.type][recoverable][detailed]

Default

All files.all types:CART

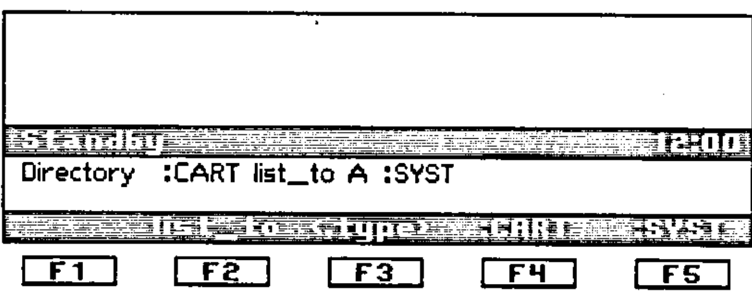
Examples

Directory .SEQ:CART/detailed
Directory :CART/recoverable
Directory fileA
Directory :SYST

Description

The Directory function shows, on the screen, the names of files in cartridge or system RAM, along with the space they occupy in memory. The final line shows the total space used and available. As the names are scrolling through the screen, <<pause>> will alternately freeze and resume the display. <ESC> will abort the directory function.

The machine will provide a limited directory of files by type (.SEQ, .LOC, .LIB, .LOG and .LST), by source (:CART, :SYST) and by name. A special file type known as a "list file" with extension .LST is reserved for a listing of the directory itself. Once a directory listing is assigned to a list file, it may be printed out on hardcopy like any other file. As an example, the following entry on the command line causes a file, A.LST, to be created in system RAM which contains a listing of the directory of the current cartridge.

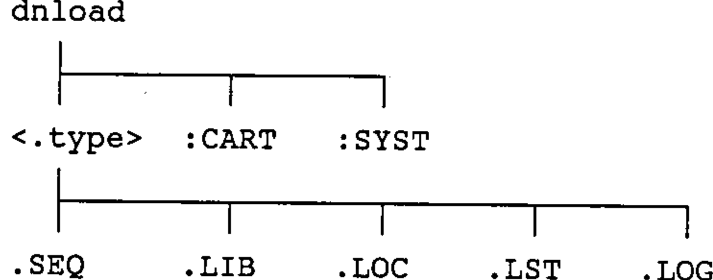


Creating a File of Directory Listing

The second menu level, accessed with <ETC> provides two more options. <<recov>> lists previously deleted files that may still be recovered. <<detail>> is a modifier to the directory function that provides the time and date that the file was last modified.

6.6 DOWNLOAD <<dnload>>

Menu



Syntax

Download to {filename}[.type][:destination]

Default

Download to {filename}.SEQ:CART

Examples

Download to FILE1.LOC:CART
Download to FILE1.LOC:SYST

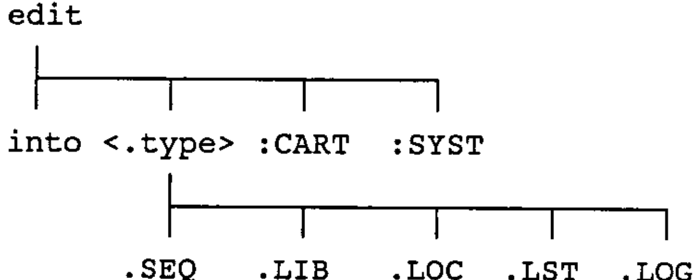
DOWNLOAD

Description

Download routes received serial data to a specified file. It may be terminated by receipt of a SUB, ETX or EOT character or by manually pressing <ESC>. Unrecognized codes are converted to "I" meaning "Illegal" for screen display. If a download operation exceeds available filespace, the file is closed and the operation aborted.

6.7 EDIT <<edit>>

Menu



Syntax

Edit [into]{filename}[.type][:source/destination]

Default

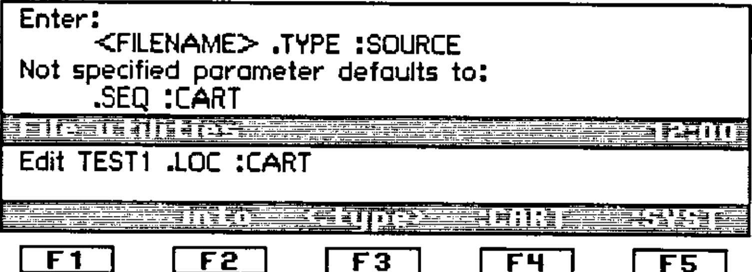
{filename}.SEQ:CART

Examples

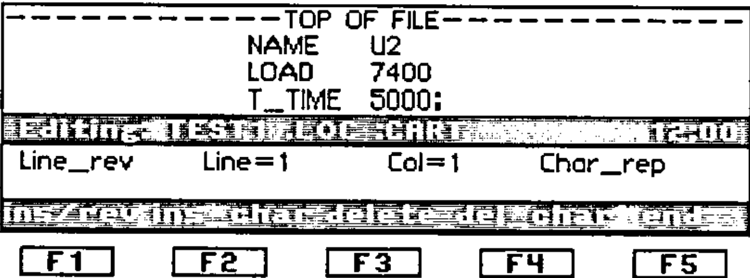
Edit into FILE1.SEQ:SYST
Edit FILE2
Edit FILE_3.LOC

Description

Any file with an upper case extension may be modified using the editor. The filename, type and source are entered on the command line.



Note that when creating a new file you must edit into a filename. For an existing file, you can merely edit a filename. This syntax permits you to keep a backup file of the original file you are changing (i.e. Edit A into B will define B as an edited version of A while retaining A).



The top four lines form a scrollable window to view the file contents. Maximum line width is 254 characters and file length is limited only by available memory space. Note that because an edited file is first stored to the specified destination before the original is deleted, a certain amount of free memory space must be available to end the edit mode after a change in file contents. For :CART this is typically double the file size; for :SYST it is triple. A warning appears if there is insufficient space.

The cursor may be moved with the arrow keys, one space at a time. The tab arrow key moves the cursor forward 8 columns (or backwards with the SHIFT key). Use of the SHIFT key with the arrow keys permits scrolling up or down by a 4 line page and immediate cursor positioning at the beginning or end of a line. Use of the Cntr key with the up and down arrow keys permits immediate scrolling to the top or bottom of a file. The current cursor line and column are displayed on the first command line.

F1 key, <<ins/rev>>, alternately specifies line insert or line revision as indicated on the command line. With line revision, characters will overwrite the existing display as they are typed. With line insert, pressing <ENTER> will insert a blank line and allow new data to be typed in.

F2 key, <<ins_char>>, alternately specifies replacing or inserting a new character at the cursor position as indicated on the command line. In replacement mode the cursor is an underline, while for insert mode it is a block.

F3 key, <<delete>, deletes the line the cursor is on.

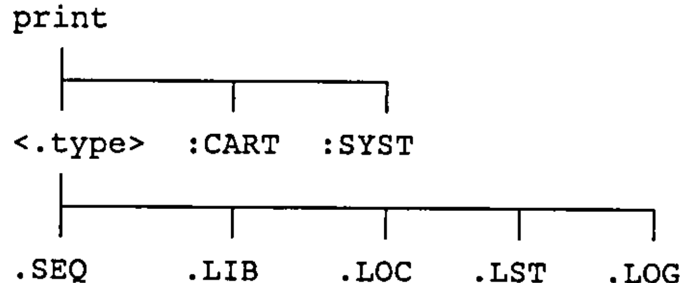
F4 key, <<del_char>>, deletes the character at the cursor position and moves the remainder of the line to the left.

F5 key, <<end>>, exits the editor and updates the changes. Note that <ESC> will also exit the editor after a confirmation, but will not update the file.

<CE> performs a backspace function, deleting the character before the cursor, shifting the remainder to the left and deleting the remainder after the left margin is reached.

6.8 PRINT <<print>>

Menu



Syntax

Print {filenames}[.type][:source]

Default

{filename}.SEQ:CART

Examples

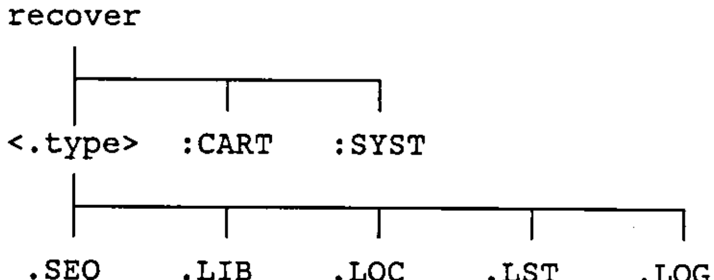
Print FILE1
Print FILE2.LOC:SYST

Description

The print function sends an ASCII data stream to the serial port for a printer to provide a hardcopy file listing. Any file with an upper case extension may be printed. Note that a LOG file may also be printed from <<prt_log>> in Manual or Sequence mode. The serial communication attributes are set using <<rs232c>> in System mode and the printout formatting is established using <<printer>> in System mode.

6.9 RECOVER <<recover>>

Menu



Syntax

Recover {filename}[.type][:source]

Default

Recover {filename}.SEQ:CART

Examples

Recover FILE1.SEQ:CART
Recover FILE1.seq:CART
Recover FILE1
Note: this recovers all types of the specified file.

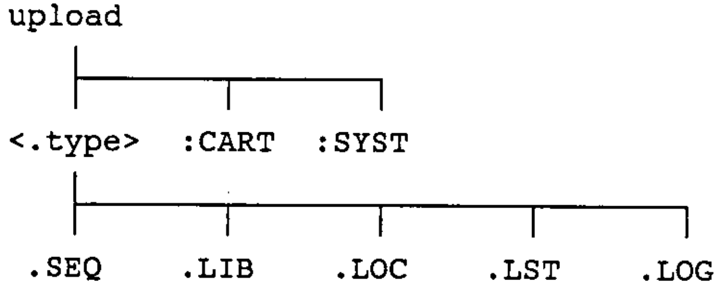
RECOVER

Description

The Recover function restores a deleted file to the directory if it has not been overwritten. It is used when a file is deleted by mistake. A list of recoverable files may be viewed with the directory function. Note that Recover should be performed as soon as possible since other storage into the cartridge could overwrite a deleted file and make it unrecoverable.

6.10 UPLOAD <<upload>>

Menu



Syntax

Upload {filename}[.type][:source]

Default

{filename}.SEQ:CART

Examples

Upload FILE1
Upload FILE1.LOC:SYST
Upload FILE1.loc:CART

UPLOAD

Description

Upload takes a specified file and sends it as a serial data stream out the communication port, following the attributes set with <<rs232c>> in System Mode. Transmission is terminated by sending three additional characters: SUB, ETX, EOT (CntrlZ, CntrlC, CntrlD). Source files are sent as ASCII characters with each line terminated by CR, LF. Lower case file types are sent as ASCII, HEX character pairs with space separators. Note that because of data compression techniques, uploaded files will be larger than they appear according to the storage space they occupy on :CART or :SYST.

7 DEVICE LIBRARIES

The device library is a database containing information for in and out of circuit testing. It is comprised of a ROM-resident file containing a large number of standard devices and expandable user files resident in system RAM or cartridge. The ROM library contains test patterns for verifying the functional integrity of reference devices in the ZIF socket. The revision level of the library is indicated on the main power up screen.

The FLUKE 900 is capable of verifying the functional integrity of reference devices in the ZIF socket by exercising them with preprogrammed test vectors (RD Test). These are stored in a library ROM for a large number of standard devices. The revision level of the library is indicated on the main power up screen. Additional devices may be added by the user to a library file.

7.1 LIBRARY FORMAT

The library files appear similar to .LOC files that were described earlier. The first command in a command group is NAME followed by the device number designator. The next command either specifies the device size or references an equivalent device that is already contained in the library.

Examples:

```
NAME     7400X
SIZE       14:

NAME     S7777
LOAD     7400:
```

Note that the first example defines a 14 pin device known as 7400X, unrelated to a 7400. The second example defines a device known as S7777 to be equivalent to a 7400.

Test parameters may also be added to the command group to account for characteristics such as drive capability (RD_DRV), parameters and RD Test specification. Note that parameters specified in a .LIB file should be general for all applications while those in a .LOC file are unique to a particular circuit.

RD test information is listed to verify the device in the socket. For PROM'S and PAL's, a C_SUM is calculated and verified. About a hundred cells throughout the PROM address range are combined for its C_SUM (enough to characterize, but not fully evaluate it). A standard pattern is

applied to a PAL. For all combinatorial and many sequential PAL's, this generates a single characteristic C_SUM. For more complex PAL's, a unique C_SUM is not obtainable by this method. The only recourse would be to write specific vectors based on a knowledge of the PAL, as described later.

Example of custom PAL library with C_SUM:

```
NAME     PAL1
LOAD     PAL20L10
C_SUM    11781:
```

The creation of RD test vector patterns is covered in Section 7.3.3. As a final general word on libraries, when a user enters a device type number, the system first checks :CART resident .LIB files, then :SYST resident .LIB files, before checking the ROM resident library. Thus, device numbers duplicated on both ROM and cartridge will be chosen from cartridge, effectively ignoring the ROM library.

7.2 SPECIFYING RD/DUT SYNCHRONIZATION

For most devices other than combinatorial and RAM chips, it is necessary to define the synchronization requirements between RD and DUT. This is because the two devices must first be in synchronism before a PASS/FAIL determination can be made. To ensure synchronization, we must be able to observe or infer all the internal states of each device. For example, an octal latch can be checked for synchronization by merely observing its outputs when they are enabled. However, an 8 bit FIFO shift register would require 8 pulses on its serial-shift-in line before we can infer the internal hidden states.

One of the failure modes of a DUT can be that it is in an illegal state or simply cannot be synchronized. Every attempt is made, therefore, to synchronize RD and DUT before a PASS/FAIL test result is presented. Under certain rare board conditions, the DUT is not compared because of an inability to synchronize (Unable to Synchronize result). Two methods are employed during the S_Time to synchronize RD and DUT:

1. Monitor DUT for activity that guarantees synchronism.
2. Stimulate RD until it "catches up" to a static DUT.

The synchronization algorithm during S_Time is as follows:

SYNCHRONIZATION ALGORITHM

```
     |
     v
+--------------+
| Wait for     | Occurrence
| First Sync   |---------------------------------------+
| Condition    |                                       |
+--------------+                                       |
     |  No Occurrence                                  |
   More                                                |
Conditions                                             |
     |                                                 |
+--+ | +--------------------------+                    |
|  v v v                          |                    |
| +------------------+            | No                 v
| | Apply Vector     |      +------------+      +-------------+
| | to RD From:      |      | Are DUT    |      | Proceed     |
| | SYNCVECT LIST    |----->| and RD     |----->| with DRC    |
| | SYNC_PAT LIST    |      | in sync    |      | TEST        |
| | SYNC_RND LIST    |      +------------+      | (PASS or    |
| +------------------+                    +---->| FAIL)       |
|        |  No More                       |     +-------------+
|        v  Vectors                       |
| +----------------+                      |
| | Concurrent     | No                   |
| | DUT activity   |----------------------+
| +----------------+
|        |
|        v Yes                                +-------------+
|   +--------------+ Yes                      | Unable      |
+---| S-Time       |--------------------------->| to          |
 No | Expired?     |                          | Synchro-    |
    +--------------+                          | nize        |
                                              +-------------+
```

As can be seen from the diagram, synchronism is first attempted with a series of Sync Conditions. These are chosen as the necessary and sufficient conditions that would guarantee the RD and DUT would be in the same state. For example, two 7474 D flip flops would be in the same state after logic 0 on the clear or preset lines or after a rising edge on the clock lines. If any of these occur, the devices are assumed to be synchronized and comparison testing is enabled.

For devices with internal hidden memory states, Sync Conditions are the only effective technique for synchronizing and care must be taken to ensure that they are specified completely to avoid falsely assuming synchronization.

If these conditions do not occur, the second method is tried, stimulating the RD. The stimulus vectors may be defined explicitly (SYNC_VECT), chosen from the RD Test pattern (SYNC_PAT), or created randomly (SYNC_RND). If no conditions or vectors are specified, SYNC_RND is automatically chosen for the duration of S_Time (beware of false synchronization if there are hidden states).

For our 7474 example, a suitable SYNC_VECT would be a pulse on both clock lines. SYNC_PAT can be an effective synchronization definition, but it should only be used if the RD Test pattern completely exercises a device. SYNC_RND is the easiest to specify although it is less effective.

After each vector is applied, RD and DUT are checked to see that their outputs are the same and, if so, DRC testing proceeds. For tristatable devices, a further parameter,

SYNC_GATE, defines when it is valid to check for synchronism on an output pin. This is usually a logic level on the output enable pin.

When defining the synchronizing parameters for a new library device, it is best to ask yourself two questions:

1. What activity could initialize a device into a known state? The Sync Conditions should be specified accordingly.
2. How might a device be stimulated to cycle it through all possible states, or given a known state of DUT, how would an RD be put into the same state? The vectors should be specified accordingly.

For programmable devices that have hidden states, method 1 should be used alone without method 2.

The actual commands that define synchronization are:

S_TIME:	duration of synchronizing attempts
SYNC_COND:	defines conditions sufficient to synchronize from on-board activity
SYNC_VECT:	defines vectors to be sent to RD
SYNC_PAT:	specifies RD Test pattern instead of sync vectors
SYNC_RND:	specifies random pattern instead of sync vectors
SYNC_IGNORE:	specifies any pins that should be ignored while checking for synchronism
SYNC_PINS:	specifies the DUT output pins which should be inactive while RD vectors are applied during synchronization.

SYNC_GATE: defines how tristate outputs are enabled on DUT. Syntax is similar to GATE.

SYNC_RESET_OFF: disables the Reset pulse. Normally, a Reset is issued before checking for each sync condition.

SYNC_GR_END: identifies the end of the group of sync parameters.

Syntax for the preceding commands is found in Appendix II.

Example library definition of device with synchronization parameters:

```
NAME          7474
SIZE          14
SYNC_COND     <1 P1=L + P4=L + P3=R>
SYNC_VECT     1<1 H P3=C P2=D5>
SYNC_PINS     1 3 4
SYNC_GR_END:
```

NOTES:

1. For illustration purposes, only half of this dual flip flop has been specified.
2. A number before the first bracket specifies repetitions of the entire expression. A number after an opening bracket specifies repetition of the bracketed portion.
3. P#=R means check for a rising edge.
4. H alone means unspecified pins are driven high.
5. P#=C means provide a clock pulse of proper polarity.
6. P#=D# means apply a level to the RD pin that is the same as the specified DUT pin number.

EXAMPLE:

```
NAME            74374
SIZE            20
SYNC_COND       <1 P11=R>
SYNC_VECT       1<1 L P11=C P3=D2 P4=D5
                P7=D6 P8=D9 P13=D12
                P14=D15 P17=D16 P18=D19>
SYNC_GATE       P1=L
SYNC_PINS       11
SYNC_GR_END:
```

NOTES:

1. Sync Vectors are specified on one line in the editor.
2. A "Synchronization timeout" message appears if the S_TIME value is too low for the complete vector list to be executed. Possible causes include: S_Time too small, vector list too long, insufficient activity on SYNC_GATE.
3. The DUT clock (Pin 11) must be inactive while vectors are applied to RD (SYNC_PINS command).

EXAMPLE:

```
NAME            74161
SIZE            16
SYNC_COND       <1 P1=L+P9=L*P2=R>
SYNC_VECT       1 <1 H P9=L P7=L P10=L
                P2=C   P3=D14   P4=D13
                P5=D12 P6=D11>
SYNC_PINS       2 7 9 10
SYNC_GR_END:
```

NOTES:

1. The expression <A*B> means A and B must occur simultaneously. <A>*<B> means A and B must occur, independently, in any order.

EXAMPLE:

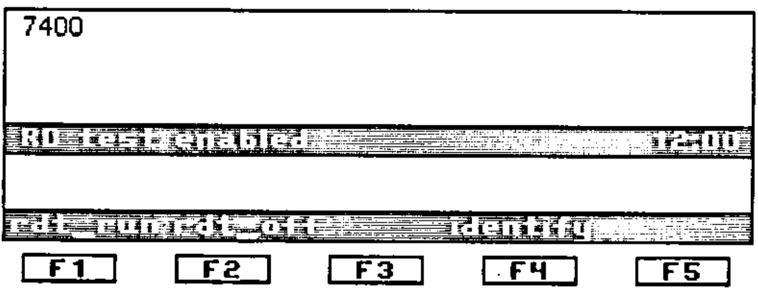
```
NAME            7490
SIZE            16
SYNC_COND       <1    P6=H*P7=H+P2=H*
                P3=H>
SYNC_VECT       2 <1 L P14=C P1=D1>/<5 L
                P1=C P14=D14>
SYNC_PINS       1 14
SYNC_GR_END:
```

NOTES:

1. Sync Vectors are applied to this Decade Counter (RD) when the DUT is inactive. Clock pin 14 receives a pulse followed by 5 pulses on clock pin 1. This is repeated twice to cycle through all 10 states.
2. The general rule for devices that have dual internal structures is to hold one in the same state as DUT while stimulating the other (eg. P1=D1).

7.3 RD TEST

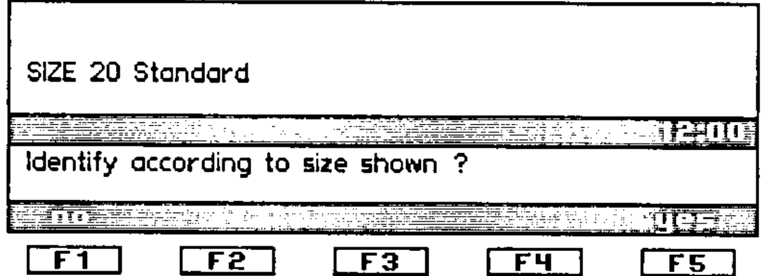
The main function of the reference device test is to verify that the correct RD has been inserted and clamped in the ZIF socket before DRC testing begins. This functionality test can also be used to verify devices out of circuit. After a device has been selected from Manual, Sequence or Develop, <ETC> will bring up a second line of F key labels that includes <<rd_test>>. The screen appears as follows after pressing this key:



<<rdt_run>> applies the library test vectors and checks the device response. Pressing <<rdt_off>> disables the RD test, as would be necessary when testing DUT signal conditions without an RD. RD Test can also be accessed under the Develop/RD Lib mode. Here it is close to the file editor level where vectors are created and allows convenient verification against an actual RD.

7.3.1 IDENTIFY

<<identify>> under RD Test appears only for devices that have a definite functionality (i.e. not PAL's or PROM's). Pressing <<identify>> brings up the following screen:

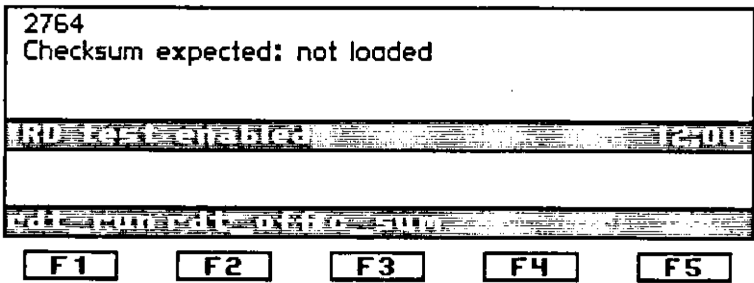


This allows an unknown device to be matched against a generic one. Pressing <<no>> will prompt for a new size and STD/NSTD power definition. Set this to the value required to match the unknown device to be identified.

Pressing <<yes>> will cause a playback of the library RD Test vectors to a device inserted into the ZIF socket. The result will appear on the status line as a device number or unknown. Note that there are sometimes several device numbers that have the same functionality. The Identify feature will provide the first positive match.

7.3.2 C_SUM

When a selected new device is a PROM or PAL, the RD Test mode screen appears as:



Pressing <<rdt_run>> will cause about 100 addresses throughout a PROM/ROM to be read and a C_SUM calculated and displayed. You may press <<c_sum>> and enter the numerical value on the command line. Subsequent RD Test trys using <<rdt_run>> or <TEST> will pass using the C_SUM entered here.

For PAL's, a C_SUM is also created, although it is arrived at differently. A standard pattern is applied based on the PAL generic number (i.e. PAL20RA10) and the output data characterized with a numeric value. Complex PAL's may produce multiple C_SUM values and so this simple RD Test creation will not work. The only alternative is to write explicit vectors based on a knowledge of the PAL the same as any standard device. The C_SUM RD Test described here is normally done in the .LOC file under Develop/New Seq mode, but may also be edited into a .LIB file as a command.

7.3.3 SPECIFYING RD TEST VECTORS

Test patterns for RD Test may be specified using conventions for two categories of devices.

1. A definite stimulus produces a predictable response.

2. A definite stimulus produces a response that:

    - is not predictable but is consistent (e.g. PROM's)
    - is not consistent (e.g. DRAM's)

Category 1 comprises most devices and the vector assignment conventions are:

- 0 - logic 0 for an input pin
- 1 - logic 1 for an input pin
- L - logic 0 for an output pin or Gnd
- H - logic 1 for an output pin or Vcc
- Z - tristated output pin
- D - disable comparison on all output pins for this vector (used for devices with internal memory that power up in indeterminate states until initialized).

In a .LIB file created from the Editor, the pins are listed in vector rows with optional spaces to improve readability. The starting command is RDTEST VECTORS and the last command is END_VECTORS.

Example:

```
NAME          7400
SIZE          14
RDTEST        VECTORS
00H 00HL      H00H00H
10H 10HL      H10H10H
01H 01HL      H01H01H
11L 11LL      L11H11H
END_VECTORS
```

Category 2 has two subcategories (CHECKSUM, PRESENCE) and uses the following vector asignment conventions:

- 0 - logic 0 for an input pin
- 1 - logic 1 for an input pin
- L - undefined pin: input, high, low or tristate output (try to drive pin low, but allow any state on pin)
- H - undefined pin: input, high, low or tristate output (try to drive pin high, but allow any state on pin)
- + - output pin with a high or low level (not tristate). Vcc and Gnd are specified with +.
- * - output pin with a high, low or tristate level
- Z - tristated output pin

For PAL's and PROM's, the header command for the vector table is:

RDTEST VECTORS CHECKSUM

The footer command is:

END_VECTORS

For DRAM's and devices with an inconsistent C_SUM, the header command is:

RDTEST VECTORS PRESENCE

The footer command is:

END_VECTORS

Note that normally, the RD Test Vectors checksum would only be created by the user for new types of PAL's and PROM's. Types that are in the library already have their RD Test specified using C_SUM as described in Section 3.3.1.

The RDTEST VECTORS PRESENCE table used on new DRAM's checks that the RD is able to drive its outputs and that it is not driving any of its inputs. The vector rate of RD Test is not fast enough to meet the timing requirements for reliable data.

7.3.4 VERIFYING AND PRINTING RD TEST VECTORS

RD Test may be run on a device from the Manual Mode that is a sublevel of develop/rd_lib. Shown on the screen will be:

- line number that failed
- pin number that failed
- C_SUM error if applicable

The RD Test patterns for user libraries and the ROM_resident system libraries may be printed. Press <<rd_lib>> in Develop mode and select the device whose pattern is to be printed under <<manual>>. Press <ESC> and <<prt_pat> which will send the formatted test pattern to the RS232C port.

7.4 OUT OF CIRCUIT DEVICE TESTING

RD Test provides a 10 KHz stored pattern functionality test of devices inserted in the socket. It primarily captures static failures such as pins that are blown low, high or open.

Reverse DRC testing may be employed to functionally compare a device in the socket within the specs of the tester to a known good device operating on a good board. The same 20 MHz signal limit and test parameters from normal DRC testing apply. This can be an effective way to screen devices such as DRAM's at the speed they will operate at in their final circuit. Note that there are a few factors that affect the quality of test:

- the test is only as comprehensive as the activity from the on-board device

- the device in the ZIF socket is only under a load of 1 LS TTL.

- a blown open fault in the socket will not be detected as a failure by DRC. The RD Test will, however, catch this, so it is an indispensible part of ZIF socket device screening.

APPENDIX I

FUNCTION KEY INDEX

KEY LABEL	DEFINITION	PAGE
advance	moves highlighted field from option to option on the display	1-27
all	is a log format that has all types of data monitored	1-31
backup	is a utility for duplicating a cartridge	6-2
cart	is a group of utilities, namely verify, fix, backup and format	6-2
:CART	specifies cartridge as the source or destination of a file	4-2
clear	changes all pins to "don't care" for trigger or gate	3-16
clip_chk	disables orientation check of test clip	3-21
col	specifies number of columns for printer page width	1-30
comment	permits keyboard entry of text into a log file	4-6

KEY LABEL	DEFINITION	PAGE
compile	produces executable files from source files	5-22 6-4
cont.	sets t_time to continuous	3-8
config_log	creates new log formats	1-31
copy	creates copies of files on cartridge or RAM	6-6
c_sum	prompts for entry of an RD Test checksum	3-20 7-13
debug	is a sublevel for generating detailed selftest results	1-16
default	re-establishes initial settings for the Reset pulse	3-14
defn_loc	prompts for entry of a device location designator	5-5
delete	is a utility for removing a file. Also erases a line in the editor.	6-8 6-18
del_char	erases a character in the editor	6-18
del_loc	prompts for entry of a location designator to remove a device	5-5
detail	specifies duty cycle information when frequency is measured. Also used for directory.	3-30 6-12

KEY LABEL	DEFINITION	PAGE
develop	is a mode for creating files and sequences	5-1
dir	provides a directory of files on cartridge or system RAM	6-10
dn_load	opens a file to receive data from RS232 port	6-13
dsp_loc	shows devices added to a new sequence file	5-6 5-8
duration	defines length of Reset pulse	3-14
edit	is a utility for changing the contents of a file	6-15
EoT	displays a screen showing states and time at end of test	3-26
ext	specifies external patch lead for frequency measurement	3-30
fix	repairs corrupted files so they may be edited or transferred	6-3
files	provides a menu of all file utilities	5-2
f_mask	permits setting the performance envelope tolerance	3-7
format	erases and initializes a cartridge	6-2

KEY LABEL	DEFINITION	PAGE
freq	permits frequency measurement on any pin and EXT lead	3-30
gate	defines a test window from states on pins and EXT	3-10
global	permits test parameters to be set for an entire board	3-2
Gnd	defines ground pin for a non-standard device	3-6
identify	matches an unknown device to RD Test library pattern	3-20
ign/comp	disables or re-enables comparison testing on a pin	3-12
indent	sets an indentation for printer output data	1-30
ins_char	spreads a line for insertion of a character	6-18
ins/rev	alternates between inserting or revising characters in the editor	6-18
into	specifies editing of a new file rather than an existing one	6-16
.LIB	specifies a library source file type	5-8

KEY LABEL	DEFINITION	PAGE
line	sets a line width for printer output data	1-30
list_seq	provides a listing of sequence names on :CART or :SYST	4-3
list_to	defines a file that will contain the directory listing	6-11
.LOC	specifies a location source file type	5-8
local	permits test parameters to be defined for a single device	3-2
log	is a mode for enabling data recording	4-9
.LOG	specifies a log file type	6-1
log_off	terminates logging of data to the highlighted file	4-11
log_on	prompts for log format and destination of data to be logged	4-9
.LST	specifies a directory listing file type	6-11
manual	is a mode for testing single devices	3-1

KEY LABEL	DEFINITION	PAGE
minimum	is a log format that monitors only failure data	1-31
new_seq	begins or continues development of a new sequence	5-5
next_freq	steps the frequency measurement to the next specified pin	3-31
no_time	specifies an indefinitely long sampling window for freq. measurement	3-31
NSTD	specifies a non-standard device power configuration	3-5
offset	shifts the comparison testing interval relative to the Reset pulse	3-14
passes	is a log format that monitors pass and failure data	1-31
pin_def	defines pins for signal conditon monitoring	3-11
polarity	defines polarity of Reset pulse	3-15
print	is a utility for printing file listings	6-19
printer	defines printer format options	1-31
prt_log	directs a log file to the RS232 port for printing	4-11

KEY LABEL	DEFINITION	PAGE
prt_pat	directs the RD Test pattern to the RS232 port for printing	7-17
prt_res	directs detailed selftest results to the RS232 port	1-16
rd_drv	permits setting a device's characteristic drive capability	3-7
rd_lib	is a mode for developing user device libraries	7-1
rd_test	is a mode for testing out-of-circuit devices	3-19
rdt_off	disables RD Test	3-20
rdt_on	re-enables RD Test	3-20
rdt_run	performs the RD Test	3-20
recov	lists recoverable files that have been deleted	6-12
recover	is a utility for recovering deleted files	6-21
reset	permits setting attributes for the Reset pulse	3-13
results	shows detailed test results on the display	3-22

KEY LABEL	DEFINITION	PAGE
roll	rotates a highlighted field through several options	1-27
run_seq	executes a test sequence	4-3
save_fil	stores a new sequence to its file destination	5-8
save_pin	defines a pin condition as the same as observed at cursor	3-24
.SEQ	specifies a sequence source file type	5-8
sequence	is a mode for preprogrammed board testing	4-1
show_log	lists log file contents on the display	4-11
size	defines number of pins and power configuration of a device	3-5
sound	sets the volume of audible feedback to keystrokes	1-29
start	restarts sequence from the beginning	4-7
state	displays a screen showing pin states during test	3-23

KEY LABEL	DEFINITION	PAGE
status	rotates through optional condition tests at cursor position	3-12
s_time	permits setting of sync time	3-9
supply	defines voltage source of Reset pulse	3-15
:SYST	specifies system RAM as source or destination of a file	4-2
system	is a mode for setting machine options	1-26
tag	is a log format suitable for a rework tag	1-31
term	sets the line termination character for printer output data	1-30
thrsld	permits setting of logic 1 level	3-10
thru	defines consecutive pins for frequency measurement	3-30
time&dat	initialize time and date information	1-26
trigger	defines a two word event that determines start of test	3-16
t_time	permits setting of test time	3-8

KEY LABEL	DEFINITION	PAGE
<.type>	reveals a menu of file types	6-1
upload	sends a file out the RS232 port	6-23
untested	shows a list of devices not yet tested in the sequence	4-7
Vcc	specifies power pin for a non-standard device	3-6
verify	a utility that identifies and flags corrupted files	6-2

APPENDIX II

COMMAND LANGUAGE

All commands may be put into a file with the Editor and they are listed here alphabetically with their required syntax. In addition, many commands may be generated from keystrokes in New_seq mode. The table on the page II - 3 lists which commands are possible under the Editor (with recommended file types) and New_seq mode. The command language follows a number of rules in addition to the specific syntax listed with each command

- Each line is a command. Blank lines have no effect, but make a listing more readable. Commands are grouped together by device or location to form a command group.
- Command groups end with a colon (:)
- Command fields are separated by at least one space. The Tab key is often used to improve readability.
- Maximum line length is 254 characters.
- Labels start with a letter and are alphanumeric (including underline character).
- A semicolon (;) denotes a comment in the file listing.

To view a file on the screen, press <<edit>> and enter the file name, type and source on the command line (e.g. A.SEQ:CART).

Example:

```
            LOC_FILE A;this comment has no effect
            TEST U1:
            TEST U22:
LABEL       TEST U46:
```

Scrolling through a listing on the screen is done with the up and down arrow keys. "Shift down arrow" scrolls a full page and "Cntr down arrow" scrolls to the end of the file.

Command parameters enclosed with { } brackets are mandatory in the syntax descriptions; parameters enclosed in [ ] brackets are optional. These brackets do not form part of the command. Conversely, for the Sync commands, < > brackets are used that do form part of the command. The syntax and examples for certain commands sometimes appear to wrap around two lines. The actual screen editor has a 254 character line length which accommodates any command on a single line.

COMMAND	EDITABLE FILE				NEW_SEQ	
	.SEQ	.LOC	.LIB			
ACTIVITY		X	X		X	
CLIP-CHK		X	X		X	
COMMENT	X				X	
COMPARE		X			X	
C_SUM		X	X		X	
DISPSLAY	X	X				
END	X					
END_FUNCTION	X					
F_MASK		X			X	
FUNCTION	X					
GATE			X	X		X
IF...THEN...	X					
IGNORE		X	X		X	
JUMP	X					
LOAD		X	X		X	
LOC_FILE	X					
NAME		X	X		X	
RD_DRV		X	X		X	
RDT_ENABLE		X	X		X	
RDTEST			X			
RESET		X			X	
SIZE		X	X		X	
SOUND	X					
S_TIME		X	X		X	
SYNC-COND				X		
SYNC_GATE				X		
SYNC_GR_END			X			
SYNC_IGNORE			X			
SYNC_PAT			X			
SYNC_PINS			X			

COMMAND	EDITABLE FILE .SEQ	.LOC	.LIB	NEW_SEQ
SYNC_RESET_OFF		X	X	
SYNC_RND			X	
SYNC_VECT			X	
TEST	X			
THRSLD		X	X	X
TRIGGER		X		X
T_TIME		X		X

ACTIVITY

Purpose: To test for signal conditions on individual pins.

Syntax: ACTIVITY {pin#=status pin#=frequency tolerance....}

Parameters: H - high
L - low
A - active
X - don't care
CLEAR - disable all condition tests
Frequency from .005 Hz to 25 MHz with units Hz, kHz, MHz
Tolerance of frequency as a percentage

Example:

ACTIVITY P3=H P7=L P9=A P12=12.325 MHz 3%
ACTIVITY CLEAR

Description: This command specifies signal condition checks to be performed on a per pin basis. They include high state, low state, active toggling and specified frequency within a tolerance. The conditions are added to any conditions that may have been specified in a previous command.

CLIP_CHK

Purpose: To verify orientation of test clip on DUT.

Syntax: CLIP_CHK {ON/OFF}

Parameters:
ON - enables clip check
OFF - disables clip check

Example: CLIP_CHK ON

Description: The clip check command enables or disables the verification of logic levels on the power pins of the DUT.

COMMENT

Purpose: To insert a message string into a log file.

Syntax: COMMENT {"message string"}

Parameter: Alphanumeric characters enclosed in quotation marks.

Example: COMMENT "THIS MESSAGE WILL APPEAR IN A LOG FILE"

Description: This command will put a message string into any currently open log files that have comments enabled in their log format.

COMPARE

Purpose: To re-enable any pins that were previously ignored.

Syntax: COMPARE {list of pins}

Parameters: pin numbers
consecutive pins

Example: COMPARE 1 3-5 11 15-13

C_SUM

Purpose: To specify the value to be used as the checksum for the RD Test of a PROM or Pal.

Syntax: C_SUM {value}

Parameter: Value in the range of 0 to 65535

Example: C_SUM 56241

Description: When the ROM-resident or user file RD Test vectors are specified in the checksum format, the C_SUM value is used to verify the RD response.

DISPLAY

Purpose: To put an operator prompt on the display.

Syntax: DISPLAY [line number] {"message string"}

Parameters:

Line number	-	1,2,3,4. Line 3 is assumed if blank.
message string	-	characters enclosed in quotation marks. The string must be less than 40 characters except for line 3 which will wrape to line 4 for a maximum of 80 characters.

Example:

DISPLAY 1 "THIS MESSAGE WILL APPEAR ON FIRST LINE"

DISPLAY "THIS WILL APPEAR ON THIRD LINE"

Description: This command puts a parameter string that is enclosed in quotation marks on the screen as an operator prompt. The string length is limited to 40 per line. Line 3 will wrap around to line 4.

END

Purpose: To return Sequence execution to the start.

Syntax: END

Parameters: None

Description: This command returns to the start and brings up the first screen when running a Sequence.

END_FUNCTION

Purpose: To return execution to the main sequence after a function subroutine.

Syntax: END_FUNCTION

Parameters: None

Example: See FUNCTION command.

F_MASK

Purpose: To set the Performance Envelope tolerance.

Syntax: F_MASK {value}

Parameter: Value is a number in the range of 20 to 200 ns with 10 ns increments.

Example: F_MASK 60

Description: This command sets the value of the comparison resolution.

FUNCTION

Purpose:	To call a subroutine consisting of one or more commands.
Syntax:	FUNCTION {label}
{label}	[command 1] [command n]
Parameter:	Label is an alphanumeric string beginning with a letter.
Example:	FUNCTION MESG1
MESG1	DISPLAY "USER PROMPT MESSAGE" END_FUNCTION
Description:	This command acts as a subroutine call, transferring execution to the labelled command and returning with END_FUNCTION. It is good practice to replace labelled functions after the END of a Sequence so they are not entered accidentally.

GATE

Purpose: To mask out indeterminate orillegal device states.

Syntax: GATE {parameter} {P#=I/O E=I/O...}

Parameters:
ON - enables or re-enables gate
OFF - disables gate
CLEAR - clear all I/O definitions
CLEAR_PINS - clear P# definitions but not E

Example:
```
GATE ON P3=1  P4=0  E=1
GATE OFF
GATE ON
```

Note: The third line re-enables the gate defined in the first line.

Description: This command masks out comparison testing whenever an event occurs defined by specified logic states on all device pins plus the EXT patch lead. It may be turned off and subsequently re-enabled.

IF...THEN...

Purpose: To redirect sequence flow based on test results.

Syntax: IF {conditions} THEN {command}

Parameters: Conditions

- PASSED
- FAILED
- RD_SHORTED
- RD_TEST_FAILED
- CLIP_TEST_FAILED
- FAILED_TO_SYNC
- NO_GATE
- NO_TRIGGER
- NO_TRIGGER_WORD1
- NO_TRIGGER_WORD2
- P#=H,L,A, frequency and tolerance

Operators

+	AND
*	OR
<	less than
>	greater than
=	equal
<>	not equal
()	precedence brackets

IF...THEN...

Examples: IF PASSED THAN DISPSLAY "ABC"
IF FAILED +P4 <> 5 MHz 5%
THEN JUMP LABEL2
IF P3-5MHz 5% *(P5=A+P6=L)
THEN JUMP LABELS

Description: This command will examine the last test result for specified conditions and, if found to be false, the rest of the command is not executed until <NEXT> command is pressed. The word THEN is interpreted as a command separator and the word following it is assumed to be a command. For the + and * operators, the evaluation takes place from left to right unless precedence is indicated by brackets.

IGNORE

Purpose: To disable comparison testing on specified pins.

Syntax: IGNORE {pins and consecutive pin groups}

Parameters: number
number - number

Example: IGNORE 1 3-5 11 15-13

JUMP

Purpose:	To unconditionally redirect sequence flow.
Syntax:	JUMP {label}
Parameter:	Label is an alphanumeric string beginning with a letter.
Example:	JUMP CONT1
	.
	.
CONT1	DISPLAY "ABC"
Description:	This will caluse the Sequence to continue execution at the label specified.

LOAD

Purpose: To load parameter data from the library for a specified device.

Syntax: LOAD {device}

Parameter: Any device named in ROM or user libraries.

Example: LOAD 7400
LOAD PAL20V10

LOC_FILE

Purpose: To reference a .loc file to a .seq file.

Syntax: LOC_FILE {file name}

Parameters: Name of .loc file

Example: LOC_FILE IBM_AT

Description: This command must be the first line of a .SEQ file. It specifies which .LOC file to use and allows different .SEQ files to use the same .LOC file. This command never has a colon (:) line terminator.

NAME

Purpose: To define a reference string for a command group in a .LOC or a .LIB file.

Syntax: NAME {string}

Parameter: Alphanumeric characters

Example: NAME U45
NAME 7400

RD_DRV

Purpose: To permit comparison testing of weak drive devices.

Syntax: RD_DRV {parameter}

Parameters: HIGH
LOW

Example: RD_DRV LOW

Description: This command changes the DIFTEST comparison circuit so it loads the RD by only about a third of the usual 1 LS load. This permits devices which have a weaker drive characteristic to be used as an RD.

RDT_ENABLE

Purpose: To enable or disable the automatic RD Test.

Syntax: RDT_ENABLE {parameter}

Parameters: ON
OFF

Example: RDT_ENABLE OFF

RDTEST

Purpose: To specify a new RD Test in a user library.

Syntax: RDTEST {VECTORS} [Vector type]
Vector Table
END_VECTORS

Parameters:

VECTORS	- evaluated by table
VECTORS CHECKSUM	- evaluated by C_SUM
VECTORS PRESENCE	- evaluated by drive detection

Note: Maximum number of vectors is 99.

Example:

```
NAME        7400
SIZE        14
RDTEST      VECTORS
00H00HL     H00H00H
10H10HL     H10H10H
01H01HL     H01H01H
11L11LL     L11H11H
END_VECTORS
```

Description: This command loads the RD test pattern. The pattern is specified as one of 3 vector table types followed by the vector table and terminated by END_VECTORS.

Vector Table Symbol Conventions

VECT	V. CHKSM V. PRNCE	PIN DESCRIPTION
0	0 (1)	LOW INPUT (VERIFIED)
1	1 (1)	HIGH INPUT (VERIFIED)
	L (2)	LOW INPUT (UNVERIFIED)
	H (2)	HIGH INPUT (UNVERIFIED)
L (6)		LOW OUTPUT OR Gnd
H (6)		HIGH OUTPUT OR Vcc
Z	Z	TRISTATED OUTPUT
	+ (7)	HIGH OR LOW OUTPUT OR Vcc,Gnd
	*	HIGH, LOW,TRISTATE OUTPUT
D	D (3)	ALL OUTPUTS UNKNOWN

Notes:

1. Pin must follow the level driven.
2. Pin need not follow driven level.
3. D on any pin causes all pins to be ignored for that vector. Used for devices that power up in indeterminate state until initialized (e.g. UART).
4. Columns of vectors are the pin numbers from left to right.
5. Spaces may separate vector elements for readability.
6. Gnd pin is assigned L for VECTORS type.
   Vcc pin is assigned H for VECTORS type.
7. Gnd and Vcc are assigned + for VECTOR CHECKSUM or PRESENCE types.

RESET

Purpose: To define the attributes of the Reset pulse.

Syntax: RESET {ON/OFF}{polarity}{source} {DUR=duration} {OFS=offset}

Parameters:

ON	enables current setting
OFF	suspends reset pulse
polarity	POS (positive) or NEG (negative)
voltage	INT (+5V) or EXT (from V patch Lead)
duration	DUR from 10 to 32767 ms in 1 ms steps
offset	OFS up to + or - 32767 ms in 1 ms steps

Example:

```
RESET     ON NEG INT  DUR=200 OFS=-30
RESET     OFF
RESET     ON
```

Description: This command issues a reset pulse on the R patch lead when <TEST> is pressed and before the S_TIME portion of the test cycle. Polarity, voltage, duration and offset relative to comparison are definable.

SIZE

Purpose: To define device size and power configuration.

Syntax: SIZE {# of pins} [VCC pin list GND pin list]

Parameters:

SIZE	VCC	GND
8	1,4,7,8	4,5,8
14	1,4,5,13,14	4,7,10,11,14
16	1,4,5,8,15,16	4,7,8,12,13,16
18	1,4,5,8,9,17,18	4,7,8,9,14,15,18
20	1,4,5,8,9,19,20	4,7,8,9,10,16,17,20
22	1,4,5,8,9,21,22	4,7,8,9,10,11,18,19,22
24	1,4,5,8,9,23,24	4,7,8,9,10,11,12,20,21,24
28	1,4,5,8,9,27,28	4,7,8,9,10,11,12,14,24,25,28

Example:

```
SIZE  16
SIZE  16 VCC 8 GND 16 4
```

Description: This command defines the size of the RD. Optionally, the Vcc and Gnd power pins may be specified. If these parameters are not included, the standard pin configuration is assumed (Vcc= highest pin#, Gnd= (highest pin#)/2).

SOUND

Purpose: To generate a sound alert when encountered in a sequence.

Syntax: SOUND {type}

Parameters: FAIL
PASS
POWER_UP
ERROR
ACK
ASCII

Example: SOUND ERROR

S_TIME

Purpose: To define the duration of a synchronizing interval prior to comparison testing.

Syntax: S_time {ON/OFF} [time]

Parameters: Time value from 10 to 9990ms in 10 ms steps.

Example:
S_TIME ON 5000
S_TIME OFF
S_TIME ON

Description: This command sets the maximum amount of time, not including Reset duration, that the machine will attempt to synchronize RD and DUT. When it is used without a time parameter, the time is chosen as the previous value used.

SYNC_COND

Purpose: To define the necessary and sufficient DUT activity for synchronizing RD and DUT.

Syntax: SYNC_COND <# cond 1> operator <# cond #2>...

Parameters:

number of occurrences of bracketed expression

Pin #=	H	high pin state
	L	low pin state
	R	rising edge
	F	falling edge
	N	not active
operator	+	logical OR
	*	logical AND
priority	< >	boundary of condition (cannot be nested)
	( )	grouping within condition or of conditions
	/	ordered AND operator that requires left condition to be satisfied before right condition is evaluated.

Example:

for 74166 Clear + Serial Shift + Parallel load

```
SYNC_COND
    <1 P9=L> +
    <8 P15=H*(P7=L*P6=R+P7=R*P6=L)> +
    <1 P15=L*(P7=L*P6=R+P7=R*P6=L)>
```

for 74595

```
SYNC_COND <8 P11=R>/<1 P12=R>
```

SYNC_IGNORE

To define the pins that should be ignored while checking if RD and DUT are synchronized.

Syntax: SYNC_DISABLE {list of pins}

Parameter: Numbers from 1 through 28

Example: SYNC_DISABLE 3 11 13

SYNC_GATE

To define the time window to check for synchronization from the DUT output enable pin(s) for tristated devices.

Syntax: SYNC_GATE {P#=1/0 P#=1/0...}

Parameters: P#= pin number
1 = logic 1
0 = logic 0

Example: SYNC_GATE P3=1 P5=0

SYNC_GR_END

To mark the end of a group of SYNC commands.

Syntax: SYNC_GR_END

Parameters: None

SYNC_PAT

To specify the use of the RD Test vector pattern for synchronization.

Syntax: SYNC_PAT

Parameters: None

SYNC_PINS

To define the list of output pins on the DUT that should be inactive during synchronization.

Syntax: SYNC_PINS {list of pins}

Parameters: Numbers from 1 through 28

Example: SYNC_PINS 2 10 13

SYNC_RESET_OFF

To disable the reset normally issued during S_TIME.

Syntax: SYNC_RESET_OFF

Parameters: None

SYNC_RND

To specify the use of random vector patterns for synchronization.

Syntax: SYNC_RND

Parameters: None

SYNC_VECT

To specify the exact vector patterns to generate for synchronization.

Syntax:

```
SYNC_VECT  N<# level Pin=attribute...>/<#...>...
```

Parameters:

< >	brackets enclosing a vector expression
/	left vector precedes right one
#	specifies number of times to repeat a bracketed expression
N	specifies number of times to repeat all expressions
H	unspecified pins are logic 1
L	unspecified pins are logic 0
C	clock pulse is provided
D#	assigns DUT pin number state to RD pin
!D#	assigns inverted DUT pin number state

Example:

```
for 7490
SYNC_VECT        2<1 L P14=C P1=D1>/
                 <5 L P1=C P14=D14>

for 7495
SYNC_VECT        1<1 L P6=1 P8=C P2=D13 P3=D12
                 P4=D11 P5=D10>
```

TEST

Purpose:	To retrieve the test parameters from a .loc file for a specified device in preparation for testing.
Syntax:	TEST {location}
Parameter:	Location is a parameter of a NAME command in a .LOC file and is the topology designator for a device on a board.
Example:	TEST U5
Description:	This .SEQ file command locates the device parameters in the .LOC file and combines them with the default parameters. The test hardware is set up awaiting the pressing of the TEST key.

THRSLD

Purpose: To define logic 1 for a device.

Syntax: THRSLD {voltage value}

Parameter: 0 to 5000 mV in 100 mV steps.

Example: THRSLD 1500

TRIGGER

Purpose: To start comparison testing after an event occurs on a device.

Syntax: TRIGGER {mode}{P#=states E=states}

Parameters:

mode	- ON	enable trigger
	- OFF	disable trigger
	- CLEAR	clear all bits to "don't care"
	- CLEAR_PINS	clear pin but not EXT
P#	- pin number	
E	- EXT lead	
states	- 1 for high 0 for low X, x for "don't care"	

Note: states are specified in pairs for word1, word2

Example:

```
TRIGGER ON P1=10 E=00 PR=1X
TRIGGER OFF
TRIGGER ON
```

Description: This command is used to define the start of comparison testing from a two word event that is a combination of the logic states on any pins of the DUT plus the EXT patch lead. It may be turned off and subsequently re-enabled.

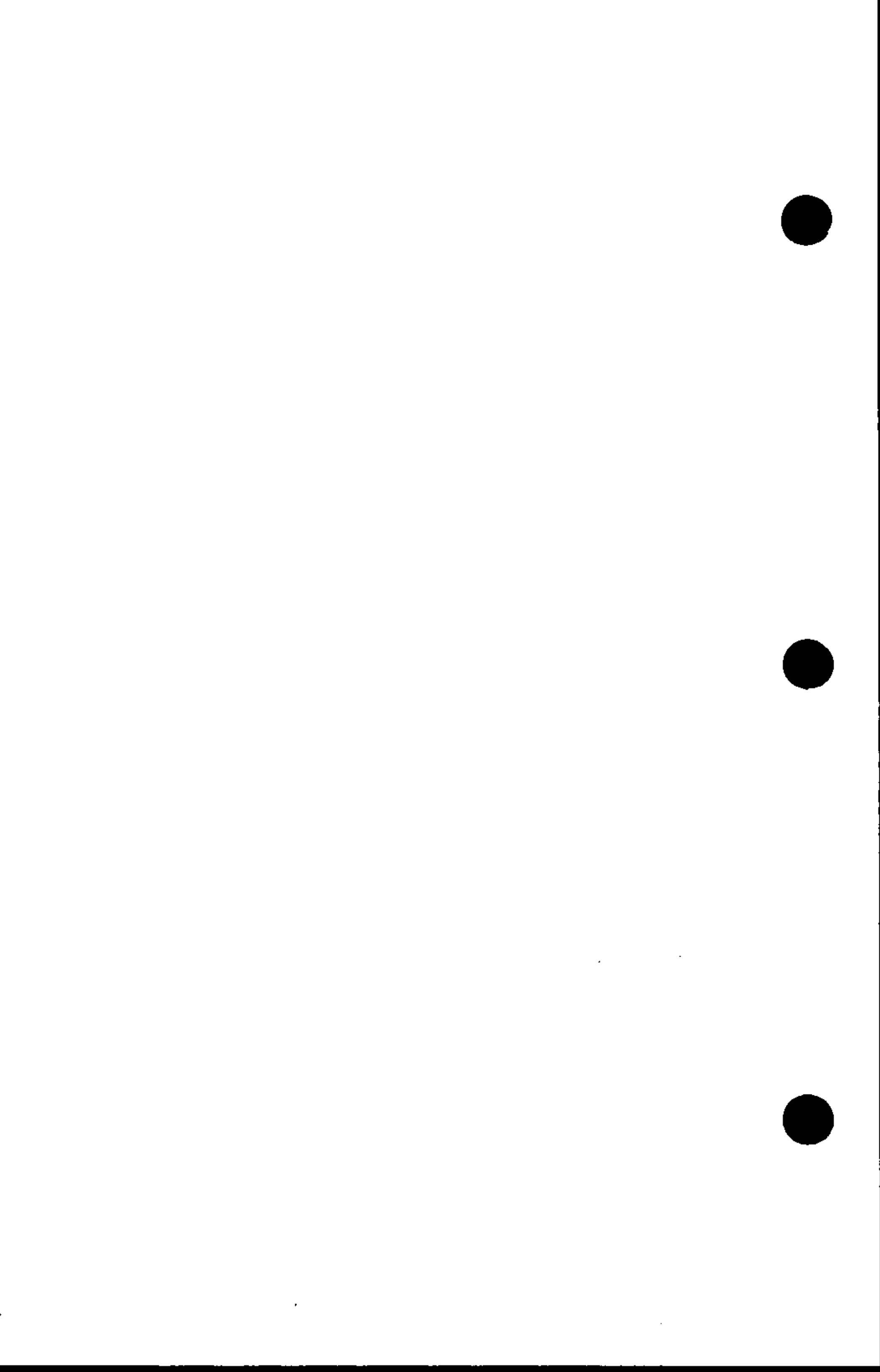
T_TIME

Purpose: To define the duration of comparison testing.

Syntax: T-TIME {value}

Parameter: A value from 10 to 9990 ms in steps of 10 ms.
CONT means a continuous test time.

Example: T_TIME 5500
T_TIME CONT



APPENDIX IV

REMOTE CONTROL PROTOCOL

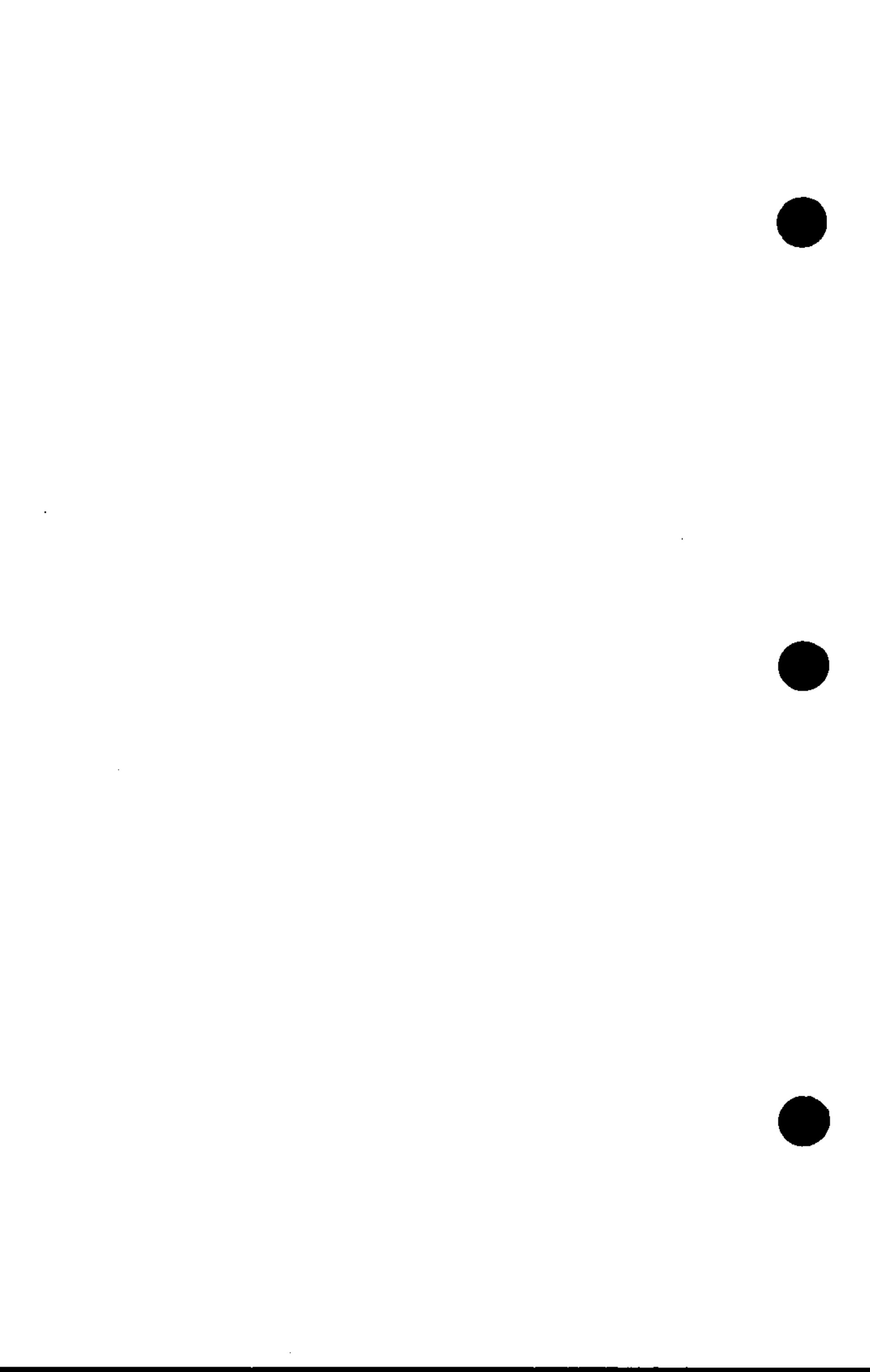
1 GENERAL DESCRIPTION

- 1.1 COMMAND STRUCTURE 2
- 1.2 ACKNOWLEDGE PROTOCOL 3
- 1.3 ACKNOWLEDGE CODES 4
- 1.4 ABORTING A COMMAND 7

2 REMOTE CONTROL COMMANDS

- 2.1 ENTERING AND EXITING REMOTE MODE 8
  - 2.1.1 identify 8
  - 2.1.2 exit remote mode 9
  - 2.1.3 reset unit - soft 9
  - 2.1.4 reset unit - full 10

- 2.2 DEVICE DEFINITION 11
  - 2.2.1 load data from a .loc file 11
  - 2.2.2 load data from a .lib file 11
  - 2.2.3 set size & power 12
  - 2.2.4 read size & power 13
  - 2.2.5 set RD drive 13
  - 2.2.6 read RD drive 14
  - 2.2.7 set c_sum 14
  - 2.2.8 read c_sum 15

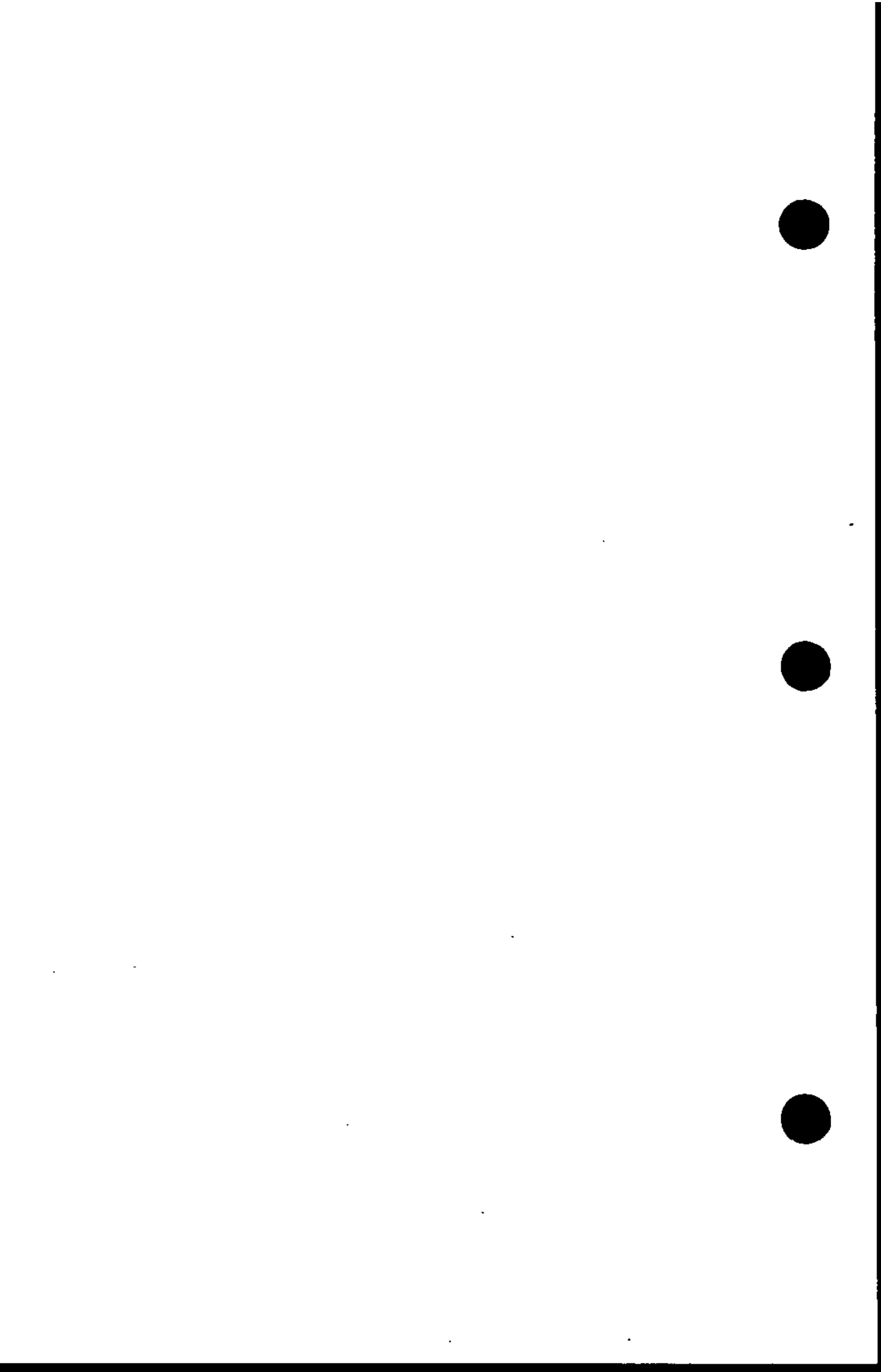


2.3 TEST PARAMETER SETUP 16
2.3.1 fmask 16
2.3.2 threshold 17
2.3.3 test time 18
2.3.4 sync time 19
2.3.5 pin_def 20
2.3.6 trigger and gate 21

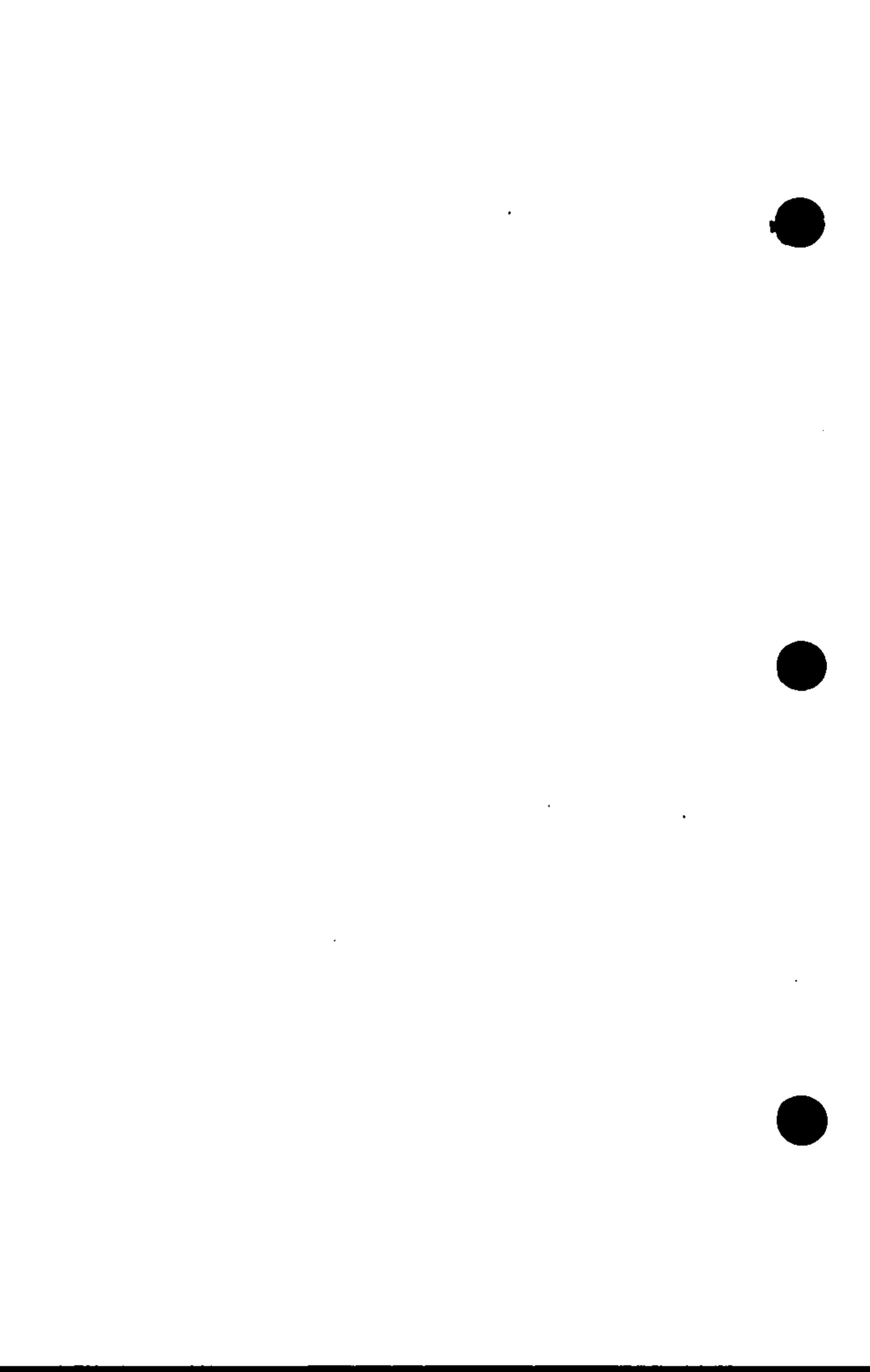
2.4 TEST CYCLE CONTROL 26
2.4.1 define reset 26
2.4.2 read reset definition 26
2.4.3 clip_chk enable/disable ... 27
2.4.4 read clip_chk status 28
2.4.5 rdtest enable/disable 28
2.4.6 read rdtest status 29
2.4.7 perform test 30
2.4.8 read test results 32
2.4.9 read status results 34

2.5 USER COMMUNICATION 36
2.5.1 display text on screen 36
2.5.2 read a keystroke 37
2.5.3 read a string of keystrokes 38
2.5.4 generate sounds 39

2.6 READING FREQUENCY AND TIMING 40
2.6.1 read frequency 40
2.6.2 read period 40
2.6.3 read high time 41
2.6.4 read low time 41
2.6.5 read duty cycle 42



2.7 FILE MANIPULATION 43
2.7.1 download to unit 43
2.7.2 upload from unit 43
2.7.3 compile file 43
2.7.4 delete a file 44
2.7.5 directory of files 44
2.7.6 format cartridge 45



REMOTE CONTROL PROTOCOL

1 GENERAL DESCRIPTION

This is a general protocol for the control of the tester by other equipment. The controller is able to control most aspects of operation, and is able to obtain complete information about tester status or results. Some macro commands are also available to make it easier/faster to drive the tester or that make programming the controller easier. All communication is done using 7 bit ASCII codes. After every command the tester sends back any results followed by an acknowledge code to indicate that it is finished and is ready to receive the next command.

To put the tester into remote mode, the identify command must first be sent while the power on screen is showing.
In the following description, these conventions are used:

- control characters are enclosed in < > brackets. e.g. <stx>
- optional parameters are indicated with [ ] brackets.
- mandatory parameters are indicated with { } brackets.
- information sent to tester is represented by
  -------->
- information sent from tester is represented by
  <--------

1.1 COMMAND STRUCTURE

byte 1	Fixed code indicating the start of a command: <stx>
byte 2	First command code.
[byte 3 to n]	Secondary command codes if required.
byte n+1 to m	Parameter bytes. <cr> is the separator between parameters.
byte m+1	End of command parameter: <etx>

General example of a command:

<stx> command code(s) [parameters] <cr> [parameters]...<etx>

1.2 ACKNOWLEDGE PROTOCOL

Every command sent to the tester will be acknowledged in a standard way. If the command will take some time, the unit will send a <sub> back to indicate it has received the command and is processing it. When it has finished it will either send an <ack> indicating it has processed the command and has no response, or it will send a string of data starting with a <stx> and ending with either a <nak> or an <ack>. The controller must not send the next command until it receives an <ack> or <nak> from the tester. The only exception to this is the abort code <dle> which can be sent by the controller at any time.

1.3 ACKNOWLEDGE CODES

02H <stx>	start of any command or result block.
03H <etx>	end of data code. Used to mark the end of any command that has a variable length.
1AH <sub>	command accepted and executing.
06H <ack>	command completed.
10H <dle>	abort current command. --------> <dle> <-------- <stx>H<nak>
0DH <cr>	used to separate parameters in a command or result.
15H <nak>	negative acknowledge. Error responses have the general form: <stx>{codes}[parameters]<nak>

<stx>A<nak>	parity error
<stx>B<nak>	break detected
<stx>C<nak>	framing error
<stx>D<nak>	overrun error
<stx>E<nak>	other communication channel errors
<stx>F<nak>	Invalid command
<stx>G<nak>	general syntax error

<stx>H<nak>	command aborted
<stx>I<nak>	device not found
<stx>J<nak>	parameter out of range
<stx>K<nak>	file not found
<stx>L<nak>	invalid file type
<stx>M<nak>	location not found
<stx>N<nak>	command execution error
<stx>Y[parameters]<nak>	errors specific to a particular command. These are described with the main data for that command.

<stx>Z[parameter]<nak>

file error parameters:

01	file is open
03	file not found
04	file already exists
05	insufficient space in destination
07	cannot write to destination
08	cartridge write protected
12	file checksum error
14	unformatted cartridge
15	file unrecoverable
1C	file too large for destination
22	invalid file type
24	corrupt file

1.4 ABORTING A COMMAND

The controller may abort many of the longer commands by sending an abort code <dle>. If the abort is recognized the response will be the abort nak string, <stx>H<nak>, and if not the response will be the standard one.

2 REMOTE CONTROL COMMANDS

2.1 ENTERING AND EXITING REMOTE MODE

2.1.1 identify

This command prompts the tester to identify its model, configuration and version numbers. This command code is always followed by the text "Identify" as insurance against some unit that is attached to the tester accidentally becoming a controller. None of the other commands will be operational until this command has been received.

Syntax

```
-------->    <stx>AIdentify<etx>
<--------    <stx>Fluke 900<cr>
             {selftest results}<cr>
             {system software version}<cr>
             {L library version}<cr>
             {M micro board rev}<cr>
             {H high speed board rev}<cr>
             {I input buffer rev}<ack>
```

Parameters

selftest	: P-pass F-fail
system s/w	: #.##
library s/w	: L#.##
micro board	: M#
high speed board	: H#
input buffer	: I#

Example

--------> <stx>AIdentify<etx>
<-------- <stx>Fluke 900<cr>P<cr>4.00<cr>
L3.00<cr>M0<cr>H1<cr>I0<ack>

2.1.2 exit remote mode

Syntax

--------> <stx>GD<ext>
<-------- <ack>

2.1.3 reset unit - soft

The tester exits remote mode and all parameters are set to their default values.

Syntax

--------> <stx>GB<etx>
<-------- <ack>

2.1.4 reset unit - full

The tester exits remote mode and executes a selftest initialization.

Syntax

-------->	<stx>GC<etx>
<--------	<ack>

2.2 DEVICE DEFINITION

One of the following three commands must be used prior to setting any test parameters.

2.2.1 load data from a .loc file

Syntax

-------->	<stx>BA{file_name}[.loc][:source]<cr> {location_name}<etx>
<--------	<sub>
<--------	<ack>

Parameter

Filename default type is .loc.

Example

-------->	<stx>BATEST1:SYST<cr>U43<etx>
<--------	<sub>
<--------	<ack>

2.2.2 load data from a .lib file

Syntax

-------->	<stx>BB{device_name}<etx>
<--------	<sub>
<--------	<ack>

Example

-------->	<stx>BB74244<etx>
<--------	<sub>
<--------	<ack>

2.2.3 set size & power

Syntax

-------->	<stx>BC{device size}
-------->	V<cr>
-------->	{Vcc pin#}<cr>
-------->	[Vcc pin#]<cr>
-------->	G<cr>
-------->	{gnd pin#}<cr>
-------->	[gnd pin#]<etx>
<--------	<ack>

Example

------->	<stx>BC20<cr>V<cr>19<cr>G<cr> 8<cr>20<etx>
<--------	<ack>

2.2.4 read size & power

Syntax

```
-------->     <stx>BD<etx>
<--------     <stx>{device size}<cr>
<--------     V<cr>
<--------     {Vcc pin#}<cr>
<--------     [Vcc pin#]<cr>
<--------     G<cr>
<--------     {gnd pin#}<cr>
<--------     [gnd pin#]<ack>
```

Example

```
-------->     <stx>BD<etx>
<--------     <stx>20<cr>V<cr>19<cr>
<--------     G<cr>8<cr>20<ack>
```

2.2.5 set RD drive

Syntax

```
-------->     <stx>IB{parameter}<etx>
<--------     <ack>
```

Parameter

H - high
L - low

Example

```
-------->     <stx>IBL<etx>
<--------     <ack>
```

2.2.6 read RD drive

Syntax

--------> <stx>IC<etx>
<-------- <stx>{parameter}<ack>

Parameter

H - high
L - low

Example

--------> <stx>IC<etx>
<-------- <stx>H<ack>

2.2.7 set c_sum

Syntax

--------> <stx>ID{value}<etx>
<-------- <ack>

Parameter

numerical c_sum value

Example

--------> <stx>ID34562<etx>
<-------- <ack>

2.2.8 read c_sum

This command causes a numerical value to be returned which is the c_sum. If there is no c_sum, no parameter appears (ie: <stx><ack>).

Syntax

-------->	<stx>IE<etx>
<--------	<stx>{value}<ack>

Example

-------->	<stx>IE<etx>
<--------	<stx>34562<ack>

2.3 TEST PARAMETER SETUP

2.3.1 fmask

2.3.1.1 set fmask

Syntax

--------> <stx>CA{value}<etx>
<-------- <ack>

Example

--------> <stx>CA120<etx>
<-------- <ack>

2.3.1.2 read fmask

Syntax

--------> <stx>CB<etx>
<-------- <stx>{value}<ack>

Example

--------> <stx>CB<etx>
<-------- <stx>120<ack>

2.3.2 threshold

2.3.2.1 set threshold

Syntax

--------> <stx>CC{value}<etx>
<-------- <ack>

Example

--------> <stx>CC2000<etx>
<-------- <ack>

2.3.2.2 read threshold

Syntax

--------> <stx>CD<etx>
<-------- <stx>{value}<ack>

Example

--------> <stx>CD<etx>
<-------- <stx>2000<ack>

2.3.3 test time

2.3.3.1 set test time

Syntax

-------->	<stx>CEA{value}<etx>
<--------	<ack>

Parameter

value: 1 to 9999
C (continuous)

Example

-------->	<stx>CEA1000<etx>
<--------	<ack>

2.3.3.2 read test time

Syntax

-------->	<stx>CEB<etx>
<--------	<stx>{value}<ack>

Parameter

value: 1 to 9999
C (continuous)

Example

-------->	<stx>CEB<etx>
<--------	<stx>C<ack>

2.3.4 sync time

2.3.4.1 set sync time

Syntax

-------->	<stx>CEC[value]<etx>
<--------	<ack>

Parameter

value: 10 to 9990
no value means off

Example

-------->	<stx>CEC2500<etx>
<--------	<ack>
-------->	<stx>CEC<etx>
<--------	<ack>

2.3.4.2 read sync time

Syntax

-------->	<stx>CED<etx>
<--------	<stx>[value]<ack>

Parameter

value: 10 to 9990
no value means off

Example

--------> <stx>CED<etx>
<-------- <stx>2500<ack>

--------> <stx>CED<etx>
<-------- <stx><ack>

2.3.5 pin_def

2.3.5.1 define a pin_def

Syntax

--------> <stx>CF{pin#}<cr>
--------> [activity or frequency]<cr>
--------> [Ignore or Compare]<etx>
<-------- <ack>

Parameter

H - high
L - low
A - active
I - ignore
C - compare

frequency	units	tolerance
##.###	Hz	#%
	kHz	
	MHz	

Example

--------> <stx>CF20<cr> (pin#)
--------> 4.5MHz4%<cr> (activity or frequency)
--------> I<etx> (Ignore/Compare)
<-------- <ack>

2.3.5.2 read a pin_def

Syntax

--------> <stx>CG{pin#}<etx>
<-------- <stx>[activity or frequency]<cr>
<-------- {Ignore or Compare}<ack>

Example

--------> <stx>CG20<etx>
<-------- <stx>4.5MHz4%<cr>
<-------- I<ack>

2.3.6 trigger and gate

Trigger and Gate are defined and enabled remotely from three words which are first assigned before being enabled. Word 0 is reserved for gate, words 1 and 2 for trigger. Each word corresponds to the pins on a device with pin 1 at the left followed by a separate bit for the EXT patch lead input.

2.3.6.1 set word definition

Syntax

```
-------->    <stx>DGA{word#}<cr>
-------->    {word}<cr>
-------->    {EXT bit}<etx>
<--------    <ack>
```

Example

```
-------->    <stx>DGA0<cr>
-------->    001X11X00XXXXXXX<cr>
-------->    1<etx>
<--------    <ack>
```

2.3.6.2 read word definition

Syntax

```
-------->    <stx>DGB{word#}<etx>
<--------    <stx>{word}<cr>
<--------    {Ext bit}<ack>
```

Example

```
-------->    <stx>DGB0<etx>
<--------    <stx>001X11X00XXXXXXX<cr>
<--------    1<ack>
```

2.3.6.3 trigger configuration

This selects the order of trigger words as 1 then 2. Present hardware limitations do not allow other orders (ie: 2\1).

Syntax

-------->	<stx>DGC1\2<etx)
<--------	<ack>

2.3.6.4 trigger enable/disable

Once trigger words have been defined and ordered, the feature is turned on or off with this command.

Syntax

-------->	<stx>DGD{parameter}<etx>
<--------	<ack>

Parameter

E - enable
D - disable

Example

-------->	<stx>DGDE<etx>
<--------	<ack>

2.3.6.5 read trigger configuration

Syntax

-------->	<stx>DGE<etx>
<--------	<stx>{E or D}<cr>
<--------	{word order}<ack>

Parameter

E - enabled
D - disabled
1\2 - present fixed word order

Example

--------> <stx>DGE<etx>
<-------- <stx>E<cr>
<-------- 1\2<ack>

2.3.6.6 gate configuration

Present hardware requires that word 0 be defined and chosen as the gate with this command.

Syntax

--------> <stx>DGF0<etx>
<-------- <ack>

2.3.6.7 gate enable/disable

Syntax

--------> <stx>DGG{E or D}<etx>
<-------- <ack>

Parameter

E - enable
D - disable

Example

--------> <stx>DGGD<etx>
<-------- <ack>

2.3.6.8 read gate configuration

Syntax

--------> <stx>DGH<etx>
<-------- <stx>{E or D}<cr>
<-------- {word#}<ack>

Parameter

E - enabled
D - disabled
0 - present fixed word #

Example

--------> <stx>DGH<etx>
<-------- <stx>D<cr>
<-------- 0<ack>

2.3.7 reset all parameters

All parameters are set to their default values and the unit remains in remote mode.

Syntax

--------> <stx>GA<etx>
<-------- <ack>

2.4 TEST CYCLE CONTROL

2.4.1 define reset

Reset pulse characteristics must be defined in the order shown. If no parameters are included, Reset is turned off.

Syntax

-------->	<stx>DA{Positive or Negative}<cr>
-------->	{Internal or External Vcc}<cr>
-------->	[-]{offset}<cr>
-------->	{duration}<etx>
<--------	<ack>

Parameters

P - positive
N - negative
I - internal
E - external

Example

-------->	<stx>DAP<cr>
-------->	I<cr>
-------->	-200<cr>
-------->	500<etx>
<--------	<ack>

2.4.2 read reset definition

Syntax

-------->	<stx>DB<etx>
<--------	<stx>{Positive or Negative}<cr>
<--------	{internal or External Vcc}<cr>
<--------	[-]{offset}<cr>
<--------	{duration}<etx>

Parameter

P - positive
N - negative
I - internal
E - external

Example

-------->	<stx>DB<etx>
<--------	<stx>P<cr>
<--------	I<cr>
<--------	-200<cr>
<--------	500<etx>

2.4.3 clip_chk enable/disable

Syntax

-------->	<stx>DC{parameter}<etx>
<--------	<ack>

Parameter

D - disable

E - enable

Example

-------->	<stx>DCD<etx>
<--------	<ack>

2.4.4 read clip_chk status

Syntax

-------->	<stx>DD<etx>
<--------	<stx>{parameter}<ack>

Parameter

D - disable
E - enable

Example

-------->	<stx>DD<etx>
<--------	<stx>D<ack>

2.4.5 rdtest enable/disable

Syntax

-------->	<stx>DE{parameter}<etx>
<--------	<ack>

Parameter

D - disable

E - enable

Example

```
-------->      <stx>DEE<etx>
<--------      <ack>
```

2.4.6 read rdtest status

Syntax

```
-------->      <stx>DF<etx>
<--------      <stx>{parameter}<ack>
```

Parameter

D - disable
E - enable

Example

```
-------->      <stx>DF<etx>
<--------      <stx>E<ack>
```

2.4.7 perform test

This command is equivalent to pressing the TEST key. The 3 last parameters are decimal numbers that, when converted to binary, are flags indicating failure details.

Syntax

-------->	<stx>DH<etx>
<--------	<sub>
<--------	<stx>{result}<cr>
<--------	[time to fault]<cr>
<--------	[type of fault parameter]<cr>
<--------	[condition details parameter]<cr>
<--------	[clip check parameter]<ack>

One of the following error responses will occur if an attemptis made to test in spite of an initial selftest error:

<stx>YT<nak>	Threshold improperly calibrated.
<stx>YF<nak>	FMASK improperly calibrated.

Parameter

result	P - Pass F - Fail U - Unable to test: sync, trigger or gate N - Not tested. RD test or clip check failed.	
time to fault	##.###	ns for nanoseconds us for microseconds ms for milliseconds s for seconds

type of fault (decimalized binary flags)

Bit	(1 is true, 0 is false)
1	short in RD socket
2	RD test failed
4	clip check failed
8	failed to sync
16	comparison test failed
32	condition test failed or missing trigger or gate
64	sync conditions satisfied
128	sync time expired

condition details (decimalized binary flags)

Bit	(1 is true, 0 is false)
1	frequency test failed
2	active pin check failed
4	high pin check failed
8	low pin check failed
16	no gate present
32	no trigger present
64	missing trig word 1
128	missing trig word 2

clip check (decimalized binary flags)

Bit	
1	clip not inserted
2	wrong clip
4	clip check fail
8	Vcc/gnd check failed
16	clip moved during test

Example

```
-------->     <stx>DH<etx>
<--------     <sub>
<--------     <stx>F<cr>
<--------     3.440s<cr>
<--------     48<cr>
<--------     16<cr>
<--------     0<etx>
```

2.4.8 read test results

This command prompts for the pin by pin failure results. A blank space <sp> is used to separate pin numbers from their result.

Syntax

--------> <stx>DIF<etx>
<-------- <stx>{pin#}[<sp>F]<sp>condition fail]<cr>
.
.
additional pin results <ack>

The following error response will be sent if there are no test results available:

<-------- <stx>YN<nak>

Parameters

Pin # - ##
F - fail comparison

condition fails:

L - fail low pin check
H - fail high pin check
A - fail active pin check

##.###[Hz][kHz][MHz]
- fail frequency check

Example

--------> <stx>DIF<etx>
<-------- <stx>1 A<cr>
<-------- 12<sp>4.34kHz<cr>
<-------- 4<sp>F<sp>H<ack>

2.4.9 read status results

This command prompts for the pin by pin status results during the previous test. These results are obtained from the keyboard by pressing the state key.

Syntax

```
-------->     <stx>DIS<etx>
<--------     <stx>{pin1 state}<cr>
              {pin 2 state}<cr>
              .
              .
              {last pin state}<ack>
```

No test results available:

```
<--------     <stx>YN<nak>
```

Parameter

H - high
L - low
A - active
##.###[Hz][kHz][MHz]
 - frequency mismatch

Example

```
-------->      <stx>DIS<etx>
<--------      <stx>A<cr>
               H<cr>
               5.3MHz<cr>
               .
               .
               L<etx>
```

2.5 USER COMMUNICATION

2.5.1 display text on screen

Syntax

-------->	<stx>EA{line #}<cr>
-------->	{column #}<cr>
-------->	{Normal or Reverse highlight}<cr>
-------->	{ASCII text}<etx>
<--------	<ack>

Parameter

The line # and column # specify the starting position of the text on the screen.

Example

-------->	<stx>EA5<cr>
-------->	21<cr>
-------->	R<cr>
-------->	hello<etx>
<--------	<ack>

2.5.2 read a keystroke

Syntax

-------->	<stx>EB<etx>
<--------	<sub>
<--------	<stx>{key}<ack>

Parameter

If the key pressed is a standard ASCII character, that character is returned. If the key is a special key a "*" is returned followed by a character representing that key.

Table of Special Key Codes:

	KEY	<SHFT>-KEY	<CNTR>-KEY
F1	1	D	W
F2	2	E	X
F3	3	F	Y
F4	4	G	Z
F5	5	H	[
CE	6	I	\
NEXT	7	J	]
ENTER	8	K	^
TEST	9	L	_
ESC	:	soft reset	full reset
ETC	;	N	a
CURSR UP	<	O	b
CURSR DOWN	=	P	c
CURSR LEFT	>	Q	d
CURSR RIGHT	?	R	e
TAB	@	S	f

Note: The system reset function caused by SHIFT ESC keys and CNTR ESC keys puts the tester out of remote mode.

ASCII Key A

-------->	<stx>EB<etx>
<--------	<sub>
<--------	<stx>A<ack>

Function Key F1

-------->	<stx>EB<etx>
<--------	<sub>
<--------	<stx>*1<ack>

2.5.3 read a string of keystrokes

Syntax

-------->	<stx>EC{column #}<cr>
-------->	{row #}<cr>
-------->	{max number of characters}<etx>
<--------	<sub>
<--------	<stx>{text string}<ack>

Note: Pressing ENTER key causes preceding text to be sent to the controller regardless of whether it has reached the maximum string length specified. The column and row numbers specify where the beginning of the string appears on the screen.

Example

-------->	<stx>EC5<cr>
-------->	21<cr>
-------->	5<etx>
<--------	<sub>
<--------	<stx>hello<ack>

2.5.4 generate sounds

Syntax

-------->	<stx>EDA{sound code}<etx>
<--------	<sub>
<--------	<ack>

Parameter

Sound codes: 0 - Power up
1 - Acknowledge
2 - ASCII Key
3 - Error
4 - Pass
5 - Fail

Example

-------->	<stx>EDA4<etx>
<--------	<sub>
<--------	<ack>

2.6 READING FREQUENCY AND TIMING

2.6.1 read frequency

The parameters consist of the pin number to be read preceded by an optional "S" to indicate that a slow signal is expected and a time-out should not be used. To read the frequency, timing or duty cycle on the external lead specify "E" instead of pin #.

Syntax

```
-------->    <stx>FA[S]{pin #}<etx>
<--------    <sub>
<--------    <stx>{frequency, Hz, kHz, MHz}<ack>
```

Example

```
-------->    <stx>FA15<etx>
<--------    <sub>
<--------    <stx>4.345kHz<ack>
```

2.6.2 read period

Syntax

```
-------->    <stx>FB[S]{pin #}<etx>
<--------    <sub>
<--------    <stx>{4 digit time, ns, us, ms, s}<ack>
```

Example

--------> <stx>FB15<etx>
<-------- <sub>
<-------- <stx>4.345ms<ack>

2.6.3 read high time

Syntax

--------> <stx>FC[S]{pin #}<etx>
<-------- <sub>
<-------- <stx>{4 digit time, ns, us, ms, s}<ack>

Example

--------> <stx>FCS15<etx>
<-------- <sub>
<-------- <stx>4.345ms<ack>

2.6.4 read low time

Syntax

--------> <stx>FD[S]{pin #}<etx>
<-------- <sub>
<-------- <stx>{4 digit time, ns, us, ms, s}<ack>

Example

-------->	<stx>FD15<etx>
<--------	<sub>
<--------	<stx>4.345ms<ack>

2.6.5 read duty cycle

Syntax

-------->	<stx>FE[S]{pin #}<etx>
<--------	<sub>
<--------	<stx>{4 digit ratio}<ack>

Example

-------->	<stx>FE15<etx>
<--------	<sub>
<--------	<stx>0.345<ack>

2.7 FILE MANIPULATION

2.7.1 download to unit

Syntax

--------> <stx>HA{file_name}[.type][:destination]<etx>
<-------- <sub>
--------> <stx>(file is sent)<etx>
<-------- <ack>

2.7.2 upload from unit

Syntax

--------> <stx>HB{file_name}[.type][:source]<etx>
<-------- <stx>
<-------- (file is sent)
<-------- <ack>

2.7.3 compile file

Syntax

--------> <stx>HC{file_name}[.type][:source]<etx>
<-------- <sub>
<-------- <ack>

Compiler Error Responses

<--------	<stx>Y<cr>	
<--------	S{line # of syntax error}<cr>	
<--------	D<cr>	{label found in more than one line}
<--------	U<cr>	{label not found on any line, yet referenced}
<--------	N<cr>	-.defaults missing in .loc file
<--------	L<cr>	-.loc file command missing in .seq file
<--------	<nak>	

Example of an Error Response:

-------->	<stx>HCTEST1.SEQ<etx>
<--------	<sub>
<--------	<stx>Y<cr>S15<cr>L<cr><nak>

2.7.4 delete a file

Syntax

-------->	<stx>{file_name}[.type][:device]<etx>
<--------	<ack>

2.7.5 directory of files

Directory normally returns a directory of the cartridge. Specifying the optional "S" directs it to return a directory of system memory.

Syntax

```
-------->     <stx>HE[S]<etx>
<--------     <stx> {first file name}<sp>size<cr>
<--------           {next file name}<sp>size<cr>
                    .
                    .
<--------           {last file name}<sp>size<cr>
<--------           {# bytes used}<sp>{# bytes free
                    space}<cr><ack>
```

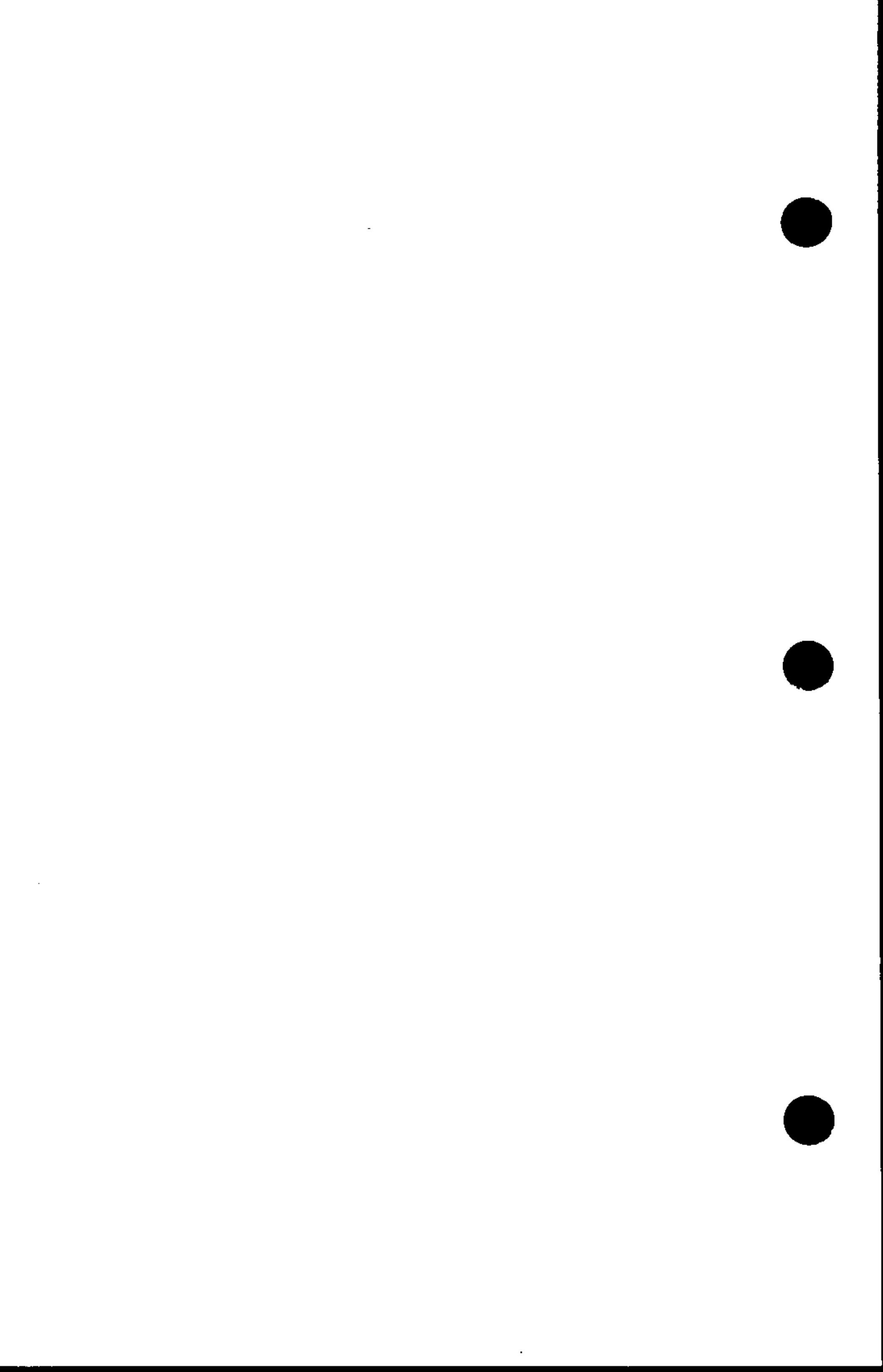
Example

```
-------->     <stx>HE<etx>
<--------     <stx>TEST1.SEQ:CART<sp>2857<cr>
<--------     TEST1.LOC:CART<sp>5326<cr>
<--------     8183<sp>Bytes<sp>used,
              <sp>24561<sp>Left<cr><ack>
```

2.7.6 format cartridge

Syntax

```
-------->     <stx>HH<etx>
<--------     <sub>
<--------     <ack>
```



Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
2509	20	1	QUAD 2-PORT REG	Yes			x						S
2522	20	1	8-BIT SER/PAR REG WITH SIGN EXT	Yes			x						
2523	20	1	8-BIT SHIFT REG/COMM I/O PINS	Yes			x						
252538	20	1	1-OF 8 DEMUX TS	Yes			x						
252548	20	1	1-OF 8 DEMUX WITH ACK	Yes			x						
25381	20	1	4-BIT ALU	Yes			x						
2923	16	1	8-INPUT MUX TS	Yes	x								
2947	20	1	OCT BIDIR TRANSC TS	Yes	x								
2965	20	2	4-BIT INV BUF/LINE DR TS	Yes	x								
2966	20	2	4-BIT NON INV BUF/LINE DR TS	Yes	x								
6330	16	1	QUAD TS BUS TRANSC INV	Yes									S
6889	16	1	QUAD TS BUS TRANSC NON INV	Yes									S
7400	14	4	2-IN NAND	Yes	x	x	x	x	x	x	x	x	AS,ACT,AC,H
7401	14	4	2-IN NAND OC	Yes	x	x	x			x	x	x	
74H01	14	4	2-IN NAND OC	Yes									H
7402	14	4	2-IN NOR	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
7403	14	4	2-IN NAND OC	Yes	x	x	x		x	x	x	x	S
7404	14	6	INVERTER	Yes	x		x	x	x	x	x	x	S,AS,ACT,AC,H
7405	14	6	INVERTER OC	Yes	x	x	x		x	x	x	x	S,ACT,AC,H
7406	14	6	INVERTER OC	Yes	x		x						
7407	14	6	BUFFER OC	Yes	x		x		x	x			

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
7408	14	4	2-IN AND	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC,H
7409	14	4	2-IN AND OC	Yes	.:	x	x			x	x	x	S,H
7410	14	3	3-IN NAND	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC,H
7411	14	3	3-IN AND	Yes	x	x	x	x	x	x	x	x	S,AC,H
7412	14	3	3-IN NAND OC	Yes	x	x	x				x	x	
7413	14	2	4-IN NAND ST	Yes	x	x	x	x			x		
7414	14	6	INVERTER ST	Yes	x	x	x	x	x	x			ACT,AC
7416	14	6	INVERTER/BUF OC	Yes	x		x						
7417	14	6	BUFFER OC	Yes	x		x						
7418	14	2	4-IN NAND	Yes			x						
7420	14	2	4-IN NAND	Yes	x	x	x	x	x	x			S,AS,ACT,AC,H
7421	14	2	4-IN AND	Yes	x	x	x	x	x	x			AS,ACT,AC,H
7424	14	4	2-IN NAND	Yes			x						
7425	14	2	4-INP NOR WITH STROBE	Yes	x								
7426	14	4	2-INPUT NAND OC	Yes	x		x						
7427	14	3	3-IN NOR	Yes	x	x	x	x	x	x			AS
7428	14	4	2-IN NOR	Yes	x	x	x						
7430	14	1	8-IN NAND	Yes	x	x	x	x	x	x			S,AS
7432	14	4	2-IN OR	Yes	x	x	x	x	x	x	x		S,AS,ACT,AC
7437	14	4	2-INPUT NAND	Yes	x	x	x	x	x				S
7438	14	4	2-INPUT NAND BUFF OC	Yes	x	x	x	x	x				S
7440	14	2	4-INP NAND BUF	Yes	x	x	x	x	x				S
7442	16	1	1_OF_10_DEMUX	Yes	x		x		x	x			
7451	14	2	AND OR INV	Yes	x					x			S,H
74L51	14	2	AND OR INV	Yes									L
74LS51	14	2	AND OR INV	Yes			x						

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
7464	14	1	4-2-3-2 IN AND/NOR	Yes									S
7473	14	2	JK-FF WITH CLEAR	Yes	x		x		x	x	x	x	
7473A	14	2	JK-FF WITH CLEAR	Yes	x								
7474	14	2	D-FF	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
7483	16	1	4-BIT ADDER WITH CARRY	Yes	x		x			x			
7485	16	1	4-BIT MAGNITUDE COMP	Yes	x		x	x	x	x			S
7486	14	4	2-INP XOR	Yes	x	x	x	x	x	x	x	x	S,ACT,AC
7490	14	1	DECADE COUNTER	Yes	x		x			x			L
7491	14	1	8-BIT SHIFT REG	Yes	x		x						L
7492	14	1	DIV BY TWELVE COUNT	Yes	x		x						
7493	14	1	BINARY COUNTER	Yes	x		x			x			
7494	16	1	4-BIT SH FT REG	Yes	x		x						
7495	14	1	4-BIT PAR-ACC SHIFT REG	Yes	x		x			x			L,AS
74107	14	2	JK-NEG EDGE FF	Yes	x		x		x	x		x	
74109	16	2	JK-FF WITH PRE AND CLR	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74112	16	2	JK-NE FF	Yes		x	x	x	x	x	x		S,ACT
74114	14	2	JK-NEG ED FF	Yes		x	x	x		x			S
74125	14	4	BUF/SEPARATE TS CONTR	Yes	x								
74126	14	4	BUF/SEPARATE TS CONTR	Yes	x								
74128	14	4	2-IN NOR LINE DRIV	Yes	x								
74132	14	4	2-IN NAND SCHMITT	Yes	x	x	x	x	x	x	x		S
74133	16	1	13-INPUT NAND	Yes	x	x	x			x			S
74134	16	1	12-INPUT NAND TS	Yes									S
74135	16	2	QUAD EX OR/NOR	Yes									S
74138	16	1	3-8 DEMUX	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
74139	16	2	2-4 DEMUX	Yes	x	x	x	x	x	x	x	x	S,ACT,AC
74140	14	2	4-IN NAND 50 OHM LINE DR	Yes									S
74147	16	1	10 TO 4 PRIORITY ENCODER	Yes	x		x		x	x			
74148	16	1	10 TO 4 LINE PRIOR ENC	Yes	x		x	x	x	x	x		S
74150	24	1	1 OF 16 SEL/MUX	Yes	x								
74151	16	1	1 OF 8 SEL/MUX	Yes	x	x	x	x	x				S,AS,ACT,AC
74153	16	2	4 TO 1 SEL/MUX	Yes	x	x	x	x	x	x			S,AS,ACT,AC
74154	24	1	4 OF 16 DEMUX	Yes	x		x	x	x	x	x		
74155	16	2	2 TO 4 DEMUX	Yes	x		x		x	x	x	x	
74156	16	2	2 TO 4 DEMUX OC	Yes	x		x						
74157	16	4	2 TO 1 MUX	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74158	16	4	2 TO 1 MUX INV	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74159	24	1	4 OF 16 DEMUX OC	Yes	x								
74160	16	1	4-BIT SYNC DEC COUNT	Yes	x	x	x	x	x	x	x	x	S,AS,AC
74161	16	1	4-BIT SYNC BIN COUNT/ASYN CLR	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74162	16	1	4-BIT SYNC DEC COUNT	Yes	x	x	x	x	x	x	x	x	S,AS
74163	16	1	4-BIT BIN COUNT	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74164	14	1	8-BIT PAR-OUT SHIFT REG	Yes	x	x	x	x	x	x	x	x	ACT,AC
74165	16	1	8-BIT SHIFT REG	Yes	x	x	x	x	x	x	x	x	
74166	16	1	8-BIT SHIFT REG	Yes	x	x	x	x		x	x	x	
74168	16	1	4-BIT U/D DEC SYN COUNT	Yes		x	x	x		x			AC
74169	16	1	4-BIT BIN SYN COUNT	Yes		x	x	x	x	x			S,AS
74173	16	1	4-BIT D TYPE REG TS	Yes	x		x			x	x	x	
74174	16	6	D-FF	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74175	16	4	D-FF	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
74180	14	1	9-BIT ODD/EVEN PAR GEN/CHEC	Yes	x					x			
74181	24	1	ARITH/LOG UNIT/FUNC GEN	Yes	x	x	x	x	x	x			S,AS
74182	16	1	LOOK AHEAD CARRY GENERATOR	Yes	x		x	x	x	x	x		S,AS
74190	16	1	SYNC UP/DOWN DEC COUNT	Yes	x	x	x	x	x	x			
74191	16	1	SYNC UP/DOWN BIN COUNT	Yes	x	x	x	x	x	x	x	x	ACT,AC
74192	16	1	SYNC U/D DEC COUNT DUAL CLOCK	Yes	x	x	x	x	x	x			
74193	16	1	SYNC U/D BIN COUNT DUAL CLOCK	Yes	x	x	x	x	x	x	x	x	ACT,AC
74194	16	1	4-BIT LEFT/RIGHT SHIFT REG	Yes	x	x	x	x	x	x			S,AS
74195	16	1	4-BIT PAR ACCESS SHIFT REG	Yes	x		x	x	x	x			S
74240	20	2	4-BIT BUFFER	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74241	20	2	4-BIT BUFFER	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74242	14	1	QUAD INV BUS TRANS	Yes	x	x	x	x	x	x			S,AS
74243	14	1	QUAD NON INV BUS TRANS	Yes	x	x	x	x	x	x			S,AS
74244	20	2	4-BIT BUFFER	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74245	20	1	8-BIT TRANCEVER	Yes	x	x	x	x	x	x	x	x	AS,ACT,AC
74251	16	1	8-TO 1 MUX TS	Yes	x	x	x	x	x	x			S,AS,ACT
74253	16	1	DUAL 4 TO 1 MUX TS	Yes	x	x	x		x	x			S,AS,ACT,AC
74257	16	4	2 TO 1 MUX TS	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74258	16	4	2 TO 1 MUX TS	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74259	16	1	8-BIT ADDRESSABLE LATCHES	Yes	x	x	x	x	x	x	x	x	
74260	14	2	5-INPUT NOR	Yes			x	x					S
74265	16	4	COMPLEMENTARY OUT ELEMENTS	Yes	x								
74266	14	4	2-INPUT XOR OC	Yes						x	x	x	
74273	20	8	D-FF WITH COMM CLR AND CLOCK	Yes	x	x	x	x	x	x	x	x	S,ACT,AC
74280	14	1	9-BIT PARITY GENERATOR	Yes	x	x	x	x	x	x	x		S,AS,ACT,AC

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
74287	16	1	PROM 256x4	Yes									S
74298	16	1	QUAD 2 INP MUX WITH STORAGE	Yes	x		x						AS
74299	20	1	8-BIT SHIFT REG 3 ST OUT	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74322	20	1	8-BIT SER/PAR REG/SIGN EXT	Yes			x	x	x	x			
74323	20	1	8-BIT SHIFT/STORE REG TS	Yes		x	x	x		x			AS,ACT
74352	16	1	DUAL 4 IN MUX	Yes	x	x	x	x		x			AS
74353	16	1	DUAL 4 IN MUX TS	Yes		x	x	x		x			
74354	20	1	8 TO 1 MUX/REG	Yes			x		x	x			
74356	20	1	8 TO 1 MUX/REG	Yes			x		x	x			
74365	16	1	6-BIT BUFFER	Yes	x		x	x		x	x	x	
74366	16	1	6-BIT INV BUF	Yes	x		x	x		x	x	x	
74367	16	1	4-BIT TS+2-BIT TS BUF	Yes	x		x	x		x	x	x	
74368	16	1	4-BIT INV 3 ST+2-BIT INV 3 ST BUF	Yes	x		x	x	x	x	x	x	
74373	20	1	8-BIT D LATCH TS	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74374	20	1	8-BIT D-FF TS	Yes	x	x	x	x	x	x	x	x	S,AS,ACT,AC
74377	20	1	OCTAL D FF WITH ENABLE	Yes	x	x	x	x	x	x	x		ACT,AC
74378	16	6	D FF WITH ENABLE	Yes		x	x	x		x			ACT,S
74379	16	1	QUAD D FF WITH ENABLE	Yes			x	x		x			AC
74381	20	1	4-BIT ALU	Yes			x	x	x				S
74390	16	2	4-BIT DECADE CUNTER	Yes	x		x		x	x			
74393	16	2	4-BIT BIN COUNTER	Yes	x		x	x	x	x	x	x	
74395	16	1	4-BIT SHIFT REG TS	Yes			x	x					
74398	20	1	QUAD 2-PORTS REG	Yes			x	x					
74399	16	1	QUAD 2-PORTS REG	Yes			x	x			x	x	S
74412	24	1	MULTIMODE BUFFERED LATCHES	Yes				x					S

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
74425	14	4	BUS BUFFERS INV TS	Yes	x								
74426	14	4	BUS BUFFERS TS	Yes	x								
74521	20	1	8_BIT_IDENTITY_COMP	Yes		x		x					
74533	20	1	OCT D TYPE TRANS LATCH TS	Yes		x	x	x	x	x	x	x	S,AS,ACT,AC
74534	20	1	OCT D TYPE FF TS	Yes		x	x	x	x	x	x	x	S,AS,ACT,AC
74537	20	1	1-OF 10 DEMUX TS	Yes				x					
74538	20	1	1OF 8 DEMUX TS	Yes				x					
74539	20	2	1 OF 4 DEMUX TS	Yes				x					
74540	20	1	OCT INV BUFF LINE DR TS	Yes		x	x	x	x	x	x	x	ACT,AC
74541	20	1	OCT NON INV BUFF LINE DRV TS	Yes		x	x	x	x	x	x	x	ACT,AC
74545	20	1	OCT TRANS TS BIDIREC	Yes				x	x				
74547	20	1	1-OF 8 DEMUX&LATCH AND ACKN	Yes				x					
74548	20	1	1-OF 8 DEMUX WITH ACKN	Yes			x	x					
74564	20	1	OCT_D_TYPE_FF_3_ST_OUT	Yes		x	x	x	x	x	x	x	ACT,AC
74573	20	1	8-BIT D-LATCH TS	Yes		x	x	x	x	x	x	x	S,AS,ACT,AC
74574	20	1	8-BIT D FF TS	Yes		x	x	x	x	x	x	x	AS,ACT,AC
74588	20	1	OCT TRANS TS BIDIREC	Yes				x					
74595	16	1	8_BIT_SH_RG_WITH_OUTPUT_LATCHES	Yes			x			x			
74620	20	1	OCT INV BUS TRANSC TS	Yes		x	x	x	x	x			ACT,AC,AS
74621	20	1	OCT BUS TRANSC OC	Yes		x	x	x					AS
74622	20	1	OCT INV BUS TRANSC OC	Yes		x	x	x					AS
74623	20	1	OCT BUS TRANSC TS	Yes		x	x	x	x	x			ACT,AC
74638	20	1	OCT_BUS_TRANSC	Yes		x	x						
74639	20	1	OCT_BUS_TRANSC	Yes		x	x						
74640	20	1	OCT BUS TRANSC INV	Yes		x	x	x	x	x	x	x	AS,ACT,AC
74641	20	1	OCT BUS TRANS INV OC	Yes		x	x	x					AS

Library version 4.00

CHIP NAME	CHIP SIZE	DEV/p CHIP	DEVICE TYPE	PATTERN	STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
					TECHNOLOGY								
74642	20	1	OCT BUS TRANS INV OC	Yes		x	x	x					AS
74643	20	1	OCT BUF A->B INV & B->A NON-INV TS	Yes		x	x	x	x	x	x	x	AS,ACT,AC
74644	20	1	OCT BUF A->B INV & B->A NON-INV OC	Yes		x	x						
74645	20	1	OCT BUS TRANS TS	Yes	x	x	x	x	x	x	x		AS,ACT,AC
74646	24	1	OCT TRANSCEVER/REGISTER	Yes		x	x	x	x	x	x	x	AS,ACT,AC
74648	24	1	OCT TRANSCEVER/REGISTER	Yes		x	x	x	x	x	x	x	AS,ACT,AC
74670	16	1	4-BY 4 REG WITH TS OUT	Yes	x		x		x	x	x		
74688	20	1	8 BIT IDENTITY COMP	Yes	x	x	x		x	x		x	
74689	20	1	8 BIT IDENTITY COMP OC	Yes		x	x						
741000	14	4	2-IN NAND	Yes		x							AS
741002	14	4	2-IN NOR	Yes		x							
741003	14	4	2-IN NAND OC	Yes		x							
741004	14	6	INVERT	Yes		x							AS
741005	14	6	INVERT OC	Yes		x							
741008	14	4	2-IN AND	Yes		x							AS
741010	14	3	3-IN NAND	Yes		x							ACT,AC
741011	14	3	3-IN NAND	Yes		x							
741020	14	2	4-IN NAND	Yes		x							
741032	14	4	2-IN OR	Yes		x							AS
741035	14	6	BUFFER OC	Yes		x							
741181	24	1	ARITH/LOGIC UNIT/FUNC GEN	Yes		x							AS
741240	20	2	4-BIT BUFF LINE DR TS	Yes		x							
741241	20	2	4-BIT BUFF TS	Yes		x							
741242	14	1	4-BIT BUS TRANSC TS	Yes		x							

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
741244	20	2	4-BIT BUFF/DR TS	Yes		x							
741245	20	1	OCT BUS TRANSC TS	Yes		x							
741640	20	1	OCT_BUS_TRANSC_INV	Yes		x							
741645	20	1	OCT BUS TRANSC NON INV TS	Yes		x							
9002	14	4	2-IN NAND	Yes	x								
9003	14	3	3-IN NAND	Yes	x								
9004	14	2	4-IN NAND	Yes	x								
9009	14	2	4-IN NAND	Yes	x								
9012	14	4	2-IN NAND	Yes	x								
9016	14	6	INVERTER	Yes	x								
9024	16	2	J-K/ FF WITH PRE & CLR	Yes	x								
9311	24	1	1-OF 16 DEMUX	Yes	x								L
9334	24	1	1-OF 16 DEMUX	Yes	x								L
9341	24	1	ALU/FUNCTION GENERATOR	Yes	x								S,L

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
DYNAMIC RAMS													
2164	16	1	64Kx1	Yes		MOS							
2620	18	1	16Kx4	Yes		MOS							
2800	16	1	256Kx1	Yes		MOS							
41256	16	1	256Kx1	Yes		MOS							
41416	18	1	16Kx4	Yes		MOS							
4164	16	1	64Kx1	Yes		MOS							
41128	16	1	128Kx1	No		MOS							
41256	16	1	256Kx1	Yes		MOS							
4416	18	1	16Kx4	Yes		MOS							
6256	16	1	256Kx1	Yes		MOS							
6665	16	1	64Kx1	Yes		MOS							
81416	18	1	16Kx4	Yes		MOS							
81256	16	1	256Kx1	Yes		MOS							
8264	16	1	64Kx1	Yes		MOS							

(TECHNOLOGY spans STAND through OTHERS)

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
EPROMS													
2732	24	1	4Kx8 EPROM	Yes		MOS							
2764	28	1	8Kx8 EPROM	Yes		MOS							
27128	28	1	16Kx8 EPROM	Yes		MOS							
27256	28	1	32Kx8 EPROM	Yes		MOS							
27512	28	1	64Kx8 EPROM	Yes		MOS							
PROMS													
82S129	16	1	PROM_256x4	Yes									S
TBP24S10	16	1	PROM_256x4	Yes									S

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY STAND	ALS	LS	F	HCT	HC	AHCT	HCTLS	OTHERS
STATIC RAMS													
1400	20	1	16Kx1	Yes		MOS							
1822	22	1	256x4	Yes		MOS							
2016	24	1	2Kx8	Yes		MOS							
2018	24	1	2Kx8	Yes		MOS							
2068	28	1	8Kx8	Yes		MOS							
2101	22	1	256x4	Yes		MOS							
2114	18	1	1Kx4	Yes		MOS							
2147	18	1	4Kx1	Yes		MOS							
2148	18	1	1Kx4	Yes		MOS							
2167	20	1	16Kx4	Yes		MOS							
4016	24	1	2Kx8	Yes		MOS							
4311	20	1	16Kx4	Yes		MOS							
5047	20	1	1Kx4	Yes		MOS							
5114	18	1	1Kx4	Yes		MOS							
5517	24	1	2Kx8	Yes		MOS							
5564	28	1	8Kx8	Yes		MOS							
5565	28	1	8Kx8	Yes		MOS							
6147	18	1	4Kx1	Yes		MOS							
6164	28	1	8Kx8	Yes		MOS							
8167	20	1	16Kx4	Yes		MOS							
9044	18	1	4Kx1	Yes		MOS							
9101	22	1	256x4	Yes		MOS							
9111	18	1	256x4	Yes		MOS							
9124	18	1	1Kx4	Yes		MOS							
93415	16	1	1Kx1	Yes		MOS							
93422	22	1	256x4	Yes		MOS							
93425	16	1	1Kx1	Yes		MOS							
9983	28	1	8Kx8	Yes		MOS							

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY
LSI					
8212	24	1	MULTIMODE BUFFERED LATCHES	Yes	TTL
8253	24	1	PROGRAMMABLE INTERVAL TIMER	No	MOS
8259	28	1	PROGRAMMABLE INTERRUPT CONTROLLER	Yes	MOS
8286	20	1	OCT BUS TRANSCEIVER	Yes	TTL
8288	20	1	BUS CONTROLLER FOR 86/88	No	MOS
PALS					
PAL10H8	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL12H6	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL14H4	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL16H2	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL16C1	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL10L8	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL12L6	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL14L4	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL16L2	20	1	AND OR PAL	Yes	BIPOLAR TTL
PAL16L8	20	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL16R8	20	1	AND OR D FF PAL	Yes	BIPOLAR TTL
PAL16R6	20	1	AND OR D FF PAL	Yes	BIPOLAR TTL
PAL16R4	20	1	AND OR D FF PAL	Yes	BIPOLAR TTL
PAL16X4	20	1	AND OR XOR D FF PAL	Yes	BIPOLAR TTL
PAL16A4	20	1	AND OR XOR D FF PAL	Yes	BIPOLAR TTL

Library version 4.00

CHIP NAME	CHIP SIZE	DEVp CHIP	DEVICE TYPE	PATTERN	TECHNOLOGY
PAL16P8	20	1	AND OR/NOR PAL	Yes	BIPOLAR TTL
PAL16RP8	20	1	AND OR/NOR D FF PAL	Yes	BIPOLAR TTL
PAL16RP6	20	1	AND OR/NOR D FF PAL	Yes	BIPOLAR TTL
PAL16RP4	20	1	AND OR/NOR D FF PAL	Yes	BIPOLAR TTL
PAL12L10	24	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL14L8	24	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL16L6	24	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL18L4	24	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL20L2	24	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL20C1	24	1	AND OR/NOR PAL	Yes	BIPOLAR TTL
PAL20L10	24	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL20X10	24	1	AND OR XOR D FF PAL	Yes	BIPOLAR TTL
PAL20X8	24	1	AND OR XOR D FF PAL	Yes	BIPOLAR TTL
PAL20X4	24	1	AND OR XOR D FF PAL	Yes	BIPOLAR TTL
PAL20L8	24	1	AND NOR PAL	Yes	BIPOLAR TTL
PAL20R8	24	1	AND OR D FF PAL	Yes	BIPOLAR TTL
PAL20R6	24	1	AND OR D FF PAL	Yes	BIPOLAR TTL
PAL20R4	24	1	AND OR D FF PAL	Yes	BIPOLAR TTL
PAL20S10	24	1	AND OR/NOR PAL	Yes	BIPOLAR TTL
PAL20RS10	24	1	AND OR/NOR D FF PAL	Yes	BIPOLAR TTL
PAL20RS8	24	1	AND OR/NOR D FF PAL	Yes	BIPOLAR TTL
PAL20RS4	24	1	AND OR/NOR D FF PAL	Yes	BIPOLAR TTL
PAL20RA10	24	1	AND OR/NOR D FF PAL	Yes	BIPOLAR TTL
PAL22V10	24	1	AND OR D FF MUX PAL	Yes	BIPOLAR TTL