

RK06

DISK DRIVE DIAG PART 2
MD-11-DZR6I-B

EP-DZR6I-B-DL-A
COPYRIGHT © 1976
FICHE 1 OF 2

NOV 1976
digital
MADE IN USA

This microfiche card contains a grid of 24 columns and 16 rows of frames. Each frame is a small, square image, likely a page from a technical manual or document. The text within the frames is too small to be legible, but the layout suggests a structured document with multiple pages of content. The card is dark blue with white frames, and the overall appearance is that of a standard microfiche format used for data storage and distribution in the 1970s.

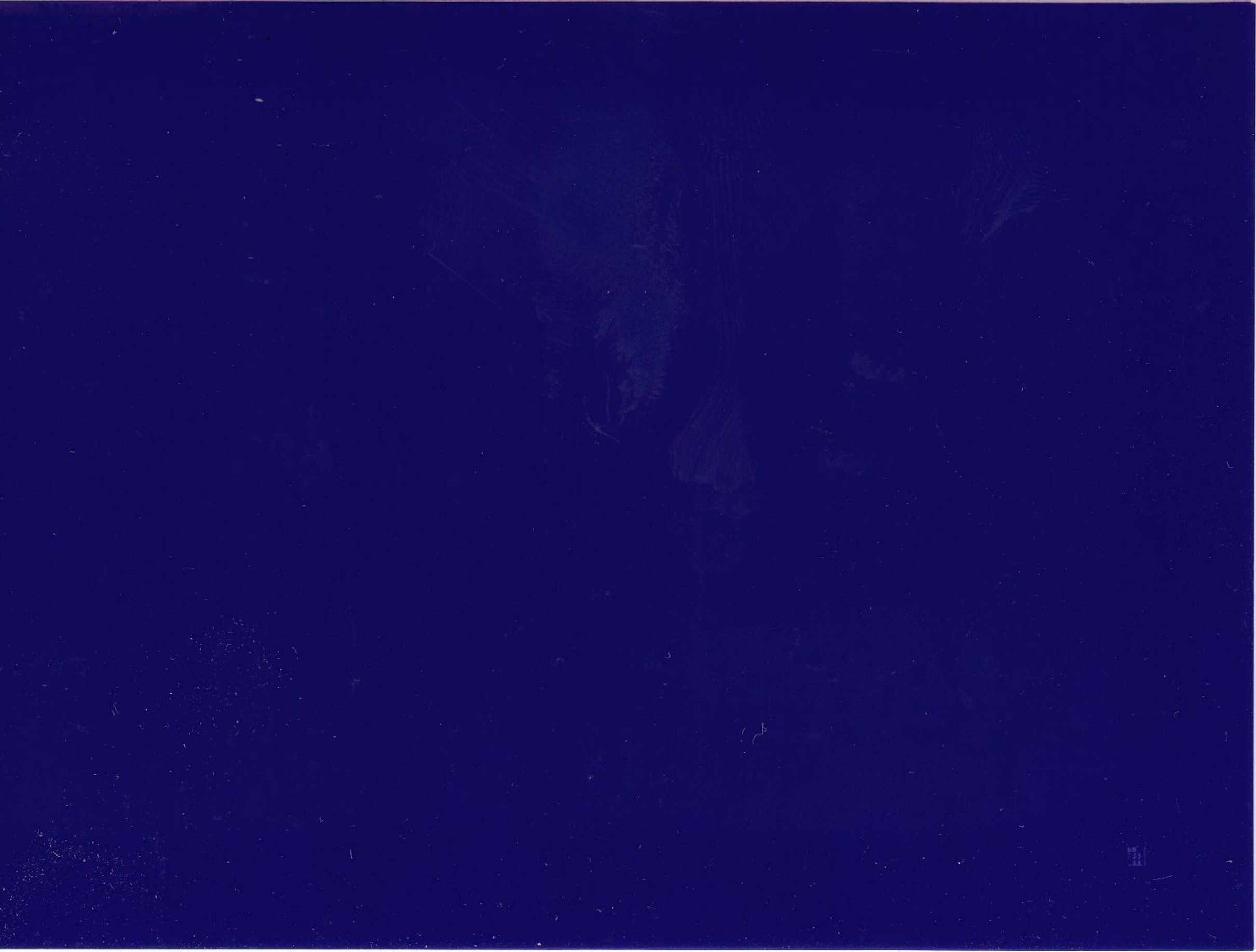
RK06

UNIBUS DIAG. PART 2
MD-11-DZR6I-B

EP-DZR6I-B-DL-A
COPYRIGHT © 1976
FICHE 2 OF 2

NOV 1976
digital
MADE IN US

This vertical strip contains 16 small, illegible diagnostic charts or data tables. Each chart appears to have a grid-like structure with some text or numbers, but the details are too small to read. They are arranged in two columns of eight.



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-CZR6I-B-D
PRODUCT NAME: UNIBUS RK06 DISK DRIVE DIAGNOSTIC: PART 2
DATE: AUGUST 1976
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: GARY PAPAIZIAN

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 HARDWARE
 - 2.2 PRELIMINARY TESTING & PROGRAMS
- 3.0 PROGRAM CONSIDERATIONS
 - 3.1 PDP-11 FAMILY COMPATIBILITY
 - 3.2 XDP
 - 3.3 ACT/APT
 - 3.4 DUAL ACCESS
 - 3.5 MEMORY MANAGEMENT
 - 3.6 PARITY CHECK ENABLED
 - 3.7 BAD SECTORS
 - 3.8 EXECUTION TIME
 - 3.9 FAULT ISOLATION
 - 3.10 ERROR CORRECTION & FAILURE RATE ANALYSIS
 - 3.11 DEFAULT UNIBUS ADDRESSES & VECTORS
- 4.0 OPERATING PROCEDURE & CONTROL FUNCTIONS
 - 4.1 PROGRAM LOADING
 - 4.2 STARTING LOCATIONS
 - 4.3 CONSOLE SWITCH REGISTERS
 - 4.4 SOFTWARE SWITCH REGISTER
 - 4.5 INPUT DIALOGUE
 - 4.6 PROGRAM EXAMPLE
- 5.0 DRIVE DIAGNOSTIC FUNCTIONAL DESCRIPTION
 - 5.1 GENERAL
 - 5.2 TEST DESCRIPTIONS
- 6.0 ERROR REPORTING
 - 6.1 ERROR INTERPRETATION
 - 6.2 ERROR PRINTOUT EXAMPLE

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 2 OF THE DRIVE DIAGNOSTICS TO INSURE THAT THE DISK IS CAPABLE OF PERFORMING READ AND WRITE DATA OPERATIONS IN BOTH 20 AND 22 SECTOR FORMATS. WORST CASE PATTERNS, SPIRAL WRITING AND READING, AND ALL OFFSET OPERATIONS ARE PERFORMED. ALSO, UNLOADING AND LOADING TIMES ARE REPORTED ALONG WITH ROTATIONAL MIN, MAX, 137 CYLINDER & MAX VELOCITY TIMES. ERROR DETECTION LOGIC IS CHECKED BY SOFTWARE ERROR FORCING.

AFTER A SUCCESSFUL RUN (WITH NO ERRORS) OF PART 2, THE DRIVE IS READY FOR PART 3 OF THE DRIVE DIAGNOSTICS.

TESTING IS BASED ON A HIERARCHY APPROACH STARTING WITH BASIC LOGIC TESTS AND PROCEEDING THRU DYNAMIC TESTING. THE TESTS WILL BE KEPT SMALL TO FACILITATE SCOPING LOGPS.

*****CAUTION*****

HALTING THIS PROGRAM ANYWHERE BUT AT THE END OF A PASS, MAY LEAVE THE HEADERS IN THE DISK CARTRIDGE IN AN UNDETERMINED STATE.

2.0 REQUIREMENTS

2.1 HARDWARE

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE DISK DIAGNOSTIC:

PDP-11
CONSOLE TELETYPE
16K MEMORY
KH11-L OR KH11-P CLOCK
RK06 UNIBUS CONTROLLER (RK611)
1 TO 8 RK06 DRIVES

- NOTES: 1. IF NEITHER KH11-L OR P CLOCK IS USED, ALL TIMING TESTS WILL BE BYPASSED. A MESSAGE AT THE BEGINNING OF THE TESTS WILL CONFIRM THIS.
2. THE PROGRAM CAN WORK OFF EITHER FORMATTED OR NON-FORMATTED PACKS.

2.2 PRELIMINARY TESTING & PROGRAMS

THE RK611 DISKLESS CONTROLLER DIAGNOSTICS (ALL PARTS) SHOULD

FIRST RUN SUCCESSFULLY FOLLOWED BY THE RK06 DRIVE DIAGNOSTIC- PART 1.

3.0 PROGRAM CONSIDERATIONS

3.1 PDP-11 FAMILY COMPATIBILITY

THIS PROGRAM CAN BE USED BY THE PDP-11/04,05,10,20,
34,35,40,45,50, & 70.

IT IS COMPATABLE WITH THE LSI-11 INSTRUCTION SET AND CAN TEST
THE RK06 ONLY IF THE DRIVE CONTROLLER FOR THE LSI-11 IS
DESIGNED TO BE DIAGNOSTICALLY COMPATABLE WITH THE RK611.

3.2 XXDP

THIS PROGRAM CAN BE CHAINED BY XXDP & WILL NOT OVERLAY THE
LOADER.

CHAIN MODE OPERATION (MONITOR)

1. THE INPUT DIALOGUE IS BYPASSED.
2. THE BUSS ADDRESS & CONTROLLER INTERRUPT VECTOR IS
DEFAULTED.
3. DRIVE 0 WILL NOT BE TESTED.
4. ALL OTHER DRIVES IN THE 'DRIVE PRESENT' CONDITION WILL
BE TESTED.

NOTE: THE DRIVE PRESENT CONDITION IS:

- A. HEADS MANUALLY LOADED
- B. CORRECT PORT SELECTED
- C. WRITE LOCK DISABLED
- D. DRIVE READY INDICATOR ON

DUMP MODE OPERATION (MANUAL)

1. INPUT DIALOGUE IF STARTED FROM 220.
2. DRIVE 0 CAN BE TESTED, BUT THE OPERATOR IS FIRST GIVEN
A MESSAGE TO REPLACE THE PACK IN DPO WITH A SCRATCH
PACK & TYPE <CR> WHEN DONE.

3.3 ACT/APT

THIS PROGRAM IS ACT COMPATIBLE. IT IS APT

COMPATABLE TO THE EXTENT THAT APT HOOKS WILL BE IN THE PROGRAM & WILL WORK THRU THE 'UPTON INTERFACE'.

FOR OTHER INTERFACES, APT MAY ONLY LOAD & START THE PROGRAM. I.E. LOAD & DUMP MODE.

AUTOMATIC MODE (MONITOR)

1. THE INPUT DIALOGUE IS BYPASSED.
2. THE BUSS ADDRESS & CONTROLLER INTEPRUPT VECTOR IS DEFAULTED.
3. ALL DRIVES IN THE 'DRIVE PRESENT' CONDITION WILL BE TESTED.

NOTE: THE DRIVE PRESENT CONDITION IS:

- A. HEADS MANUALLY LOADED
- B. CORRECT PORT SELECTED
- C. WRITE LOCK DISABLED
- D. DRIVE READY INDICATOR ON

NOTE: SEVERAL APT CONSIDERATIONS ARE STILL TO BE DEFINED.

DUMP MODE (MANUAL): INPUT DIALOGUE IF STARTED FROM 220.

3.4 DUAL ACCESS

THIS PROGRAM WILL NOT TEST OR SUPPORT DUAL-ACCESS. A DRIVE EQUIPED WITH DUAL ACCESS MUST BE SWITCHED TO THE PORT UNDER TEST TO PREVENT CONTENTION WITH THE OTHER PORT.

DUAL ACCESS TESTS WILL BE INCORPORATED IN A SEPARATE PROGRAM AT A LATER DATE.

3.5 MEMORY MANAGEMENT

IF THE MEMORY PARITY CHECK OPTION IS AVAILABLE ON THE SYSTEM, THE PROGRAM WILL RUN WITH MEMORY CHECK ENABLED.

3.6 PARITY CHECK ENABLED

IF THE MEMORY PARITY CHECK OPTION IS AVAILABLE ON THE SYSTEM, THE PROGRAM WILL RUN WITH MEMCRY CHECK ENABLED.

3.7 BAD SECTOR

THE PROGRAM WILL COMPARE DATA ERRORS WITH THE BAD SECTOR INFORMATION CONTAINED ON CYLINDER 410, HEAD 2. PRINTOUTS OF DATA ERRORS DUE TO BAD SECTORS/TRACKS WILL BE MASKED OUT.

3.8 EXECUTION TIME

THE EXECUTION TIMES SHOWN BELOW ARE BASED ON THE PDP 11/50.

TOTAL TIME: 2 MIN, 40 SEC

A BREAKDOWN OF THE MORE LENGTHY TESTS ARE SHOWN BELOW:

TEST 20	DRIVE OFF TRACK	45 SEC
TEST 21	UNLOAD HEADS	10 SEC
TEST 22	LOAD HEADS	20 SEC
TEST 23	ROTATIONAL TIMING	15 SEC
TEST 24	MAX SEEK TIMES	15 SEC
TEST 26	137 CYL SEEK TIMES	10 SEC
TEST 27	MAX VELOCITY TIMES	15 SEC

3.9 FAULT ISOLATION

FAULT ISOLATION WILL NOT BE PERFORMED FOR THE FIRST STAGE RELEASE BUT WILL BE INCLUDED FOR THE SECOND STAGE RELEASE.

3.10 ERROR CORRECTION AND FAILURE RATE ANALYSIS

THIS PROGRAM WILL NOT DO ERROR CORRECTION OF FAILURE RATE ANALYSIS.

3.11 DEFAULT UNIBUS ADDRESSES & VECTORS

THE FOLLOWING IS A LIST OF ALL DEFAULT ADDRESSES & VECTORS OF ALL HARDWARE TO BE USED & THEIR MEMORY ADDRESSES WHERE THEY CAN BE CHANGED.

	LOCATION	DEFAULT CONTENTS
RK06 BUSS ADDRESS	1264	177440
CONTROLLER INTERRUPT VECTOR	1334	210
CONTROLLER PRIORITY	1336	240
P-CLOCK STATUS REG	1340	172540
P-CLOCK SET BUFFER	1342	172542
P-CLOCK READ BUFFER	1344	172544
L-CLOCK STATUS REG	1346	177546
L-CLOCK INTERRUPT VECTOR	1350	100

P-CLOCK INTERRUPT VECTOR	1352	104
TTY KB STATUS REG	1144	177560
TTY KB BUFFER	1146	177562
TTY PRINTER STATUS REG	1150	177564
TTY PRINTER BUFFER	1152	177566

4.0 OPERATING PROCEDURE & CONTROL FUNCTIONS

4.1 PROGRAM LOADING

THE PROGRAM CAN BE LOADED FROM PAPER TAPE USING STANDARD PROCEDURE FOR ABSOLUTE LOADER TAPES; OR FROM ANY MEDIA SUPPORTED BY XXDP.

4.1.1 LOAD THE STARTING ADDRESS (SEE SEC 4.2).

4.1.2 SET SWITCH REGISTERS AS DESIRED (SEE SEC 4.3).

4.1.3 SET DRIVES TO BE TESTED IN THE 'LOAD' CONDITION & WITH THE APPROPRIATE PORT SELECTED & WRITE LOCK DISABLED. DRIVES NOT TO BE TESTED MUST HAVE BOTH PORTS DESELECTED.

NOTE: THE DRIVE WILL NOT RESPOND TO THE 'START SPINDLE' COMMAND IF THE RUN/STOP SWITCH IS IN THE 'STOP' POSITION.

4.1.4 PRESS 'START'

THE PROGRAM WILL IDENTIFY ITSELF AND WILL BEGIN A DIALOGUE WITH THE OPERATOR TO DETERMINE DRIVES TO BE TESTED (SEE SEC 4.5).

THE PROGRAM BEGINS TESTING ONLY THOSE DRIVES SPECIFIED BY THE INPUT DIALOGUE. IF A SPECIFIED DRIVE CANNOT BE FOUND BY THE PROGRAM IT WILL BE FLAGGED AS AN ERROR THAT THE DRIVE WAS NOT AVAILABLE. THEN BEGINNING WITH THE LOWEST NUMERICAL DRIVE AND PROCEEDING IN SEQUENTIAL ORDER, ALL VALID DRIVES WILL BE TESTED. ONE PASS THROUGH THE TEST SEQUENCE WILL BE PERFORMED ON EACH DRIVE BEFORE MOVING TO THE NEXT DRIVE IN SEQUENCE. THE DRIVE TO BE TESTED WILL BE TYPED AT THE BEGINNING OF EACH PASS. "END OF PASS" WILL BE TYPED AFTER TESTING ALL DRIVES.

4.2 STARTING LOCATIONS

LOCATION 200 - STARTING ADDRESS TO DEFAULT THE BUSS ADDRESS & THE CONTROLLER INTERRUPT VECTOR & TEST ALL DRIVES IN THE 'DRIVE PRESENT' CONDITION.

NOTE: THE DRIVE PRESENT CONDITION IS:

- A. HEADS MANUALLY LOADED
- B. CORRECT PORT SELECTED
- C. WRITE LOCK DISABLED
- D. DRIVE READY INDICATOR ON

LOCATION 204 - SAME AS 200 START BUT BYPASS WRITE TESTS.

LOCATION 214 - SAME AS 200 START BUT BYPASS TIMING TESTS.

LOCATION 220 - STARTING ADDRESS TO INPUT TESTING PARAMETERS VIA THE INPUT DIALOGUE. BUSS ADDRESS & CONT. INTERRUPT VECTOR INPUTTED ONLY ON 1ST PASS.

LOCATION 224 - SAME AS 220 START BUT BYPASS WRITE TESTS.

LOCATION 230 - SAME AS 220 START BUT BYPASS TIMING TESTS.

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT CONSIDERATIONS IN SECTIONS 3.2 & 3.3.

4.3 SWITCH REGISTER

THE SWITCHES ARE USED TO PROVIDE CONTROL FUNCTIONS.

SWITCH	FUNCTION
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUT
12	BYPASS DRIVE AFTER 20 ERRORS
11	INHIBIT ITERATION
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SW<07:00>

J01

4.3.1 SW<15>

THE PROGRAM HALTS ON ENCOUNTERING AN ERROR, AFTER TYPING OUT THE ERROR MESSAGE AND PERTINENT INFORMATION, IF SW13=0. PRESSING "CONTINUE" RESTORES NORMAL OPERATION OF THE PROGRAM.

4.3.2 SW<14>

THE PROGRAM LOOPS ON THE TEST THAT IS BEING EXECUTED WHEN THE SWITCH IS PUT ON. THIS SWITCH IS NORMALLY USED ALONG WITH SW15.

4.3.3 SW<13>

THIS SWITCH INHIBITS ALL ERROR MESSAGES. NORMALLY USED WHEN LOOPING ON TEST (SW14) OR LOOPING ON ERROR (SW9).

4.3.4 SW<12>

THIS SWITCH BYPASSES A GIVEN DRIVE AFTER 20 ERRORS HAVE BEEN DETECTED.

4.3.5 SW<11>

EACH TEST WILL BE EXECUTED ONLY ONCE. NORMALLY AFTER THE FIRST PASS, EACH SUBTEST IS ITERATED A NUMBER OF TIMES (USUALLY 50, 5 IN SOME CASES). SETTING THIS SWITCH INHIBITS ITERATIONS, SO THAT QUICK PASSES CAN BE MADE.

4.3.6 SW<10>

RINGS A BELL ON ERROR. USEFUL WHEN ERROR TYPEOUT IS INHIBITED.

4.3.7 SW<09>

THIS SWITCH PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP FOR ERRORS. IF THE PROGRAM DETECTS AN ERROR, IT WILL LOOP BACK TO THE BEGINNING OF TEST.

4.3.8 SW<08>

THIS SWITCH IS USED TO SELECT A PARTICULAR TEST (AS PER SW<00-7>) FOR EXECUTION AND SUBSEQUENT LOOPING. THUS IF

TEST 15 IS TO BE SELECTED THE SWITCH SETTING WOULD BE 000415. IT SHOULD BE NOTED THAT BEFORE SELECTING TEST 15, ALL THE PREVIOUS TESTS (1-14) WILL BE EXECUTED.

4.4 'SOFTWARE' SWITCH REGISTER

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/04 OR 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RK06 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

SWR = NNNNNN NEW =

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

4.5 INPUT DIALOGUE

THE DIALOGUE WILL BE DONE INTERACTIVELY. THE PROGRAM WILL REQUEST A PARAMETER BY CONSOLE TYPEOUT. THE PARAMETER MAY THEN BE ENTERED AS SPECIFIED BELOW OR ALLOWED TO DEFAULT BY A CARRIAGE RETURN. UNRECOGNIZED OR ILLEGAL RESPONSES WILL BE ECHOED BACK FOLLOWED BY "?". THE PROPER RESPONSE MAY THEN BE ENTERED.

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT CONSIDERATIONS IN SECTIONS 3.2 & 3.4.

4.5.1 DRIVE SELECTION

THE REQUEST WILL BE:

DRIVES TO BE TESTED:

THE DEFAULT RESPONSE IS CARRIAGE RETURN TO TEST ALL DRIVES
IN THE 'DRIVE PRESENT' CONDITION.

THE OPERATOR CAN ALSO TYPE IN THE SPECIFIC DRIVE NUMBERS
TO BE TESTED, SEPARATED BY COMMAS & TERMINATED BY A CARRIAGE
RETURN.

E.G. DRIVES TO BE TESTED: 1,2,4,6

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT
CONSIDERATIONS IN SECTIONS 3.2 & 3.3.

4.5.2 BUS ADDRESS

THE REQUEST WILL BE:

TYPE IN BUSS ADDRESS IF NOT 177440

THE DEFAULT IS A CARRIAGE RETURN

4.5.3 CONTROLLER INTERRUPT VECTOR

THE REQUEST WILL BE:

TYPE IN CONTROLLER INTERRUPT VECTOR IF NOT 210

THE DEFAULT IS A CARRIAGE RETURN.

4.5.4 EXAMPLE OF PROGRAM DIALOGUE

THE EXAMPLE SHOWN IS FOR A PROGRAM STARTED AT ADDRESS 220.
ALL OPERATOR RESPONSES ARE UNDERLINED.

UNIBUS RK06 DRIVE DIAGNOSTIC
PART 2
MAINDEC-11-DZR6I-B-PB

DRIVES TO BE TESTED: 1,3<CR>

TYPE IN BUSS ADDRESS IF NOT 177440 <CR>

TYPE IN CONTROLLER INTERRUPT VECTOR IF NOT 210 <CR>

WILL TEST DRIVES:

1
3

DRIVE 1

(THE REST IS IDENTICAL TO THE EXAMPLE SHOWN IN 4.6 BELOW)

4.6 PROGRAM EXAMPLE

THE FOLLOWING IS AN EXAMPLE OF A PROGRAM STARTED AT THE
DEFAULT ADDRESS (200) & WITH 2 DRIVES ON THE LINE.

UNIBUS RK06 DRIVE DIAGNOSTIC
PART 2
MAINDEC-11-DZR6I-B-PB

WILL TEST DRIVES:

0
1

DRIVE 0

DRIVE SERIAL NO. AAA
CARTRIDGE SERIAL NO. BBB

DRIVE 1

DRIVE SERIAL NO. CCC
CARTRIDGE SERIAL NO. DDD

END PASS #1

WILL TEST DRIVES:

0
1

DRIVE 0

DRIVE SERIAL NO. AAA
CARTRIDGE SERIAL NO. BBB

DRIVE 1

DRIVE SERIAL NO. CCC
CARTRIDGE SERIAL NO. DDD

END PASS # 2

(ETC)

THE ABOVE ASSUMES NO ERRORS DETECTED.
THE NUMBER OF PASSES IS DETERMINED BY ACT/APT/XXDP

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT
CONSIDERATIONS IN SECTIONS 3.2 & 3.3.

5.0 DRIVE DIAGNOSTIC FUNCTIONAL DESCRIPTION

5.1 GENERAL

A. WRITE TESTS

THESE TESTS CHECK THE ABILITY OF THE DRIVE TO WRITE & READ
WORSE CASE PATTERNS; PERFORM ALL OFFSETS & PERFORM ALL
SPIRAL WRITING.

B. SERVO & SPINDLE TIMING TESTS

THESE TESTS CHECK & TYPE HEAD LOAD, UNLOAD & INDEX TIMING,
ALSO MIN, MAX, AND AVERAGE SEEK TIMES, AND MAX VELOCITY
OF THE HEADS ARE MEASURED & TYPED.

5.2 TEST DESCRIPTIONS

BASIC CONTROLLER TESTS, SIZING & SETUP

TEST 1 REFERENCE ALL CONTROLLER REGISTERS

THIS TEST VERIFIES THAT ALL THE CONTROLLER REGISTERS
CAN BE ACCESSED. THE INABILITY TO BE ACCESSED WILL
RESULT IN A TIMEOUT TRAP WITH AN ERROR MESSAGE. ANY
ERROR IN THIS TEST WILL RESULT IN ABORTING ALL OTHER
TESTS AND JUMPING TO 'END OF PASS'

TEST 2 SIZE THE BUSS

THIS TEST IS ENTERED ONLY IF 'DRIVE SELECTION' IS DEFAULTED

EITHER BY RUNNING IN THE AUTO MODE OR A 200 START IN THE MANUAL MODE.

EVERY DRIVE FROM 0 THRU 7 IS ADDRESSED. CONTROLLER ERROR (CERR) IS EXAMINED AND IF NOT SET, THE DRIVE WILL BE TESTED. IF SET, THE PROGRAM WILL BYPASS TESTING THAT DRIVE ONLY IF THE ERROR WAS A RESULT OF MDS, LIFE OR NED BEING SET; OR BOTH NED & DRA RESET INDICATING THE OTHER PORT IS ACCESSED.

TEST 3 VERIFY OPERATOR DRIVE SELECTIONS

THIS TEST IS ENTERED ONLY IF DRIVE SELECTION IS NOT DEFAULTED. EVERY DRIVE FROM 0 TO 7 IS ADDRESSED & CONTROLLER ERROR (CERR) IS EXAMINED. IF NOT SET THE PROGRAM WILL ASSUME THE DRIVE IS PRESENT. IT WILL THEN CHECK TO SEE THAT THE DRIVE WAS INPUTTED FOR TESTING. IF NOT, IT WILL BE AN ERROR. IF CERR WAS SET, THAT DRIVE WILL BE BYPASSED ONLY IF THE ERROR WAS A RESULT OF MDS OR LIFE SET OR BOTH NED & DRA RESET (WRONG PORT). IF CERR IS A RESULT OF NED ONLY, IT IS CHECKED AGAINST THE INPUTTED INFOR TO VERIFY IT WAS NOT SPECIFIED.

TEST 4 FIND NEXT DRIVE TO BE TESTED

THIS TEST FINDS THE NEXT DRIVE PRESENT & PUTS THAT ADDRESS IN 'DRVAD'. THROUGHOUT THE FOLLOWING TESTS, THE DRIVE TESTED IS THE DRIVE WHOSE ADDRESS IS IN 'DRVAD'.

TEST 5 PRINT DRIVE SERIAL NUMBER

THIS TEST READS & PRINTS THE DRIVE SERIAL # FROM MSG A, WORD 11 IN DECIMAL & IS PERFORMED ON THE 1ST PASS ONLY

TEST 6 SET VV WITH PACK COMMAND

IF VV IS RESET, THE PACK COMMAND IS USED TO SET IT.

TEST 7 READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL

THIS TEST VERIFIES THAT CYL 410, TRACK 2 CAN BE READ. THIS AREA CONTAINS BAD SECTOR INFO WHICH IS WRITTEN BY THE FACTORY DURING MANF. ALL BAD SECTOR INFO (BSE) WILL BE STORED AT THIS TIME TO MASK FUTURE READ HEADER OR DATA ERROR PRINTOUTS. IF BSE INFO CANNOT BE READ, OR IF AFTER READING THE BSE INFO

IF IT IS DETERMINED THAT AN ALIGNMENT CARTRIDGE IS USED,
A MESSAGE WILL BE TYPED INDICATING THAT ALL
FUTURE FORMAT AND READ-WRITE TESTS WILL BE BYPASSED.
THIS IS DONE SO AS NOT TO DESTROY BSE INFO OR AN ALIGNMENT PACK BY WRITING
THE PACK SERIAL # IS TYPED IN OCTAL 8 FOR THE FIRST PASS ONLY.

WRITE TESTS

TEST 10 BASIC WRITE DATA TEST; 1 WORD

THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE JUST ONE WORD,
ALL SECTORS ON CYL 0 ARE GIVEN IDENTICAL HEADERS &
A WRITE COMMAND IS ISSUED. READ & WRITE CHECK COMMANDS ARE NOT
PERFORMED. THIS TEST PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP
FOR A WRITE ERROR.

TEST 11 BASIC WRITE DATA TEST; FULL SECTOR

THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE
A FULL SECTOR. ALL ZEROS ARE WRITTEN BY THE WRITE DATA COMMAND
& CHECKED BY A RD DATA COMMAND. A FURTHER CHECK IS PERFORMED
BY THE WRT CHK COMMAND.
THE ABOVE IS REPEATED FOR AN ALL ONES PATTERN.

TEST 12 20 SECTOR FORMAT TEST

DATA IS WRITTEN ON A FULL TRACK IN 20 SECTOR FORMAT.
MSG B0,B1 ARE CHECKED FOR ANY ERROR CONDITION

TEST 13 TEST OFFSET & RTC LOGIC

THE HEADS ARE FIRST OFFSET BY OFFSET COMMANDS.
THIS TEST CHECKS THE RTC LOGIC BY VERIFYING THAT THE
'OFFSET ON' BIT (MSG A,00) RESETS AND THE OFFSET REG

BECOMES THE CYL DIFF INFO WHEN A SEEK CMD TO A
DIFFERENT CYLINDER IS ISSUED
IT ALSO TESTS THAT DRIVE CLEAR & SEEK TO SELF WILL NOT
CLEAR THE 'OFFSET ON' BIT OR THE OFFSET REG.
ALL OFFSET POSITIONS IN BOTH DIRECTIONS ARE CHECKED

TEST 14 TEST READ DATA AT ALL HEAD OFFSET POSITIONS

THIS TEST VERIFIES THAT THE HEAD OFFSET LOGIC IS OPERATIONAL BY WRITING ALL 1'S PATTERNS ON CYLINDER 0, HEAD 0. THEN PERFORMING READ DATA FROM CENTERLINE AND MOVING OUT + AND - OFFSET POSITIONS UNTIL A FAILURE OCCURES. THE FAILING OFFSET POSITIONS ARE PRINTED OUT IF LESS THAN THE OFFSET TOLERANCE TO BE SPECIFIED. OFFSET CODES ARE ALSO VERIFIED BY READING MSG A, STATUS 00 & 10. ALL HEADS ARE TESTED AT CYL 0.

IF THERE ARE NO FAILURES AT ALL, THIS INDICATES THAT

OR A. HEADS DID NOT MOVE AT ALL
 B. THE COMBINATION OF DISC SURFACE, HEADS, R/W AMP
 ARE EXCEPTIONALLY GOOD.

AN APPROPRIATE MESSAGE WILL BE TYPED.

TEST 15 WRITE WITH HEADS OFFSET

THIS TEST VERIFIES THAT WHEN ATTEMPTING TO WRITE WITH HEADS OFFSET THAT THE OFFSET WILL CLEAR & THE DRIVE WILL WRITE SINCE THE WRITE COMMAND HAS AN IMPLIED RTC. THIS TEST IS PERFORMED FOR MAX POS & NEG OFFSETS ONLY

TEST 16 TEST CURRENT CROSS-OVER C YLINDERS

THIS TEST VERIFIES THAT THE DRIVE CAN WRITE & READ OFF CURRENT CHANGE CYLINDERS X & Y IN THE FOLLOWING WAY:

SPIRAL WRITING IS PERFORMED FROM CYLINDER X TO CYLINDER Y WITH A DATA PATTERN FILLING THE ENTIRE 2 CYLINDERS.

A WRITE CHECK IS THEN PERFORMED TO VERIFY DATA WAS PROPERLY WRITTEN. THIS TEST IS PERFORMED FOR ALL 3 HEADS.

CYLINDER X: 63 127 191 255 319 383
 CYLINDER Y: 64 128 192 256 320 384

TEST 17 TEST HEAD SWITCHING TIME

TESTS THE ABILITY TO SWITCH HEADS IN LESS THEN 10MS WHEN HEADS SPIRAL.

1. SECTOR 17 IS FIRST LOCATED AND A WRITE DATA COMMAND OF 512 WORDS TO SECTOR 21 IS ISSUED.
2. THE PROGRAM NOW KNOWS THAT THE DRIVE WILL NOT HAVE TO TRAVEL

E02

- A FULL REVOLUTION BEFORE FINDING SECTOR 21.
3. SINCE EACH SECTOR TAKES APPROX. 1.2MS, THE TIME BETWEEN THE START OF THE WRITE COMMAND (FROM SECTOR 21, HEAD 0; TO SECTOR 0, HEAD 1) AND CONTROLLER READY SHOULD BE APPROX 6MS

THE ABOVE IS REPEATED FOR HEAD SWITCHING BETWEEN 1 TO 2

THIS TEST IS BYPASSED IF NEITHER L OR P CLOCK IS PRESENT

TEST 20 DRIVE OFF TRACK TEST

THIS TEST CHECKS FOR SERVO OSCILLATIONS DURING SETTLING TIME BEYOND THE ALLOTTED 3MS.

1. INITIALLY, EVERY CYLINDER IS FORMATTED WITH IDENTICAL HEADERS (UNIQUE TO EACH CYLINDER)
2. A FULL SECTOR WRITE COMMAND IS ISSUED BY A SINGLE CYL SEEK FROM 0 TO 1. AS HEADERS ARE IDENTICAL, THE NEXT SECTOR TO COME UNDER THE HEADS WILL IMMEDIATELY BE WRITTEN.
3. IF THERE IS OSCILLATION SENSED BY READING THE TRIBITS, DRIVE OFF TRACK ERROR WILL SET.

IN THIS MANNER OSCILLATING SEEKS ARE PERFORMED BETWEEN ALL MAJOR CYLINDERS. 100 OSCILLATIONS ARE PERFORMED AT EACH MAJOR CYLINDER BEFORE DOING THE NEXT CYLINDER

SERVO & SPINDLE TIMING TESTS.

TEST 21 TIME BETWEEN OUTER LIMIT TO HEADS HOME DURING UNLOAD

TIME IS MEASURED FROM ATTN ASSERTING (APPROXIMATES REV & OUTER LIMIT) TO HEADS HOME ASSERTING. EXPECTED TIME APPROX 500MS

ALL TIMING TESTS ARE BYPASSED IF NEITHER L OR P CLOCK IS PRESENT & WILL BE INDICATED BY A MESSAGE

TEST 22 TEST LOW VELOCITY TIMES DURING LOADING

THIS TEST ISSUES A START SPINDLE COMMAND AFTER 'HEADS HOME' HAS BEEN DETECTED FROM THE PREVIOUS TEST. THE FOLLOWING "LOW VELOCITY" TIMES ARE CHECKED AGAINST LIMITS TO BE DEFINED:

F02

TIME 1: TIME BETWEEN HEADS HOME NEGATING & TRACK FOLL OK ASSERTING
EXPECTED TIME APPROX 500 MS

TIME 2: TIME BETWEEN OUTER LIMIT & INNER LIMIT

TIME IS MEASURED FROM TRACK FOLL OK (APPROX OUTER LIMIT)
TO REV (APPROX INNER LIMIT)
EXPECTED TIME APPROX 2 SEC

TIME 3: TIME BETWEEN INNER LIMIT & OUTER LIMIT

TIME IS MEASURED FROM REV ASSERTING (FROM ABOVE)
TO REV NEGATING (APPROX OUTER LIMIT)
EXPECTED TIME APPROX 2 SEC

TEST 23 MEASURE ROTATIONAL SPEED

THIS TEST MEASURES INDEX TIMING BY:

- A. CHANGE FORMAT TO 20 SECTOR & READ HEADER.
CONTROLLER RDY STARTS THE TIMER
- B. CHANGE FORMAT TO 22 SECTOR & READ HEADER.
CONTROLLER RDY ENDS THE TIMER.

INDEX TIMING IS THE TIME BETWEEN THE 2 CONT. RDY TIMES.
THIS IS BECAUSE A CHANGE OF FORMAT INHIBITS SECTOR PULSES
UNTIL THE NEXT INDEX APPEARS-THUS KEEPING A GIVEN
FORMAT COMPLETE THROUGHOUT AN ENTIRE CYLINDER

THE TIME IS THE AVERAGE OF 100 READINGS.

TEST 24 MEASURE MAX SEEK TIME

THIS TEST MEASURES THE MAX SEEK TIME BETWEEN CYLINDERS 0 & 410
THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
IF NOT WITHIN LIMITS TO BE SUPPLIED.
MAX SEEK TIME SHOULD BE LESS THAN 70MS.

TEST 25 MEASURE MIN SEEK TIME

THIS TEST MEASURES THE MIN SEEK TIME BETWEEN CYLINDER 0 & 1
THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
IF NOT WITHIN LIMITS TO BE SUPPLIED.
MIN SEEK TIME SHOULD BE LESS THAN 10MS.

TEST 26 MEASURE 137 CYLINDER SEEK TIME

THIS TEST MEASURES THE AVERAGE SEEK TIME BETWEEN CYLINDERS 0 & 137
THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED

IF NOT WITHIN LIMITS TO BE SUPPLIED.
AVERAGE SEEK TIME SHOULD BE LESS THAN 40MS

TEST 27 MEASURE MAX VELOCITY OF HEADS

THIS TESTS MAX VELOCITY BY DOING SEEKS BETWEEN
CYL 0 & 383 AND MEASURING THE TIME BETWEEN CYLINDERS
128 & 256. SINCE THE DISTANCE BETWEEN CYL 128 & 256 IS KNOWN,
THE AVERAGE VELOCITY OF 100 SEEKS IS CALCULATED & TYPED
IF NOT WITHIN THE SPECIFIED LIMITS TO BE SUPPLIED.

THE PROGRAM DOES NOT REQUIRE FORMATTED PACKS AS FORMATTING
IS PERFORMED IN ANY CASE.

ANY TEST THAT MODIFIES STANDARD FORMATTING IS FOLLOWED BY A
'CLEAN UP' TEST TO PUT THOSE CYLINDERS BACK TO STANDARD
FORMAT.

6.0 ERROR REPORTING

6.1 ERROR INTERPRETATION

WHENEVER AN ERROR MESSAGE IS PRINTED OUT, ALL REGISTERS
AND OTHER DATA PERTAINING TO THE ERROR ARE ALSO GIVEN.
MSG A(00), MSG B(01), RKER, RKBA...ETC, INDICATE THE
CONTENTS OF THE CORRESPONDING REGISTERS AT THE TIME OF
ERROR.

EVERY ERROR MESSAGE CONTAINS A PC. THIS PC INDICATES THE
POSITION IN PROGRAM WHERE THE ERROR CALL IS LOCATED. THE
ERROR MESSAGE, BECAUSE OF PRACTICAL CONSIDERATIONS IS MADE
SHORT AND MEANINGFUL. THE USER IS ADVISED TO LOOK UP THE
PC IN THE PROGRAM LISTING, WHERE HE WILL FIND MORE INFORMATION
ABOUT THE ERROR. IN MANY INSTANCES, A SINGLE FAULT WILL
GIVE RISE TO MORE THAN ONE ERROR REPORT. A LITTLE DELIBERATION
AND CAREFUL EXAMINATION OF THE DATA GIVEN WILL BE CERTAINLY
VERY HELPFUL IN PINPOINTING THE FAULT. A BRIEF EXPLANATION
OF WHAT IS BEING CHECKED IN THE TEST IS GIVEN AT THE
BEGINNING OF EVERY TEST. ALL THE NUMBERS GIVEN WITH ERROR
MESSAGES ARE IN OCTAL.

NOTE

NO ERROR LOGGING OR OPERATION HISTORY IS PROVIDED.

6.2 ERROR PRINTOUT EXAMPLE

MESSAGE A0 ERROR
AT END OF HEAD LOADING

```
TEST NO.  PC
000014  013432
RKMR2   RKMR3   RKER   RKDS   RKCS1  RKCS2
140141  100000  000000  140101  040200  000101
SHOULD BE
150341
```

MESSAGE B1 ERROR
AFTER LIMIT DETECT

```
TEST NO.  PC
000023  023316
RKMR2   RKMR3   RKER   RKDS   RKCS1  RKCS2
045720  030001  000000  040000  040200  000100
SHOULD BE
045720  120001
```

IN THE FIRST EXAMPLE, RKMR2 (MESSAGE A0) DID NOT READ BACK CORRECTLY. THE CORRECT "SHOULD BE" CONTENTS IS UNDER 'RKMR2'.

IN THE 2ND EXAMPLE, RKMR3 (MESSAGE B1) DID NOT READ BACK CORRECTLY WITH THE CORRECT 'SHOULD BE' CONTENTS UNDER 'RKMR3'.

30	OPERATIONAL SWITCH SETTINGS
44	SUMMARY OF STARTING LOCATIONS
55	BASIC DEFINITIONS
166	RK06 CONTROLLER REGISTER DEFINITION
186	CONTROL AND STATUS REGISTER 1 BITS (RKCS1:0)
217	CONTROL AND STATUS REGISTER 2 BITS (RKCS2:10)
234	ERROR REGISTER BIT DEFINITION (RKER:14)
253	STATUS REGISTER BIT DEFINITION (RKDS:12)
269	MAINTENANCE REGISTER 1 BIT DEFINITION (RKMR1:22)
285	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A (RKMR2:34)
298	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A (RKMR2:34)
312	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B (RKMR3:36)
325	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B (RKMR3:36)
339	COMMON MASKS AND OTHER BITS: MESSAGE A (RKMR2:34)
346	COMMON MASKS AND OTHER BITS: MESSAGE B (RKMR3:36)
355	TRAP CATCHER
364	STARTING ADDRESS(ES)
380	ACT11 HOOKS
391	RPT PARAMETER BLOCK
414	COMMON TAGS
464	RPT MAILBOX-ETABLE
1194	ERROR POINTER TABLE
2221	PROGRAM SETUP
2258	INITIALIZE THE COMMON TAGS
2406	BASIC CONTROLLER TESTS, SIZING & SETUP
2408	T1 REFERENCE ALL CONTROLLER REGISTERS
2451	T2 SIZE THE BUSS
2559	T3 VERIFY OPERATOR DRIVE SELECTIONS
2661	T4 FIND NEXT DRIVE TO BE TESTED
2704	T5 PRINT DRIVE SERIAL NUMBER
2746	T6 SET VV WITH PACK COMMAND
2780	T7 READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL #
2869	WRITE TESTS
2872	T10 BASIC WRITE DATA TEST; 1 WORD
2970	T11 BASIC WRITE DATA TEST; FULL SECTOR
3218	T12 20 SECTOR FORMAT TEST
3391	T13 TEST OFFSET & RTC LOGIC
3678	T14 TEST READ DATA AT ALL HEAD OFFSET POSITIONS
3970	T15 WRITE WITH HEADS OFFSET
4198	T16 TEST CURRENT CROSS-OVER CYLINDERS
4347	T17 TEST HEAD SWITCHING TIME
4404	T20 DRIVE OFF TRACK TEST
4611	SERVO & SPINDLE TIMING TESTS
4618	T21 TIME BETWEEN OUTER LIMIT TO HEADS HOME DURING UNLOAD
4706	T22 TEST LOW VELOCITY TIMES DURING LOADING
4826	T23 MEASURE ROTATIONAL SPEED
4915	T24 MEASURE MAX SEEK TIME
5033	T25 MEASURE MIN SEEK TIME
5151	T26 MEASURE 137 CYLINDER SEEK TIME
5269	T27 MEASURE MAX VELOCITY OF HEADS
5435	END OF PASS ROUTINE
5480	SUBROUTINES
6359	UNEXPECTED TIMEOUT HANDLER
6396	UNEXPECTED INTERRUPT HANDLER
6432	MEMORY CHECK ENABLE TRAP
6446	RK06 INTERRUPT HANDLER

6464	POWER DOWN AND UP ROUTINES
6546	SCOPE HANDLER ROUTINE
6611	ERROR HANDLER ROUTINE
6665	TYPE ROUTINE
6744	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
6811	APT COMMUNICATIONS ROUTINE
6869	BINARY TO OCTAL (ASCII) AND TYPE
6945	TTY INPUT ROUTINE
7112	READ AN OCTAL NUMBER FROM THE TTY
7165	DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
7204	DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
7266	SINGLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
7284	TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS
7307	INTEGER MULTIPLY ROUTINE
7354	SAVE AND RESTORE RO-R5 ROUTINES
7399	TRAP DECODER
7422	TRAP TABLE
7447	SERVICE MESSAGES
7824	ERROR MESSAGES
8136	DATA HEADERS
8354	ERROR OUTPUT DATA
8388	ERROR DATA FORMATS
8480	TYPE ERROR ROUTINE

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

167400
000001

```

.NLIST CND,MC,MD
.LIST ME
.ENABL ABS,AMA

;DEFINE SYSMAC MACPOS

.MCALL .HEADER .SWRHI .SWRLO .EQUAT .SETUP .SCATCH .SCMTAG
.MCALL .SEOP $SCOPE $ERROR $TYPE $TYPOCT $SAVE
.MCALL .STYPDEC .SREAD .SROCT .STRAP .SDB20 $MULT
.MCALL $ACT11 .SAPTHDR .SAPTYPE $DB20 $SB20 $SUPRS
$SWR= 167400 ;DEFINE SWITCHES 15,14,13,11,10,9,8
$TN= 1 ;SET FIRST TEST NO. TO 1

```

```

.TITLE UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
;*COPYRIGHT (C) 1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY GARY PAPAIZIAN
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C1),MAR 24, 1976.
;*

```

```

.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;* SWITCH USE
;* -----
;* 15 HALT ON ERROR
;* 14 LOOP ON TEST
;* 13 INHIBIT ERROR TYPEOUTS
;* 12 ABORT DRIVE AFTER 20 ERRORS
;* 11 INHIBIT ITERATIONS
;* 10 BELL ON ERROR
;* 9 LOOP ON ERROR
;* 8 LOOP ON TEST IN SWR<7:0>

```

```

.SBTTL SUMMARY OF STARTING LOCATIONS
;*
;*
;* 200 DEFAULT PARAMETERS
;* 204 DEFAULT PARAMETERS & BYPASS WRITE TESTS
;* 214 DEFAULT PARAMETERS & BYPASS TIMING TESTS
;* 220 INPUT PARAMETERS
;* 224 INPUT PARAMETERS & BYPASS WRITE TESTS
;* 230 INPUT PARAMETERS & BYPASS TIMING TESTS
;* 240 ODT11
;*

```

.SBTTL BASIC DEFINITIONS

;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

001100

STACK= 1100

.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL

.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL

;*MISCELLANEOUS DEFINITIONS

000011

HT= 11 ;;CODE FOR HORIZONTAL TAB

000012

LF= 12 ;;CODE FOR LINE FEED

000015

CR= 15 ;;CODE FOR CARRIAGE RETURN

000200

CRLF= 20 ;;CODE FOR CARRIAGE RETURN-LINE FEED

177776

PS= 177776 ;;PROCESSOR STATUS WORD

177774

.EQUIV PS,PSW

STKLMT= 177774 ;;STACK LIMIT REGISTER

177772

PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER

177570

DSWR= 177570 ;;HARDWARE SWITCH REGISTER

177570

DDISP= 177570 ;;HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS

000000

R0= %0 ;;GENERAL REGISTER

000001

R1= %1 ;;GENERAL REGISTER

000002

R2= %2 ;;GENERAL REGISTER

000003

R3= %3 ;;GENERAL REGISTER

000004

R4= %4 ;;GENERAL REGISTER

000005

R5= %5 ;;GENERAL REGISTER

000006

R6= %6 ;;GENERAL REGISTER

000007

R7= %7 ;;GENERAL REGISTER

.EQUIV R6,SP ;;STACK POINTER

.EQUIV R7,PC ;;PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS

000000

PR0= 0 ;;PRIORITY LEVEL 0

000040

PR1= 40 ;;PRIORITY LEVEL 1

000100

PR2= 100 ;;PRIORITY LEVEL 2

000140

PR3= 140 ;;PRIORITY LEVEL 3

000200

PR4= 200 ;;PRIORITY LEVEL 4

000240

PR5= 240 ;;PRIORITY LEVEL 5

000300

PR6= 300 ;;PRIORITY LEVEL 6

000340

PR7= 340 ;;PRIORITY LEVEL 7

;*SWITCH REGISTER SWITCH DEFINITIONS

100000

SW15= 100000

040000

SW14= 40000

020000

SW13= 20000

010000

SW12= 10000

004000

SW11= 4000

002000

SW10= 2000

001000

SW09= 1000

000400

SW08= 400

000200

SW07= 200

000100

SW06= 100

000040

SW05= 40

000020

SW04= 20

000010

SW03= 10

000004

SW02= 4

54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109

```

110      000002      SW01= 2
111      000001      SW00= 1
112      .EQUIV      SW09,SW9
113      .EQUIV      SW08,SW8
114      .EQUIV      SW07,SW7
115      .EQUIV      SW06,SW6
116      .EQUIV      SW05,SW5
117      .EQUIV      SW04,SW4
118      .EQUIV      SW03,SW3
119      .EQUIV      SW02,SW2
120      .EQUIV      SW01,SW1
121      .EQUIV      SW00,SW0
122
123      .:DATA BIT DEFINITIONS (BIT00 TO BIT15)
124      100000      BIT15= 100000
125      040000      BIT14= 40000
126      020000      BIT13= 20000
127      010000      BIT12= 10000
128      004000      BIT11= 4000
129      002000      BIT10= 2000
130      001000      BIT09= 1000
131      000400      BIT08= 400
132      000200      BIT07= 200
133      000100      BIT06= 100
134      000040      BIT05= 40
135      000020      BIT04= 20
136      000010      BIT03= 10
137      000004      BIT02= 4
138      000002      BIT01= 2
139      000001      BIT00= 1
140      .EQUIV      BIT09,BIT9
141      .EQUIV      BIT08,BIT8
142      .EQUIV      BIT07,BIT7
143      .EQUIV      BIT06,BIT6
144      .EQUIV      BIT05,BIT5
145      .EQUIV      BIT04,BIT4
146      .EQUIV      BIT03,BIT3
147      .EQUIV      BIT02,BIT2
148      .EQUIV      BIT01,BIT1
149      .EQUIV      BIT00,BIT0
150
151      .:BASIC "CPU" TRAP VECTOR ADDRESSES
152      000004      ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
153      000010      RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
154      000014      TBITVEC= 14 ;: "T" BIT
155      000014      TRTVEC= 14 ;: TRACE TRAP
156      000014      BPTVEC= 14 ;: BREAKPOINT TRAP (BPT)
157      000020      IOTVEC= 20 ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
158      000024      PWRVEC= 24 ;: POWER FAIL
159      000030      EMTVEC= 30 ;: EMULATOR TRAP (EMT) **ERROR**
160      000034      TRAPVEC= 34 ;: "TRAP" TRAP
161      000060      TKVEC= 60 ;: TTY KEYBOARD VECTOR
162      000064      TPVEC= 64 ;: TTY PRINTER VECTOR
163      000240      PIRQVEC= 240 ;: PROGRAM INTERRUPT REQUEST VECTOR
164
165      .SBTTL RK06 CONTROLLER REGISTER DEFINITION

```

```

166
167 ;          $BASE=177440
168
169 000000 RKCS1= 0 ;CONTROL AND STATUS REGISTER 1
170 000002 RKWC= 2 ;WORD COUNT REGISTER
171 000004 RKBA= 4 ;BUS ADDRESS REGISTER
172 000006 RKDA= 6 ;DESIRED TRACK SECTOR REGISTER
173 000010 RKCS2= 10 ;CONTROL AND STATUS REGISTER 2
174 000012 RFD5= 12 ;DRIVE STATUS REGISTER
175 000014 RKER= 14 ;ERROR REGISTER
176 000016 RKASOF= 16 ;ATTENTION SUMMARY AND OFFSET REGISTER
177 000020 RKDC= 20 ;DESIRED CYLINDER REGISTER
178 000024 RKDB= 24 ;DATA BUFFER
179 000026 RKMR1= 26 ;MAINTENANCE REGISTER 1
180 000034 RKMR2= 34 ;MAINTENANCE REGISTER 2 (MESSAGE LINE A)
181 000036 RKMR3= 36 ;MAINTENANCE REGISTER 3 (MESSAGE LINE B)
182 000030 RKECPS= 30 ;ECC POSITION INFORMATION
183 000032 RKECPT= 32 ;ECC PATTERN INFORMATION
184
185 .SBTTL CONTROL AND STATUS REGISTER 1 BITS (RKCS1:0)
186
187 ;          DRIVE COMMANDS
188
189 000001 SELDRV= 1 ;SELECT DRIVE (GET STATUS)
190 000003 PACK= 3 ;PACK ACKNOWLEDGE
191 000005 CLEAR= 5 ;DRIVE CLEAR
192 000007 UNLOAD= 7 ;UNLOAD
193 000011 SRTSPL= 11 ;START SPINDLE
194 000013 RECAL= 13 ;RECALIBRATE
195 000015 OFFSET= 15 ;OFFSET
196 000017 SEEK= 17 ;SEEK
197 000021 RDDATA= 21 ;READ DATA
198 000023 WRDATA= 23 ;WRITE DATA
199 000025 RDHEAD= 25 ;READ HEADER
200 000027 WRHEAD= 27 ;WRITE HEADER AND DATA
201 000031 WRTCHK= 31 ;WRITE CHECK
202
203 000001 GO= BIT0 ;GO BIT
204 000100 IE= BIT6 ;INTERRUPT ENABLE
205 000200 RDY= BIT7 ;CONTROLLER READY
206 000400 BA16= BIT8 ;BUS ADDRESS BIT 16
207 001000 BA17= BIT9 ;BUS ADDRESS BIT 17
208 002000 CDT= BIT10 ;CONTROLLER DRIVE TYPE (0=RK06)
209 004000 CTO= BIT11 ;CONTROLLER TIMEOUT
210 010000 CFMT= BIT12 ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
211 020000 SPAR= BIT13 ;SERCON PARITY ERROR DETECTED BY CONTROLLER
212 040000 DI= BIT14 ;DRIVE INTERRUPT
213 100000 CERR= BIT15 ;CONTROLLER ERROR
214 100000 CCLR= BIT15 ;CONTROLLER CLEAR
215
216 .SBTTL CONTROL AND STATUS REGISTER 2 BITS (RKCS2:10)
217
218 000007 DRVMSK= 7 ;MASK FOR DRIVE SELECTION CODE
219 000010 RLS= BIT3 ;DESELECT OR RELEASE DRIVE IN BITS 0-2
220 000020 BAI= BIT4 ;BUS ADDRESS INCREMENT INHIBIT
221 000040 SCLR= BITS ;SUBSYSTEM CLEAR CONTROLLER AND ALL DRIVES
  
```

268
269
270
271
272
273
274
275
276
277

000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

IR= BIT6
OR= BIT7
JFE= BIT8
MDS= BIT9
PGE= BIT10
NEM= BIT11
NED= BIT12
UPE= BIT13
WCE= BIT14
DLT= BIT15

: INPUT READY
: OUTPUT READY
: UNIT FIELD ERROR
: MULTIPLE DRIVE SELECT
: PROGRAMMING ERROR
: NON-EXISTENT MEMORY
: NON-EXISTENT DRIVE
: UNIBUS PARITY ERROR
: WRITE CHECK ERROR
: DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION (PKER:14)

000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

ILF= BIT0
SKI= BIT1
NXF= BIT2
DRPAR= BIT3
FMTE= BIT4
DTYE= BIT5
ECH= BIT6
BSE= BIT7
HVRC= BIT8
COE= BIT9
IDAE= BIT10
MLE= BIT11
DTE= BIT12
OPI= BIT13
LNS= BIT14
DCK= BIT15

: ILLEGAL FUNCTION CODE
: SEEK INCOMPLETE
: NON-EXECUTABLE FUNCTION
: DRIVE DETECTED SERCON PARITY ERROR
: FORMAT ERROR
: DRIVE TYPE ERROR
: ECC HARD
: BAD SECTOR ERROR
: HEADER VRC ERROR
: CYLINDER ADDRESS OVERFLOW ERROR
: INVALID DISK ADDRESS ERROR: HEAD/CYL
: WRITE LOCK ERROR
: DRIVE TIMING ERROR
: OPERATION (SEARCH) INCOMPLETE
: DRIVE UNSAFE
: DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION (RKDS:12)

000001
000004
000010
000020
000040
000100
000200
000400
004000
020000
040000
100000

DRA= BIT0
OFST= BIT2
ACLO= BIT3
DCLO= BIT4
DROT= BIT5
VV= BIT6
DROY= BIT7
DDT= BIT8
WRL= BIT11
PIP= BIT13
DSC= BIT14
SVAL= BIT15

: DRIVE AVAILABLE (CONTROLLER IS SET IF
: THIS BIT IS RESET)
: DRIVE OFFSET
: AC LOW
: DC LOW
: DRIVE OFF TRACK
: VOLUME VALID
: DRIVE READY
: DRIVE TYPE (0=RK06)
: WRITE LOCK
: POSITIONING IN PROGRESS
: DRIVE STATUS CHANGE
: STATUS VALID

.SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION (RKMRI:22)

000017
000020
000040
000100
000200
000400
001000
002000

MESMSK= 17
PAT= BIT4
DMD= BIT5
MSP= BIT6
MIND= BIT7
MCLK= BIT8
MERD= BIT9
MEWD= BIT10

: MESSAGE MASK
: FORCE EVEN PARITY ON SERCON MESSAGE LINES
: DIAGNOSTIC MODE
: MAINTENANCE SECTOR PULSE
: MAINTENANCE INDEX
: MAINTENANCE CLOCK
: MAINTENANCE ENCODED READ DATA
: MAINTENANCE ENCODED WRITE DATA

278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333

004000
010000
020000
040000
100000

000040
000100
000200
000400
001000
002000
004000
010000
020000
040000

000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000

000040
000100
000200
000400
001000
002000
004000
010000
020000
040000

000020
000040
000100
000200
000400
001000
002000
004000

PCA= BIT11
PCD= BIT12
ECCW= BIT13
WRTGAT= BIT14
RDGATE= BIT15

D.DRA= BITS
D.VV= BIT6
D.DROY= BIT7
D.DCT= BIT8
D.FORM= BIT9
D.OFF= BIT10
D.WRL= BIT11
D.SPIN= BIT12
D.PIP= BIT13
D.DSC= BIT14

D.TFOK= BIT4
D.HOHM= BITS
D.BRHM= BIT6
D.DOOR= BIT7
D.CART= BIT8
D.SPOK= BIT9
D.FWD= BIT10
D.REV= BIT11
D.LOAD= BIT12
D.RTZ= BIT13
D.UNLD= BIT14

D.IDAE= BITS
D.ACLO= BIT6
D.FLT= BIT7
D.ILF= BIT8
D.PAR= BIT9
D.SKI= BIT10
D.WLE= BIT11
D.SPLS= BIT12
D.DROT= BIT13
D.UNS= BIT14

D.SECT= BIT4
D.WCUR= BITS
D.WGAT= BIT6
D.HOFL= BIT7
D.MHD= BIT8
D.XERR= BIT9
D.TIB= BIT10
D.FLO= BIT11

:PRECOMPENSATION ADVANCE
:PRECOMPENSATION DELAY
:ECC WORD IS BEING READ OR WRITTEN
:WRITE GATE
:READ GATE

.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A (RKMR2:34)

:DRIVE AVAILABLE
:VOLUME VALID
:DRIVE READY
:DRIVE TYPE (0=RK06)
:DRIVE FORMAT
:OFFSET ON
:WRITE LOCK
:SPINDLE ON
:POSITIONING IN PROGRESS
:DRIVE STATUS CHANGE

.SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A (RKMR2:34)

:TRACK FOLLOWING OK
:HEADS HOME
:BRUSHES HOME
:L/JOR INTERLOCKED
:CARTRIDGE INTERLOCK
:SPEED OK
:FORWARD
:REVERSE
:HEADS LOADING
:RETURN TO ZERO
:HEADS UNLOADING

.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B (RKMR3:36)

:INVALID DISK ADDRESS ERROR:HEAD/CYL
:AC LOW
:DRIVE FAULT
:ILLEGAL FUNCTION CODE
:DRIVE DETECTED SERCON PARITY ERROR
:SEEK INCOMPLETE
:WRITE LOCK ERROR
:SPEED LOSS
:DRIVE OFF TRACK
:R/W UNSAFE

.SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B (RKMR3:36)

:SECTOR ERROR
:WRITE CURRENT AND NO WRITE GATE
:WRITE GATE AND NO TRANSISTIONS
:HEAD FAULT
:MULTIPLE HEAD SELECT
:INDEX ERROR
:TRIBIT ERROR
:PLO ERROR

332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352

010000
020000
040000

D.NMOV= BIT12
D.LIMD= BIT13
D.SUNS= BIT14

;SEEK AND NO MOTION
;LIMIT DETECT ON SEEK
;SERVO UNSAFE

.SBTTL COMMON MASKS AND OTHER BITS: MESSAGE A (PKMR2:34)

000007
017760
017760
077770

M.DRV= 7
M.CDIF= 17760
M.OFST= 17760
M.SER= 77770

;DRIVE CODE, ALL BYTES
;CYLINDER DIFF, BYTE 10
;OFFSET VALUE, BYTE 10
;DRIVE SERIAL #, BYTE 11

.SBTTL COMMON MASKS AND OTHER BITS: MESSAGE B (RKMR3:36)

000003
017760
040000
000760
007000
100000

M.ID= 3
M.CADD= 17760
M.ALGN= BIT14
M.SECT= 760
M.HEAD= 7000
M.PAR= BIT 5

;BYTE ID, ALL BYTES
;CYLINDER ADDRESS, BYTE 10
;ALIGN SIGN, BYTE 10
;SECTOR COUNT, BYTE 11
;HEAD DECODE, BYTE 11
;PARITY, MESS A/B, ALL BYTES

353
354
355
356 000000
357
358
359
360 000174
361 000174 000000
362 000176 000000
363
364 000200 000137 010710
365 000204 000204
366 000204 000137 010604
367 000214 000214
368 000214 000137 010624
369 000220 000220
370 000220 000137 010564
371 000224 000224
372 000224 000137 010644
373 000230 000230
374 000230 000137 010666
375
376 000240 000240
377 000240 000137 054724
378
379
380
381
382
383 000244
384 000046 000046
385 000046 031550
386 000052 000052
387 000052 100000
388 000244
389 001000
390
391
392
393
394
395 001000
396 000024 000024
397 000024 000200
398 000044 000044
399 000044 001000
400 001000
401
402
403
404
405 001000
406 001000 000000
407 001002 001210
408 001004 000170

.SBTTL TRAP CATCHER

```

      =0
      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
      =174
DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
.SBTTL  STARTING ADDRESS(ES)
      JMP 2*START ;;JUMP TO STARTING ADDRESS OF PROGRAM
      =204
      JMP BYWRT
      =214
      JMP BYTIM
      =220
      JMP PARSRT      ;INPUT ALL PARAMETERS & START TESTING
      =224
      JMP BYWRTA
      =230
      JMP BYTIMA
      =240
      JMP 0.ODT      ;ENTER ODT11

```

.SBTTL ACT11 HOOKS

```

      ;*****
      ;HOOKS REQUIRED BY ACT11
      $SVPC=.      ;SAVE PC
      =46
      SENDAD      ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
      =52
      .WORD 100000      ;;2)SET LOC.52 TO 100000
      =52
      .=$SVPC      ;;RESTORE PC
      =1000

```

.SBTTL APT PARAMETER BLOCK

```

      ;*****
      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
      ;*****
      .SX=.      ;;SAVE CURRENT LOCATION
      =24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
      200      ;;FOR APT START UP
      =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
      $APTHDR      ;;POINT TO APT HEADER BLOCK
      =.SX      ;;RESET LOCATION COUNTER
      ;*****
      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
      ;INTERFACE SPEC.

```

```

$APTHD:
$HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MADDR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT:  .WORD 120.  ;;RUN TIM OF LONGEST TEST

```


F03

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IB.P11 APT PARAMETER BLOCK

MACY11 27(732) 01-OCT-76 10:38 PAGE 10

SEQ 0011

409 001006 000454
410 001010 000454
411 001012 000052
412

\$PASTM: .WORD 300. ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD 300. ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
\$ETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

413
414
415
416
417
418
419
420 001100 001100
421 001100 000000
422 001102 000
423 001103 000
424 001104 000000
425 001106 000000
426 001110 000000
427 001112 000000
428 001114 000
429 001115 001
430 001116 000000
431 001120 000000
432 001122 000000
433 001124 000000
434 001126 000000
435 001130 000000
436 001132 000000
437 001134 000
438 001135 000
439 001136 000000
440 001140 177570
441 001142 177570
442 001144 177560
443 001146 177562
444 001150 177564
445 001152 177566
446 001154 000
447 001155 002
448 001156 012
449 001157 000
450 001160 000000
451 001162 000000
452 001164 000000
453 001166 000000
454 001170 000000
455 001172 000000
456 001174 000000
457 001176 000000
458 001200 177607 000377
459 001204 077
460 001205 015
461 001206 000012
462
463
464
465
466
467 001210
468 001210 000000

```

.SBTTL COMMON TAGS
;*****
;THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
;USED IN THE PROGRAM.

.SCMTAG:      .=1100
;:START OF COMMON TAGS

$TSTNM: .WORD 0 ;:CONTAINS THE TEST NUMBER
$ERFLG: .BYTE 0 ;:CONTAINS ERROR FLAG
$ICNT: .WORD 0 ;:CONTAINS SUBTEST ITERATION COUNT
$LPADR: .WORD 0 ;:CONTAINS SCOPE LOOP ADDRESS
$LPERR: .WORD 0 ;:CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL: .WORD 0 ;:CONTAINS TOTAL ERRORS DETECTED
$ITEMB: .BYTE 0 ;:CONTAINS ITEM CONTROL BYTE
$ERMAX: .BYTE 1 ;:CONTAINS MAX. ERRORS PER TEST
$ERRPC: .WORD 0 ;:CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'BAD' DATA
$GDADR: .WORD 0 ;:CONTAINS 'GOOD' DATA
$BDADR: .WORD 0 ;:CONTAINS 'BAD' DATA
;:RESERVED--NOT TO BE USED

$AUTOB: .BYTE 0 ;:AUTOMATIC MODE INDICATOR
$INTAG: .BYTE 0 ;:INTERRUPT MODE INDICATOR

$SWR: .WORD 0 ;:ADDRESS OF SWITCH REGISTER
$DISP: .WORD 0 ;:ADDRESS OF DISPLAY REGISTER
$TKS: 177560 ;:TTY KBD STATUS
$TKB: 177562 ;:TTY KBD BUFFER
$TPS: 177564 ;:TTY PRINTER STATUS REG. ADDRESS
$TPB: 177566 ;:TTY PRINTER BUFFER REG. ADDRESS
$NULL: .BYTE 0 ;:CONTAINS NULL CHARACTER FOR FILLS
$FILLS: .BYTE 2 ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC: .BYTE 12 ;:INSERT FILL CHARS. AFTER A "LINE FEED"
$STPFLG: .BYTE 0 ;:"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
$STMP0: .WORD 0 ;:USER DEFINED
$STMP1: .WORD 0 ;:USER DEFINED
$STMP2: .WORD 0 ;:USER DEFINED
$STMP3: .WORD 0 ;:USER DEFINED
$STMP4: .WORD 0 ;:USER DEFINED
$STMP5: .WORD 0 ;:USER DEFINED
$TIMES: 0 ;:MAX. NUMBER OF ITERATIONS
$ESCAPE: 0 ;:ESCAPE ON ERROR ADDRESS
$BELL: .ASCIZ <207><377><377> ;:CODE FOR BELL
$QUES: .ASCII '?' ;:QUESTION MARK
$CRLF: .ASCII <15> ;:CARRIAGE RETURN
$LF: .ASCIZ <12> ;:LINE FEED
;*****
.SBTTL APT MAILBOX-ETABLE
;*****
.EVEN
$MAIL: ;:APT MAILBOX
$MSGTY: .WORD AMSGTY ;:MESSAGE TYPE CODE

```

H03

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZR618.P11 APT MAILBOX-ETABLE

MACY11 27(732) 01-OCT-76 10:38 PAGE 12

SEQ 0013

469	001212	000000	\$FATAL: .WORD	AFATAL	:: FATAL ERROR NUMBER
470	001214	000000	\$TESTN: .WORD	ATESTN	:: TEST NUMBER
471	001216	000000	\$PASS: .WORD	APASS	:: PASS COUNT
472	001220	000000	\$DEVCT: .WORD	ADEVCT	:: DEVICE COUNT
473	001222	000000	\$UNIT: .WORD	AUNIT	:: I/O UNIT NUMBER
474	001224	000000	\$MSGAD: .WORD	AMSGAD	:: MESSAGE ADDRESS
475	001226	000000	\$MSGLG: .WORD	AMSLG	:: MESSAGE LENGTH
476	001230		\$ETABLE:		:: APT ENVIRONMENT TABLE
477	001230	000	\$ENV: .BYTE	AENV	:: ENVIRONMENT BYTE
478	001231	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT MODE BITS
479	001232	000000	\$SWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
480	001234	000000	\$USWR: .WORD	AUSWR	:: USER SWITCHES
481	001236	000000	\$CPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTICNS
482			::*		BITS 15-11=CPU TYPE
483			::*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
484			::*		11/70=06, P00=07, Q=10
485			::*		BIT 10=REAL TIME CLOCK
486			::*		BIT 9=FLOATING POINT PROCESSOR
487			::*		BIT 8=MEMORY MANAGEMENT
488	001240	000	\$MAMS1: .BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
489	001241	000	\$MTYP1: .BYTE	AMTYP1	:: MEM. TYPE, BLK#1
490			::*		MEM. TYPE BYTE -- (HIGH BYTE)
491			::*		900 NSEC CORE=001
492			::*		300 NSEC BIPOLAR=002
493			::*		500 NSEC MOS=003
494	001242	000000	\$MADR1: .WORD	AMADR1	:: HIGH ADDRESS, BLK#1
495			::*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
496	001244	000	\$MAMS2: .BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
497	001245	000	\$MTYP2: .BYTE	AMTYP2	:: MEM. TYPE, BLK#2
498	001246	000000	\$MADR2: .WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
499	001250	000	\$MAMS3: .BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
500	001251	000	\$MTYP3: .BYTE	AMTYP3	:: MEM. TYPE, BLK#3
501	001252	000000	\$MADR3: .WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
502	001254	000	\$MAMS4: .BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
503	001255	000	\$MTYP4: .BYTE	AMTYP4	:: MEM. TYPE, BLK#4
504	001256	000000	\$MADR4: .WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
505	001260	000000	\$VECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
506	001262	000000	\$VECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
507	001264	177440	\$BASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
508	001266	000000	\$DEVN: .WORD	ADEVN	:: DEVICE MAP
509	001270	000000	\$CDW1: .WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
510	001272	000000	\$CDW2: .WORD	ACDW2	:: CONTROLLER DESCRIPTION WORD#2
511	001274	000000	\$DDW0: .WORD	ADDW0	:: DEVICE DESCRIPTOR WORD#0
512	001276	000000	\$DDW1: .WORD	ADDW1	:: DEVICE DESCRIPTOR WORD#1
513	001300	000000	\$DDW2: .WORD	ADDW2	:: DEVICE DESCRIPTOR WORD#2
514	001302	000000	\$DDW3: .WORD	ADDW3	:: DEVICE DESCRIPTOR WORD#3
515	001304	000000	\$DDW4: .WORD	ADDW4	:: DEVICE DESCRIPTOR WORD#4
516	001306	000000	\$DDW5: .WORD	ADDW5	:: DEVICE DESCRIPTOR WORD#5
517	001310	000000	\$DDW6: .WORD	ADDW6	:: DEVICE DESCRIPTOR WORD#6
518	001312	000000	\$DDW7: .WORD	ADDW7	:: DEVICE DESCRIPTOR WORD#7
519	001314	000000	\$DDW8: .WORD	ADDW8	:: DEVICE DESCRIPTOR WORD#8
520	001316	000000	\$DDW9: .WORD	ADDW9	:: DEVICE DESCRIPTOR WORD#9
521	001320	000000	\$DDW10: .WORD	ADDW10	:: DEVICE DESCRIPTOR WORD#10
522	001322	000000	\$DDW11: .WORD	ADDW11	:: DEVICE DESCRIPTOR WORD#11
523	001324	000000	\$DDW12: .WORD	ADDW12	:: DEVICE DESCRIPTOR WORD#12
524	001326	000000	\$DDW13: .WORD	ADDW13	:: DEVICE DESCRIPTOR WORD#13

581	001446	000077	CYL:	63.	;CYLINDER NUMBERS USED IN
582	001450	000177		127.	;CURRENT CROSSOVER TEST
583	001452	000277		191.	
584	001454	000377		255.	
585	001456	000477		319.	
586	001460	000577		383.	
587					
588	001462	000000	TIM1:	0	;USED IN TIMING TESTS
589	001464	000000	TIM2:	0	
590	001466	000000	TIM3:	0	
591	001470	000000	TIM4:	0	
592					
593	001472	000000	LPCNT:	0	;LOOP CTR USED IN CALCLK
594	001474	000000	LPTIM:	0	;LOOP TIME IN USEC
595					
596	001476	000000	SUM:	0	;LO ORDER FOR TIMING TESTS
597	001500	000000		0	;HI ORDER FOR TIMING TESTS
598	001502	000000	SUM1:	0	;LO ORDER FOR TIMING TESTS
599	001504	000000		0	;HI ORDER FOR TIMING TESTS
600					
601	001506	000000	WD1:	0	;ACTUAL HEADER/DATA WORD
602	001510	000000	WD2:	0	;EXPECTED DATA WORD
603					
604	001512	000000	OFFERR:	0	;SET WHEN WRITE CHECK ERROR ON OFFSET
605					
606					
607	001514	000000	HEAD:	0	;HEAD NUMBER
608	001516	000000	HD1:	0	;SHIFTED HEAD# FOR FORMATTER ROUTINE
609	001520	000000	FORMAT:	0	;FORMAT TYPE
610	001522	000000	FMT1:	0	;SHIFTED FORMAT FOR FORMATTER ROUTINE
611	001524	000000	WDCNT:	0	;WORD COUNT
612					
613	001526	000000	DATA0:	0	;ALL 0'S
614	001530	052525	DATA01:	52525	;0101 PATT
615	001532	177777	DATA1:	177777	;ALL 1'S
616	001534	133467	DPAT1:	133467	
617	001536	070627	DPAT2:	70627	
618					
619	001540	000000	WORD:	0	;HEADER/DATA WORD
620	001542	000000	HDWD:	0	;HEADER WORD FROM RK06
621					
622	001544	000000	BSERR:	0	;CANNOT READ BSE INFO WHEN SET
623	001546	000000	LIMERR:	0	;LIMIT DETECT ERROR FLAG
624					
625	001550	000102	HDTAB:	.BLKW 66.	;CALCULATED HEADER WORD TABLE
626	001754	000102	RHTAB:	.BLKW 66.	;FILLED AFTER READ HEADER CMD
627	002160	000102	SRTTAB:	.BLKW 66.	;ABOVE RHTAB SORTED STARTING FROM
628					;SECTOR 0 BY SORT ROUTINE
629					
630	002364	000400	BSE20:	.BLKW 256.	;20 SECTOR BSE INFO
631	003364	000400	BSE22:	.BLKW 256.	;22 SECTOR BSE INFO.
632	004364	000400	RDTAB:	.BLKW 256.	;FILLED AFTER READ DATA CMD
633					
634	005364	000633	INVCYL:	411.	;INVALID CYLINDER ADDR
635	005366	000634		412.	
636	005370	000640		416.	
637	005372	000700		448.	

637 005374 000740

480.

638
 639
 640 005376 001 002 004
 641 005401 010 020 040
 642 005404 100 200

ATTN: .BYTE 1,2,4,10,20,40,100,200 ;ATN 0-7 RESP.

643 .EVEN

644
 645 .LIST MD

646
 647
 648 :USE LOOP X TO OMIT SUBCLR
 649 :
 650

651 .MACRO LOOP A
 652 SCOP1
 653 MOV #STACK,SP ;RESTORE STK PTR

654
 655 .IF B A
 656 JSR PC,SUBCLR
 657 ERROR 24 ;CERR AFTER SCLR

658
 659 .ENDC
 660 .ENDM LOOP

661
 662 :
 663 :A=MSG A0 ERROR#, B=MSG B0 ERROR#
 664 :C=MSG A1 ERROR#, D=MSG B1 ERROR#
 665 :
 666 :E=<ERROR CONDITION DESCRIPTION>
 667 :F=D.DSC AFTER ATTN OR 0 AFTER DRIVE CLEAR OR ANY IMPLIED SEEKS

668
 669 :NOTE: F CAN BE ANY E.I COMBINATION DESIRED

670
 671 .MACRO CKWD12 A,B,C,D,E,F

672
 673 MOV #CCLR,RKCS1(R5) ;CONTR CLEAR
 674 JSR PC,GSTAT ;GET LATEST STATUS
 675 MOV #<F!D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
 676 JSR PC,CKMR2 ;CHECK MR2
 677 ERROR A ;MSG A0 ERROR E

678 CLR SBMR3
 679 JSR PC,CKMR3 ;CHECK MR3
 680 ERROR B ;MSG B0 ERROR E

681
 682 MOV #CCLR,RKCS1(R5)
 683 MOV #1,RKMR1(R5) ;SELECT WORD 1
 684 JSR PC,GSTAT
 685 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2

686 JSR PC,CKMR2
 687 ERROR C ;MSG A1 ERROR E

688 CLR SE MR3
 689 JSR PC,CKMR3
 690 ERROR D ;MSG B1 ERROR E

691
 692 .ENDM CKWD12

693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748

```

;A=CYL DIFF/OFFSET ERROR#
;B=CYL ADDR ERROR#
;C=<ERROR CONDITION DESCRIPTION>
.MACRO CKWD3 A,B,C,'D','E'

        MOV      #CCLR,RKCS1(R5)
        JSR      PC,RDCYLD      ;READ CYL DIFF IN RKMR2
        TST      CYLDIF
        BEQ      D
        ERROR    A              ;CYL DIFF/OFFSET NOT CLEARED C
D:       JSR      PC,RDCYLA      ;READ CYL ADDR IN RKMR3
        TST      CYLADD
        BEQ      E
        ERROR    B              ;CYL ADDR NOT CLEARED C
E:

.ENDM   CKWD3

.MACRO DRCLR 'A'

        MOV      #CCLR,RKCS1(R5)
        MOV      $UNIT,RKCS2(R5) ;DRIVE#
        MOV      #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
        MOV      T10,TEMP1      ;SETUP TIMEOUT
        JSR      PC,FRDY        ;FIND RDY
        ERROR    151            ;NO RDY AFTER DRIVE CLEAR CMD
        JSR      PC,TSTATN      ;TEST FOR ATTN
        BR       A
        ERROR    154            ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
A:

.ENDM   DRCLR

;USE CALIB X TO OMIT CKWD12 & CKWD3
.MACRO CALIB A,'C'

        MOV      #CCLR,RKCS1(R5)
        MOV      $UNIT,RKCS2(R5)
        MOV      #RECAL,RKCS1(R5) ;RECAL CMD
        ;RESET CYL DIFF/OFFSET & CYL ADDR REG
        ;IN RKMR2 & RKMR3 RESP.
        MOV      T10,TEMP1      ;SETUP TIMEOUT
        JSR      PC,FRDY        ;FIND RDY
        ERROR    124            ;RDY NOT SET AFTER RECAL CMD

        MOV      #1,RKMR1(R5)   ;SELECT WORD 1
        JSR      PC,GSTAT
        BIT      #D.RTZ,HMR2
        BNE      C
        ERROR    244            ;RTZ NOT SET DURING RECAL CMD
    
```

```

749 C:      MOV      T10,TEMP2      ;SETUP TIMEOUT
750      JSR      PC,FATT1        ;FIND ATTN
751      ERROR    55              ;NO ATTN AFTER RECAL CMD
752 .IF B  A
753      CKWD12   221,275,222,276,<AFTER RECAL CMD>,D.DSC
754      CKWD3    47,50,<AFTER RECAL CMD>
755 .ENDC
756      DRCLR
757
758 .ENDM  CALIB
759
760
761 ;
762 ; A=WRHEAD/<CFMT!WRHEAD>
763 ; USE WRHDR <A>,X TO OMIT CKWD12
764 ;
765 .MACRO  WRHDR  A,C,'D
766
767      MOV      #<A>,RKCS1(R5)    ;WRITE HEADER CMD
768      MOV      T50000,TEMP1      ;SETUP TIMEOUT
769      JSR      PC,FRDY           ;FIND RDY
770      ERROR    200              ;NO RDY AFTER WRITE HEADER CMD
771      JSR      PC,GSTAT         ;GET FRESH STATUS
772      BIT      #CERR,HCS1
773      BEQ      D
774      ERROR    201              ;CERR AFTER WRITE HEADER CMD
775      TYPE     MSG26            ;ABORTING DATA TESTS TO DO TIMING TESTS
776      JMP      TIMING
777
778 D:
779 .IF B  C
780      CKWD12   277,267,300,<70,<AFTER WRITE HEADER CMD>,D
781 .ENDC
782 .ENDM  WRHDR
783
784 ;
785 ; A=RDHEAD/<CFMT!RDHEAD>
786 ; USE RDHDR <A>,X TO OMIT CKWD12
787 ;
788 .MACRO  RDHDR  A,C,'D','E
789
790      MOV      #RHTAB,RO
791      MOV      #<A>,RKCS1(R5)    ;READ HEADER CMD
792      MOV      T50000,TEMP1      ;SETUP TIMEOUT
793      JSR      PC,FRDY           ;FIND RDY
794      ERROR    171              ;NO RDY AFTER READ HEADER CMD
795      BIT      #CERR,HCS1
796      BEQ      D
797      ERROR    174              ;CERR AFTER READ HEADER CMD
798      TYPE     MSG26            ;ABORTING DATA TESTS TO DO TIMING TESTS
799      JMP      TIMING
800
801 D:      MOV      RKDB(R5),(RO)+    ;1'ST WORD FROM SILO TO RHTAB
802      MOV      RKDB(R5),(RO)+    ;2'ND WORD
803      MOV      RKDB(R5),(RO)+    ;3'RD WORD
804

```


805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860

```

      BIT      #DLT,RKCS2(R5)
      BEQ      E
      JSR      PC,GSTAT
      ERROR    173          ;DLT AFTER READ HEADER CMD
      TYPE     MSG26       ;ABORTING DATA TESTS TO DO TIMING TESTS
      JMP      TIMING
E:
      .IF      B           C
      .ENDC
      .ENDM    RDHDR
      .MACRO   HDCHK3     ?A
      RDHDR   RDHEAD,X
      CMP     RHTAB,TOCYL ;CHECK WORD 0 ONLY, CYL#
      BEQ     A           ;BR IF SAME
      ERROR   S1         ;WRONG CYL# ON HEADER
A:
      .ENDM    HDCHK3
      ; A=TOCYL/FRCYL , B=HEAD#, C = 0 FOR 22 SECTOR, 1 FOR 20 SECTOR
      .MACRO   HDTBL     A,B,C
      MOV     A,CALADD    ;SETUP
      MOV     #B,HEAD    ;TO FILL
      MOV     #C,FORMAT  ;HEADER
      JSR     PC,FHDTAB  ;TABLE
      .ENDM    HDTBL
      ; QUICK SEEK. ENTER WITH CYL# IN RKDC
      .MACRO   QKSEEK    ?A
      MOV     #SEEK,RKCS1(R5) ;SEEK CMD
      MOV     T50,TEMP1    ;SETUP TIMEOUT
      JSR     PC,FRDY     ;FIND RDY
      ERROR   131         ;NO RDY AFTER SEEK CMD
      MOV     T50000,TEMP1 ;SETUP TIMEOUT
      JSR     PC,FATT2    ;FIND ATTN
      ERROR   132         ;NO ATTN AFTER SEEK CMD
      BIT     #CERR,HCS1
      BEQ     A
      ERROR   210         ;CERR AFTER SEEK CMD

```

A:

861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916

.ENDM QKSEEK

:A=WRDATA/<CFMT!WRDATA>
:USE WDATA <A>,X TO OMIT CKWD12

:MACRO WDATA A,C,'D

```

MOV      #<A>,RKCS1(R5) ;WRITE DATA CMD
MOV      T50000,TEMP1   ;SETUP TIMEOUT
JSR      PC,FRDY        ;FIND ROY
ERROR    11              ;NO ROY AFTER WRITE DATA CMD
JSR      PC,GSTAT       ;GET FRESH STATUS
BIT      #CERR,HCS1
BEQ      D
ERROR    12              ;CERR AFTER WRITE DATA CMD
TYPE     MSG26          ;ABORTING DATA TESTS TO DO TIMING
JMP      TIMING

```

```

D:
:IF      B              C
:        CKWD12        52,23,53,25,<AFTER WRITE DATA CMD>,D
.ENDC
.ENDM    WDATA

```

:A=RDDATA/<CFMT!RDATA>
:USE RDATA <A>,X TO OMIT CKWD12

:MACRO RDATA A,C,'D','E

```

MOV      #<A>,RKCS1(R5) ;READ DATA CMD
MOV      T50000,TEMP1   ;SETUP TIMEOUT
JSR      PC,FRDY        ;FIND ROY
ERROR    13              ;NO ROY AFTER READ DATA CMD
JSR      PC,GSTAT       ;GET FRESH STATUS
BIT      #DCK,HER
BEQ      D
ERROR    21              ;DATA CHECK ERROR AFTER READ DATA CMD
:        ;ECC ERROR DETECTED
TYPE     MSG26          ;ABORTING DATA TESTS TO DO TIMING
JMP      TIMING

```

```

D:
BIT      #CERR,HCS1
BEQ      E
ERROR    14              ;CERR AFTER READ DATA CMD
TYPE     MSG26          ;ABORTING DATA TESTS TO DO TIMING
JMP      TIMING

```

```

E:
:IF      B              C
:        CKWD12        54,26,56,30,<AFTER READ DATA CMD>,D
.ENDC

```

917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972

```

.ENDM RDATA

;A=WRCHK/<CFMT!WRCHK>
;C=EXPECTED DATA FOR TYPEOUT
;USE WRCHK <A>,DATA0,X TO OMIT CKWD12
.MACRO WRCHK A,C,D,'E','F'

    MOV    #<A>,RKCS1(R5) ;WRITE CHECK CMD
    MOV    T5000,TEMP1 ;SETUP TIMEOUT
    JSR    PC,FRDY ;FIND RDY
    ERROR  15 ;NO RDY AFTER WRITE CHECK CMD

    JSR    PC,GSTAT ;GET FRESH STATUS
    BIT    #CERR,HCS1
    BEQ    F

    BIT    #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
    BEQ    E
    MOV    RKDB(R5),WD1 ;ACTUAL WORD FOR TYPEOUT
    MOV    C,WD2 ;EXPECTED WORD FOR TYPEOUT
    JSR    PC,TRUERR ;CHECK AGAINST BSE INFO
    ERROR  16 ;WCE AFTER WRITE CMD
    BR     F

E:    ERROR  22 ;CERR AFTER WRITE CHK CMD

F:
    .IF    B D
        CKWD12 57,31,60,32,<AFTER WRITE CHECK CMD>,D
    .ENDC

.ENDM WRCHK

.MACRO OFFSET 'A'

    MOV    #A,SESCAPE
    MOV    #OFFSET,RKCS1(R5) ;OFFSET CMD
    MOV    T100,TEMP1 ;SETUP TIMEOUT
    JSR    PC,FRDY
    ERROR  33 ;NO RDY AFTER OFFSET CMD

    CLR    RKM1(R5) ;GET WORD 0
    JSR    PC,GSTAT
    MOV    #<D.PIP!D.SPIN!D.OFF!D.VV!D.DRA>,SBMR2
    JSR    PC,CKMR2 ;CHECK MR2
    ERROR  35 ;MSG A0 ERROR DURING OFFSET CMD

    CLR    SBMR3
    JSR    PC,CKMR3 ;CHECK MR3
    ERROR  61 ;MSG B0 ERROR DURING OFFSET CMD

```

973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028

```

MOV      #CCLR,RKCS1(R5)
MOV      #1,RKMR1(R5) ;SELECT WORD 1
JSR      PC,GSTAT
MOV      #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2
JSR      PC,CKMR2
ERROR    36 ;MSG A1 ERROR DURING OFFSET

CLR      SBMR3
JSR      PC,CKMR3
ERROR    62 ;MSG B1 ERROR DURING OFFSET

A:       CLR      $ESCAPE
MOV      T5000,TEMP1 ;SETUP TIMEOUT
JSR      PC,FATT2 ;FIND ATTN
ERROR    34 ;NO ATTN AFTER OFFSET CMD

CKWD12  260,261,37,40,<AFTER OFFSET CMD>,<D.DSC!D.OFF>

.ENDM    OFFSET

;A=CYL # TO GO TO
;B=FWD DIRECTION MSG
;C=REV DIRECTION MSG
.MACRO   SKTIM  A,B,C

JSR      PC,CALCLK ;CALIB TIME TO GO THRU 'LOOP'

CLR      R0 ;ITERATION COUNTER
CLR      SUM ;LO WORD OF FOWARD SEEK TIME
CLR      SUM+2 ;HI WORD OF FOWARD SEEK TIME
CLR      SUM1 ;LO WORD OF REVERSE SEEK TIME
CLR      SUM1+2 ;HI WORD OF REVERSE SEEK TIME

15:     JSR      PC,SUBCLR
ERROR    24 ;CERR AFTER SCLR

MOV      #7$, $ESCAPE

25:     MOV      #A,RKDC(R5) ;SETUP FOR CYL A
MOV      #SEEK,RKCS1(R5) ;SEEK CMD
MOV      T50,TEMP1
JSR      PC,FROY ;FIND ROY
ERROR    131 ;NO ROY AFTER SEEK CMD

CLR      LPCNT ;COUNT PASSES THRU LOOP
MOV      $UNIT,R4
JSR      PC,FATT3 ;FIND ATTN AND STEP 'LPCNT'
ERROR    132 ;NO ATTN AFTER SEEK CMD

JSR      PC,GSTAT
BIT      #CERR,HCS1
BEQ      55
ERROR    210 ;CERR AFTER SEEK CMD.

```

```

1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084

5S:  MOV    LPCNT,$TMP0    ;FROM LOOP
      MOV    LPTIM,$TMP1   ;FROM CALCLK
      MULT   $TMP0,$TMP1

      TST    RKDC(R5)
      BEQ    6S            ;BR IF THIS SEEK WAS REVERSE TO CYL 0

      ADD    $TMP1,SUM      ;SUM UP FOWARD SEEK TIMES (LSB)
      ADC    SUM+2
      ADD    $TMP2,SUM+2    ;MSB

      JSR    PC,SUBCLR
      ERROR  24            ;CERR AFTER SCLR
      BR     2S            ;SEEK TO CYL 0

6S:  ADD    $TMP1,SUM1      ;SUM UP REVERSE SEEK TIMES (LSB)
      ADC    SUM1+2
      ADD    $TMP2,SUM1+2  ;MSB

      INC    RO
      CMP    RO,#100.      ;ALL SEEKS DONE?
      BNE    1S            ;BR & REPEAT IF NO TO CYL A

      TYPE   B             ;FOWARD SEEK TIME
      JSR    PC,AVGTIM     ;CALC & TYPE AVG TIME (FOWARD)

      TYPE   C             ;REVERSE SEEK TIME
      MOV    $SUM1,SUM     ;SETUP FOR
      MOV    SUM+2,SUM+2   ;AVGTIM ROUTINE
      JSR    PC,AVGTIM     ;CALC & TYPE AVG TIME (REVERSE)

      TYPE   ,SCLRF
      TYPE   ,SCLRF
      CLR    $ESCAPE
      SKIP   R,<GO TO NEXT TEST>

7S:  CLR    $ESCAPE
      CALIB X

.ENDM SKTIM
.MACRO EOPGM

SCOPE
MOV    #STACK,SP
INC    $DEVCT             ;INCR COUNT FOR # DRIVES CHECKED
CMP    DRIVS,$DEVCT      ;ARE ALL DRIVES PRESENT TESTED?
BEQ    $EOP1+2           ;BR IF YES
JMP    NUDRV             ;ELSE TEST NEXT DRIVE PRESENT
$EOP1: SCOPE
.ENDM EOPGM

;A= ERROR #
;B = ERROR CONDITION

```

```

1085 .MACRO OFFDIR A,B,?C,?D
1086 MOV R2,RKASOF(R5) ;REFRESH RKASOF
1087
1088 BIT #BIT7,R2
1089 BNE C ;BR IF NEG OFFSET
1090
1091 JSR PC,RDCYLD
1092 CMP R2,CYLDIF ;CHECK POS OFFSET
1093 BEQ D
1094 ERROR A ;OFFSET IN RKMR2 NOT = RKASOF
1095 BR D ;B
1096
1097 C: JSR PC,RDCYLD
1098 CMP R1,CYLDIF ;CHECK NEG OFFSET
1099 BEQ D
1100 ERROR A ;OFFSET IN RKMR2 NOT = RKASOF
1101 ;B
1102
1103 D:
1104 .ENDM OFFDIR
1105
1106 ;A&B=ERROR # & CYL #
1107 ;C&D=ERROR # & CYL #
1108
1109 .MACRO AVGVEL A,B,C,D,?E,?F,?G,?H
1110
1111 MOV T500,TEMP1
1112 JSR PC,QKCYLD ;READ CYL DIFF (NO SHIFTING)
1113 BIT #BIT11,CYLDIF ;CHECK BIT 7 (NO SHIFTING)
1114 BNE F ;BR IF SET
1115 DEC TEMP1
1116 BNE E
1117 ERROR A ;CANNOT FIND CYL B
1118
1119 F: CLR LPCNT
1120 G: JSR PC,QKCYLD
1121 BIT #BIT11,CYLDIF ;CHECK BIT 7 (NO SHIFTING)
1122 BEQ H
1123
1124 JSR PC,LOOP
1125 TST LPCNT
1126 BNE G
1127 ERROR C ;CANNOT FIND CYL D
1128
1129 H:
1130 .ENDM AVGVEL
1131
1132 .NLIST MD
1133
1134 ;
1135 ;THE FOLLOWING ARE HOLDING REGISTERS FOR THE RK611 REGISTERS
1136 ;THEY ARE LOADED AFTER RDY IS REC'D FROM WRDY ROUTINE.
1137 ;
1138
1139 005406 000000 HCS1: 0 ;HOLD RKCS1
1140 005410 000000 HCS2: 0 ;HOLD RKCS2

```

1141	005412	000000	HWC:	0	;HOLD RKWC
1142	005414	000000	HBA:	0	;ETC.
1143	005416	000000	HDA:	0	
1144	005420	000000	HDS:	0	
1145	005422	000000	HER:	0	
1146	005424	000000	HASOF:	0	
1147	005426	000000	HDC:	0	
1146	005430	000000	HDB:	0	
1149	005432	000000	HMR1:	0	
1150	005434	000000	HMR2:	0	
1151	005436	000000	HMR3:	0	
1152	005440	000000	HPOS:	0	
1153	005442	000000	HPAT:	0	
1154					
1155					
1156			;THE FOLLOWING ARE "SHOULD BE" REGISTERS FOR THE ABOVE REGISTERS		
1157					
1158	005444	000000	SBMR2:	0	; 'SHOULD BE' FOR HMR2
1159	005446	000000	SBMR3:	0	;ETC
1160					
1161	005450	000000	TEMP1:	0	;TEMPORARY STORAGE.
1162	005452	000000	TEMP2:	0	
1163	005454	000000	TEMP3:	0	
1164	005456	000000	TEMP4:	0	
1165	005460	000000	TEMPS:	0	
1166					
1167					
1168			;ALL THE FLAGS BELOW ARE CLEARED INITIALLY BY THE CLRFLG ROUTINE.		
1169					
1170					
1171	005462	000000	DDUMP:	0	;FLAG - SET WHEN IN DDP DUMP MODE
1172	005464	000000	DDPCH:	0	;FLAG - SET WHEN IN DDP CHAIN MODE
1173	005466	000000	ACT11:	0	;FLAG - SET WHEN IN ACT11 MODE OF OPERATION
1174	005470	000000	PPTP:	0	;FLAG - SET WHEN PROGRAM LOADED BY PAPER TAPE
1175	005472	000000	DRIVS:	0	;CONTAINS THE NUMBER OF DRIVES PRESENT
1176					
1177			;THE FLAGS BELOW ARE SET TO 1 TO INDICATE THAT A PARTICULAR DRIVE		
1178			;IS PRESENT AND IS TO BE TESTED.		
1179					
1180	005474	000000	DRIV0:	0	;FLAG SET TO 1 WHEN DRIVE 0 PRESENT
1181	005476	000000	DRIV1:	0	;FOR DRIVE 1
1182	005500	000000	DRIV2:	0	;FOR DRIVE 2
1183	005502	000000	DRIV3:	0	;FOR DRIVE 3
1184	005504	000000	DRIV4:	0	;FOR DRIVE 4
1185	005506	000000	DRIV5:	0	;FOR DRIVE 5
1186	005510	000000	DRIV6:	0	;FOR DRIVE 6
1187	005512	000000	DRIV7:	0	;FOR DRIVE 7
1188					
1189	005514	000000	LCLKF:	0	;L-CLOCK FLAG PRESENT FLAG
1190	005516	000000	PCLKF:	0	;P-CLOCK FLAG PRESENT FLAG
1191	005520	000000	DOTIM:	0	;SET IF EITHER CLOCK PRESENT FOR TIMING TESTS.
1192	005522	000000	SIZFLG:	0	;SET IF DEFAULT DO SIZING IN TEST 1

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ;;POINTS TO THE ERROR MESSAGE
;* DH ;;POINTS TO THE DATA HEADER
;* DT ;;POINTS TO THE DATA
;* DF ;;POINTS TO THE DATA FORMAT

\$ERRTB:

1193					
1194					
1195					
1196					
1197					
1198					
1199					
1200					
1201					
1202					
1203					
1204					
1205					
1206					
1207	005524				
1208					
1209					
1210	005524	046275			
1211	005526	051464	EM2		;DR # IN RKCS2 CANNOT BE READ BACK CORRECTLY IN RKMR2
1212	005530	053676	DH1		
1213	005532	054134	DT1		
1214			DF1		
1215					
1216	005534	046514			
1217	005536	051464	EM5		;DETECTED MDS
1218	005540	053676	DH1		
1219	005542	054134	DT1		
1220			DF1		
1221					
1222	005544	046535			
1223	005546	051464	EM6		;DETECTED UFE
1224	005550	053676	DH1		
1225	005552	054134	DT1		
1226			DF1		
1227					
1228	005554	046556			
1229	005556	051464	EM7		;DETECTED DRA & NED RESET (WRONG PORT SELECTED?)
1230	005560	053676	DH1		
1231	005562	054134	DT1		
1232			DF1		
1233	005564	046645			
1234	005566	051464	EM8		;DR PRESENT BUT NOT SPECIFIED BY OPERATOR
1235	005570	053676	DH1		
1236	005572	054134	DT1		
1237			DF1		
1238					
1239	005574	046721			
1240	005576	051464	EM9		;DR NOT PRESENT BUT SPECIFIED BY OPERATOR
1241	005600	053676	DH1		
1242	005602	054134	DT1		
1243			DF1		
1244					
1245	005604	046775			
1246	005606	051464	EM10		;ABORT TEST, COULD NOT REFERENCE CONTROLLER REGISTER
1247	005610	053676	DH1		
1248	005612	054134	DT1		
			DF1		

1249				
1250			;ERROR 10	
1251	005614	047060	EM11	;DRA & NED BOTH SET
1252	005616	051464	DH1	
1253	005620	053676	DT1	
1254	005622	054134	DF1	
1255			;ERR 11	
1256	005624	047124	EM12	;NO RDY
1257	005626	052525	DH27	;AFTER WRITE DATA CMD
1258	005630	053676	DT1	
1259	005632	054250	DF10	
1260			;ERR 12	
1261	005634	047520	EM21	;CERR SET
1262	005636	052525	DH27	
1263	005640	053676	DT1	
1264	005642	054250	DF10	
1265			;ERR 13	
1266	005644	047124	EM12	;NO RDY
1267	005646	052475	DH26	;AFTER READ DATA CMD
1268	005650	053676	DT1	
1269	005652	054250	DF10	
1270			;ERR 14	
1271	005654	047520	EM21	;CERR SET
1272	005656	052475	DH26	
1273	005660	053676	DT1	
1274	005662	054250	DF10	
1275			;ERR 15	
1276	005664	047124	EM12	;NO RDY
1277	005666	052610	DH32	;AFTER WRITE CHECK CMD
1278	005670	053676	DT1	
1279	005672	054250	DF10	
1280			;ERR 16	
1281	005674	050304	EM80	;WRITE CHECK ERROR SET
1282	005676	052610	DH32	;AFTER WRITE CHECK CMD
1283	005700	054016	DT6	
1284	005702	054150	DF3	
1285			;ERR 17	
1286	005704	050343	EM81	;WRITE CHECK CMD NOT FUNCTIONING
1287	005706	053410	DH52	;WITH INTENTIONAL MISCOMPARE
1288	005710	051464	DH1	
1289	005712	054250	DF10	
1290			;ERR 20	
1291	005714	050407	EM82	;READ DATA NOT COMPARE WITH WRITE DATA
1292	005716	052475	DH26	;AFTER READ DATA CMD
1293	005720	054016	DT6	
1294	005722	054150	DF3	
1295			;ERR 21	
1296	005724	050461	EM83	;DATA CHECK ERROR
1297	005726	052475	DH26	
1298	005730	053676	DT1	
1299	005732	054250	DF10	
1300			;ERR 22	
1301	005734	047520	EM21	;CERR SET
1302	005736	052610	DH32	;AFTER WRITE CHECK CMD
1303	005740	053676	DT1	
1304	005742	054250	DF10	

1305			:ERR 23		
1306	005744	047435		EM18	:MSG B0 ERROR
1307	005746	052525		DH27	:AFTER WRITE DATA CMD
1308	005750	053676		DT1	
1309	005752	054230		DF9	
1310			:ERROR 24		
1311	005754	047520		EM21	:CERR SET
1312	005756	052345		DH21	:AFTER SCLR
1313	005760	053676		DT1	
1314	005762	054250		DF10	
1315			:ERR 25		
1316	005764	047477		EM20	:MSG B1 ERROR
1317	005766	052525		DH27	
1318	005770	053676		DT1	
1319	005772	054230		DF9	
1320			:ERR 26		
1321	005774	047435		EM18	
1322	005776	052475		DH26	:AFTER READ DATA CMD
1323	006000	053676		DT1	
1324	006002	054230		DF9	
1325			:ERROR 27		
1326				EM24	:VOL VALID NOT SET
1327	006004	047542		DH19	:AFTER PACK CMD
1328	006006	052267		DT1	
1329	006010	053676		DF10	
1330	006012	054250			
1331			:ERR 30		
1332	006014	047477		EM20	:MSG B1 ERROR
1333	006016	052475		DH26	:AFTER READ DATA CMD.
1334	006020	053676		DT1	
1335	006022	054230		DF9	
1336			:ERR 31		
1337	006024	047435		EM18	:MSG B0 ERROR
1338	006026	052610		DH32	:AFTER WRITE CHECK CMD
1339	006030	053676		DT1	
1340	006032	054230		DF9	
1341			:ERR 32		
1342	006034	047477		EM20	:MSG B1 ERROR
1343	006036	052610		DH32	
1344	006040	053676		DT1	
1345	006042	054230		DF9	
1346			:ERR 33		
1347	006044	047124		EM12	:CONTR NOT READY
1348	006046	052425		DH24	:AFTER OFFSET CMD
1349	006050	053722		DT2	
1350	006052	054264		DF11	
1351			:ERR 34		
1352	006054	047162		EM13	:NO ATTN
1353	006056	052425		DH24	
1354	006060	053722		DT2	
1355	006062	054264		DF11	
1356			:ERR 35		
1357	006064	047414		EM17	:MSG A0 ERROR
1358	006066	053444		DH53	:DURING OFFSET COMMAND
1359	006070	053676		DT1	
1360	006072	054210		DF8	

1361					
1362	006074	047456			
1363	006076	053444			
1364	006100	053676			
1365	006102	054210			
1366					
1367	006104	047456			
1368	006106	052425			
1369	006110	053676			
1370	006112	054210			
1371					
1372	006114	047477			
1373	006116	052425			
1374	006120	053676			
1375	006122	054230			
1376					
1377	006124	047204			
1378	006126	051764			
1379	006130	053744			
1380	006132	054144			
1381					
1382	006134	047745			
1383	006136	052425			
1384	006140	053676			
1385	006142	054250			
1386					
1387	006144	050516			
1388	006146	052373			
1389	006150	053722			
1390	006152	054264			
1391					
1392	006154	047242			
1393	006156	051464			
1394	006160	054114			
1395	006162	054164			
1396					
1397	006164	050516			
1398	006166	053355			
1399	006170	053722			
1400	006172	054264			
1401					
1402	006174	000000			
1403	006176	000000			
1404	006200	000000			
1405	006202	000000			
1406					
1407	006204	047640			
1408	006206	052216			
1409	006210	053676			
1410	006212	054250			
1411					
1412	006214	047707			
1413	006216	052216			
1414	006220	053676			
1415	006222	054250			
1416					

```

;ERR 36
EM19 ;MSG A1 ERROR
DH53
DT1
DF8

;ERR 37
EM19 ;MSG A1 ERROR
DH24 ;AFTER OFFSET CMD
DT1
DF8

;ERR 40
EM20 ;MSG B1 ERROR
DH24
DT1
DF9

;ERR 41
EM14 ;UNEXP MEM PCORITY TRAP
CH8 ;TEST #, TRAP PC
DT3
DF2

;ERR 42
EM41 ;CYL ADDR IN RKMR2 DID NOT REMAIN CLEARED
DH24
DT1
DF10

;ERR 43
EM85
DH22
DT2
DF11

;ERR 44
EM15 ;WCE AT CYL 411,TRK 2, SEC 21
DH1
DT10
DF4

;ERR 45
EM85 ;OFFSET BIT IN RKMR2 CLEARED
DH51 ;AFTER SEEK TO SELF
DT2
DF11

;ERR 46
0
0
0
0

;ERROR 47
EM39 ;CYL DIFF/OFFSET IN RKMR2 NOT CLEARED
DH17 ;AFTER RECAL CMD
DT1
DF10

;ERROR 50
EM40 ;CYL ADDR IN RKMR3 NOT CLEARED
DH17 ;AFTER RECAL COMD
DT1
DF10

;ERR 51

```

1417	006224	051040	EM93		
1418	006226	052452	DH25		;WRONG CYL# IN HEADER WORD (MISPOSITION)
1419	006230	054076	DT9		;AFTER SEEK CMD
1420	006232	054350	DF20		
1421				;ERR 52	
1422	006234	047414	EM17		;MSG A0 ERROR
1423	006236	052525	DH27		;AFTER WRITE DATA CMD
1424	006240	053676	DT1		
1425	006242	054210	DF8		
1426				;ERR 53	
1427	006244	047456	EM19		;MSG A1 ERROR
1428	006246	052525	DH27		
1429	006250	053676	DT1		
1430	006252	054210	DF8		
1431				;ERR 54	
1432	006254	047414	EM17		;MSG A0 ERROR
1433	006256	052475	DH26		;AFTER READ DATA CMD
1434	006260	053676	DT1		
1435	006262	054210	DF8		
1436				;ERROR 55	
1437	006264	047162	EM13		;NO ATTN
1438	006266	052216	DH17		;AFTER RECAL CMD
1439	006270	053676	DT1		
1440	006272	054250	DF10		
1441				;ERR 56	
1442	006274	047456	EM19		;MSG A1 ERROR
1443	006276	052475	DH26		
1444	006300	053676	DT1		
1445	006302	054210	DF8		
1446				;ERR 57	
1447	006304	047414	EM17		;MSG A0 ERROR
1448	006306	052610	DH32		;AFTER WRITE CHECK CMD
1449	006310	053676	DT1		
1450	006312	054210	DF8		
1451				;ERR 60	
1452	006314	047456	EM19		;MSG A1 ERROR
1453	006316	052610	DH32		
1454	006320	053676	DT1		
1455	006322	054210	DF8		
1456				;ERR 61	
1457	006324	047435	EM18		;MSG B0 ERROR
1458	006326	053444	DH53		;DURING OFFSET CMD
1459	006330	053676	DT1		
1460	006332	054230	DF9		
1461				;ERR 62	
1462	006334	047477	EM20		;MSG B1 ERROR
1463	006336	053444	DH53		
1464	006340	053676	DT1		
1465	006342	054230	DF9		
1466				;ERR 63	
1467	006344	000000	0		
1468	006346	000000	0		
1469	006350	000000	0		
1470	006352	000000	0		
1471				;ERR 64	
1472	006354	050622	EM88		;DID NOT FIND SECTOR 0 FROM INDEX

1473	006356	053472	DH54	;AFTER FORMAT CHANGE AND READY REC'D
1474	006360	053676	DT1	
1475	006362	054250	DF10	
1476				
1477	006364	000000	0	
1478	006366	000000	0	
1479	006370	000000	0	
1480	006372	000000	0	
1481				
1482	006374	050051	EM60	;NO HEAD HOME
1483	006376	052242	DH18	;AFTER UNLOAD CMD
1484	006400	053676	DT1	
1485	006402	054250	DF10	
1486				
1487	006404	050663	EM89	;HDS HOME NOT CLEARED
1488	006406	052005	DH9	;DURING START SPIN CMD
1489	006410	053676	DT1	
1490	006412	054250	DF10	
1491				
1492	006414	050723	EM90	;TRACK FOLL OK NOT SET
1493	006416	052005	DH9	
1494	006420	053676	DT1	
1495	006422	054250	DF10	
1496				
1497	006424	050762	EM91	;REV NOT SET
1498	006426	052005	DH9	
1499	006430	053676	DT1	
1500	006432	054250	DF10	
1501				
1502	006434	051007	EM92	;REV NOT CLEARED
1503	006436	052005	DH9	
1504	006440	053676	DT1	
1505	006442	054250	DF10	
1506				
1507	006444	047162	EM13	;NO ATTN
1508	006446	052242	DH18	;AFTER UNLOAD CMD
1509	006450	053722	DT2	
1510	006452	054264	DF11	
1511				
1512	006454	047162	EM13	;NO ATTN
1513	006456	052060	DH10	;AT END OF HEAD LOADING
1514	006460	053676	DT1	
1515	006462	054250	DF10	
1516				
1517	006464	047414	EM17	;MSG A0 ERROR
1518	006466	052060	DH10	;AT END OF HEAD LOADIG
1519	006470	053676	DT1	
1520	006472	054210	DF8	
1521				
1522	006474	047435	EM18	;MSG B0 ERROR
1523	006476	052060	DH10	
1524	006500	053676	DT1	
1525	006502	054230	DF9	
1526				
1527	006504	047456	EM19	;MSG A1 ERROR
1528	006506	052060	DH10	

1529	006510	053676	DT1	
1530	006512	054210	DF8	
1531			:ERR 100	
1532	006514	047477	EM20	;MSG B1 ERROR
1533	006516	052060	DH1C	
1534	006520	053676	DT1	
1535	006522	054230	DF9	
1536			:ERROR 101	
1537	006524	051121	EM94	;OFFSET NOT CLEARED
1538	006526	053305	DH47	;AFTER READ HEADER WITH MOVEMENT
1539	006530	053676	DT1	
1540	006532	054250	DF10	
1541			:ERROR 102	
1542	006534	051170	EM95	;FORMAT NOT SET
1543	006536	052525	DH27	;AFTER WRITE DATA CMD
1544	006540	053676	DT1	
1545	006542	054250	DF10	
1546			:ERR 103	
1547	006544	051170	EM95	
1548	006546	052610	DH32	;AFTER WRITE CHECK CMD
1549	006550	053676	DT1	
1550	006552	054250	DF10	
1551			:ERR 104	
1552	006554	047640	EM39	;OFFSET NOT RESET
1553	006556	053611	DH57	;AFTER WRITE CMD WITH OFFSET
1554	006560	053676	DT1	
1555	006562	054250	DF10	
1556			:ERR 105	
1557	006564	047707	EM40	;CYL ADDR NOT 0
1558	006566	053611	DH57	
1559	006570	053676	DT1	
1560	006572	054250	DF10	
1561			:ERR 106	
1562	006574	051224	EM96	;CANNOT FIND SECTOR 17
1563	006576	051464	DH1	
1564	006600	053676	DT1	
1565	006602	054134	DF1	
1566			:ERR 107	
1567	006604	051252	EM97	;HEAD SWITCHING > 16MS
1568	006606	052525	DH27	;AFTER WRITE DATA CMD
1569	006610	053770	DT5	
1570	006612	054314	DF15	
1571			:ERR 110	
1572	006614	051313	EM98	;CANNOT FIND CYL 128
1573	006616	052452	DH25	;AFTER SEEK CMD
1574	006620	053676	DT1	
1575	006622	054250	DF10	
1576			:ERR 111	
1577	006624	051344	EM99	;CANNOT FIND CYL 256
1578	006626	052452	DH25	
1579	006630	053676	DT1	
1580	006632	054250	DF10	
1581			:ERR 112	
1582	006634	051375	EM100	;DRIVE OFF TRACK SET
1583	006636	052525	DH27	;AFTER WRITE DATA CMD
1584	006640	053750	DT4	

1585	006642	054174		
1586			:ERR 113	
1587	006644	047575	DF6	
1588	006646	052525	EM36	:CYL ADDR IN RKMR3 INCORRECT
1589	006650	053750	DH27	
1590	006652	054174	DT4	
1591			DF6	
1592	006654	050561	:ERROR 114	
1593	006656	052425	EM86	:OFFSET IN RKMR2 NOT = RKASOF
1594	006660	053722	DH24	:AFTER OFFSET CMD
1595	006662	054264	DT2	
1596			DF11	
1597	006664	050561	:ERR 115	
1598	006666	052373	EM86	
1599	006670	053722	DH22	:AFTER DRIVE CLEAR CMD
1600	006672	054264	DT2	
1601			DF11	
1602	006674	047124	:ERROR 116	
1603	006676	052267	EM12	:CONT NOT RDY
1604	006700	053676	DH19	:AFTER PACK CMD
1605	006702	054250	DT1	
1606			DF10	
1607	006704	047124	:ERROR 117	
1608	006706	052312	EM12	:CONT NOT RDY
1609	006710	053676	DH20	:AFTER SEL DR CMD
1610	006712	054250	DT1	
1611			DF10	
1612	006714	047124	:ERROR 120	
1613	006716	052345	EM12	
1614	006720	053676	DH21	:AFTER SUBSYS CLEAR
1615	006722	054250	DT1	
1616			DF10	
1617	006724	047124	:ERROR 121	
1618	006726	052005	EM12	
1619	006730	053676	DH9	:AFTER START SPINDLE CMD
1620	006732	054250	DT1	
1621			DF10	
1622	006734	051432	:ERROR 122	
1623	006736	052556	EM101	:DID NOT GO TO CYL 10
1624	006740	053676	DH30	:AFTER READ HEADER CMD
1625	006742	054250	DT1	
1626			DF10	
1627	006744	050561	:ERROR 123	
1628	006746	053355	EM86	:RKMR2 OFFSET NOT = RKASOF
1629	006750	053722	DH51	:AFTER SEEK TO SELF
1630	006752	054264	DT2	
1631			DF11	
1632	006754	047124	:ERROR 124	
1633	006756	052216	EM12	
1634	006760	053676	DH17	:AFTER RECAL CMD
1635	006762	054250	DT1	
1636			DF10	
1637	006764	000000	:ERROR 125	
1638	006766	000000	0	
1639	006770	000000	0	
1640	006772	000000	0	

1641			:ERROR 126	
1642	006774	000000	0	
1643	006776	000000	0	
1644	007000	000000	0	
1645	007002	000000	0	
1646			:ERROR 127	
1647	007004	000000	0	
1648	007006	000000	0	
1649	007010	000000	0	
1650	007012	000000	0	
1651			:ERROR 130	
1652	007014	000000	0	
1653	007016	000000	0	
1654	007020	000000	0	
1655	007022	000000	0	
1656			:ERROR 131	
1657	007024	047124	EM12	;NO RDY
1658	007026	052452	DH25	;AFTER SEEK CMD
1659	007030	053676	DT1	
1660	007032	054250	DF10	
1661			:ERROR 132	
1662	007034	047162	EM13	;NO ATTN
1663	007036	052452	DH25	
1664	007040	053676	DT1	
1665	007042	054250	DF10	
1666			:ERROR 133	
1667	007044	047414	EM17	;MSG A0 ERROR
1668	007046	052452	DH25	
1669	007050	053676	DT1	
1670	007052	054210	DF8	
1671			:ERROR 134	
1672	007054	047435	EM18	;MSG B0 ERROR
1673	007056	052452	DH25	
1674	007060	053676	DT1	
1675	007062	054230	DF9	
1676			:ERROR 135	
1677	007064	047456	EM19	;MSG A1 ERROR
1678	007066	052452	DH25	
1679	007070	053676	DT1	
1680	007072	054210	DF8	
1681			:ERROR 136	
1682	007074	047477	EM20	;MSG B1 ERROR.
1683	007076	052452	DH25	
1684	007100	053676	DT1	
1685	007102	054230	DF9	
1686			:ERROR 137	
1687	007104	047640	EM39	;CYL DIFF/OFFSET IN RKM2 NOT CLEARED
1688	007106	052452	DH25	
1689	007110	053676	DT1	
1690	007112	054250	DF10	
1691			:ERROR 140	
1692	007114	000000	0	
1693	007116	000000	0	
1694	007120	000000	0	
1695	007122	000000	0	
1696			:ERROR 141	

1697	007124	000000	0
1698	007126	000000	0
1699	007130	000000	0
1700	007132	000000	0
1701			;ERROR 142
1702	007134	000000	0
1703	007136	000000	0
1704	007140	000000	0
1705	007142	000000	0
1706			;ERROR 143
1707	007144	000000	0
1708	007146	000000	0
1709	007150	000000	0
1710	007152	000000	0
1711			;ERROR 144
1712	007154	000000	0
1713	007156	000000	0
1714	007160	000000	0
1715	007162	000000	0
1716			;ERROR 145
1717	007164	000000	0
1718	007166	000000	0
1719	007170	000000	0
1720	007172	000000	0
1721			;ERROR 146
1722	007174	000000	0
1723	007176	000000	0
1724	007200	000000	0
1725	007202	000000	0
1726			;ERROR 147
1727	007204	000000	0
1728	007206	000000	0
1729	007210	000000	0
1730	007212	000000	0
1731			;ERROR 150
1732	007214	000000	0
1733	007216	000000	0
1734	007220	000000	0
1735	007222	000000	0
1736			;ERROR 151
1737	007224	047124	EM12
1738	007226	052373	DH22
1739	007230	053676	DT1
1740	007232	054250	DF10
1741			;ERROR 152
1742	007234	000000	0
1743	007236	000000	0
1744	007240	000000	0
1745	007242	000000	0
1746			;ERROR 153
1747	007244	000000	0
1748	007246	000000	0
1749	007250	000000	0
1750	007252	000000	0
1751			;ERROR 154
1752	007254	050016	EM55

;NO RDY
;AFTER CLEAR CMD

;ATTN NOT CLEARED

1753	007256	052373		DH22
1754	007260	053722		DT2
1755	007262	054264		DF11
1756			;ERROR 155	0
1757	007264	000000		0
1758	007266	000000		0
1759	007270	000000		0
1760	007272	000000		0
1761			;ERROR 156	0
1762	007274	000000		0
1763	007276	000000		0
1764	007300	000000		0
1765	007302	000000		0
1766			;ERROR 157	0
1767	007304	000000		0
1768	007306	000000		0
1769	007310	000000		0
1770	007312	000000		0
1771			;ERROR 160	0
1772	007314	000000		0
1773	007316	000000		0
1774	007320	000000		0
1775	007322	000000		0
1776			;ERROR 161	0
1777	007324	000000		0
1778	007326	000000		0
1779	007330	000000		0
1780	007332	000000		0
1781			;ERROR 162	0
1782	007334	000000		0
1783	007336	000000		0
1784	007340	000000		0
1785	007342	000000		0
1786			;ERROR 163	0
1787	007344	000000		0
1788	007346	000000		0
1789	007350	000000		0
1790				0
1791				
1792				
1793				
1794				
1795				
1796				
1797				
1798				
1799				
1800				
1801				
1802				
1803				
1804				
1805				
1806				
1807				
1808				

1809					
1810					
1811	007352	000000			
1812			;ERROR	164	
1813	007354	000000			
1814	007356	000000			
1815	007360	000000			
1816	007362	000000			
1817			;ERROR	165	
1818	007364	000000			
1819	007366	000000			
1820	007370	000000			
1821	007372	000000			
1822			;ERROR	166	
1823	007374	000000			
1824	007376	000000			
1825	007400	000000			
1826	007402	000000			
1827			;ERROR	167	
1828	007404	000000			
1829	007406	000000			
1830	007410	000000			
1831	007412	000000			
1832			;ERROR	170	
1833	007414	000000			
1834	007416	000000			
1835	007420	000000			
1836	007422	000000			
1837			;ERROR	171	
1838	007424	047124			;NO RDY
1839	007426	052556			;AFTER READ HEADER CMD
1840	007430	053676			
1841	007432	054250			
1842			;ERROR	172	
1843	007434	000000			
1844	007436	000000			
1845	007440	000000			
1846	007442	000000			
1847			;ERROR	173	
1848	007444	050127			;DLT SET
1849	007446	052556			
1850	007450	053770			
1851	007452	054314			
1852			;ERROR	174	
1853	007454	047520			;CERR SET
1854	007456	052556			
1855	007460	053770			
1856	007462	054314			
1857			;ERROR	175	
1858	007464	047640			;CYL DIFF NOT CLEARED
1859	007466	052060			;AT END OF HEAD LOADING
1860	007470	053676			
1861	007472	054250			
1862			;ERROR	176	
1863	007474	047707			;CYL ADDR NOT CLEARED.
1864	007476	052060			

1865	007500	053676	011	
1866	007502	054250	DF10	
1867			;ERROR 177	
1868	007504	000000	0	
1869	007506	000000	0	
1870	007510	000000	0	
1871	007512	000000	0	
1872			;ERROR 200	
1873	007514	047124	EM12	;NO RDY
1874	007516	052750	DH39	;AFTER WRITE HEADER CMD
1875	007520	053770	DT5	
1876	007522	054314	DF15	
1877			;ERROR 201	
1878	007524	047520	EM21	;CERR SET
1879	007526	052750	DH39	
1880	007530	053770	DT5	
1881	007532	054314	DF15	
1882			;ERROR 202	
1883	007534	050150	EM65	;READ HEADER ERROR
1884	007536	051464	DH1	
1885	007540	054036	DT7	
1886	007542	054304	DF14	
1887			;ERROR 203	
1888	007544	047414	EM17	;MSG A0 ERROR
1889	007546	052642	DH33	;DURING SEEK CMD
1890	007550	053676	DT1	
1891	007552	054210	DF8	
1892			;ERROR 204	
1893	007554	047435	EM18	;MSG B0 ERROR
1894	007556	052642	DH33	
1895	007560	053676	DT1	
1896	007562	054230	DF9	
1897			;ERROR 205	
1898	007564	047456	EM19	;MSG A1 ERROR
1899	007566	052642	DH33	
1900	007570	053676	DT1	
1901	007572	054210	DF8	
1902			;ERROR 206	
1903	007574	047477	EM20	;MSG B1 ERROR
1904	007576	052642	DH33	
1905	007600	053676	DT1	
1906	007602	054230	DF9	
1907			;ERROR 207	
1908	007604	047575	EM36	;CYL ADDR IN RKMR3 INCORRECT
1909	007606	052452	DH25	;AFTER SEEK CMD
1910	007610	053750	DT4	
1911	007612	054174	DF6	
1912			;ERROR 210	
1913	007614	047520	EM21	;CERR SET
1914	007616	052452	DH25	
1915	007620	053676	DT1	
1916	007622	054250	DF10	
1917			;ERROR 211	
1918	007624	000000	0	
1919	007626	000000	0	
1920	007630	000000	0	

1921	007632	000000	0	
1922			;ERROR 212	
1923	007634	000000	0	
1924	007636	000000	0	
1925	007640	000000	0	
1926	007642	000000	0	
1927			;ERROR 213	
1928	007644	000000	0	
1929	007646	000000	0	
1930	007650	000000	0	
1931	007652	000000	0	
1932			;ERROR 214	
1933	007654	000000	0	
1934	007656	000000	0	
1935	007660	000000	0	
1936	007662	000000	0	
1937			;ERROR 215	
1938	007664	000000	0	
1939	007666	000000	0	
1940	007670	000000	0	
1941	007672	000000	0	
1942			;ERROR 216	
1943	007674	000000	0	
1944	007676	000000	0	
1945	007700	000000	0	
1946	007702	000000	0	
1947			;ERROR 217	
1948	007704	000000	0	
1949	007706	000000	0	
1950	007710	000000	0	
1951	007712	000000	0	
1952			;ERROR 220	
1953	007714	000000	0	
1954	007716	000000	0	
1955	007720	000000	0	
1956	007722	000000	0	
1957			;ERROR 221	
1958	007724	047414	EM17	;MSG A0 ERROR
1959	007726	052216	DH17	
1960	007730	053676	DT1	
1961	007732	054210	DF8	
1962			;ERROR 222	
1963	007734	047456	EM19	;MSG A1 ERROR
1964	007736	052216	DH17	
1965	007740	053676	DT1	
1966	007742	054210	DF8	
1967			;ERROR 223	
1968	007744	000000	0	
1969	007746	000000	0	
1970	007750	000000	0	
1971	007752	000000	0	
1972			;ERROR 224	
1973	007754	000000	0	
1974	007756	000000	0	
1975	007760	000000	0	
1976	007762	000000	0	

1977			;ERROR 225	
1978	007764	000000	0	
1979	007766	000000	0	
1980	007770	000000	0	
1981	007772	000000	0	
1982			;ERROR 226	
1983	007774	047124	EM12	;NO RDY
1984	007776	052475	DH26	;AFTER READ DATA CMD
1985	010000	053676	DT1	
1986	010002	054250	DF10	
1987			;ERROR 227	
1988	010004	047520	EM21	;CERR SET
1989	010006	052475	DH26	
1990	010010	053770	DT5	
1991	010012	054314	DF15	
1992			;ERROR 230	
1993	010014	050107	EM62	;DTE SET
1994	010016	052475	DH26	
1995	010020	053770	DT5	
1996	010022	054314	DF15	
1997			;ERROR 231	
1998	010024	050127	EM63	;DLT SET
1999	010026	052475	DH26	
2000	010030	053770	DT5	
2001	010032	054314	DF15	
2002			;ERROR 232	
2003	010034	000000	0	
2004	010036	000000	0	
2005	010040	000000	0	
2006	010042	000000	0	
2007			;ERROR 233	
2008	010044	047351	EM16	;CANNOT READ BSE INFO
2009	010046	053105	DH42	;ON SECT 0,2,4,6,8
2010	010050	053676	DT1	
2011	010052	054330	DF17	
2012			;ERROR 234	
2013	010054	047351	EM16	
2014	010056	053156	DH43	;ON SECT 1,3,5,7,9
2015	010060	053676	DT1	
2016	010062	054330	DF17	
2017			;ERROR 235	
2018	010064	050226	EM69	;ALIGN CARTRIDGE USED
2019	010066	053226	DH44	;WILL BYPASS FORMAT & ALL R/W TESTS
2020	010070	053676	DT1	
2021	010072	054250	DF10	
2022			;ERROR 236	
2023	010074	000000	0	
2024	010076	000000	0	
2025	010100	000000	0	
2026	010102	000000	0	
2027			;ERROR 237	
2028	010104	000000	0	
2029	010106	000000	0	
2030	010110	000000	0	
2031	010112	000000	0	
2032			;ERROR 240	

2033 010114 000000
 2034 010116 000000
 2035 010120 000000
 2036 010122 000000
 2037
 2038 010124 000000
 2039 010126 000000
 2040 010130 000000
 2041 010132 000000
 2042
 2043 010134 000000
 2044 010136 000000
 2045 010140 000000
 2046 010142 000000
 2047
 2048
 2049 010144 047575
 2050 010146 052452
 2051 010150 054056
 2052 010152 054174
 2053
 2054 010154 050257
 2055 010156 053060
 2056 010160 053676
 2057 010162 054250
 2058
 2059 010164 000000
 2060 010166 000000
 2061 010170 000000
 2062 010172 000000
 2063
 2064 010174 000000
 2065 010176 000000
 2066 010200 000000
 2067 010202 000000
 2068
 2069 010204 000000
 2070 010206 000000
 2071 010210 000000
 2072 010212 000000
 2073
 2074 010214 000000
 2075 010216 000000
 2076 010220 000000
 2077 010222 000000
 2078
 2079 010224 000000
 2080 010226 000000
 2081 010230 000000
 2082 010232 000000
 2083
 2084 010234 000000
 2085 010236 000000
 2086 010240 000000
 2087 010242 000000
 2088

0
 0
 0
 0
 ;ERROR 241
 0
 0
 0
 0
 ;ERROR 242
 0
 0
 0
 0
 ;ERROR 243
 EM36
 DH25
 DT8
 DF6
 ;ERR 244
 EM74
 DH41
 DT1
 DF10
 ;ERR 245
 0
 0
 0
 0
 ;ERR 246
 0
 0
 0
 0
 ;ERR 247
 0
 0
 0
 0
 ;ERR 250
 0
 0
 0
 0
 ;ERR 251
 0
 0
 0
 0
 ;ERR 252
 0
 0
 0
 0
 ;ERR 253

;CYL ADDR IN RKMR3 INCORRECT
 ;AFTER SEEK CMD

 ;RTZ NOT SET
 ;DURING RECAL CMD

2089	010244	000000	0	
2090	010246	000000	0	
2091	010250	000000	0	
2092	010252	000000	0	
2093			0	
2094	010254	000000	0	
2095	010256	000000	0	
2096	010260	000000	0	
2097	010262	000000	0	
2098			0	
2099	010264	000000	0	
2100	010266	000000	0	
2101	010270	000000	0	
2102	010272	000000	0	
2103			0	
2104	010274	000000	0	
2105	010276	000000	0	
2106	010300	000000	0	
2107	010302	000000	0	
2108			0	
2109	010304	000000	0	
2110	010306	000000	0	
2111	010310	000000	0	
2112	010312	000000	0	
2113			0	
2114	010314	047414	EM17	;MSG A0 ERROR
2115	010316	052425	DH24	;AFTER OFFSET CMD
2116	010320	053676	DT1	
2117	010322	054210	DF8	
2118			0	
2119	010324	047435	EM18	;MSG B0 ERROR
2120	010326	052425	DH24	
2121	010330	053676	DT1	
2122	010332	054230	DF9	
2123			0	
2124	010334	000000	0	
2125	010336	000000	0	
2126	010340	000000	0	
2127	010342	000000	0	
2128			0	
2129	010344	000000	0	
2130	010346	000000	0	
2131	010350	000000	0	
2132	010352	000000	0	
2133			0	
2134	010354	000000	0	
2135	010356	000000	0	
2136	010360	000000	0	
2137	010362	000000	0	
2138			0	
2139	010364	047435	EM18	;MSG B0 ERROR
2140	010366	052373	DH22	;AFTER DRIVE CLEAR CMD
2141	010370	053676	DT1	
2142	010372	054230	DF9	
2143			0	
2144	010374	047477	EM20	;MSG B1 ERROR

2145	010376	052373	DH22	
2146	010400	053676	DT1	
2147	010402	054230	DF9	
2148				;ERR 267
2149	010404	047435	EM18	;MSG B0 ERROR
2150	010406	052750	DH39	;AFTER WRITE HEADER CMD
2151	010410	053676	DT1	
2152	010412	054230	DF9	
2153				;ERR 270
2154	010414	047477	EM20	;MSG B1 ERROR
2155	010416	052750	DH39	
2156	010420	053676	DT1	
2157	010422	054230	DF9	
2158				;ERR 271
2159	010424	000000	0	
2160	010426	000000	0	
2161	010430	000000	0	
2162	010432	000000	0	
2163				;ERR 272
2164	010434	000000	0	
2165	010436	000000	0	
2166	010440	000000	0	
2167	010442	000000	0	
2168				;ERR 273
2169	010444	047414	EM17	;MSG A0 ERROR
2170	010446	052373	DH22	;AFTER DRV CLR CMD
2171	010450	053676	DT1	
2172	010452	054210	DF8	
2173				;ERR 274
2174	010454	047456	EM19	;MSG A1 ERROR
2175	010456	052373	DH22	
2176	010460	053676	DT1	
2177	010462	054210	DF8	
2178				;ERR 275
2179	010464	047435	EM18	;MSG B0 ERROR
2180	010466	052216	DH17	;AFTER RECAL CMD
2181	010470	053676	DT1	
2182	010472	054230	DF9	
2183				;ERR 276
2184	010474	047477	EM20	;MSG B1 ERROR
2185	010476	052216	DH17	
2186	010500	053676	DT1	
2187	010502	054230	DF9	
2188				;ERR 277
2189	010504	047414	EM17	;MSG A0 ERROR
2190	010506	052750	DH39	;AFTER WRITE HEADER CMD
2191	010510	053676	DT1	
2192	010512	054210	DF8	
2193				;ERR 300
2194	010514	047456	EM19	;MSG A1 ERROR
2195	010516	052750	DH39	
2196	010520	053676	DT1	
2197	010522	054210	DF8	
2198				;ERR 301
2199	010524	000000	0	
2200	010526	000000	0	

2201	010530	000000	
2202	010532	000000	
2203			:ERR 302
2204	010534	000000	
2205	010536	000000	
2206	010540	000000	
2207	010542	000000	
2208			:ERR 303
2209	010544	000000	
2210	010546	000000	
2211	010550	000000	
2212	010552	000000	
2213			:ERR 304
2214	010554	000000	
2215	010556	000000	
2216	010560	000000	
2217	010562	000000	
2218			



```

2219
2220 .SBTTL PROGRAM SETUP
2221
2222 010564 012737 000001 001356 PARSRT: MOV #1,PARAM ;SET FLAG FOR 220 START
2223 010572 005037 001360 CLR BYPWRT
2224 010576 005037 001362 CLR BYPTIM
2225 010602 000450 BR PRGSRT ;START PROGRAM
2226
2227 010604 105037 001356 BYWRT: CLRB PARAM
2228 010610 012737 000001 001360 MOV #1,BYPWRT ;BYPASS W .E TESTS
2229 010616 105037 001362 CLRB BYPTIM
2230 010622 000440 BR PRGSRT
2231
2232 010624 105037 001356 BYTIM: CLRB PARAM
2233 010630 105037 001360 CLRB BYPWRT
2234 010634 012737 000001 001362 MOV #1,BYPTIM ;BYPASS TIMING TESTS
2235 010642 000430 BR PRGSRT
2236
2237 010644 012737 000001 001356 BYWRTA: MOV #1,PARAM
2238 010652 012737 000001 001360 MOV #1,BYPWRT
2239 010660 105037 001362 CLRB BYPTIM
2240 010664 000417 BR PRGSRT
2241
2242 010666 012737 000001 001356 BYTIMA: MOV #1,PARAM
2243 010674 105037 001360 CLRB BYPWRT
2244 010700 012737 000001 001362 MOV #1,BYPTIM
2245 010706 000406 BR PRGSRT
2246
2247 010710 105037 001356 START: CLRB PARAM ;CLEAR FOR 200 START
2248 010714 005037 001360 CLR BYPWRT
2249 010720 005037 001362 CLR BYPTIM
2250 010724 000005 PRGSRT: RESET ;CLEAR ALL INT ENABLE & INIT
2251 010726 012706 001100 MOV #STACK,SP ;SETUP STACK POINTER
2252 010732 012746 000340 MOV #PR7,-(SP) ;PSW LOADED TO BE
2253 010736 012746 010744 MOV #15,-(SP) ;LSI-11 COMPATABLE
2254 010742 000002 RTI ;LOCKOUT ALL INTERRUPTS
2255
2256 010744
2257 IS:
2258 .SBTTL INITIALIZE THE COMMON TAGS
2259 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2260 010744 012706 001100 MOV #CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2261 010750 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
2262 010752 022706 001140 CMP #SWR,R6 ;;DONE?
2263 010756 001374 BNE -6 ;;LOOP BACK IF NO
2264 010760 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER
2265 ;;INITIALIZE A FEW VECTORS
2265 010764 012737 036346 000020 MOV #SCOPE,#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2266 010772 012737 000340 000022 MOV #340,#IOTVEC+2 ;;LEVEL 7
2267 011000 012737 036626 000030 MOV #ERROR,#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2268 011006 012737 000340 000032 MOV #340,#EMTVEC+2 ;;LEVEL 7
2269 011014 012737 042122 000034 MOV #TRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2270 011022 012737 000340 000036 MOV #340,#TRAPVEC+2 ;;LEVEL 7
2271 011030 012737 036102 000024 MOV #SPWRDN,#PWRVEC ;;POWER FAILURE VECTOR
2272 011036 012737 000340 000026 MOV #340,#PWRVEC+2 ;;LEVEL 7
2273 011044 013737 031516 031510 MOV SENDCT,SEOPCT ;;SETUP END-OF-PROGRAM COUNTER
2274 011052 005037 001174 CLR STIMES ;;INITIALIZE NUMBER OF ITERATIONS

```

```

2275 011056 005037 001176 CLR $ESCAPE ;: CLEAR THE ESCAPE ON ERROR ADDRESS
2276 011062 112737 000001 001115 MOVB #1,$ERMAX ;: ALLOW ONE ERROR PER TEST
2277 011070 012737 011070 001106 MOV #,$SLPADR ;: INITIALIZE THE LOOP ADDRESS FOR SCOPE
2278 011076 012737 011076 001110 MOV #,$SLPERR ;: SETUP THE ERROR LOOP ADDRESS
2279 ;: SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2280 ;: EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
2281 011104 013746 000004 MOV #ERRVEC, -(SP) ;: SAVE ERROR VECTOR
2282 011110 012737 011144 000004 MOV #64$, #ERRVEC ;: SET UP ERROR VECTOR
2283 011116 012737 177570 001140 MOV #DSWR, SWR ;: SETUP FOR A HARDWARE SWICH REGISTER
2284 011124 012737 177570 001142 MOV #DDISP, DISPLAY ;: AND A HARDWARE DISPLAY REGISTER
2285 011132 022777 177777 170000 CMP # -1, #SWR ;: TRY TO REFERENCE HARDWARE SWR
2286 011140 001012 BNE 66$ ;: BRANCH IF NO TIMEOUT TRAP OCCURRED
2287 ;: AND THE HARDWARE SWR IS NOT = -1
2288 011142 000403 BR 65$ ;: BRANCH IF NO TIMEOUT
2289 011144 012716 011152 64$: MOV #65$, (SP) ;: SET UP FOR TRAP RETURN
2290 011150 000002 RTI
2291 011152 012737 000176 001140 65$: MOV #SWREG, SWR ;: POINT TO SOFTWARE SWR
2292 011160 012737 000174 001142 MOV #DISPREG, DISPLAY
2293 011166 012637 000004 66$: MOV (SP)+, #ERRVEC ;: RESTORE ERROR VECTOR
2294
2295 011172 005037 001216 CLR $PASS ;: CLEAR PASS COUNT
2296 011176 132737 000200 001231 BITB #APT$SIZE, $ENVH ;: TEST USER SIZE UNDER APT
2297 011204 001403 BEQ 67$ ;: YES, USE NON-APT SWIT
2298 011206 012737 001232 001140 MOV #SSWREG, SWR ;: NO, USE APT SWITCH REGISTER
2299 011214
2300 011214 012737 011254 000004 67$: MEMPAR: MOV #1$, ERRVEC ;: SETUP TIMEOUT VECTOR
2301 011222 012737 000340 000006 MOV #PR7, ERRVEC+2
2302
2303 011230 012737 000001 172100 MOV #1, MEMBAS ;: LOAD REG TO DET IF
2304 ;: MEM CHECK ENABLE AVAIL
2305 011236 012737 036004 000114 MOV #MEMERR, MEMVEC ;: LOAD VECTOR IF NO TIMEOUT
2306 011244 012737 000340 000116 MOV #PR7, MEMVEC+2
2307 011252 000401 BR 2$
2308
2309 011254 022626 1$: CMP (SP)+, (SP)+ ;: ADJ STACK
2310 011256 012737 000006 000004 2$: MOV #ERRVEC+2, ERRVEC ;: RESTRE TRAP CATCHER
2311 011264 005037 000006 CLR ERRVEC+2
2312
2313 011270 004737 031604 JSR PC, CLRFLG ;: CLEAR DDUMP THRU SIZFLG
2314 011274 005037 001220 CLR $DEVCT
2315 011300 005037 001222 CLR $UNIT
2316
2317 ;:
2318 ;: FIND OUT IF XXDP, ACT, APT; CHAIN OR DUMP MODE
2319 ;:
2320
2321 011304 005737 000042 START1: TST 42
2322 011310 001014 BNE 1$ ;: BR IF AUTO
2323 011312 004737 031624 JSR PC, TITLE ;: MANUAL TYPE PROG ID
2324 011316 123727 000041 000013 CMPB 41, #13 ;: 13=LOADED BY XXDP
2325 011324 001010 BNE 2$
2326 011326 005237 005462 INC DDUMP ;: SET RMO6 DUMP MODE FLAG
2327 011332 104401 043062 TYPE MSG2 ;: REPLACE DRO PACK W/SCRATCH & DO<CR>
2328 011336 000137 011352 JMP $T2
2329 011342 000137 011416 1$: JMP $T3
2330 011346 005237 005470 2$: INC PPTP ;: SET ACT/APT/PTP DUMP MODE FLAG

```

```

2331
2332
2333      ; CHECK IF ALL PARAMETERS DEFAULTED. IF NOT, BEGIN INPUT DIALOGUE
2334      ; WITH OPERATOR. THE REPLY TO 'DRIVES TO BE TESTED' SHOULD BE
2335      ; DRIVE NOS. SEPERATED BY COMMAS & TERMINATED BY <CR>
2336      ; EX:      DRIVES TO BE TESTED: 1,2,4<CR>
2337
2338
2339      011352 005737 001356      ST2:   TST      PARAM
2340      011356 001002              BNE      1$      ; BR IF 220 START
2341      011360 000137 011450      JMP      ST4     ; 200 START, DEFAULT & SIZE THE BUSS
2342      011364 104401 043133      ;:     TYPE    MSG3  ; DRIVES TO BE TESTED
2343      011370 004737 031644      JSR     PC_GDRVS ; GET DR NOS.
2344      011374 104401 043165      TYPE    MSG4     ; BUSS ADDR
2345      011400 004737 032004      JSR     PC_GBA   ; GET BA
2346      011404 104401 043230      TYPE    MSG5     ; CONT INT VECTOR
2347      011410 004737 032032      JSR     PC_GINT  ; GET INT VECTOR
2348      011414 000427      BR      ST5
2349
2350
2351      ; AUTO MODE
2352      ; CHECK IF LOADED BY XXDP OR OTHER. SET FLAGS & NO INPUT DIALOGUE.
2353      ; DEFAULT ALL PARAMETERS. TEST ONLY THOSE DRIVES THAT ARE READY
2354      ; ON THE BUSS
2355
2356
2357      011416 123727 000041 000013 ST3:   CMPB    41, #13 ; 13=LOADED BY XXDP
2358      011424 001007              BNE      1$
2359      011426 005237 005464              INC     DDPCH   ; SET RK06 CHAIN MODE FLAG
2360      011432 004737 031624      JSR     PC_TITLE
2361      011436 104401 043343      TYPE    MSG7
2362      011442 000402              BR      ST4
2363      011444 005237 005466      1$:    INC     ACT11 ; SET ACT AUTO FLAG.
2364
2365      011450 012737 177440 001264 ST4:   MOV     #177440, $BASE ; DEFAULT VALUE
2366      011456 012737 000210 001334 MOV     #210, RKVEC  ; DEFAULT VALUE
2367      011464 004737 032064      JSR     PC_SETINT
2368      011470 005237 005522      INC     SIZEFLG  ; DO "SIZE THE BUSS" TEST
2369
2370
2371      011474 012706 001100      ST5:   MOV     #STACK, SP ; INIT STACK
2372      011500 012746 000340      MOV     #PR7, -(SP) ; PSW LOADED TO BE
2373      011504 012746 011512      MOV     #5$, -(SP) ; LSI-11 COMPATABLE
2374      011510 000002              RTI
2375      011512 012737 005474 001366 5$:   MOV     #DRIVO, DRVPTR ; LOCKOUT ALL INTERRUPTS
2376      011520 005037 001220      CLR     $DEVCT   ; SETUP
2377      011524 005037 001222      CLR     $UNIT    ; NO. OF DRVS DONE
2378      011530 012737 011576 000004 MOV     #1$, ERRVEC ; CURRENT DRV UNDER TEST
2379      011536 005777 167604      TST     $LK$     ; SETUP TIMEOUT ERROR VECTOR
2380      011542 005237 005514      INC     LCLKF    ; SEE IF L-CLOCK THERE
2381      011546 013700 001350      MOV     LCVEC, RO ; PRESENT, SET FLAG.
2382      011552 012737 011540 000004 MOV     #2$, ERRVEC ; VECTOR ADDR
2383      011560 005777 167554      TST     $PK$     ; SEE IF P-CLOCK THERE
2384      011564 005237 005516      INC     PCLKF    ; PRESENT, SET FLAG
2385      011570 013700 001352      MOV     PCVEC, RO ; VECTOR ADDR
2386      011574 000412      BR
  
```

```

2387 011576 022626          15:  CMP      (SP)+,(SP)+      ;L-CLOCK NOT THERE, CLEAR STACK
2388 011600 012737 011644 000004  MOV      #45,ERR,EC
2389 011606 005777 167526      TST      @PKS           ;SEE IF P-CLOCK THERE
2390 011612 005237 005516      INC      PCLKF         ;PRESENT, SET FLAG
2391 011616 013700 001352      MOV      PCVEC,RO      ;VECTOR ADDR
2392 011622 005237 005520      35:  INC      DOTIM        ;INDICATES TIMING TESTS CAN BE DONE
2393 011626 012720 035254      MOV      @CLOCK,(RO)+ ;SERVICE ROUTINE FOR CLOCKS
2394 011632 012710 00034C      MOV      @PR7,(RO)
2395 011636 0004J7          BR       TST1          ;;GO TO NEXT TEST
2396
2397 011640 022626          25:  CMP      (SP)+,(SP)+      ;P-CLOCK NOT THERE, CLEAR STACK
2398 011642 000767          BR
2399
2400 011644 022626          45:  CMP      (SP)+,(SP)+      ;NEITHER CLOCK THERE, CLEAR STACK
2401 011646 005237 005520      CLR      DOTIM        ;TIMING TESTS CANNOT BE DONE.
2402 011652 104401 043546      TYPE    ,MSG13       ;ALL TIMING TESTS BYPASSED
2403
2404
  
```

.SBTTL BASIC CONTROLLER TESTS, SIZING & SETUP

*TEST 1 REFERENCE ALL CONTROLLER REGISTERS

* THIS TEST VERIFIES THAT ALL THE CONTROLLER REGISTERS
* CAN BE ACCESSED. THE INABILITY TO BE ACCESSED WILL
* RESULT IN A TIMEOUT TRAP WITH AN ERROR MESSAGE. ANY
* ERROR IN THIS TEST WILL RESULT IN ABORTING ALL OTHER
* TESTS AND JUMPING TO 'END OF PASS'

2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000

```
TEST1: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR

MOV #IS,ERRVEC ;SETUP TIMEOUT ERROR VECTOR
MOV $BASE,R5 ;SETUP INDEX REG.
TST RKCS1(R5) ;REFERENCE ALL THE
TST RKCS2(R5) ;CONTROLLER REGISTERS
TST RKWC(R5)
TST RKBA(R5)
TST RKDA(R5)
TST RKDS(R5) ;TIMEOUTS IN THIS SECTION
TST RKER(R5) ;INDICATE THAT THE CONTROLLER
TST RKASOF(R5) ;REGISTERS CANNOT BE READ.
TST RKDC(R5) ;TESTING SHOULD NOT PROCEED
TST RKDB(R5) ;UNTIL THIS IS REMEDIED.
TST RKMR1(R5)
TST RKMR2(R5)
TST RKMR3(R5)
TST RKECPS(R5)
TST RKECPT(R5)

MOV #BADTMO,ERRVEC ;SETUP TIMEOUT HANDLER
BR TST2 ;GO TO NEXT TEST

IS: CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
ERROR 7 ;ABORT-COULD NOT REFERENCE CONTROLLER REGISTER
JMP $EOP1
```

*TEST 2 SIZE THE BUSS

* THIS TEST IS ENTERED ONLY IF 'DRIVE SELECTION' IS DEFAULTED
* EITHER BY RUNNING IN THE AUTO MODE OR A 200 START IN THE
* MANUAL MODE.
* EVERY DRIVE FROM 0 THRU 7 IS ADDRESSED.
* CONTROLLER ERROR (CERR) IS EXAMINED AND IF NOT SET, THE
* DRIVE WILL BE TESTED. IF SET, THE PROGRAM WILL BYPASS
* TESTING THAT DRIVE ONLY IF THE ERROR WAS A RESULT OF
* MDS, UFE OR NED BEING SET; OR BOTH NED & DRA RESET IN-
* DICATING THE OTHER PORT IS ACCESSED.

F06

UNIBUS RKO6 DRIVE DIAGNOSTIC PART 2
 DZR6IB.P11 T2 SIZE THE BUSS

MACY11 27(732) 01-OCT-76 10:38 PAGE 49

SEQ 0050

```

2511 012120 000004
2512 012122 012737 000001 001174
2513 012130 012706 001100
2514 012034 012765 000040 000010
2515 012042 013737 001432 005450
2516 012050 004737 032102
2517 012054 104120
2518 012056 005737 005522
2519 012062 001562
2520 012064 104401 043452
2521 012070 005037 005472
2522 012074 005000
2523 012076 012701 005474
2524 012102
2525 012102 104415
2526 012104 012706 001100
2527 012110 012765 000040 000010
2528 012116 013737 001432 005450
2529 012124 004737 032102
2530 012130 104120
2531 012132 010065 000010
2532 012136 012765 000001 000000
2533 012144 013737 001432 005450
2534 012152 004737 032102
2535 012156 104117
2536 012160 032737 100000 005406
2537 012166 001046
2538 012170 013737 005434 005450
2539 012176 042737 177770 005450
2540 012204 020037 005450
2541 012210 001016
2542 012212 005700
2543 012214 001003
2544 012216 005737 005464
2545 012222 001014
2546 012224 005237 005472
2547 012230 005211
2548 012232 104401 001205
2549 012236 010046
2550 012240 104403
2551 012242 001
2552 012243 000
2553 012244 000403
2554 012246 004737 032512
2555 012252 104001
2556 012254 005721
2557 012256 005200
2558 012260 022700 000010
2559 012264 001306

```

```

: *
: *****
1$: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR
MOV #SCLR,RKCS2(R5) ;SUBSYSTEM CLEAR
MOV T10,TEMP1 ;SET TIMEOUT
JSR PC,FRDY ;FIND RDY
ERROR 120 ;RDY NOT SET BY END OF SCLR
TST SIZEFLG
BEQ TST3 ;DO NOT SIZE, GOTO NEXT TEST
TYPE MSG10 ;WILL TEST DRIVES
CLR DRIVS ;# OF DRIVES PRESENT
CLR RO ;DRV ADDR
MOV #DRIVO,R1 ;DRV FLAG
1$: SCOP1
MOV #STACK,SP ;RESTORE STK PTR
MOV #SCLR,RKCS2(R5) ;SUBSYSTEM CLEAR
MOV T10,TEMP1 ;SET TIMEOUT
JSR PC,FRDY ;FIND RDY
ERROR 120 ;RDY NOT SET BY END OF SCLR
MOV RO,RKCS2(R5) ;SELECT THE DRIVE ADDR
MOV #SELDRV,RKCS1(R5) ;SELECT DRIVE CMD
MOV T10,TEMP1
JSR PC,FRDY ;FIND RDY
ERROR 117 ;NO RDY AFTER SELECT DRIVE CMD.
BIT #CERR,HCS1
BNE 2$
MOV HMR2,TEMP1
BIC #1<DRVMSK>,TEMP1
CMP RO,TEMP1 ;S/B SAME
BNE 3$
TST RO
BNE 4$
TST DDPCH ;SEE IF XXDP CHAIN MODE
BNE 5$
4$: INC DRIVS ;INC DRIVE COUNT.
INC (R1) ;SET DRIVE PRESENT FLAG
TYPE SCRLF
MOV RO,-(SP) ;SAVE RO FOR TYPEOUT
;TYPE DR #
;GO TYPE--OCTAL ASCII
;TYPE 1 DIGIT(S)
;SUPPRESS LEADING ZEROS
5$: JSR PC,BYP ;TYPE BYPASS DR #
ERROR 1 ;WRITTEN DR # DOES NOT MATCH RKMR2 DR #
5$: TST (R1)+ ;SHIFT PTR TO NEXT DR. FLAG
INC RO ;INC DR #
CMP #8.,RO
BNE 1$ ;MORE LEFT.

```



```

2517 012266 005737 005472 TST DRIVS
2518 012272 001054 BNE 10$
2519 012274 104401 043537 TYPE ,MSG12 ;NONE
2520 012300 000137 031462 JMP $EOP1
2521
2522 012304 032737 001000 005410 2$: BIT #MDS,HCS2
2523 012312 001015 BNE 6$
2524 012314 032737 000400 005410 BIT #UFE,HCS2
2525 012322 001015 BNE 7$
2526 012324 032737 000001 005420 BIT #DRA,HDS
2527 012332 001015 BNE 8$
2528 012334 032737 010000 005410 BIT #NED,HCS2
2529 012342 001424 BEQ 9$
2530 012344 000743 BR 5$
2531
2532 012346 004737 032512 6$: JSR PC,BYP ;TYPE BYP DR #
2533 012352 104002 ERROR 2 ;MDS DETECTED
2534 012354 000737 BR 5$
2535
2536 012356 004737 032512 7$: JSR PC,BYP
2537 012362 104003 ERROR 3 ;UFE DETECTED
2538 012364 000733 BR 5$
2539
2540 012366 032737 010000 005410 8$: BIT #NED,HCS2
2541 012374 001713 BEQ 4$
2542 012376 104401 043657 TYPE ,MSG15 ;DRV#
2543 012402 010046 MOV RO,-(SP) ;;SAVE RO FOR TYPEOUT
2544 ;;TYPE DR#
2545 012404 104403 TYPOS ;;GO TYPE--OCTAL ASCII
2546 012406 001 .BYTE 1 ;;TYPE 1 DIGIT(S)
2547 012407 000 .BYTE 0 ;;SUPPRESS LEADING ZEROS
2548 012410 104010 ERROR 10 ;;DRA & NED BOTH SET
2549 012412 000720 BR 5$
2550
2551 012414 004737 032512 9$: JSR PC,BYP
2552 012420 104004 ERROR 4 ;NO DRA & NO NED = OTHER PORT SELECTED
2553 012422 000714 BR 5$
2554 012424 000137 012754 10$: JMP NUDRV

```

*TEST 3 VERIFY OPERATOR DRIVE SELECTIONS

```

*
* THIS TEST IS ENTERED ONLY IF DRIVE SELECTION IS NOT
* DEFAULTED. EVERY DRIVE FROM 0 TO 7 IS ADDRESSED &
* CONTROLLER ERROR (CERR) IS EXAMINED. IF NOT SET, THE
* PROGRAM WILL ASSUME THE DRIVE IS PRESENT. IT WILL THEN CHECK
* TO SEE THAT THE DRIVE WAS INPUTTED FOR TESTING. IF NOT, IT WILL
* BE AN ERROR. IF CERR WAS SET, THAT DRIVE WILL BE BYPASSED
* ONLY IF THE ERROR WAS A RESULT OF MDS OR UFE SET OR BOTH
* NED & DRA RESET (WRONG PORT). IF CERR IS A RESULT OF
* NED ONLY, IT IS CHECKED AGAINST THE INPUTTED INFOR TO
* VERIFY IT WAS NOT SPECIFIED.

```

```

2571 012430 000004 TST3: SCOPE
2572 012432 012737 000001 001174 MOV #1,$TIMES ;;DO 1 ITERATION

```

H06

2573	012440	012706	001100		MOV	#STACK,SP	;RESTORE STK PTR
2574	012444	005000			CLR	RO	;DRIVE ADDR
2575	012446	012701	005474		MOV	#DRIVO,R1	;DRIVE FLAG
2576	012452			1\$:			
2577	012452	104415			SCOP1		
2578	012454	012706	001100		MOV	#STACK,SP	;RESTORE STK PTR
2579							
2580	012460	012765	000040	000010	MOV	#SCLR,RKCS2(R5)	;SUBSYSTEM CLEAR
2581	012466	013737	001432	005450	MOV	T10,TEMP1	;SET TIME OUT
2582	012474	004737	032102		JSR	PC,FRDY	;FIND RDY
2583	012500	104120			ERROR	120	;NO RDY AFTER SCLR
2584	012502	010065	000010		MOV	RO,RKCS2(R5)	;DRV ADDR
2585	012506	012765	000001	000000	MOV	#SELDRV,RKCS1(R5)	;SELECT DRIVE CMD
2586	012514	013737	001432	005450	MOV	T10,TEMP1	
2587	012522	004737	032102		JSR	PC,FRDY	;FIND RDY
2588	012526	104117			ERROR	117	;NO RDY AFTER SELECT DRIVE CMD.
2589	012530	032737	100000	005406	BIT	#CERR,HCS1	
2590	012536	001036			BNE	2\$	
2591	012540	013737	005434	005450	MOV	HMR2,TEMP1	
2592	012546	042737	177770	005450	BIC	#C<DRVMSK>,TEMP1	
2593	012554	020037	005450		CMP	RO,TEMP1	;S/B SAME
2594	012560	001010			BNE	3\$	
2595	012562	005711		11\$:	TST	(R1)	
2596	012564	001417			BEQ	5\$	
2597	012566	005721		4\$:	TST	(R1)+	;SHIFT PTR TO NEXT DR FLAG
2598	012570	005200			INC	RO	;INC DR#
2599	012572	022700	000010		CMP	#8.,RO	
2600	012576	001325			BNE	1\$;MORE LEFT
2601	012600	000465			BR	TST4	;GO TO NEXT TEST
2602							
2603	012602	004737	032512	3\$:	JSR	PC,BYP	;TRY BYPASS DRIVE#
2604	012606	104001			ERROR	1	;WRITTEN DR# DOES NOT MATCH RKMR2 DR#
2605	012610	005711			TST	(R1)	
2606	012612	001765			BEQ	4\$;BRANCH IF NOT SPEC BY INPUT
2607	012614	005337	005472	12\$:	DEC	DRIVS	;DECREMENT TOTAL DRIVS
2608	012620	005011			CLR	(R1)	;CLEAR DRIVE FLAG
2609	012622	000761			BR	4\$	
2610							
2611	012624	004737	032512	5\$:	JSR	PC,BYP	
2612	012630	104005			ERROR	5	;DR PRESENT BUT NOT SPECIFIED BY OPERATOR
2613	012632	000755			BR	4\$	
2614							
2615	012634	032737	001000	005410	2\$:	BIT	#MDS,HCS2
2616	012642	001027			BNE	6\$	
2617	012644	032737	000400	005410	BIT	#UFE,HCS2	
2618	012652	001027			BNE	7\$	
2619	012654	032737	000001	005420	BIT	#DRA,HDS	
2620	012662	001005			BNE	8\$	
2621	012664	032737	010000	005410	BIT	#NED,HCS2	
2622	012672	001423			BEQ	9\$	
2623	012674	000404			BR	10\$	
2624	012676	032737	010000	005410	8\$:	BIT	#NED,HCS2
2625	012704	001726			BEQ	11\$	
2626	012706	005711		10\$:	TST	(R1)	
2627	012710	001726			BEQ	4\$	
2628							

```

2629 012712 004737 032512 JSR PC,BYP ;TYPE BYPASS DRIVE#
2630 012716 104006 ERROR 6
2631 012720 000735 BR 12$
2632
2633 012722 004737 032512 6$: JSR PC,BYP ;TYPE BYPASS DRIVE#
2634 012726 104002 ERROR 2 ;MDS DETECTED
2635 012730 000762 BR 8$
2636
2637 012732 004737 032512 7$: JSR PC,BYP
2638 012736 104003 ERROR 3 ;UFE DETECTED
2639 012740 000756 BR 8$
2640
2641 012742 004737 032512 9$: JSR PC,BYP
2642 012746 104004 ERROR 4 ;DRA & NED RESET - OTHER PORT SELECTED
2643 012750 000752 BR 8$
2644
2645 012752 001237 BNE 1$ ;BRANCH IF MORE LEFT.
2646
2647
2648 ;
2649 ; THIS PART OF THE PROGRAM WILL BE REPEATED FOR EACH
2650 ; DRIVE PRESENT
2651 ;
2652 ; 'SUNIT' CONTAINS THE ADDRESS OF THE DRIVE CURRENTLY
2653 ; UNDER TEST
2654
2655 012754 NUDRV: ;ENTER HERE FROM LAST TEST
2656
2657 ;*****
2658 ;TEST 4 FIND NEXT DRIVE TO BE TESTED
2659 ;
2660 ; THIS TEST FINDS THE NEXT DRIVE PRESENT & PUTS THAT
2661 ; ADDRESS IN 'SUNIT'.
2662 ; THROUGHOUT THE FOLLOWING TESTS, THE DRIVE TESTED IS
2663 ; THE DRIVE WHOSE ADDRESS IS IN 'SUNIT'.
2664 ;*****
2665 ;*****
2666 012754 000004 TST4: SCOPE
2667 012756 012737 000001 001174 MOV #1,$TIMES ;:DO 1 ITERATION
2668 012764 012706 001100 MOV #STACK,$P ;:RESTORE STK PTR
2669 012770 012737 000004 001214 MOV #STN-1,$TESTN
2670 012776 012737 000004 001102 MOV #STN-1,$TSTNM
2671
2672 013004 005737 005472 TST DRVS ;:ANY DRIVES PRESENT?
2673 013010 001004 BNE 4$ ;:YES BRANCH
2674 013012 104401 044572 TYPE MSG27 ;:ALL DRIVES TESTED
2675 013016 000137 031462 JMP $EOPI ;:NO, GO TO END
2676
2677 013022 013701 001366 4$: MOV DRVPTR,R1 ;:ADDR OF NEXT DRIVE FLAG
2678 013026 005737 001220 TST $DEVCT ;:IS FIRST DRIVE BEING CHECKED
2679 013032 001402 BEQ 2$ ;:YES, BRANCH
2680 013034 005237 001222 1$: INC SUNIT ;:INCR DRIVE ADDR TO NEXT DRIVE
2681 013040 005721 2$: TST (R1)+ ;:IS DRIVE PRESENT?
2682 013042 001774 BEQ 1$ ;:NO, FIND NEXT DRIVE PRESENT
2683 013044 005737 005464 TST DDPCH ;:DDP CHAIN MODE?
2684 013050 001403 BEQ 3$ ;:NO, BRANCH

```

```

2685 013052 005737 001222          TST    $UNIT          ;YES, IS IT DRIVE 0?
2686 013056 001766                    BEQ    1$              ;IF YES, DON'T TEST DR 0
2687 013060 010137 001366          3$:   MOV    R1,DRVPTR ;STORE POINTER TO THE NEXT DR. FLAG
2688 013064 104401 043657          TYPE   MSG15         ;"DRIVE"
2689 013070 013700 001222          MOV    $UNIT,R0
2690 013074 010046          MOV    R0,-($P)      ;;SAVE R0 FOR TYPECUT
2691                    ;DRIVE #
2692 013076 104403          TYPOS                    ;GO TYPE--OCTAL ASCII
2693 013100 001          .BYTE 1              ;TYPE 1 DIGIT(S)
2694 013101 000          .BYTE 0              ;SUPPRESS LEADING ZEROS
2695
2696 013102 104401 001205          TYPE   ,SCLRF
2697
2698 013106          PFSRT:                ;ENTER HERE FOR POWER FAIL RESTART
2699                    ;*****
2700                    ;*TEST 5          PRINT DRIVE SERIAL NUMBER
2701                    ;*
2702                    ;*          THIS TEST READS & PRINTS THE DRIVE SERIAL # FROM MSG A, WORD 11
2703                    ;*          IN BCD & IS PERFORMED ON THE 1ST PASS ONLY
2704                    ;*
2705                    ;*****
2706 013106 000004          TSTS:  SCOPE
2707 013110 012737 000001 001174    MOV    #1,$TIMES      ;;DO 1 ITERATION
2708 013116 012706 001100          MOV    #STACK,$P     ;RESTORE STK PTR
2709
2710 013122 005737 001216          TST    $PASS
2711 013126 001046          BNE    TST6           ;;GO TO NEXT IF NOT FIRST PASS
2712 013130 004737 032570          JSR    PC,SUBCLR     ;DO SUBSYS CLEAR
2713 013134 104024          ERROR  24            ;CERR AFTER SCLR
2714
2715 013136 104401 043671          TYPE   MSG16         ;DRIVE SERIAL NO.
2716 013142 012765 000003 000026    MOV    #3,RKMR1(R5)  ;SELECT BYTE 3
2717 013150 004737 032526          JSR    PC,GSTAT      ;GET STATUS
2718 013154 013701 005434          MOV    HMR2,R1       ;GET SERIAL #
2719 013160 012704 041406          MOV    #SOCTVL,R4    ;GET ADDR CHAR BUFF
2720 013164 010446          MOV    R4,-($P)     ;STORE ON STACK FOR $SUPRS
2721 013166 012703 000003          MOV    #3,R3         ;SETUP CHAR COOUNT
2722 013172 006101          ROL    R1            ;INITIALIZE BIT POSITIONS
2723 013174 006101          ROL    R1
2724 013176 006101          1$:   ROL    R1         ;GET NEXT 4 BITS
2725 013200 006101          ROL    R1
2726 013202 006101          ROL    R1
2727 013204 006101          ROL    R1
2728 013206 010100          MOV    R1,R0         ;GET WORKING COPY
2729 013210 042700 177760          BIC    #177760,R0    ;CLEAR ALL BUT LOW 4 BITS
2730 013214 052700 000060          BIS    #60,R0        ;CONVERT TO ASCII DIGIT
2731 013220 110024          MOV    R0,(R4)+     ;PUT ASCII DIGIT INTO CHAR BUFF
2732 013222 005303          DEC    R3
2733 013224 001364          BNE    1$           ;BR IF ALL 3 CHARS NOT DONE
2734 013226 105014          CLRB   (R4)         ;ELSE INSERT NULL TERMINATOR
2735
2736 013230 004737 041654          JSR    PC,$SUPRS    ;TYPE
2737 013234 104401 001205          TYPE   ,SCLRF
2738 013240 104401 001205          TYPE   ,SCLRF
2739
2740                    ;*****

```

2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796

```

:*TEST 6          SET VV WITH PACK COMMAND
:*
:*          IF VV IS RESET, THE PACK COMMAND IS USED TO SET IT.
:*
:*****
TST6:  SCOPE
      MOV      #1,STIMES          ;;DO 1 ITERATION
      MOV      #STACK,SP          ;RESTORE STK PTR
      JSR      PC,SUBCLR
      ERROR    24                  ;CERR AFTER SCLR
      BIT      #D.VV,HMR2
      BNE     TST7                ;;GO TO NEXT TEST IF VV SET
      SCOP1
      MOV      #STACK,SP          ;RESTORE STK PTR
      JSR      PC,SUBCLR
      ERROR    24                  ;CERR AFTER SCLR
      MOV      #PACK,RKCS1(R5)    ;CMD TO SET VV
      MOV      #10,TEMP1
      JSR      PC,FRDY
      ERROR    116                ;RDY NOT SET AFTER PACK CMD
      BIT      #D.VV,HMR2
      BNE     TST7                ;;GO TO NEXT TEST IF VV NOW SET
      ERROR    27                  ;PACK DID NOT SET V.V.

```

```

:*****
:*TEST 7          READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL #
:*
:*          THIS TEST VERIFIES THAT CYL 410, TRACK 2 CAN BE READ.
:*          THIS AREA CONTAINS BAD SECTOR INFO WHICH IS WRITTEN BY THE
:*          FACTORY DURING MANF. ALL BAD SECTOR INFO (BSE) WILL BE STORED
:*          AT THIS TIME TO MASK FUTURE READ HEADER OR DATA ERROR PRINTOUTS.
:*          IF BSE INFO CANNOT BE READ, OR IF AFTER READING THE BSE INFO
:*          IT IS DETERMINED THAT AN ALIGNMENT CARTRIDGE IS USED,
:*          A MESSAGE WILL BE TYPED INDICATING THAT ALL
:*          FUTURE FORMAT AND READ-WRITE TESTS WILL BE BYPASSED.
:*          THIS IS DONE SO AS NOT TO DESTROY BSE INFO OR AN ALIGNMENT PACK BY WRITING
:*
:*          THE PACK SERIAL # IS TYPED IN OCTAL & FOR THE FIRST PASS ONLY.

```

```

:*****
TST7:  SCOPE
      MOV      #1,STIMES          ;;DO 1 ITERATION
      MOV      #STACK,SP          ;RESTORE STK PTR
      JSR      PC,SUBCLR
      ERROR    24                  ;CERR AFTER SCLR
      CLR      TEMP2
      CLR      TEMP3
      ;SECTOR CTR
      ;0=22 SEC, 1=20 SEC, 2=DONE

```

```

2797 013400 005037 001544 CLR BSERR ;BSE INFO NO GOOD IF SET
2798
2799 013404 012765 003364 000004 MOV #BSE22,RKBA(R5) ;BSE TABLE FOR 22 SECTOR FORMAT
2800 013412 012765 001000 000006 MOV #1000,RKDA(R5) ;HEAD 2, SECTOR 0
2801 013420 012765 000632 000020 MOV #410.,RKDC(R5) ;CYL 410
2802
2803 013426 012765 177400 000002 1$: MOV #-256.,RKWC(R5) ;LOAD WORD CT
2804 013434 012765 000021 000000 MOV #RDATA,RKCS1(R5) ;READ DATA COMMAND
2805 013442 013737 001444 005450 MOV T50000,TEMP1 ;SETUP TIMEOUT
2806 013450 004737 032102 JSR PC,FRDY ;FIND RDY
2807 013454 104226 ERROR 226 ;NO RDY AFTER READ DATA CMD
2808 013456 004737 032526 JSR PC,GSTAT ;GET FRESH DATA
2809 013462 032737 100000 005406 BIT #CERR,HCS1
2810 013470 001416 BEQ 3$
2811 013472 104227 ERROR 227 ;CERR AFTER READ DATA CMD
2812 013474 005237 001544 INC BSERR ;SET BSE ERROR FLAG
2813 013500 032737 010000 005422 BIT #DTE,HER
2814 013506 001401 BEQ 2$
2815 013510 104230 ERROR 230 ;DTE AFTER READ DATA CMD
2816 013512 032737 100000 005410 2$: BIT #DLT,HCS2
2817 013520 001404 BEQ 4$
2818 013522 104231 ERROR 231 ;DLT AFTER READ DATA CMD
2819 013524 000402 BR 4$
2820
2821 013526 005037 001544 3$: CLR BSERR ;BSE INFO OK
2822 013532 005737 001544 4$: TST BSERR ;BSE READ OK?
2823 013536 001420 BEQ 8$ ;BR IF YES
2824 013540 062765 000002 000006 ADD #2,RKDA(R5) ;TRY NEXT SECTOR
2825 013546 005237 005452 INC TEMP2
2826 013552 023727 005452 000005 CMP TEMP2,#5 ;READ ALL 5 SECTORS?
2827 013560 001322 BNE 1$ ;BR IF NO
2828 013562 005737 005454 TST TEMP3
2829 013566 001002 BNE 6$
2830 013570 104233 ERROR 233 ;CANT READ BSE ON SECTORS 0,2,4,6,8
2831 013572 000452 BR TST10 ;GO TO NEXT TEST
2832 013574 104234 6$: ERROR 234 ;CANT READ BSE ON SECTORS 1,3,5,7,9
2833 013576 000450 BR TST10 ;GO TO NEXT TEST
2834
2835 013600 012700 000006 8$: MOV #6,RO ;SETUP FOR WORD 3 OF BSE INFO
2836 013604 016037 003364 005456 MOV BSE22(RO),TEMP4 ;PULL OUT CARTRIDGE TYPE INFO.
2837 013612 001404 BEQ 9$ ;BRANCH IF DATA CARTRIDGE
2838 013614 104235 ERROR 235 ;ALIGNMENT CARTRIDGE USED
2839 013616 005237 001544 INC BSERR ;SET BSE ERROR FLAG
2840 013622 000417 BR 10$
2841
2842 013624 005237 005454 9$: INC TEMP3
2843 013630 023727 005454 000002 CMP TEMP3,#2
2844 013636 001411 BEQ 10$
2845
2846 013640 005037 005452 CLR TEMP2
2847 013644 012765 002364 000004 MOV #BSE20,RKBA(R5) ;BSE TABLE FOR 20 SECTOR FORMAT
2848 013652 012765 001001 000006 MOV #1001,RKDA(R5) ;HEAD 2, SECTOR 1
2849 013660 000662 BR 1$
2850
2851 013662 005737 001216 10$: TST $PASS
2852 013666 001014 BNE TST10 ;GO TO NEXT TST IF NOT 1'ST PASS
  
```

2853	013670	104401	043715	TYPE	MSG17	;CART SERIAL #
2854	013674	012746	003364	MOV	BASE22, -(SP)	
2855	013700	004737	041304	JSR	PC, \$DB20	;CONVERT DBL BINARY WORD TO OCTAL
2856	013704	004737	041654	JSR	PC, \$SUPRS	;TYPE SERIAL #
2857	013710	104401	001205	TYPE	, \$CRLF	
2858	013714	104401	001205	TYPE	, \$CRLF	

.SBTTL WRITE TESTS

TEST 10 BASIC WRITE DATA TEST; 1 WORD

THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE JUST ONE WORD,
ALL SECTORS ON CYL 0 ARE GIVEN IDENTICAL HEADERS &
A WRITE COMMAND IS ISSUED. READ & WRITE CHECK COMMANDS ARE NOT
PERFORMED. THIS TEST PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP
FOR A WRITE ERROR.

2874	013720	000004		TST10:	SCOPE	
2875	013722	012737	000001	MOV	#1, \$TIMES	;DO 1 ITERATION
2876	013730	012706	001100	MOV	#STACK, SP	;RESTORE STK PTR
2878	013734	005737	001360	TST	BYPWRT	
2879	013740	001404		BEG	DOWRT	
2880	013742	104401	044040	TYPE	MSG19	;BYPASSING WRITE PTESTS
2881	013746	000137	025264	JMP	TIMING	
2882	013752			DOWRT:		
2884	013752	005737	001544	TST	BSERR	;SEE IF ALIGN CART
2885	013756	001406		BEG	25	;BR IF NO
2886	013760	104401	045676	TYPE	, MSG40	;BSE OR ALIGN CART USED
2887	013764	104401	044510	TYPE	, MSG26	;ABORTING DATA TESTS
2888	013770	000137	025264	JMP	TIMING	
2890	013774	004737	032570	25:	JSR	PC, SUBCLR
2891	014000	104024		ERROR	24	;CERR AFTER SCLR
2893	014002	012700	001550	MOV	#HDTAB, RO	;MAKE ALL CYL 0 HEADERS IDENTICAL
2895	014006	005020		15:	CLR	(RO)+
2896	014010	012720	140000	MOV	#140000, (RO)+	;HEADER WORD 1: SECTOR 0
2897	014014	012720	140000	MOV	#140000, (RO)+	;HEADER WORD 2: XOR OF 1 & 2
2899	014020	020027	001754	CMP	RO, #HDTAB+132.	;ALL HEADERS DONE? (22X6=132)
2900	014024	001370		BNE	15	;BR IF NO
2902	014026	012765	001550	MOV	#HDTAB, RKBA(R5)	;HEADER TABLE
2903	014034	012765	177676	MOV	#-56., RKWC(R5)	;WORD COUNT
2905	014042	012765	000027	MOV	#(WRHEAD), RKCS1(R5)	;WRITE HEADER CMD
2906	014050	013737	001444	MOV	T5000, TEMP1	;SETUP TIMEOUT
2907	014056	004737	032102	JSR	PC, FRDY	;FIND RDY
2908	014062	104200		ERROR	200	;NO RDY AFTER WRITE HEADER CMD

N06

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 57
 DZR618.P11 T10 BASIC WRITE DATA TEST; 1 WORD

SEG 0058

```

2909 014064 004737 032526 JSR PC,GSTAT ;GET FRESH STATUS
2910 014070 032737 100000 005406 BIT #CERR,HCS1
2911 014076 001405 BEQ 64$
2912 014100 104201 ERROR 201 ;CERR AFTER WRITE HEADER CMD
2913 014102 104401 044510 TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS
2914 014106 000137 025264 JMP TIMING
2915 014112 64$:
2916
2917
2918 014112 104415 SCOP1
2919 014114 012706 001100 MOV #STACK,SP ;RESTORE STK PTR
2920
2921 014120 004737 032570 JSR PC,SUBCLR
2922 014124 104024 ERROR 24 ;CERR AFTER SCLR
2923
2924 014126 012765 001532 000004 MOV #DATA1,RKBA(R5) ;DATA TO BE ALL 1'S
2925 014134 012765 177777 000002 MOV #-1,RKWC(R5) ;WORD COUNT=1
2926
2927
2928 014142 012765 000023 000000 MOV #<WRDATA>,RKCS1(R5) ;WRITE DATA CMD
2929 014150 013737 001444 005450 MOV T5000,TEMP1 ;SETUP TIMEOUT
2930 014156 004737 032102 JSR PC,FRDY ;FIND RDY
2931 014162 104011 ERROR 11 ;NO RDY AFTER WRITE DATA CMD
2932 014164 004737 032526 JSR PC,GSTAT ;GET FRESH STATUS
2933 014170 032737 100000 005406 BIT #CERR,HCS1
2934 014176 001405 BEQ 65$
2935 014200 104012 ERROR 12 ;CERR AFTER WRITE DATA CMD
2936 014202 104401 044510 TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING
2937 014206 000137 025264 JMP TIMING
2938
2939 014212 65$:
2940
2941 014212 012765 100000 000000 MOV #CLR,RKCS1(R5) ;CONTR CLEAR
2942 014220 004737 032526 JSR PC,GSTAT ;GET LATEST STATUS
2943 014224 012737 010340 005444 MOV #<D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
2944 014232 004737 035344 JSR PC,CKMR2 ;CHECK MR2
2945 014236 104052 ERROR 52 ;MSG A0 ERROR AFTER WRITE DATA CMD
2946 014240 005037 005446 CLR SBMR3
2947 014244 004737 035426 JSR PC,CKMR3 ;CHECK MR3
2948 014250 104023 ERROR 23 ;MSG B0 ERROR AFTER WRITE DATA CMD
2949
2950 014252 012765 100000 000000 MOV #CLR,RKCS1(R5)
2951 014260 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
2952 014266 004737 032526 JSR PC,GSTAT
2953 014272 012737 001720 005444 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2
2954 014300 004737 035344 JSR PC,CKMR2
2955 014304 104053 ERROR 53 ;MSG A1 ERROR AFTER WRITE DATA CMD
2956 014306 005037 005446 CLR SBMR3
2957 014312 004737 035426 JSR PC,CKMR3
2958 014316 104025 ERROR 25 ;MSG B1 ERROR AFTER WRITE DATA CMD
2959
2960
2961 ;*****
2962 ;*TEST 11 BASIC WRITE DATA TEST; FULL SECTOR
2963 ;*
2964 ;* THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE

```



```

2965
2966
2967
2968
2969
2970
2971 014320 000004
2972 014322 012737 000001 001174
2973 014330 012706 001100
2974 014334 004737 032570
2975 014340 104024
2976
2977 014342 012765 001550 000004
2978 014350 012765 177676 000002
2979 014356 005037 201372
2980
2981 014362 013737 001372 001406
2982 014370 012737 000000 001514
2983 014376 012737 000000 001520
2984 014404 004737 033356
2985
2986
2987 014410 012765 000027 000000
2988 014416 013737 001444 005450
2989 014424 004737 032102
2990 014430 104200
2991 014432 004737 032526
2992 014436 032737 100000 005406
2993 014444 001405
2994 014446 104201
2995 014450 104401 044510
2996 014454 000137 025264
2997 014460
2998
2999
3000 014460 104415
3001 014462 012706 001100
3002
3003 014466 004737 032570
3004 014472 104024
3005
3006 014474 012765 001526 000004
3007 014502 013700 001526
3008 014506 052765 000020 000010
3009 014514 012765 177400 000002
3010
3011 014522 012765 000023 000000
3012 014530 013737 001444 005450
3013 014536 004737 032102
3014 014542 104011
3015 014544 004737 032526
3016 014550 032737 100000 005406
3017 014556 001405
3018 014560 104012
3019 014562 104401 044510
3020 014566 000137 025264

```

```

:* A FULL SECTOR. ALL ZEROS ARE WRITTEN BY THE WRITE DATA COMMAND
:* & CHECKED BY A RD DATA COMMAND. A FURTHER CHECK IS PERFORMED
:* BY THE WRT CHK COMMAND.
:* THE ABOVE IS REPEATED FOR AN ALL ONES PATTERN.

```

```

*****

```

```

↑ST11: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR
JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR
MOV #HDTAB,RKBA(R5) ;RESTORE TO 22 SECTOR
MOV #-66.,RKWC(R5) ;STANDARD FORMAT
CLR TOCYL
MOV TOCYL,CALADD ;SETUP
MOV #0,HEAD ;TO FILL
MOV #0,FORMAT ;HEADER
JSR PC,FHDTAB ;TABLE
MOV #(<WRHEAD>,RKCS1(R5)) ;WRITE HEADER CMD
MOV T50000,TEMP1 ;SETUP TIMEOUT
JSR PC,FRDY ;FIND ROY
ERROR 200 ;NO ROY AFTER WRITE HEADER CMD
JSR PC,GSTAT ;GET FRESH STATUS
BIT #CERR,HCS1
BEQ 64$
ERROR 201 ;CERR AFTER WRITE HEADER CMD
TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS
JMP TIMING
64$:
SCOPE1
MOV #STACK,SP ;RESTORE STK PTR
JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR
MOV #DATA0,RKBA(R5) ;WRITE ALL 0'S
MOV DATA0,R0
BIS #BA1,RKCS2(R5)
MOV #-256.,RKWC(R5)
1$:
MOV #(<WRDATA>,RKCS1(R5)) ;WRITE DATA CMD
MOV T50000,TEMP1 ;SETUP TIMEOUT
JSR PC,FRDY ;FIND ROY
ERROR 11 ;NO ROY AFTER WRITE DATA CMD
JSR PC,GSTAT ;GET FRESH STATUS
BIT #CERR,HCS1
BEQ 65$
ERROR 12 ;CERR AFTER WRITE DATA CMD
TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING
JMP TIMING

```

```

3021
3022 014572          65$:
3023
3024 014572 012765 100000 000000      MOV      #CCLR,RKCS1(R5) ;CONTR CLEAR
3025 014600 004737 032526          JSR      PC,GSTAT      ;GET LATEST STATUS
3026 014604 012737 010340 005444      MOV      #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
3027 014612 004737 035344          JSR      PC,CKMR2      ;CHECK MR2
3028 014616 104052          ERROR    52            ;MSG A0 ERROR AFTER WRITE DATA CMD
3029 014620 005037 005446          CLR      SBMR3
3030 014624 004737 035426          JSR      PC,CKMR3      ;CHECK MR3
3031 014630 104023          ERROR    23            ;MSG B0 ERROR AFTER WRITE DATA CMD
3032
3033 014632 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
3034 014640 012765 000001 000026      MOV      #1,RKMR1(R5) ;SELECT WORD 1
3035 014646 004737 032526          JSR      PC,GSTAT
3036 014652 012737 001720 005444      MOV      #<0.SPOK!D.CART!D.DOOR!D.BRMH!D.TFOK>,SBMR2
3037 014660 004737 035344          JSR      PC,CKMR2
3038 014664 104053          ERROR    53            ;MSG A1 ERROR AFTER WRITE DATA CMD
3039 014666 005037 005446          CLR      SBMR3
3040 014672 004737 035426          JSR      PC,CKMR3
3041 014676 104025          ERROR    25            ;MSG B1 ERROR AFTER WRITE DATA CMD
3042
3043 014700 104415          SCOP1
3044 014702 012706 001100          MOV      #STAC,SP      ;RESTORE STK PTR
3045
3046 014706 004737 032570          JSR      PC,SUBCLR
3047 014712 104024          ERROR    24            ;CERR AFTER SCLR
3048
3049 014714 012765 004364 000004      MOV      #ROTAB,RKBA(R5)
3050 014722 012765 177400 000002      MOV      #-256.,RKWC(R5)
3051
3052
3053 014730 012765 000021 000000      MOV      #<RDATA>,RKCS1(R5) ;READ DATA CMD
3054 014736 013737 001444 005450      MOV      T5000,TEMP1 ;SETUP TIMEOUT
3055 014744 004737 032102          JSR      PC,FRDY      ;FIND RDY
3056 014750 104013          ERROR    13            ;NO RDY AFTER READ DATA CMD
3057 014752 004737 032526          JSR      PC,GSTAT      ;GET FRESH STATUS
3058 014756 032737 100000 005422      BIT      #DCK,MR
3059 014764 001405          BEQ     66$
3060 014766 104021          ERROR    21            ;DATA CHECK ERROR AFTER READ DATA CMD
3061                          ;ECC ERROR DETECTED
3062 014770 104401 044510          TYPE    MSG26          ;ABORTING DATA TESTS TO DO TIMING
3063 014774 000137 025264          JMP     TIMING
3064
3065 015000 032737 100000 005406 66$:      BIT      #CERR,HCS1
3066 015006 001405          BEQ     67$
3067 015010 104014          ERROR    14            ;CERR AFTER READ DATA CMD
3068 015012 104401 044510          TYPE    MSG26          ;ABORTING DATA TESTS TO DO TIMING
3069 015016 000137 025264          JMP     TIMING
3070
3071 015022          67$:
3072
3073 015022 012765 100000 000000      MOV      #CCLR,RKCS1(R5) ;CONTR CLEAR
3074 015030 004737 032526          JSR      PC,GSTAT      ;GET LATEST STATUS
3075 015034 012737 010340 005444      MOV      #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
3076 015042 004737 035344          JSR      PC,CKMR2      ;CHECK MR2

```

3077	015046	104054			ERROR	54		;MSG A0 ERROR AFTER READ DATA CMD
3078	015050	005037	005446		CLR	SBMR3		
3079	015054	004737	035426		JSR	PC,CKMR3		;CHECK MR3
3080	015060	104026			ERROR	26		;MSG B0 ERROR AFTER READ DATA CMD
3081								
3082	015062	012765	100000	000000	MOV	#CCLR,RKCS1(R5)		
3083	015070	012765	000001	000026	MOV	#1,RKMR1(R5)		;SELECT WORD 1
3084	015076	004737	032526		JSR	PC,GSTAT		
3085	015102	012737	001720	005444	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK),SBMR2		
3086	015110	004737	035344		JSR	PC,CKMR2		
3087	015114	104056			ERROR	56		;MSG A1 ERROR AFTER READ DATA CMD
3088	015116	005037	005446		CLR	SBMR3		
3089	015122	004737	035426		JSR	PC,CKMR3		
3090	015126	104030			ERROR	30		;MSG B1 ERROR AFTER READ DATA CMD
3091								
3092								
3093								
3094	015130	012701	004364		MOV	#RDTAB,R1		
3095	015134	011137	001506		MOV	(R1),WD1		;ACTUAL WORD FOR TYPEOUT
3096	015140	010037	001510		MOV	RO,WD2		;EXPECTED DATA FOR TYPEOUT
3097								
3098	015144	020021			CMP	RO,(R1)+		;COMPARE READ DATA TABLE AGAINST
3099	015146	001403			BEQ	3\$;THE 'SHOULD BE' VALUE
3100	015150	004737	033666		JSR	PC,TRUERR		
3101	015154	104020			ERROR	20		;READ DATA DID NOT COMPARE WITH WRITE DATA
3102								
3103	015156	020127	005364		CMP	R1,#RDTAB+512.		
3104	015162	001364			BNE	2\$		
3105								
3106	015164	020037	001526		CMP	RO,DATA0		
3107	015170	001401			BEQ	4\$		
3108	015172	000407			BR	5\$		
3109								
3110	015174	012765	001532	000004	MOV	#DATA1,RKBA(R5)		;WRITE ALL 1'S
3111	015202	013700	001532		MOV	DATA1,RO		
3112	015206	000137	014506		JMP	1\$		
3113								
3114	015212							
3115	015212	104415			SCOP1			
3116	015214	012706	001100		MOV	#STACK,SP		;RESTORE STK PTR
3117								
3118	015220	004737	032570		JSR	PC,SUBCLR		
3119	015224	104024			ERROR	24		;CERR AFTER SCLR
3120								
3121	015226	052765	000020	000010	BIS	#BA1,RKCS2(R5)		;THIS PORTION OF THE TEST CHECKS
3122	015234	012765	001532	000004	MOV	#DATA1,RKBA(R5)		;OUT THE WRITE CHECK CMD
3123	015242	012765	177400	000002	MOV	#-256.,RKWC(R5)		;ALL 1'S WERE PREVIOUSLY WRITTEN
3124								
3125								
3126	015250	012765	000031	000000	MOV	#(WRTCHK),RKCS1(R5)		;WRITE CHECK CMD
3127	015256	013737	001444	005450	MOV	T5000,TEMP1		;SETUP TIMEOUT
3128	015264	004737	032102		JSR	PC,FRDY		;FIND RDY
3129	015270	104015			ERROR	15		;NO RDY AFTER WRITE CHECK CMD
3130								
3131	015272	004737	032526		JSR	PC,GSTAT		;GET FRESH STATUS
3132	015276	032737	100000	005406	BIT	#CERR,HCS1		

E07

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 61
 DZR6IB.P11 T11 BASIC WRITE DATA TEST; FULL SECTOR

SEQ 0062

```

3133 015304 001417          BEQ      69$
3134
3135 015306 032737 040000 005410      BIT      #WCE,HCS2      ;SEE IF WRITE CHECK ERROR
3136 015314 001412          BEQ      68$
3137 015316 016537 000024 001506      MOV      RKDB(R5),WD1 ;ACTUAL WORD FOR TYPEOUT
3138 015324 013737 001532 001510      MOV      DATA1,WD2  ;EXPECTED WORD FOR TYPEOUT
3139 015332 004737 033666          JSR      PC,TRUERR   ;CHECK AGAINST BSE INFO
3140 015336 104016          ERROR   16          ;WCE AFTER WRITE CMD
3141 015340 000401          BR      69$
3142
3143 015342 104022          68$:  ERROR   22          ;CERR AFTER WRITE CHK CMD
3144
3145 015344          69$:
3146
3147 015344 012765 100000 000000      MOV      #CCLR,RKCS1(R5) ;CONTR CLEAR
3148 015352 004737 032526          JSR      PC,GSTAT   ;GET LATEST STATUS
3149 015356 012737 010340 005444      MOV      #<D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
3150 015364 004737 035344          JSR      PC,CKMR2   ;CHECK MR2
3151 015370 104057          ERROR   57          ;MSG A0 ERROR AFTER WRITE CHECK CMD
3152 015372 005037 005446          CLR      SBMR3
3153 015376 004737 035426          JSR      PC,CKMR3   ;CHECK MR3
3154 015402 104031          ERROR   31          ;MSG B0 ERROR AFTER WRITE CHECK CMD
3155
3156 015404 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
3157 015412 012765 000001 000026      MOV      #1,RKMR1(R5) ;SELECT WORD 1
3158 015420 004737 032526          JSR      PC,GSTAT
3159 015424 012737 001720 005444      MOV      #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2
3160 015432 004737 035344          JSR      PC,CKMR2
3161 015436 104060          ERROR   60          ;MSG A1 ERROR AFTER WRITE CHECK CMD
3162 015440 005037 005446          CLR      SBMR3
3163 015444 004737 035426          JSR      PC,CKMR3
3164 015450 104032          ERROR   32          ;MSG B1 ERROR AFTER WRITE CHECK CMD
3165
3166
3167
3168 015452 104415          SCOP1
3169 015454 012706 001100      MOV      #STACK,SP   ;RESTORE STK PTR
3170
3171 015460 004737 032570          JSR      PC,SUBCLR
3172 015464 104024          ERROR   24          ;CERR AFTER SCLR
3173
3174 015466 012765 001526 000004      MOV      #DATA0,RKBA(R5) ;SETUP TO CHECK AGAINST WRONG DATA
3175 015474 012765 177400 000002      MOV      #-256.,RKWC(R5)
3176
3177 015502 012765 000031 000000      MOV      #WRTCHK,RKCS1(R5)
3178 015510 012737 141520 005450      MOV      #50000.,TEMP1
3179 015516 004737 032102          JSR      PC,FROY
3180 015522 104015          ERROR   15          ;NO RDY AFTER WRITE CHECK CMD
3181 015524 004737 032526          JSR      PC,GSTAT   ;GET FRESH STATUS
3182 015530 032737 040000 005410      BIT      #WCE,HCS2   ;EXPECT MISCOMPARE
3183 015536 001001          BNE     6$
3184 015540 104017          ERROR   17          ;WRITE CHECK CMD NOT FUNCTIONING
3185
3186 015542          6$:
3187
3188 015542 012765 100000 000000      MOV      #CCLR,RKCS1(R5) ;CONTR CLEAR

```

F07

```

3189 015550 004737 032526 JSR PC,GSTAT ;GET LATEST STATUS
3190 015554 012737 010340 005444 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
3191 015562 004737 035344 JSR PC,CKMR2 ;CHECK MR2
3192 015565 104057 ERROR 57 ;MSG A0 ERROR AFTER WRITE CHK CMD WITH INTENTIONAL ERROR
3193 015570 005037 005446 CLR SBMR3
3194 015574 004737 035426 JSR PC,CKMR3 ;CHECK MR3
3195 015600 104031 ERROR 31 ;MSG B0 ERROR AFTER WRITE CHK CMD WITH INTENTIONAL ERROR
3196
3197 015602 012765 100000 000000 MOV #CLR,RKCSI(R5)
3198 015610 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
3199 015616 004737 032526 JSR PC,GSTAT
3200 015622 012737 001720 005444 MOV #<0.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2
3201 015630 004737 035344 JSR PC,CKMR2
3202 015634 104060 ERROR 60 ;MSG A1 ERROR AFTER WRITE CHK CMD WITH INTENTIONAL ERROR
3203 015636 005037 005446 CLR SBMR3
3204 015642 004737 035426 JSR PC,CKMR3
3205 015646 104032 ERROR 32 ;MSG B1 ERROR AFTER WRITE CHK CMD WITH INTENTIONAL ERROR
3206
3207
3208
3209
3210
3211
3212
3213
3214

```

```

*****
*TEST 12 20 SECTOR FORMAT TEST
*
* DATA IS WRITTEN ON A FULL TRACK IN 20 SECTOR FORMAT.
* MSG B0,B1 ARE CHECKED FOR ANY ERROR CONDITION
*
*****

```

```

3215 015650 000004 000001 001174 *TEST 12: SCOPE
3216 015652 012737 000001 001174 MOV #1,STIMES ;DO 1 ITERATION
3217 015660 012706 001100 MOV #STACK,SP ;RESTORE STK PTR
3218 015664 004737 032570 JSR PC,SUBCLR
3219 015670 104024 ERROR 24 ;CERR AFTER SCLR
3220 015672 012765 001550 000004 MOV #HDTAB,RK9A(R5) ;HEADER WORD TABLE
3221 015700 012765 177704 000002 MOV #-60.,RKWC(R5) ;WORD COUNT FOR 20 SECTOR FMT
3222 015706 005037 001372 CLR TOCYL
3223
3224
3225 015712 013737 001372 001406 MOV TOCYL,CALADD ;SETUP
3226 015720 012737 000000 001514 MOV #0,HEAD ;TO FILL
3227 015726 012737 000001 001520 MOV #1,FORMAT ;HEADER
3228 015734 004737 033356 JSR PC,FHDTAB ;TABLE
3229
3230
3231 015740 012765 013027 000000 MOV #<CFMT!WRHEAD>,RKCSI(R5) ;WRITE HEADER CMD
3232 015746 013737 001444 005450 MOV T5000,TEMP1 ;SETUP TIMEOUT
3233 015754 004737 032102 JSR PC,FRDY ;FIND RDY
3234 015760 104200 ERROR 200 ;NO RDY AFTER WRITE HEADER CMD
3235 015762 004737 032526 JSR PC,GSTAT ;GET FRESH STATUS
3236 015766 032737 100000 005406 BIT #CERR,HCSI
3237 015774 001405 BEQ 645
3238 015776 104201 ERROR 201 ;CERR AFTER WRITE HEADER CMD
3239 016000 104401 044510 TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS
3240 016004 000137 025264 JMP TIMING
3241 016010
3242
3243 016010 104415 SDCPI
3244 016012 012706 001100 MOV #STACK,SP ;RESTORE STK PTR

```

645:

```

3245
3246 016016 004737 032570 JSR PC,SUBCLR
3247 016022 104024 ERROR 24 ;CERR AFTER SCLR
3248
3249 016024 012765 001532 000004 MOV #DATA1,RKBA(R5) ;WRITE ALL 1'S
3250 016032 052765 000020 000010 BIS #BAI,RKCS2(R5) ;BUSS ADDR INCR INHIBIT
3251 016040 012765 166000 000002 MOV #-20.*256.,RKWC(R5) ;DO FULL TRACK.
3252
3253
3254 016046 012765 010023 000000 MOV #<CFMT!WRDATA>,RKCS1(R5) ;WRITE DATA CMD
3255 016054 013737 001444 005450 MOV T5000,TEMP1 ;SETUP TIMEOUT
3256 016062 004737 032102 JSR PC,FRDY ;FIND RDY
3257 016066 104011 ERROR 11 ;NO RDY AFTER WRITE DATA CMD
3258 016070 004737 032526 JSR PC,GSTAT ;GET FRESH STATUS
3259 016074 032737 100000 005406 BIT #CERR,HCS1
3260 016102 001405 BEQ 65$
3261 016104 104012 ERROR 12 ;CERR AFTER WRITE DATA CMD
3262 016106 104401 TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING
3263 016112 000137 025264 JMP TIMING
3264
3265 016116 65$:
3266 016116 012765 010001 000000 MOV #<CFMT!SELDRV>,RKCS1(R5)
3267 016124 013737 001432 005450 MOV T10,TEMP1
3268 016132 004737 032102 JSR PC,FRDY ;FIND RDY
3269 016136 104117 ERROR 117 ;RDY NOT FOUND AFTER SELDRV CMD
3270 016140 032737 001000 005434 BIT #D.FORM,HMR2
3271 016146 001001 BNE 1$
3272 016150 104102 ERROR 102 ;FORMAT NOT SET
3273
3274 016152 1$:
3275
3276 016152 012765 100000 000000 MOV #CCLR,RKCS1(R5) ;CONTR CLEAR
3277 016160 004737 032526 JSR PC,GSTAT ;GET LATEST STATUS
3278 016164 012737 010340 005444 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
3279 016172 004737 035344 JSR PC,CKMR2 ;CHECK MR2
3280 016176 104052 ERROR 52 ;MSG A0 ERROR AFTER WRITE DATA CMD
3281 016200 005037 005446 CLR SBMR3
3282 016204 004737 035426 JSR PC,CKMR3 ;CHECK MR3
3283 016210 104023 ERROR 23 ;MSG B0 ERROR AFTER WRITE DATA CMD
3284
3285 016212 012765 100000 000000 MOV #CCLR,RKCS1(R5)
3286 016220 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
3287 016226 004737 032526 JSR PC,GSTAT
3288 016232 012737 001720 005444 MOV #<0.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2
3289 016240 004737 035344 JSR PC,CKMR2
3290 016244 104053 ERROR 53 ;MSG A1 ERROR AFTER WRITE DATA CMD
3291 016246 005037 005446 CLR SBMR3
3292 016252 004737 035426 JSR PC,CKMR3
3293 016256 104025 ERROR 25 ;MSG B1 ERROR AFTER WRITE DATA CMD
3294
3295 016260 012765 166000 000002 MOV #-20.*256.,RKWC(R5)
3296 016266 012765 001532 000004 MOV #DATA1,RKBA(R5)
3297 016274 052765 000020 000010 BIS #BAI,RKCS2(R5)
3298
3299 016302 012765 010031 000000 MOV #<CFMT!WRTCHK>,RKCS1(R5) ;WRITE CHECK CMD
3300 016310 013737 001444 005450 MOV T5000,TEMP1 ;SETUP TIMEOUT

```

H07

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 64
 DZR6IB.P11 T12 20 SECTOR FORMAT TEST

SEQ 0065

3301	016316	004737	032102		JSR	PC,FRDY	;FIND RDY
3302	016322	104015			ERROR	15	;NO RDY AFTER WRITE CHECK CMD
3303							
3304	016324	004737	032526		JSR	PC,GSTAT	;GET FRESH STATUS
3305	016330	032737	100000	005406	BIT	#CERR,HCS1	
3306	016336	001417			BEQ	67\$	
3307							
3308	016340	032737	040000	005410	BIT	#WCE,HCS2	;SEE IF WRITE CHECK ERROR
3309	016346	001412			BEQ	66\$	
3310	016350	016537	000024	001506	MOV	RKDB(R5),WD1	;ACTUAL WORD FOR TYPEOUT
3311	016356	013737	001532	001510	MOV	DATA1,WD2	;EXPECTED WORD FOR TYPEOUT
3312	016364	004737	033666		JSR	PC,TRUERR	;CHECK AGAINST BSE INFO
3313	016370	104016			ERROR	16	;WCE AFTER WRITE CMD
3314	016372	000401			BR	67\$	
3315							
3316	016374	104022			66\$:	ERROR	22 ;CERR AFTER WRITE CHK CMD
3317							
3318	016376				67\$:		
3319							
3320	016376	012765	010001	000000	MOV	#<CFMT!SELDRV>,RKCS1(R5)	
3321	016404	013737	001432	005450	MOV	1:7 TEMP1	
3322	016412	004737	032102		JSR	PC,FRDY	;FIND RDY
3323	016416	104117			ERROR	117	;NO RDY AFTER SELDRV CMD
3324	016420	032737	001000	005434	BIT	#D.FORM,HMR2	
3325	016426	001001			BNE	2\$	
3326	016430	104103			ERROR	103	;FORMAT NOT SET
3327							
3328	016432				2\$:		
3329							
3330	016432	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	;CONTR CLEAR
3331	016440	004737	032526		JSR	PC,GSTAT	;GET LATEST STATUS
3332	016444	012737	010340	005444	MOV	#<D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2	;SHOULD BE VALUE
3333	016452	004737	035344		JSR	PC,CKMR2	;CHECK MR2
3334	016456	104057			ERROR	57	;MSG A0 ERROR AFTER WRITE CHECK CMD
3335	016460	005037	005446		CLR	SBMR3	
3336	016464	004737	035426		JSR	PC,CKMR3	;CHECK MR3
3337	016470	104031			ERROR	31	;MSG B0 ERROR AFTER WRITE CHECK CMD
3338							
3339	016472	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3340	016500	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
3341	016506	004737	032526		JSR	PC,GSTAT	
3342	016512	012737	001720	005444	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2	
3343	016520	004737	035344		JSR	PC,CKMR2	
3344	016524	104060			ERROR	60	;MSG A1 ERROR AFTER WRITE CHECK CMD
3345	016526	005037	005446		CLR	SBMR3	
3346	016532	004737	035426		JSR	PC,CKMR3	
3347	016536	104032			ERROR	32	;MSG B1 ERROR AFTER WRITE CHECK CMD
3348							
3349	016540	104415			SCOP1		
3350	016542	012706	001100		MOV	#STACK,SP	;RESTORE STK PTR
3351							
3352	016546	004737	032570		JSR	PC,SUBCLR	
3353	016552	104024			ERROR	24	;CERR AFTER SCLR
3354							
3355	016554	012765	001550	000004	MOV	#HDTAB,RKBA(R5)	;RESTORE CYL 0 TO 22 SECTOR FMT
3356	016562	012765	177676	000002	MOV	#-66.,RKWC(R5)	

```

3357 016570 005037 001372 CLR TOCYL
3358
3359
3360 016574 013737 001372 001406 MOV TOCYL,CALADD ;SETUP
3361 016602 012737 000000 001514 MOV #0,HEAD ;TO FILL
3362 016610 012737 000000 001520 MOV #0,FORMAT ;HEADER
3363 016616 004737 033356 JSR PC,FHDTAB ;TABLE
3364
3365
3366 016622 012765 000027 000000 MOV #<WRHEAD>,RKCSI(R5) ;WRITE HEADER CMD
3367 016630 013737 001444 005450 MOV T5000,TEMP1 ;SETUP TIMEOUT
3368 016636 004737 032102 JSR PC,FRDY ;FIND RDY
3369 016642 104200 ERROR 200 ;NO RDY AFTER WRITE HEADER CMD
3370 016644 004737 032526 JSR PC,GSTAT ;GET FRESH STATUS
3371 016650 032737 100000 005406 BIT #CERR,HCSI
3372 016656 001405 BEQ 685
3373 016660 104201 ERROR 201 ;CERR AFTER WRITE HEADER CMD
3374 016662 104401 044510 TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS
3375 016666 000137 025264 JMP TIMING
3376 016672

```

685:

```

3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392

```

```

*****
;TEST 13 TEST OFFSET & RTC LOGIC
;
; THE HEADS ARE FIRST OFFSET BY OFFSET COMMANDS.
; THIS TEST CHECKS THE RTC LOGIC BY VERIFYING THAT THE
; 'OFFSET ON' BIT (MSG A,00) RESETS AND THE OFFSET REG
; BECOMES THE CYL DIFF INFO WHEN A SEEK CMD TO A
; DIFFERENT CYLINDER IS ISSUED
; IT ALSO TESTS THAT DRIVE CLEAR & SEEK TO SELF WILL NOT
; CLEAR THE 'OFFSET ON' BIT OR THE OFFSET REG.
; ALL OFFSET POSITIONS IN BOTH DIRECTIONS ARE CHECKED
*****

```

```

3393 016672 000004 ST13: SCOPE
3394 016674 012737 000001 001174 MOV #1,STIMES ;DO 1 ITERATION
3395 016702 012706 001100 MOV #STACK,SP ;RESTORE STK PTR
3396
3397 016706 012702 000001 MOV #1,R2 ;MIN POS OFFSET
3398
3399 016712 004737 032570 15: JSR PC,SUBCLR
3400 016716 104024 ERROR 24 ;CERR AFTER SCLR
3401
3402 016720 010265 000016 MOV R2,RKASOF(R5) ;SET OFFSET
3403
3404 016724 012737 017060 001176 MOV #64$,SESCAPE
3405 016732 012765 000015 000000 MOV #OFFSET,RKCSI(R5) ;OFFSET CMD
3406 016740 013737 001440 005450 MOV T100,TEMP1 ;SETUP TIMEOUT
3407 016746 004737 032102 JSR PC,FRDY
3408 016752 104033 ERROR 33 ;NO RDY AFTER OFFSET CMD
3409
3410 016754 005065 000026 CLR RKMR1(R5) ;GET WORD 0
3411 016760 004737 032526 JSR PC,GSTAT
3412 016764 012737 032140 005444 MOV #<D.PIP!D.SPIN!D.OFF!D.VV!D.DRA>,SBMR2

```


3413	016772	004737	035344		JSR	PC,CKMR2	;CHECK MR2
3414	016776	104035			ERROR	35	;MSG A0 ERROR DURING OFFSET CMD
3415	017000	005037	005446		CLR	SBMR3	
3416	017004	004737	035426		JSR	PC,CKMR3	;CHECK MR3
3417	017010	104061			ERROR	61	;MSG B0 ERROR DURING OFFSET CMD
3418	017012	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3419	017020	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
3420	017026	004737	032526		JSR	PC,GSTAT	
3421	017032	012737	001720	005444	MOV	#<D.SP0K!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2	
3422	017040	004737	035344		JSR	PC,CKMR2	
3423	017044	104036			ERROR	36	;MSG A1 ERROR DURING OFFSET
3424	017046	005037	005446		CLR	SBMR3	
3425	017052	004737	035426		JSR	PC,CKMR3	
3426	017056	104062			ERROR	62	;MSG B1 ERROR DURING OFFSET
3427	017060	005037	001176		CLR	\$ESCAPE	
3428	017064	013737	001442	005450	MOV	T5000,TEMP1	;SETUP TIMEOUT
3429	017072	004737	032414		JSR	PC,FATT2	;FIND ATTN
3430	017076	104034			ERROR	34	;NO ATTN AFTER OFFSET CMD
3431	017100	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	;CONTR CLEAR
3432	017106	004737	032526		JSR	PC,GSTAT	;GET LATEST STATUS
3433	017112	012737	052340	005444	MOV	#<D.DSC!D.OFF!D.SPIN!D.DRY!D.VV!D.DRA>,SBMR2	;SHOULD BE VALUE
3434	017120	004737	035344		JSR	PC,CKMR2	;CHECK MR2
3435	017124	104260			ERROR	260	;MSG A0 ERROR AFTER OFFSET CMD
3436	017126	005037	005446		CLR	SBMR3	
3437	017132	004737	035426		JSR	PC,CKMR3	;CHECK MR3
3438	017136	104261			ERROR	261	;MSG B0 ERROR AFTER OFFSET CMD
3439	017140	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3440	017146	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
3441	017154	004737	032526		JSR	PC,GSTAT	
3442	017160	012737	001720	005444	MOV	#<D.SP0K!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2	
3443	017166	004737	035344		JSR	PC,CKMR2	
3444	017172	104037			ERROR	37	;MSG A1 ERROR AFTER OFFSET CMD
3445	017174	005037	005446		CLR	SBMR3	
3446	017200	004737	035426		JSR	PC,CKMR3	
3447	017204	104040			ERROR	40	;MSG B1 ERROR AFTER OFFSET CMD
3448	017206	004737	033074		JSR	PC,RDCYLA	
3449	017212	005737	001404		TST	CYLADD	
3450	017216	001401			BEQ	17\$	
3451	017220	104042			ERROR	42	;CYL ADDR IN RKMR3 WAS NOT 0 ;AFTER OFFSET CMD FROM CYL 0
3452	017222						
3453	017222	010265	000016		MOV	R2,RKASOF(R5)	;REFRESH RKASOF
3454	017226	032702	000200		BIT	#BIT7,R2	
3455	017232	001007			BNE	65\$;BR IF NEG OFFSET
3456	017234	004737	032720		JSR	PC,RDCYLD	

K07

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 67
 DZR618.P11 T13 TEST OFFSET & RTC LOGIC

SEQ 0068

3469	017240	020237	001402		CMP	R2,CYLDIF	;CHECK POS OFFSET
3470	017244	001410			BEQ	66\$	
3471	017246	104114			ERROR	114	;OFFSET IN RKMR2 NOT = RKASCF
3472	017250	000406			BR	66\$;AFTER OFFSET CMD
3473							
3474	017252	004737	032720	66\$:	JSR	PC,RDCYLD	
3475	017256	020137	001402		CMP	R1,CYLDIF	;CHECK NEG OFFSET
3476	017262	001401			BEQ	66\$	
3477	017264	104114			ERROR	114	;OFFSET IN RKMR2 NOT = RKASCF
3478							;AFTER OFFSET CMD
3479	017266			66\$:			
3480							
3481	017266	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3482	017274	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	;DRIVE#
3483	017302	012765	000005	000000	MOV	#CLEAR,RKCS1(R5)	;DRIVE CLEAR CMD
3484	017310	013737	001432	005450	MOV	T10,TEMP1	;SETUP TIMEOUT
3485	017316	004737	032102		JSR	PC,FRDY	;FIND RDY
3486	017322	104151			ERROR	151	;NO RDY AFTER DRIVE CLEAR CMD
3487	017324	004737	032266		JSR	PC,TSTATN	;TEST FOR ATTN
3488	017330	000401			BR	67\$	
3489	017332	104154			ERROR	154	;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
3490	017334			67\$:			
3491							
3492	017334	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3493	017342	004737	032526		JSR	PC,GSTAT	
3494	017346	032737	002000	005434	BIT	#D.OFF,HMR2	
3495	017354	001001			BNE	4\$	
3496	017356	104043			ERROR	43	;OFFSET BIT IN RKMR2 CLEARED
3497							;AFTER DRIVE CLEAR CMD & SELECT DRV CMD
3498	017360			4\$:			
3499							
3500	017360	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	;CONTR CLEAR
3501	017366	004737	032526		JSR	PC,GSTAT	;GET LATEST STATUS
3502	017372	012737	012340	005444	MOV	#(D.OFF!D.SPIN!D.DRDY!D.VV!D.DRA),SBMR2	;SHOULD BE VALUE
3503	017400	004737	035344		JSR	PC,CKMR2	;CHECK MR2
3504	017404	104273			ERROR	273	;MSG A0 ERROR AFTER DRIVE CLEAR CMD
3505	017406	005037	005446		CLR	SBMR3	
3506	017412	004737	035426		JSR	PC,CKMR3	;CHECK MR3
3507	017416	104265			ERROR	265	;MSG B0 ERROR AFTER DRIVE CLEAR CMD
3508							
3509	017420	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3510	017426	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
3511	017434	004737	032526		JSR	PC,GSTAT	
3512	017440	012737	001720	005444	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK),SBMR2	
3513	017446	004737	035344		JSR	PC,CKMR2	
3514	017452	104274			ERROR	274	;MSG A1 ERROR AFTER DRIVE CLEAR CMD
3515	017454	005037	005446		CLR	SBMR3	
3516	017460	004737	035426		JSR	PC,CKMR3	
3517	017464	104266			ERROR	266	;MSG B1 ERROR AFTER DRIVE CLEAR CMD
3518							
3519	017466	010265	000016		MOV	R2,RKASOF(R5)	;REFRESH RKASOF
3520							
3521	017472	032702	000200		BIT	#BIT7,R2	
3522	017476	001007			BNE	68\$;BR IF NEG OFFSET
3523							
3524	017500	004737	032720		JSR	PC,RDCYLD	

3525	017504	020237	001402		CMP	R2,CYLDIF	;CHECK POS OFFSET
3526	017510	001410			BEQ	69\$	
3527	017512	104115			ERROR	115	;OFFSET IN RKMR2 NOT = RKASOF
3528	017514	000406			BR	69\$;AFTER DRIVE CLEAR CMD
3529							
3530	017516	004737	032720	68\$:	JSR	PC,RDCYLD	
3531	017522	020137	001402		CMP	R1,CYLDIF	;CHECK NEG OFFSET
3532	017526	001401			BEQ	69\$	
3533	017530	104115			ERROR	115	;OFFSET IN RKMR2 NOT = RKASOF
3534							;AFTER DRIVE CLEAR CMD
3535	017532			69\$:			
3536							
3537	017532	012765	000017	000000	MOV	#SEEK,RKCS1(R5)	;SEEK CMD
3538	017540	013737	001434	005450	MOV	T50,TEMP1	;SETUP TIMEOUT
3539	017546	004737	032102		JSR	PC,FRDY	;FIND RDY
3540	017552	104131			ERROR	131	;NO RDY AFTER SEEK CMD
3541							
3542	017554	013737	001444	005450	MOV	T50000,TEMP1	;SETUP TIMEOUT
3543	017562	004737	032414		JSR	PC,ATTN	;FIND ATTN
3544	017566	104132			ERROR	132	;NO ATTN AFTER SEEK CMD
3545							
3546	017570	032737	100000	005406	BIT	#CERR,HCS1	
3547	017576	001401			BEQ	70\$	
3548	017600	104210			ERROR	210	;CERR AFTER SEEK CMD
3549							
3550	017602			70\$:			
3551							
3552							
3553	017602	032737	002000	005434	BIT	#D.OFF,HMR2	
3554	017610	001001			BNE	7\$	
3555	017612	104045			ERROR	45	;OFFSET BIT CLEARED IN RKMR2 AFTER SEEK TO SELF.
3556							
3557	017614			7\$:			
3558	017614	010265	000016		MOV	R2,RKASOF(R5)	;REFRESH RKASOF
3559							
3560	017620	032702	000200		BIT	#BIT7,R2	
3561	017624	001007			BNE	71\$;BR IF NEG OFFSET
3562							
3563	017626	004737	032720		JSR	PC,RDCYLD	
3564	017632	020237	001402		CMP	R2,CYLDIF	;CHECK POS OFFSET
3565	017636	001410			BEQ	72\$	
3566	017640	104123			ERROR	123	;OFFSET IN RKMR2 NOT = RKASOF
3567	017642	000406			BR	72\$;AFTER SEEK TO SELF
3568							
3569	017644	004737	032720	71\$:	JSR	PC,RDCYLD	
3570	017650	020137	001402		CMP	R1,CYLDIF	;CHECK NEG OFFSET
3571	017654	001401			BEQ	72\$	
3572	017656	104123			ERROR	123	;OFFSET IN RKMR2 NOT = RKASOF
3573							;AFTER SEEK TO SELF
3574	017660			72\$:			
3575							
3576	017660	004737	032570		JSR	PC,SUBCLR	
3577	017664	104024			ERROR	24	;CERR AFTER SCLR
3578							
3579	017666	012737	000012	001372	MOV	#10.,TOCYL	
3580	017674	012765	000012	000020	MOV	#10.,RKDC(R5)	;SETUP CYL 10


```

3637 020122 013737 001444 005450      MOV    T50000,TEMP1      ;SETUP TIMEOUT
3638 020130 004737 032414              JSR    PC,FATT2         ;FIND ATTN
3639 020134 104132                      ERROR  132             ;NO ATTN AFTER SEEK CMD
3640
3641 020136 032737 100000 005406      BIT    #CERR,HCS1
3642 020144 001401                      BEQ    76$
3643 020146 104210                      ERROR  210             ;CERR AFTER SEEK CMD
3644
3645 020150                                76$:
3646
3647
3648 020150 032702 000200      BIT    #BIT7,R2         ;SEE IF DOING NEG OFFSETS
3649 020154 001014                      BNE    18$             ;BR IF YES
3650
3651 020156 005202                      INC    R2
3652 020160 020227 000061      CMP    R2,#61          ;SEE IF JUST DID MAX POS OFFSET
3653 020164 001402                      BEQ    20$             ;BR IF YES
3654 020166 000137 016712      JMP    1$              ;ELSE DO NEXT P'S OFFSET
3655
3656 020172 012702 000201      20$: MOV    #201,R2      ;SETUP NEG OFFSET FOR RKASOF
3657 020176 012701 000101      MOV    #101,R1         ;SETUP NEG OFFSET OFOR MSG A
3658 020202 000137 016712      JMP    1$              ;DO NEG OFFSET
3659
3660 020206 005201      18$: INC    R1
3661 020210 005202                      INC    R2
3662 020212 020227 000261      CMP    R2,#261        ;SEE IF ALL NEG OFFSETS DONE
3663 020216 001402                      BEQ    TST14          ;GO TO NEXT TST
3664 020220 000137 016712      JMP    1$              ;DO ANOTHER
3665

```

```

*****
;TEST 14      TEST READ DATA AT ALL HEAD OFFSET POSITIONS

```

```

;
; THIS TEST VERIFIES THAT THE HEAD OFFSET LOGIC IS OPERATIONAL BY
; WRITING ALL 1'S PATTERNS ON CYLINDER 0, HEAD 0. THEN
; PERFORMING READ DATA FROM CENTERLINE AND MOVING OUT + AND - OFFSET
; POSITIONS UNTIL A FAILURE OCCURES. THE FAILING OFFSET POSITIONS
; ARE PRINTED OUT IF LESS THAN THE OFFSET TOLERANCE TO BE SPECIFIED.
; OFFSET CODES ARE ALSO VERIFIED BY READING MSG A, STAJ S 00 & 10.

```

```

; ALL HEADS ARE TESTED AT CYLINDER 0
; IF THERE ARE NO FAILURES AT ALL, THIS INDICATES THAT

```

```

; OR
; A. HEADS DID NOT MOVE AT ALL
; B. THE COMBINATION OF DISC SURFACE, HEADS, R/W AMP
; ARE EXCEPTIONALLY GOOD.

```

```

; AN APPROPRIATE MESSAGE WILL BE TYPED.

```

```

*****
TST14:

```

```

3686 020224 000004      SCOPE
3687 020226 012737 000001 001174      MOV    #1,STIMES      ;DO 1 ITERATION
3688 020234 012706 001100      MOV    #STACK,SP      ;RESTORE STK PTR
3689
3690 020240 004737 032570      JSR    PC,SUBCLR
3691 020244 104024                      ERROR  24              ;CERR AFTER SCLR
3692

```

3693	020246	012765	001532	000004	MOV	#DATA1,RKBA(R5)	:WRITE ALL 1'S
3694	020254	052765	000020	000010	BIS	#BA1,RKCS2(R5)	:BUSS ADDR INCR INHIBIT
3695	020262	012765	177400	000002	MOV	#-256.,RKWC(R5)	:SECTOR 0 ONLY
3696							:WILL DO AN IMPLIED SEEK TO CYL 0.
3697							:WAS ON CYL 1 FROM LAST TEST
3698							
3699	020270	012765	000023	000000	MOV	#(WRDATA),RKCS1(R5)	:WRITE DATA CMD
3700	020276	013737	001444	005450	MOV	T50000,TEMP1	:SETUP TIMEOUT
3701	020304	004737	032102		JSR	PC,FRDY	:FIND RDY
3702	020310	104011			ERROR	11	:NO RDY AFTER WRITE DATA CMD
3703	020312	004737	032526		JSR	PC,GSTAT	:GET FRESH STATUS
3704	020316	032737	100000	005406	BIT	#CERR,HCS1	
3705	020324	001465			BEQ	64\$	
3706	020326	104012			ERROR	12	:CERR AFTER WRITE DATA CMD
3707	020330	104401	044510		TYPE	MSG26	:ABORTING DATA TESTS TO DO TIMING
3708	020334	000137	025264		JMP	TIMING	
3709							
3710	020340					64\$:	
3711							
3712	020340	012765	100000	000000	MOV	#CLR,RKCS1(R5)	:CONTR CLEAR
3713	020346	004737	032526		JSR	PC,GSTAT	:GET LATEST STATUS
3714	020352	012737	010340	005444	MOV	#(0!D.SPIN!D.DRDY!D.VV!D.DRA),SBMR2	:SHOULD BE VALUE
3715	020360	004737	035344		JSR	PC,CKMR2	:CHECK MR2
3716	020364	104052			ERROR	52	:MSG A0 ERROR AFTER WRITE DATA CMD
3717	020366	005037	005446		CLR	SBMR3	
3718	020372	004737	035426		JSR	PC,CKMR3	:CHECK MR3
3719	020376	104023			ERROR	23	:MSG B0 ERROR AFTER WRITE DATA CMD
3720							
3721	020400	012765	100000	000000	MOV	#CLR,RKCS1(R5)	
3722	020406	012765	000001	000026	MOV	#1,RKMR1(R5)	:SELECT WORD 1
3723	020414	004737	032526		JSR	PC,GSTAT	
3724	020420	012737	001720	005444	MOV	#(0.SPOK!D.CART!D.DOOR!D.BRM!D.TFOK),SBMR2	
3725	020426	004737	035344		JSR	PC,CKMR2	
3726	020432	104053			ERROR	53	:MSG A1 ERROR AFTER WRITE DATA CMD
3727	020434	005037	005446		CLR	SBMR3	
3728	020440	004737	035426		JSR	PC,CKMR3	
3729	020444	104025			ERROR	25	:MSG B1 ERROR AFTER WRITE DATA CMD
3730							
3731	020446	012765	001532	000004	MOV	#DATA1,RKBA(R5)	
3732	020454	052765	000020	000010	BIS	#BA1,RKCS2(R5)	
3733	020462	012765	177400	000002	MOV	#-256.,RKWC(R5)	
3734							
3735	020470	012765	000031	000000	MOV	#(VRTCHK),RKCS1(R5)	:WRITE CHECK CMD
3736	020476	013737	001444	005450	MOV	T50000,TEMP1	:SETUP TIMEOUT
3737	020504	004737	032102		JSR	PC,FRDY	:FIND RDY
3738	020510	104015			ERROR	15	:NO RDY AFTER WRITE CHECK CMD
3739							
3740	020512	004737	032526		JSR	PC,GSTAT	:GET FRESH STATUS
3741	020516	032737	100000	005406	BIT	#CERR,HCS1	
3742	020524	001417			BEQ	66\$	
3743							
3744	020526	032737	040000	005410	BIT	#WCE,HCS2	:SEE IF WRITE CHECK ERROR
3745	020534	001412			BEQ	65\$	
3746	020536	016537	000024	001506	MOV	RKDB(R5),WD1	:ACTUAL WORD FOR TYPEOUT
3747	020544	013737	001532	001510	MOV	DATA1,WD2	:EXPECTED WORD FOR TYPEOUT
3748	020552	004737	033666		JSR	PC,TRUERR	:CHECK AGAINST BSE INFO

3805	021030	005065	000026		CLR	RKMR1(R5)	;GET WORD 0
3806	021004	004737	032526		JSR	PC,GSTAT	
3807	021010	012737	032140	005444	MOV	#<D.PIP!D.SPIN!D.OFF!D.VV!D.DRA>	,SBMR2
3808	021016	004737	035344		JSR	PC,CKMR2	;CHECK MR2
3809	021022	104035			ERROR	35	;MSG A0 ERROR DURING OFFSET CMD
3810							
3811	021024	005037	005446		CLR	SBMR3	
3812	021030	004737	035426		JSR	PC,CKMR3	;CHECK MR3
3813	021034	104061			ERROR	61	;MSG B0 ERROR DURING OFFSET CMD
3814							
3815	021036	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3816	021044	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
3817	021052	004737	032526		JSR	PC,GSTAT	
3818	021056	012737	001720	005444	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>	,SBMR2
3819	021064	004737	035344		JSR	PC,CKMR2	
3820	021070	104036			ERROR	36	;MSG A1 ERROR DURING OFFSET
3821							
3822	021072	005037	005446		CLR	SBMR3	
3823	021076	004737	035426		JSR	PC,CKMR3	
3824	021102	104062			ERROR	62	;MSG B1 ERROR DURING OFFSET
3825							
3826	021104	005037	001176		CLR	\$ESCAPE	
3827	021110	013737	001442	005450	MOV	T5000,TEMP1	;SETUP TIMEOUT
3828	021116	004737	032414		JSR	PC,FATT2	;FIND ATTN
3829	021122	104034			ERROR	34	;NO ATTN AFTER OFFSET CMD
3830							
3831							
3832	021124	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	;CONTR CLEAR
3833	021132	004737	032526		JSR	PC,GSTAT	;GET LATEST STATUS
3834	021136	012737	052340	005444	MOV	#<D.DSC!D.OFF!D.SPIN!D.DRDY!D.VV!D.DRA>	,SBMR2 ;SHOULD BE VALUE
3835	021144	004737	035344		JSR	PC,CKMR2	;CHECK MR2
3836	021150	104260			ERROR	260	;MSG A0 ERROR AFTER OFFSET CMD
3837	021152	005037	005446		CLR	SBMR3	
3838	021156	004737	035426		JSR	PC,CKMR3	;CHECK MR3
3839	021162	104261			ERROR	261	;MSG B0 ERROR AFTER OFFSET CMD
3840							
3841	021164	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3842	021172	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
3843	021200	004737	032526		JSR	PC,GSTAT	
3844	021204	012737	001720	005444	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>	,SBMR2
3845	021212	004737	035344		JSR	PC,CKMR2	
3846	021216	104037			ERROR	37	;MSG A1 ERROR AFTER OFFSET CMD
3847	021220	005037	005446		CLR	SBMR3	
3848	021224	004737	035426		JSR	PC,CKMR3	
3849	021230	104040			ERROR	40	;MSG B1 ERROR AFTER OFFSET CMD
3850							
3851							
3852							
3853	021232	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
3854	021240	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	;DRIVE#
3855	021246	012765	000005	000000	MOV	#CLEAR,RKCS1(R5)	;DRIVE CLEAR CMD
3856	021254	013737	001432	005450	MOV	T10,TEMP1	;SETUP TIMEOUT
3857	021262	004737	032102		JSR	PC,FRDY	;FIND RDY
3858	021266	104151			ERROR	151	;NO RDY AFTER DRIVE CLEAR CMD
3859	021270	004737	032266		JSR	PC,TSTATN	;TEST FOR ATTN
3860	021274	000401			BR	68\$	


```

3861 021276 104154          ERROR 154          ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
3862 021300          68$:
3863
3864 021300 012765 001532 000004      MOV      #DATA1,RKBA(R5)
3865 021306 052765 000020 000010      BIS      #BA1,RKCS2(R5)
3866 021314 012765 177400 000002      MOV      #-256.,RKWC(R5)
3867 021322 012765 000031 000000      MOV      #WRCHK,RKCS1(R5) ;WRITE CHECK CMD
3868 021330 012737 141520 005450      MOV      #50000.,TEMP1 ;SETUP TIMEOUT
3869 021336 004737 032102          JSR      PC,FRDY ;FIND RDY
3870 021342 104015          ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
3871 021344 004737 032526          JSR      PC,GSTAT ;GET FRESH STATUS
3872 021350 032737 040000 005410      BIT      #WCE,HCS2
3873 021356 001423          BEQ      2$
3874
3875 021360 016537 000024 001506      MOV      RKDB(R5),WD1 ;GET MISCOMPARED WORD
3876 021366 004737 033666          JSR      PC,TRUEFR ;CHECK AGAINST BSE INFO
3877 021372 005237 001512          INC      OFFERR ;BAD WRITE CHK ERROR=SET ERR FLG.
3878
3879 021376 005737 001512          TST      OFFERR
3880 021402 001411          BEQ      2$
3881 021404 104401 045634          TYPE    MSG39 ;WRITE CHECK FAILURE AT OFFSET
3882 021410 010046          MOV      RD,-(SP) ;SAVE RD FOR TYPEOUT
3883 ;TYPE OFFSET VALUE
3884 021412 104403          TYPOS   ;GO TYPE--OCTAL ASCII
3885 021414 006 ;TYPE 6 DIGITS
3886 021415 000 ;SUPPRESS LEADING ZEROS
3887 021416 104401 001205      TYPE    ,SCRLF
3888 021422 104401 001205      TYPE    ,SCRLF
3889
3890 021426 032700 000200          2$: BIT      #BIT7,RO ;SEE IF OFFSET IS + OR -
3891 021432 001023          BNE     5$ ;BR IF - OFFSET
3892
3893 021434 020027 000060          CMP      RO,#60
3894 021440 001412          BEQ      4$
3895 021442 005737 001512          TST      OFFERR
3896 021446 001404          BEQ      3$
3897 021450 012700 000200          8$: MOV      #200,RO ;SETUP FOR NEG OFFSET
3898 021454 000137 020722          JMP      1$
3899
3900 021460 005200          3$: INC      RO
3901 021462 000137 020722          JMP      1$
3902
3903 021466 005737 001512          4$: TST      OFFERR
3904 021472 001366          BNE     8$ ;DO NEG OFFSETS
3905 021474 104401 045476          TYPE    ,MSG37 ;NO WRITE CHECK ERROR AT MAX POS OFFSET
3906 ;NOTE! EITHER HEADS DID NOT MOVE
3907 ;OR READ/WRITE AMP IS EXCEPTIONALLY GOOD.
3908 021500 000763          BR      8$ ;DO NEG OFFSETS
3909
3910 021502 020027 000260          5$: CMP      RO,#260
3911 021506 001404          BEQ      6$
3912 021510 005737 001512          TST      OFFERR
3913 021514 001076          BNE     TST15 ;GO TO NEXT TST
3914 021516 000760          BR      3$
3915
3916 021520 000737 001512          6$: TST      OFFERR

```

```

3917 021524 001002          BNE      75
3918 021526 104401 045554   TYPE    ,MSG38          ;NO WRITE CHECK ERROR AT MAX NEG OFFSET
3919                                     ;NOTE! EITHER HEADS DID NOT MOVE
3920                                     ;OR READ/WRITE AMP IS EXCEPTIONALLY GOOD.
3921 021532          75:
3922
3923 02 532 012765 100000 000000   MOV     #CCLR,RKCS1(R5)
3924 02 540 013765 001222 000010   MOV     $UNIT,RKCS2(R5)
3925 02 546 012765 000013 000000   MOV     #RECAL,RKCS1(R5)          ;RECAL CMD
3926                                     ;RESET CYL DIFF/OFFSET & CYL ADDR REG
3927                                     ;IN RKMR2 & RKMR3 RESP.
3928 021554 013737 001432 005450   MOV     T10,TEMP1
3929 021562 004737 032102          JSR     PC,FRDY
3930 021566 104124          ERROR   124              ;SETUP TIMEOUT
3931                                     ;FIND RDY
3932 021570 012765 000001 000026   MOV     #1,RKMR1(R5)          ;SELECT WORD 1
3933 021576 004737 032526          JSR     PC,GSTAT
3934 021602 032737 020000 005434   BIT     #D.RTZ,HMR2
3935 021610 001001          BNE     69$
3936 021612 104244          ERROR   244              ;RTZ NOT SET DURING RECAL CMD
3937 021614 013737 001432 005452 69$:   MOV     T10,TEMP2
3938 021622 004737 032320          JSR     PC,FATT1
3939 021626 104055          ERROR   55              ;SETUP TIMEOUT
3940                                     ;FIND ATTN
3941 021630 012765 100000 000000   MOV     #CCLR,RKCS1(R5)
3942 021636 013765 001222 000010   MOV     $UNIT,RKCS2(R5)          ;DRIVE#
3943 021644 012765 000005 000000   MOV     #CLEAR,RKCS1(R5)          ;DRIVE CLEAR CMD
3944 021652 013737 001432 005450   MOV     T10,TEMP1
3945 021660 004737 032102          JSR     PC,FRDY
3946 021664 104151          ERROR   151              ;SETUP TIMEOUT
3947 021666 004737 032266          JSR     PC,TSTATN
3948 021672 000401          BR     70$
3949 021674 104154          ERROR   154              ;NO RDY AFTER DRIVE CLEAR CMD
3950                                     ;TEST FOR ATTN
3951 021676          70$:
3952                                     ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
3953 021676 005201          INC     R1
3954 021700 020127 000003          CMP     R1,#3
3955 021704 001402          BEQ     TST15
3956 021706 000137 020700          JMP     95$
3957                                     ;HEAD CTR
3958                                     ;SEE IF ALL HEADS DONE
3959                                     ;BR IF YES
3960                                     ;ELSE REPEAT ALL FOR NEXT HEAD
3961                                     ;*****
3962                                     ;TEST 15 WRITE WITH HEADS OFFSET
3963                                     ;*
3964                                     ;THIS TEST VERIFIES THAT WHEN ATTEMPTING TO
3965                                     ;WRITE WITH HEADS OFFSET THAT THE OFFSET WILL CLEAR
3966                                     ;& THE DRIVE WILL WRITE
3967                                     ;SINCE THE WRITE COMMAND HAS AN IMPLIED RTC.
3968                                     ;THIS TEST IS PERFORMED FOR MAX POS & NEG OFFSETS ONLY
3969                                     ;*****
3970 TST15: SCOPE
3971 021712 000004          MOV     #1,$TIMES          ;DO 1 ITERATION
3972 021714 012737 000001 001174   MOV     #STACK,SP          ;RESTORE STK PTR
3973 021722 012706 001100          MOV     #260,R0           ;MAX NEG OFFSET

```

G08

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 76
 DZR618.P11 T15 WRITE WITH HEADS OFFSET

SEQ 0077

3973	021732	004737	032570		!S:	JSR	PC, SUBCLR	
3974	021736	104024				ERROR	24	;CERR AFTER SCLR
3975								
3976	021740	010065	00J16			MOV	RD, RKASOF(R5)	;SET OFFSET
3977								
3978	021744	012737	022100	001176		MOV	#64\$ \$ESCAPE	
3979	021752	012765	000015	000000		MOV	#OFFSET, RKCS1(R5)	;OFFSET CMD
3980	021760	013737	001440	005450		MOV	T100, TEMP1	;SETLP TIMEOUT
3981	021766	004737	032102			JSR	PC, FRDY	
3982	021772	104033				ERROR	33	;NO RDY AFTER OFFSET CMD
3983								
3984	021774	005065	000026			CLR	RKMR1(R5)	;GET WORD 0
3985	022000	004737	032526			JSR	PC, GSTAT	
3986	022004	012737	032140	005444		MOV	#<D.PIP!D.SPIN!D.OFF!D.VV!D.DRA>, SBMR2	
3987	022012	004737	035344			JSR	PC, CKMR2	;CHECK MR2
3988	022016	104035				ERROR	35	;MSG A0 ERROR DURING OFFSET CMD
3989								
3990	022020	005037	005446			CLR	SBMR3	
3991	022024	004737	035426			JSR	PC, CKMR3	;CHECK MR3
3992	022030	104061				ERROR	61	;MSG B0 ERROR DURING OFFSET CMD
3993								
3994	022032	012765	100000	000000		MOV	#CLR, RKCS1(R5)	
3995	022040	012765	000001	000026		MOV	#1, RKMR1(R5)	;SELECT WORD 1
3996	022046	004737	032526			JSR	PC, GSTAT	
3997	022052	012737	001720	005444		MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>, SBMR2	
3998	022060	004737	035344			JSR	PC, CKMR2	
3999	022064	104036				ERROR	36	;MSG A1 ERROR DURING OFFSET
4000								
4001	022066	005037	005446			CLR	SBMR3	
4002	022072	004737	035426			JSR	PC, CKMR3	
4003	022076	104062				ERROR	62	;MSG B1 ERROR DURING OFFSET
4004								
4005	022100	005037	001176		64\$:	CLR	\$ESCAPE	
4006	022104	013737	001442	005450		MOV	T500, TEMP1	;SETUP TIMEOUT
4007	022112	004737	032414			JSR	PC, FATT2	;FIND ATTN
4008	022116	104034				ERROR	34	;NO ATTN AFTER OFFSET CMD
4009								
4010								
4011	022120	012765	100000	000000		MOV	#CLR, RKCS1(R5)	;CONTR CLEAR
4012	022126	004737	032526			JSR	PC, GSTAT	;GET LATEST STATUS
4013	022132	012737	052340	005444		MOV	#<D.DSC!D.OFF!D.SPIN!D.DRDY!D.VV!D.DRA>, SBMR2	;SHOULD BE VALUE
4014	022140	004737	035344			JSR	PC, CKMR2	;CHECK MR2
4015	022144	104260				ERROR	260	;MSG A0 ERROR AFTER OFFSET CMD
4016	022146	005037	005446			CLR	SBMR3	
4017	022152	004737	035426			JSR	PC, CKMR3	;CHECK MR3
4018	022156	104261				ERROR	261	;MSG B0 ERROR AFTER OFFSET CMD
4019								
4020	022160	012765	100000	000000		MOV	#CLR, RKCS1(R5)	
4021	022166	012765	000001	000026		MOV	#1, RKMR1(R5)	;SELECT WORD 1
4022	022174	004737	032526			JSR	PC, GSTAT	
4023	022200	012737	001720	005444		MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>, SBMR2	
4024	022206	004737	035344			JSR	PC, CKMR2	
4025	022212	104037				ERROR	37	;MSG A1 ERROR AFTER OFFSET CMD
4026	022214	005037	005446			CLR	SBMR3	
4027	022220	004737	035426			JSR	PC, CKMR3	
4028	022224	104040				ERROR	40	;MSG B1 ERROR AFTER OFFSET CMD

H08

4029						
4030						
4031						
4032	022226	012765	100000	000000	MOV	#CCLR,RKCS1(R5)
4033	022234	013765	001222	000010	MOV	\$UNIT,RKCS2(R5) ;DRIVE#
4034	022242	012765	000005	000000	MOV	#CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
4035	022250	013737	001432	005450	MOV	T10,TEMP1 ;SETUP TIMEOUT
4036	022256	004737	032102		JSR	PC,FRDY ;FIND RDY
4037	022262	104151			ERROR	151 ;NO RDY AFTER DRIVE CLEAR CMD
4038	022264	004737	032266		JSR	PC,TSTATN ;TEST FOR ATTN
4039	022270	000401			BR	65\$
4040	022272	104154			ERROR	154 ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
4041	022274					65\$:
4042						
4043						
4044	022274	012765	001526	000004	MOV	#DATA0,RKBA(R5) ;WRITE ALL 0'S
4045	022302	052765	000020	000010	BIS	#BAI,RKCS2(R5) ;BUS ADDR INCR INHIBIT
4046	022310	012765	177400	000002	MOV	#-256.,RKWC(R5) ;SECTOR 0 ONLY
4047						
4048	022316	012765	000023	000000	MOV	#<WRDATA>,RKCS1(R5) ;WRITE DATA CMD
4049	022324	013737	001444	005450	MOV	T5000,TEMP1 ;SETUP TIMEOUT
4050	022332	004737	032102		JSR	PC,FRDY ;FIND RDY
4051	022336	104011			ERROR	11 ;NO RDY AFTER WRITE DATA CMD
4052	022340	004737	032526		JSR	PC,GSTAT ;GET FRESH STATUS
4053	022344	032737	100000	005406	BIT	#CERR,HCS1
4054	022352	001405			BEQ	66\$
4055	022354	104012			ERROR	12 ;CERR AFTER WRITE DATA CMD
4056	022356	104401	044510		TYPE	MSG26 ;REPORTING DATA TESTS TO DO TIMING
4057	022362	000137	025264		JMP	TIMING
4058						
4059	022366					66\$:
4060						
4061	022366	012765	100000	000000	MOV	#CCLR,RKCS1(R5) ;CONTR CLEAR
4062	022374	004737	032526		JSR	PC,GSTAT ;GET LATEST STATUS
4063	022400	012737	010340	005444	MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
4064	022406	004737	035344		JSR	PC,CKMR2 ;CHECK MR2
4065	022412	104052			ERROR	52 ;MSG A0 ERROR AFTER WRITE DATA CMD
4066	022414	005037	005446		CLR	SBMR3
4067	022420	004737	035426		JSR	PC,CKMR3 ;CHECK MR3
4068	022424	104023			ERROR	23 ;MSG B0 ERROR AFTER WRITE DATA CMD
4069						
4070	022426	012765	100000	000000	MOV	#CCLR,RKCS1(R5)
4071	022434	012765	000001	000026	MOV	#1,RKMR1(R5) ;SELECT WORD 1
4072	022442	004737	032526		JSR	PC,GSTAT
4073	022446	012737	001720	005444	MOV	#<0.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2
4074	022454	004737	035344		JSR	PC,CKMR2
4075	022460	104053			ERROR	53 ;MSG A1 ERROR AFTER WRITE DATA CMD
4076	022462	005037	005446		CLR	SBMR3
4077	022466	004737	035426		JSR	PC,CKMR3
4078	022472	104025			ERROR	25 ;MSG B1 ERROR AFTER WRITE DATA CMD
4079						
4080						
4081	022474	012765	100000	000000	MOV	#CCLR,RKCS1(R5)
4082	022502	004737	032720		JSR	PC,RCYLD ;READ CYL DIFF IN RKMR2
4083	022506	005737	001402		TST	CYLDIF
4084	022512	001401			BEQ	67\$

4085	022514	104104			ERROR	104		;CYL DIFF/OFFSET NOT CLEARED AFTER WRITE CMD WITH OFFSET
4086	022516	004737	033074		JSR	PC,RDCYLA	67\$:	;READ CYL ADDR IN RKMR3
4087	022522	005737	001404		TST	CYLADD		
4088	022526	001401			BEQ	68\$		
4089	022530	104105			ERROR	105	68\$:	;CYL ADDR NOT CLEARED AFTER WRITE CMD WITH OFFSET
4090	022532							
4091								
4092								
4093	022532	104415			SCOP1			
4094	022534	012706	001100		MOV	#STACK,SP		;RESTORE STK PTR
4095								
4096	022540	004737	032570		JSR	PC,SUBCLR		
4097	022544	104024			ERROR	24		;CERR AFTER SCLR
4098								
4099								
4100	022546	012765	001526	000004	MOV	#DATA0,RKBA(R5)		
4101	022554	052765	000020	000010	BIS	#BA1,RKCS2(R5)		
4102	022562	012765	177400	000002	MOV	#-256.,RKWC(R5)		
4103								
4104	022570	012765	000031	000000	MOV	#(WRTCHK),RKCS1(R5)		;WRITE CHECK CMD
4105	022576	013737	001444	005450	MOV	T50000,TEMP1		;SETUP TIMEOUT
4106	022604	004737	032102		JSR	PC,FRDY		;FIND RDY
4107	022610	104015			ERROR	15		;NO RDY AFTER WRITE CHECK CMD
4108								
4109	022612	004737	032526		JSR	PC,GSTAT		;GET FRESH STATUS
4110	022616	032737	100000	005406	BIT	#CERR,HCS1		
4111	022624	001417			BEQ	70\$		
4112								
4113	022626	032737	040000	005410	BIT	#WCE,HCS2		;SEE IF WRITE CHECK ERROR
4114	022634	001412			BEQ	69\$		
4115	022636	016537	000024	001506	MOV	RKDB(R5),WD1		;ACTUAL WORD FOR TYPEOUT
4116	022644	013737	001526	001510	MOV	DATA0,WD2		;EXPECTED WORD FOR TYPEOUT
4117	022652	004737	033666		JSR	PC,TRUERR		;CHECK AGAINST BSE INFO
4118	022656	104016			ERROR	16		;WCE AFTER WRITE CMD
4119	022660	000401			BR	70\$		
4120								
4121	022662	104022			ERROR	22	69\$:	;CERR AFTER WRITE CHK CMD
4122								
4123	022664						70\$:	
4124								
4125	022664	012765	100000	000000	MOV	#CLR,RKCS1(R5)		;CONTR CLEAR
4126	022672	004737	032526		JSR	PC,GSTAT		;GET LATEST STATUS
4127	022676	012737	010340	005444	MOV	#(D.SPIN!D.DRDY!D.VV!D.DRA),SBMR2		;SHOULD BE VALUE
4128	022704	004737	035344		JSR	PC,CKMR2		;CHECK MR2
4129	022710	104057			ERROR	57		;MSG A0 ERROR AFTER WRITE CHECK CMD
4130	022712	005037	005446		CLR	SBMR3		
4131	022716	004737	035426		JSR	PC,CKMR3		;CHECK MR3
4132	022722	104031			ERROR	31		;MSG B0 ERROR AFTE WRITE CHECK CMD
4133								
4134	022724	012765	100000	000000	MOV	#CLR,RKCS1(R5)		
4135	022732	012765	000001	000026	MOV	#1,RKMR1(R5)		;SELECT WORD 1
4136	022740	004737	032526		JSR	PC,GSTAT		
4137	022744	012737	001720	005444	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK),SBMR2		
4138	022752	004737	035344		JSR	PC,CKMR2		
4139	022756	104060			ERROR	60		;MSG A1 ERROR AFTER WRITE CHECK CMD
4140	022760	005037	005446		CLR	SBMR3		

JOB

UNIBUS RKO6 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 79
 DZR6IB.P11 T15 WRITE WITH HEADS OFFSET

SEQ 0020

4141	022764	004737	035426		JSR	PC,CKMR3	
4142	022770	104032			ERROR	32	;MSG B1 ERROR AFTER WRITE CHECK CMD
4143							
4144							
4145							
4146	022772	020027	000260		CMP	RO,#260	
4147	022776	001004			SNE	25	;BR IF JUST DID POS OFFSET
4148	023000	012700	000060		MOV	#60,RO	;ELSE SETUP FOR POS OFFSET
4149	023004	000137	021732		JMP	15	
4150							
4151	023010						25:
4152							
4153	023010	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
4154	023016	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	
4155	023024	012765	000013	000000	MOV	#RECAL,RKCS1(R5)	;RECAL CMD
4156							;RESET CYL DIFF/OFFSET & CYL ADDR REG
4157							;IN RKMR2 & RKMR3 RESP.
4158	023032	013737	001432	005450	MOV	T10,TEMP1	;SETUP TIMEOUT
4159	023040	004737	032102		JSR	PC,FRDY	;FIND RDY
4160	023044	104124			ERROR	124	;RDY NOT SET AFTER RECAL CMD
4161							
4162	023046	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
4163	023054	004737	032526		JSR	PC,GSTAT	
4164	023060	032737	020000	005434	BIT	#D,RTZ,HMR2	
4165	023066	001001			BNE	715	
4166	023070	104244			ERROR	244	;RTZ NOT SET DURING RECAL CMD
4167	023072	013737	001432	005452	MOV	T10,TEMP2	;SETUP TIMEOUT
4168	023100	004737	032320		JSR	PC,FATT1	;FIND ATTN
4169	023104	104055			ERROR	55	;NO ATTN AFTER RECAL CMD
4170							
4171	023106	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
4172	023114	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	;DRIVE#
4173	023122	012765	000005	000000	MOV	#CLEAR,RKCS1(R5)	;DRIVE CLEAR CMD
4174	023130	013737	001432	005450	MOV	T10,TEMP1	;SETUP TIMEOUT
4175	023136	004737	032102		JSR	PC,FRDY	;FIND RDY
4176	023142	104151			ERROR	151	;NO RDY AFTER DRIVE CLEAR CMD
4177	023144	004737	032266		JSR	PC,TSTATN	;TEST FOR ATTN
4178	023150	000401			BR	725	
4179	023152	104154			ERROR	154	;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
4180	023154						725:

4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196

```

*****
*TEST 16      TEST CURRENT CROSS-OVER CYLINDERS
*
*   THIS TEST VERIFIES THAT THE DRIVE CAN WRITE & READ OFF
*   CURRENT CHANGE CYLINDERS X & Y IN THE FOLLOWING WAY:
*
*   SPIRAL WRITING IS PERFORMED FROM CYLINDER X TO CYLINDER Y
*   WITH A DATA PATTERN FILLING THE ENTIRE 2 CYLINDERS.
*
*   A WRITE CHECK IS THEN PERFORMED TO VERIFY DATA WAS PROPERLY WRITTEN.
*   THIS TEST IS PERFORMED FOR ALL 3 HEADS.
*
*   CYLINDER X:  63 127 191 255 319 383

```

K08

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 80
D2R618.P11 T16 TEST CURRENT CROSS-OVER CYLINDERS

SEQ 0081

```
4197          : *      CYLINDER Y: 64 128 192 256 320 384
4198          : *
4199          : *
4200 023154 000004          : *****
4201 023156 012737 0C0001 001174  †ST16: SCOPE
4202 023164 012706 001100          MOV #1,STIMES ;;DO 1 ITERATION
4203          MOV #STACK,SP
4204 023170 012700 001446          MOV #CYL,RO ;CYL ADDR TABLE
4205          IS: JSR PC,SUBCLR
4206 023174 004737 032570          ERROR 24 ;CERR AFTER SCLR
4207 023200 104024
4208          MOV (RO),RKDC(R5) ;CYL #
4209 023202 011065 000020          CLR RKDA(R5) ;HEAD/SECTOR
4210 023206 005065 000006          MOV #DPAT1,RKBA(R5) ;DATA PATTERN
4211 023212 012765 001534 000004          BIS #BAI,RKCS2(R5) ;BUSS ADDR INCREMENT INHIBIT
4212 023220 052765 000020 000010          MOV #-6*22.*256.,RKWC(R5) ;WORD COUNT TO SPIRAL & FILL 2 CYLINDERS
4213 023226 012765 076000 000002
4214          MOV #<WRDATA>,RKCS1(R5) ;WRITE DATA CMD
4215 023234 012765 000023 000000          MOV T50000,TEMP1 ;SETUP TIMEOUT
4216 023242 013737 001444 005450          JSR PC,FRDY ;FIND ROY
4217 023250 004737 032102          ERROR 11 ;NO ROY AFTER WRITE DATA CMD
4218 023254 104011          JSR PC,GSTAT ;GET FRESH STATUS
4219 023256 004737 032526          BIT #CERR,HCS1
4220 023262 032737 100000 005406          BEQ 64$
4221 023270 001405          ERROR 12 ;CERR AFTER WRITE DATA CMD
4222 023272 104012          TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING
4223 023274 104401 044510          JMP TIMING
4224 023300 000137 025264
4225          64$:
4226 023304
4227          MOV #CCLR,RKCS1(R5) ;CONTR CLEAR
4228 023304 012765 100000 000000          JSR PC,GSTAT ;GET LATEST STATUS
4229 023312 004737 032526          MOV #<D!D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2 ;SHOULD BE VALUE
4230 023316 012737 010340 005444          JSR PC,CKMR2 ;CHECK MR2
4231 023324 004737 035344          ERROR 52 ;MSG A0 ERROR AFTER WRITE DATA CMD
4232 023330 104052          CLR SBMR3
4233 023332 005037 005446          JSR PC,CKMR3 ;CHECK MR3
4234 023336 004737 035426          ERROR 23 ;MSG B0 ERROR AFTER WRITE DATA CMD
4235 023342 104023
4236          MOV #CCLR,RKCS1(R5)
4237 023344 012765 100000 000000          MOV #1,RKMR1(R5) ;SELECT WORD 1
4238 023352 012765 000001 000026          JSR PC,GSTAT
4239 023360 004737 032526          MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2
4240 023364 012737 001720 005444          JSR PC,CKMR2 ;MSG A1 ERROR AFTER WRITE DATA CMD
4241 023372 004737 035344          CLR SBMR3
4242 023376 104053          JSR PC,CKMR3 ;MSG B1 ERROR AFTER WRITE DATA CMD
4243 023400 005037 005446          ERROR 25
4244 023404 004737 035426
4245 023410 104025
4246          MOV (RO),RKDC(R5)
4247          CLR RKDA(R5) ;HEAD/SECTOR
4248 023412 011065 000020          MOV #DPAT1,RKBA(R5)
4249 023416 005065 000006          BIS #BAI,RKCS2(R5)
4250 023422 012765 0C1534 000704          MOV #-6*22.*256.,RKWC(R5)
4251 023430 052765 000020 000010
4252 023436 012765 076000 000002
```

```

4253
4254 023444 012765 000031 000000      MOV      #(<WATCH>,RKCS1(R5)) ;WRITE CHECK CMD
4255 023452 013737 001444 005450      MOV      T5000,TEMP1 ;SETUP TIMEOUT
4256 023460 004737 032102      JSR      PC,FRDY ;FIND RDY
4257 023464 104015      ERROR    15 ;NO RDY AFTER WRITE CHECK CMD
4258
4259 023466 004737 032526      JSR      PC,GSTAT ;GET FRESH STATUS
4260 023472 032737 100000 005406      BIT      #CERR,HCS1
4261 023500 001417      BEQ      66$
4262
4263 023502 032737 040000 005410      BIT      #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4264 023510 001412      BEQ      65$
4265 023512 016537 000024 001506      MOV      RKDB(R5),WD1 ;ACTUAL WORD FOR TYPEOUT
4266 023520 013737 001534 001510      MOV      DPAT1,WD2 ;EXPECTED WORD FOR TYPEOUT
4267 023526 004737 033666      JSR      PC,TRUERR ;CHECK AGAINST BSE INFO
4268 023532 104016      ERROR    16 ;WCE AFTER WRITE CMD
4269 023534 000401      BR       66$
4270
4271 023536 104022      65$:     ERROR    22 ;CERR AFTER WRITE CHK CMD
4272
4273 023540      66$:
4274
4275 023540 012765 100000 000000      MOV      #CCLR,RKCS1(R5) ;CONTR CLEAR
4276 023546 004737 032526      JSR      PC,GSTAT ;GET LATEST STATUS
4277 023552 012737 010340 005444      MOV      #(<D.SPIN!D.DRDY!D.VV!D.DRA>,SBMR2) ;SHOULD BE VALUE
4278 023560 004737 035344      JSR      PC,CKMR2 ;CHECK MR2
4279 023564 104057      ERROR    57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4280 023566 005037 005446      CLR      SBMR3
4281 023572 004737 035426      JSR      PC,CKMR3 ;CHECK MR3
4282 023576 104031      ERROR    31 ;MSG B0 ERROR AFTER WRITE CHECK CMD
4283
4284 023600 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
4285 023606 012765 000001 000026      MOV      #1,RKMR1(R5) ;SELECT WORD 1
4286 023614 004737 032526      JSR      PC,GSTAT
4287 023620 012737 001720 005444      MOV      #(<D.SPJK!D.CART!D.DOOR!D.BRHM!D.TFOK>,SBMR2)
4288 023626 004737 035344      JSR      PC,CKMR2
4289 023632 104060      ERROR    60 ;MSG A1 ERROR AFTER WRITE CHECK CMD
4290 023634 005037 005446      CLR      SBMR3
4291 023640 004737 035426      JSR      PC,CKMR3
4292 023644 104032      ERROR    32 ;MSG B1 ERROR AFTER WRITE CHECK CMD
4293
4294
4295 023646 022027 000577      3$:     CMP      (R0)+,#383. ;ALL CYLINDERS DONE?
4296 023652 001402      BEQ      4$ ;BR IF YES
4297 023654 000137 023174      JMP      1$ ;ELSE REPEAT
4298
4299 023660      4$:
4300
4301 023660 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
4302 023666 013765 001222 000010      MOV      $UNIT,RKCS2(R5)
4303 023674 012765 000013 000000      MOV      #RECAL,RKCS1(R5) ;RECAL CMD
4304 ;RESET CYL DIFF/OFFSET & CYL ADDR REG
4305 ;IN RKMR2 & RKMR3 RESP.
4306 023702 013737 001432 005450      MOV      T10,TEMP1 ;SETUP TIMEOUT
4307 023710 004737 032102      JSR      PC,FRDY ;FIND RDY
4308 023714 104124      ERROR    124 ;RDY NOT SET AFTER RECAL CMD
    
```



```

4309
4310 023716 012765 000001 000026      MOV      #1,RKMR1(R5)      ;SELECT WORD 1
4311 023724 004737 032526          JSR      PC,GSTAT
4312 023730 032737 020000 005434      BIT      #D.RTZ,HMR2
4313 023736 001001          BNE      67$
4314 023740 104244          ERROR   244              ;RTZ NOT SET DURING RECAL CMD
4315 023742 013737 001432 005452 67$:      MOV      T10,TEMP2        ;SETUP TIMEOUT
4316 023750 004737 032320          JSR      PC,FATT1        ;FIND ATTN
4317 023754 104055          ERROR   55              ;NO ATTN AFTER RECAL CMD
4318
4319 023756 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
4320 023764 013765 001222 000010      MOV      #UNIT,RKCS2(R5) ;DRIVE#
4321 023772 012765 000005 000000      MOV      #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
4322 024000 013737 001432 005450      MOV      T10,TEMP1        ;SETUP TIMEOUT
4323 024004 004737 032102          JSR      PC,FRDY         ;FIND ROY
4324 024008 104151          ERROR   151            ;NO ROY AFTER DRIVE CLEAR CMD
4325 024014 004737 032266          JSR      PC,TSTATN      ;TEST FOR ATTN
4326 024020 000401          BR       68$
4327 024022 104154          ERROR   154            ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349

```

```

*****
:TEST 17      TEST HEAD SWITCHING TIME
:
:      TESTS THE ABILITY TO SWITCH HEADS IN LESS THEN 10MS WHEN HEADS SPIRAL.
:
:      1. SECTOR 17 IS FIRST LOCATED AND A WRITE DATA COMMAND OF 512 WORDS
:         TO SECTOR 21 IS ISSUED.
:      2. THE PROGRAM NOW KNOWS THAT THE DRIVE WILL NOT HAVE TO TRAVEL
:         A FULL REVOLUTION BEFORE FINDING SECTOR 21.
:      3. SINCE EACH SECTOR TAKES APPROX. 1.2MS, THE TIME BETWEEN
:         THE START OF THE WRITE COMMAND (FROM SECTOR 21, HEAD 0; TO
:         SECTOR 0, HEAD 1) AND CONTROLLER READY SHOULD BE APPROX 6MS
:
:      THE ABOVE IS REPEATED FOR HEAD SWITCHING BETWEEN 1 TO 2
:
:      THIS TEST IS BYPASSED IF NEITHER L OR P CLOCK IS PRESENT
:
*****

```

```

4350 024024 000004          TST17: SCOPE
4351 024026 012737 000001 001174      MOV      #1,STIMES      ;;DO 1 ITERATION
4352 024034 012706 001100          MOV      #STACK,SP
4353
4354 024040 005737 005520          TST      DOTIM          ;BYPASS THIS TEST IF
4355 024044 001001          BNE      1$            ;NEITHER L OR P CLOCK PRESENT
4356 024046 000461          BR       TST20         ;;GO TO NEXT TEST
4357
4358 024050 012737 000025 005460 1$:      MOV      #25,TEMPS      ;HEAD 0, SECTOR 21 TO BE PUT IN RKDA
4359 024056 004737 032570 2$:      JSR      PC,SUBCLR
4360 024062 104024          ERROR   24              ;CERR AFTER SCLR
4361
4362 024064 004737 033144          JSR      PC,FSEC17      ;FIND SECTOR 17
4363 024070 104116          ERROR   116            ;CANNOT FIND SECTOR 17
4364 024072 000447          BR       TST20         ;;GO TO NEXT TEST

```

```

4365
4366 024074 012765 001530 000004      MOV      #DATA01,RKBA(R5) ;DATA 0101
4367 024102 052765 000020 000010      BIS      #BAI,RKCS2(R5) ;BUSS ADDR INCREMENT INHIBIT
4368 024110 012765 177000 000002      MOV      #-512,RKWC(R5) ;WORD COUNT
4369 024116 013765 005460 000006      MOV      TEMPS,RKDA(R5) ;HEAD & SECTOR
4370 024124 012765 000023 000000      MOV      #WRDATA,RKCS1(R5) ;WRITE DATA CMD
4371
4372 024132 012737 001050 005450      MOV      #1050,TEMP1 ;THIS DELAY JUST OK FOR 11/50
4373 024140 004737 032474                JSR      PC,DLY ;DO DELAY
4374
4375 024144 032765 000200 000000 7$:      BIT      #RDY,RKCS1(R5) ;LOOK FOR CONTROLLER READY
4376 024152 001004                BNE      B$
4377 024154 004737 032102                JSR      PC,FRDY ;FIND RDY AND GET FRESH STATUS
4378 024160 104011                ERROR   11 ;NO RDY AFTER SEL DRV CMD
4379 024162 104107                ERROR   107 ;HEAD SWITCHING LONGER THAN DELAY
4380
4381 024164 004737 035310                JSR      PC,CLKOF ;TURN CLOCK INTR OFF.
4382
4383 024170 023727 005460 000425      CMP      TEMPS,#425 ;HEAD 1,SECTOR 21 DONE?
4384 024176 001405                BEQ     TST20 ;GO TO NEXT TEST
4385 024200 012737 000425 005460      MOV      #425,TEMP5
4386 024206 000137 024056                JMP     2$ ;ELSE REPEAT FOR HEAD 1, SECTOR 21
4387

```

```

*****
:TEST 20 DRIVE OFF TRACK TEST

```

```

* THIS TEST CHECKS FOR SERVO OSCILLATIONS DURING SETTLING TIME BEYOND
* THE ALLOTTED 3MS.

```

- * 1. INITIALLY, EVERY CYLINDER IS FORMATTED WITH IDENTICAL HEADERS (UNIQUE TO EACH CYLINDER)
- * 2. A FULL SECTOR WRITE COMMAND IS ISSUED BY A SINGLE CYL SEEK FROM 0 TO 1. AS HEADERS ARE IDENTICAL, THE NEXT SECTOR TO COME UNDER THE HEADS WILL IMMEDIATELY BE WRITTEN.
- * 3. IF THERE IS OSCILLATION SENSED BY READING THE TRIBITS, DRIVE OFF TRACK ERROR WILL SET.

```

* IN THIS MANNER OSCILLATING SEEKS ARE PERFORMED BETWEEN ALL MAJOR CYLINDERS.
* 100 OSCILLATIONS ARE PERFORMED AT EACH MAJOR CYLINDER
* BEFORE DOING THE NEXT CYLINDER

```

```

*****
TST20: SCOPE

```

```

4407 024212 000004      TST20: SCOPE
4408 024214 012737 000001 001174      MOV      #1,STIMES ;DO 1 ITERATION
4409 024222 012706 001100                MOV      #STACK,SP ;RESTORE STK PTR
4410 024226 005037 001372                CLR      TOCYL
4411 024232 012737 100000 005460      MOV      #BIT15,TEMP5
4412
4413 024240 004737 032570 1$:      JSR      PC,SUBCLR
4414 024244 104024                ERROR   24 ;CERR AFTER SCLR
4415
4416 024246 012700 001550                MOV      #HDTAB,RO ;FORMAT HEADERS ON ALL MAJOR CYL.
4417
4418 024252 013720 001372 2$:      MOV      TOCYL,(RO)+ ;HEADER WORD 0: CYL #
4419 024256 012720 140000                MOV      #140000,(RO)+ ;HEADER WORD 1: ALL SECTOR 0
4420 024262 012710 140000                MOV      #140000,(RO) ;HEADER WORD 2: XOR OF 0 & 1

```

```

4421 024266 053720 001372      BLS      TOCYL,(RC)+      ;ADD CYL # TO WORD 2
4422
4423 024272 020027 001754      CMP      RO,#HDTAB+132. ;ALL 22 SECTORS DONE? (22X6=132)
4424 024276 001365      BNE      25          ;BR IF NO
4425
4426 024300 012765 001550 000004      MOV      #HDTAB,RKBA(R5)
4427 024306 012765 177676 000002      MOV      #-66.,RKWC(R5)
4428 024314 013765 001372 000020      MOV      TOCYL,RKDC(R5)
4429
4430 024322 012765 000027 000000      MOV      #(<WRHEAD>,RKCSI(R5) ;WRITE HEADER CMD
4431 024330 013737 001444 005450      MOV      T50000,TEMP1 ;SETUP TIMEOUT
4432 024336 004737 032102      JSR      PC,FRDY ;FIND RDY
4433 024342 104200      EFOR     200 ;NO RDY AFTER WRITE HEADER CMD
4434 024344 004737 032526      JSR      PC,GSTAT ;GET FRESH STATUS
4435 024350 032737 100000 005406      BIT      #CERR,HCSI
4436 024356 001405      BEQ     64$
4437 024360 104201      ERROR   201 ;CERR AFTER WRITE HEADER CMD
4438 024362 104401 044510      TYPE    MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS
4439 024366 000137 025264      JMP     TIMING
4440
4441 024372      64$:
4442
4443 024372 006137 005460      ROL     TEMPS ;SET CARRY ONLY ONCE
4444 024376 006137 001372      ROL     TOCYL ;SELECT NEXT MAJOR CYL
4445 024402 023727 001372 001000      CMP     TOCYL,#1000 ;ALL MAJOR CYL FORMATTED?
4446 024410 001313      BNE     1$ ;BR IF NO
4447 024412 005065 000020      CLR     RKDC(R5) ;SETUP TO RETURN TO CYL 0
4448
4449 024416 012765 000017 000000      MOV     #SEEK,RKCSI(R5) ;SEEK CMD
4450 024424 013737 001434 005450      MOV     T50,TEMP1 ;SETUP TIMEOUT
4451 024432 004737 032102      JSR     PC,FRDY ;FIND RDY
4452 024436 104131      ERROR   131 ;NO RDY AFTER SEEK CMD
4453
4454 024440 013737 001444 005450      MOV     T50000,TEMP1 ;SETUP TIMEOUT
4455 024446 004737 032414      JSR     PC,FATT2 ;FIND ATTN
4456 024452 104132      ERROR   132 ;NO ATTN AFTER SEEK CMD
4457
4458 024454 032737 100000 005406      BIT     #CERR,HCSI
4459 024462 001401      BEQ     65$
4460 024464 104210      ERROR   210 ;CERR AFTER SEEK CMD
4461
4462 024466      65$:
4463
4464 024466 012737 025024 001176      MOV     #11$,SESCAPE
4465 024474 005000      CLR     RO ;ITERATION COUNTER
4466 024476 012737 000001 001372      MOV     #1,TOCYL ;SETUP TO CYL #
4467 024504 005037 001370      CLR     FRCYL
4468
4469 024510 104415      SCOP1
4470 024512 012706 001100      MOV     #STACK,SP ;RESTORE STK PTR
4471
4472 024516 013737 001372 001400      MOV     TOCYL,CALDIF ;SETUP FOR ERROR PRINTOUT
4473
4474 024524 004737 032570      JSR     PC,SUBCLR ;CERR AFTER SCLR
4475 024530 104024      ERROR   24
4476

```

4477	024532	013765	001372	000020		MOV	TOCYL,RKDC(R5)	;GO TO CYL #
4478	024540	012765	001532	000004		MOV	#DATA1,RKBA(R5)	;ALL 1'S
4479	024546	052765	000020	000010		BIS	#BAI,RKCS2(R5)	
4480	024554	012765	177400	000002		MOV	#-256.,RKWC(R5)	;SECTOR TO BE ALL 1'S
4481								
4482	024562	012765	000022	000000		MOV	#WRDATA,RKCS1(R5)	;WRITE DATA CMD
4483	024570	013737	001444	005450		MOV	T5000,TEMP1	
4484	024576	004737	032102			JSR	PC,FRDY	;FIND RDY
4485	024602	104011				ERROR	11	;NO RDY AFTER WRITE DATA CMD.
4486								
4487	024604	004737	032526			JSR	PC,GSTAT	;GET FRESH STATUS
4488	024610	032737	020000	005436		BIT	#D.DROT,HMR3	;SEE IF DRIVE OFF TRACK
4489	024616	001401				BEQ	5\$	
4490	024620	104112				ERROR	112	;DRIVE OFF TRACK AFTER WRITE DATA CMD
4491								
4492	024622	032737	100000	005406	5\$:	BIT	#CERR,HCS1	
4493	024630	001401				BEQ	6\$	
4494	024632	104012				ERROR	12	;CERR SET AFTER WRITE DATA CMD
4495								
4496	024634	004737	033074		6\$:	JSR	PC,RDCYLA	
4497	024640	023737	001404	001372		CMP	CYLADD,TOCYL	
4498	024646	001401				BEQ	7\$	
4499	024650	104113				ERROR	113	;CYL ADDR IN RKMR3 NOT = RKDC
4500								
4501	024652				7\$:	SCOP1		
4502	024652	104415				MOV	#STACK,SP	;RESTORE STK PTR
4503	024654	012706	001100					
4504								
4505	024660	004737	032570			JSR	PC,SUBCLR	
4506	024664	104024				ERROR	24	;CERR AFTER SCLR
4507								
4508								;RETURN TO CYL 0
4509	024666	012765	001532	000004		MOV	#DATA1,RKBA(R5)	
4510	024674	052765	000020	000010		BIS	#BAI,RKCS2(R5)	
4511	024702	012765	177400	000002		MOV	#-256.,RKWC(R5)	
4512								
4513	024710	012765	000023	000000		MOV	#WRDATA,RKCS1(R5)	
4514	024716	013737	001444	005450		MOV	T5000,TEMP1	
4515	024724	004737	032102			JSR	PC,FRDY	;FIND RDY
4516	024730	104011				ERROR	11	;NO RDY AFTER WRITE DATA CMD
4517								
4518	024732	004737	032526			JSR	PC,GSTAT	;GET FRESH STATUS
4519	024736	032737	020000	005436		BIT	#D.DROT,HMR3	
4520	024744	001401				BEQ	8\$	
4521	024746	104112				ERROR	112	;DRIVE OFF TRACK AFTER WRITE DATA CMD
4522								
4523	024750	032737	100000	005406	8\$:	BIT	#CERR,HCS1	
4524	024756	001401				BEQ	9\$	
4525	024760	104012				ERROR	12	;CERR AFTER WRITE DATA CMD
4526								
4527	024762	004737	033074		9\$:	JSR	PC,RDCYLA	
4528	024766	005737	001404			TST	CYLADD	
4529	024772	001401				BEQ	10\$	
4530	024774	104042				ERROR	42	;NOT BACK TO CYL 0
4531								
4532	024776	005200			10\$:	INC	RO	

4533	025000	020027	000144		CMP	RO, #100.	: ALL ITERATIONS DONE?
4534	025004	001247			BNE	35	: BR IF NO
4535							
4536	025006	005000			CLR	RO	: RESET ITERATION CTR
4537	025010	006337	001372		ASL	TOCYL	
4538	025014	023727	001372	001000	CMP	TOCYL, #1000	: ALL MAJOR CYL DONE?
4539	025022	001240			BNE	35	: BR IF NO
4540							
4541	025024	005037	001372		CLR	TOCYL	: RESTORE TO ORIG 22 SECTOR FORMAT.
4542	025030	012737	100000	005460	MOV	#BIT15, TEMPS	
4543							
4544	025036	004737	032570		JSR	PC, SUBCLR	
4545	025042	104024			ERROR	24	: CERR AFTER SCLR
4546							
4547	025044	012765	001550	000004	MOV	#HDTAB, RKBA(R5)	
4548	025052	012765	177676	000002	MOV	#-66., RKWC(R5)	
4549	025060	013765	001372	000020	MOV	TOCYL, RKDC(R5)	
4550							
4551	025066	013737	001372	001406	MOV	TOCYL, CALADD	: SETUP
4552	025074	012737	000000	001514	MOV	#0, HEAD	: TO FILL
4553	025102	012737	000000	001520	MOV	#0, FORMAT	: HEADER
4554	025110	004737	033356		JSR	PC, FHDTAB	: TABLE
4555							
4556							
4557	025114	012765	000027	000000	MOV	#<WRHEAD>, RKCS1(R5)	: WRITE HEADER CMD
4558	025122	013737	001444	005450	MOV	T50000, TEMP1	: SETUP TIMEOUT
4559	025130	004737	032102		JSR	PC, FRDY	: FIND RDY
4560	025134	104200			ERROR	200	: NO RDY AFTER WRITE HEADER CMD
4561	025136	004737	032526		JSR	PC, GSTAT	: GET FRESH STATUS
4562	025142	032737	100000	005406	BIT	#CERR, HCS1	
4563	025150	001405			BEQ	665	
4564	025152	104201			ERROR	201	: CERR AFTER WRITE HEADER CMD
4565	025154	104401	044510		TYPE	MSG26	: ABORTING DATA TESTS TO DO TIMING TESTS
4566	025160	000137	025264		JMP	TIMING	
4567	025164						
4568							
4569							
4570	025164	006137	005460		ROL	TEMPS	
4571	025170	006137	001372		ROL	TOCYL	
4572	025174	023727	001372	001000	CMP	TOCYL, #1000	: ALL MAJOR CYL REFORMATTED?
4573	025202	001315			BNE	125	: BR IF NO
4574							
4575	025204	005065	000020		CLR	RKDC(R5)	: SETUP TO RETURN TO CYL 0
4576	025210	005037	001176		CLR	\$ESCAPE	
4577							
4578	025214	012765	000017	000000	MOV	#SEEK, RKCS1(R5)	: SEEK CMD
4579	025222	013737	001434	005450	MOV	T50, TEMP1	: SETUP TIMEOUT
4580	025230	004737	032102		JSR	PC, FRDY	: FIND RDY
4581	025234	104131			ERROR	131	: NO RDY AFTER SEEK CMD
4582							
4583	025236	013737	001444	005450	MOV	T50000, TEMP1	: SETUP TIMEOUT
4584	025244	004737	032414		JSR	PC, FAT12	: FIND ATTN
4585	025250	104132			ERROR	132	: NO ATTN AFTER SEEK CMD
4586							
4587	025252	032737	100000	005406	BIT	#CERR, HCS1	
4588	025260	001401			BEQ	675	

4589 025262 104210 ERROR 210 ;CERR AFTER SEEK CMD

4590
4591 025264 675:

4592
4593
4594 .SBTTL SERVO & SPINDLE TIMING TESTS

4595
4596 025264 TIMING:

4597
4598
4599
4600

4601 *****
4602 *TEST 21 TIME BETWEEN OUTER LIMIT TO HEADS HOME DURING UNLOAD
4603 *
4604 * TIME IS MEASURED FROM ATTN ASSERTING (APPROXIMATES REV & OUTER LIMIT)
4605 * TO HEADS HOME ASSERTING. EXPECTED TIME APPROX 500MS
4606 *
4607 * ALL TIMING TESTS ARE BYPASSED IF NEITHER
4608 * L OR P CLOCK IS PRESENT & WILL BE INDICATED BY A MESSAGE
4609 *
4610 *****

4611 *****
4612 *TEST21: SCOPE
4613 *
4614 *
4615 *
4616 *
4617 *
4618 *
4619 *
4620 *
4621 *
4622 *
4623 *
4624 *
4625 *
4626 *
4627 *
4628 *
4629 *
4630 *
4631 *
4632 *
4633 *
4634 *
4635 *
4636 *
4637 *
4638 *
4639 *
4640 *
4641 *
4642 *
4643 *
4644 *

4611 025264 000004
4612 025266 012737 000001 001174
4613 025274 012706 001100

4614
4615 025300 005737 001362
4616 025304 001404
4617 025306 104401 046011
4618 025312 000137 031434

4619 025316
4620 025316 005737 005520
4621 025322 001004
4622 025324 104401 043546
4623 025330 000137 031434

4624
4625 025334 004737 032570
4626 025340 104024
4627
4628
4629 025342 012765 000017 000000
4630 025350 013737 001434 005450
4631 025356 004737 032102
4632 025362 104131

4633
4634 025364 013737 001444 005450
4635 025372 004737 032414
4636 025376 174132
4637
4638 025400 032737 100000 005406
4639 025406 001401
4640 025410 104210

4641
4642 025412 645:
4643
4644

4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700

4681 025300 005737 001362 TST BYPTIM ;SEE IF BYPASS TIMING TESTS
4682 025304 001404 BEQ TIME1 ;BR IF NO
4683 025306 104401 046011 TYPE MSG41 ;BYPASS TIMING TESTS
4684 025312 000137 031434 JMP \$EOP
4685
4686 TIME1: TST DOTIM
4687 025316 005737 005520 BNE IS
4688 025322 001004 TYPE MSG13 ;TIMING TESTS BYPASSED
4689 025324 104401 043546 JMP \$EOP
4690 025330 000137 031434
4691
4692 IS: JSR PC,SUBCLR
4693 025334 004737 032570 ERROR 24 ;CERR AFTER SCLR
4694 025340 104024
4695
4696
4697 MOV #SEEK,RKCS1(R5) ;SEEK CMD
4698 025342 012765 000017 000000 MOV T50,TEMP1 ;SETUP TIMEOUT
4699 025350 013737 001434 005450 JSR PC,FRDY ;FIND RDY
4700 025356 004737 032102 ERROR 131 ;NO RDY AFTER SEEK CMD

4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750

4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800

F09

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 88
 DZR6IB.P11 T21 TIME BETWEEN OUTER LIMIT TO HEADS HOME DURING UNLOAD

SEG 0089

4645	025412	004737	032570		JSR	PC, SUBCLR	
4646	025416	104024			ERROR	24	;CERR AFTER SCLR
4647							
4648	025420	004737	034240		JSR	PC, CALCLK	;CALIB TIME TO GO THRU 'LOOP'
4649							
4650	025424	012737	025612	001176	MOV	#4\$, \$ESCAPE	
4651	025432	012765	000007	000000	MOV	#UNLOAD, RKCS1(R5)	;UNLOAD CMD
4652	025440	013737	001432	005450	MOV	T10, TEMP1	;SETUP TIMEOUT
4653	025446	004737	032414		JSR	PC, FATT2	;FIND ATTN
4654	025452	104073			ERROR	73	;NO ATTN AFTER UNLOAD CMD
4655							
4656							
4657	025454	012765	100000	000000	MOV	#CCLR, RKCS1(R5)	
4658	025462	013765	001222	000010	MOV	\$UNIT, RKCS2(R5)	;DRIVE#
4659	025470	012765	000005	000000	MOV	#CLEAR, RKCS1(R5)	;DRIVE CLEAR CMD
4660	025476	013737	001432	005450	MOV	T10, TEMP1	;SETUP TIMEOUT
4661	025504	004737	032102		JSR	PC, FRDY	;FIND RDY
4662	025510	104151			ERROR	151	;NO RDY AFTER DRIVE CLEAR CMD
4663	025512	004737	032266		JSR	PC, TSTATN	;TEST FOR ATTN
4664	025516	000401			BR	65\$	
4665	025520	104154			ERROR	154	;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
4666	025522						
4667							
4668							
4669	025522	005037	001472		CLR	LPCNT	
4670	025526	012765	000001	000026	MOV	#1, RKMR1(R5)	;SELECT WORD 1
4671							
4672	025534	004737	032526		JSR	PC, GSTAT	
4673	025540	032737	000040	005434	BIT	#0, HOHM, HMR2	
4674	025546	001006			BNE	3\$;BR IF GOT HEAD HOME
4675	025550	004737	034672		JSR	PC, LOOP	;ELSE GO THRU LOOP
4676	025554	005737	001472		TST	LPCNT	;TEST FOR OVERFLOW
4677	025560	001365			BNE	2\$;BR IF NO
4678	025562	104066			ERROR	66	;NO HEAD HOME AFTER UNLOAD CMD
4679							
4680	025564	104401	043745		TYPE	, MSG18	;TIME BEING MEASURED
4681	025570	013737	001472	001160	MOV	(LPCNT, \$TMP0	;SETUP FOR MULT
4682	025576	004737	034750		JSR	PC, TYPTIM	;TYPE TIME IN USEC
4683	025602	104401	001205		TYPE	, \$CRLF	
4684	025606	104401	001205		TYPE	, \$CRLF	
4685							
4686	025612	005037	001176		4\$: CLR	\$ESCAPE	
4687							

```

*****
*TEST 22      TEST LOW VELOCITY TIMES DURING LOADING
*
*      THIS TEST ISSUES A START SPINDLE COMMAND
*      AFTER 'HEADS HOME' HAS BEEN DETECTED FROM THE PREVIOUS TEST.
*      THE FOLLOWING "LOW VELOCITY" TIMES ARE CHECKED AGAINST
*      LIMITS TO BE DEFINED:
*
*      TIME 1:  TIME BETWEEN HEADS HOME NEGATING & TRACK FOLL OK ASSERTING
*                EXPECTED TIME APPROX 500 MS
*
*      TIME 2:  TIME BETWEEN OUTER LIMIT & INNER LIMIT
*
*****
  
```

4700

```

4701          : *          TIME IS MEASURED FROM TRACK FOLL OK (APPROX OUTER LIMIT)
4702          : *          TO REV (APPROX INNER LIMIT)
4703          : *          EXPECTED TIME APPROX 2 SEC
4704          : *
4705          : *          TIME 3:  TIME BETWEEN INNER LIMIT & OUTER LIMIT
4706          : *
4707          : *          TIME IS MEASURED FROM REV ASSERTING (FROM ABOVE)
4708          : *          TO REV NEGATING (APPROX OUTER LIMIT)
4709          : *          EXPECTED TIME APPROX 2 SEC
4710          : *
4711          : *          *****
4712 025616 000004          : *          †ST22:  SCOPE
4713 025620 012737 000001 001174          : *          MOV      #1,STIMES          ;:DO 1 ITERATION
4714 025626 012706 001100          : *          MOV      #STACK,SP          ;RESTORE STK PTR
4715          : *
4716 025632 004737 032570          : *          JSR      PC,SUBCLR          ;
4717 025636 104024          : *          ERROR   24          ;CERR AFTER SCLR
4718          : *
4719 025640 012765 000011 000000          : *          MOV      #SRTSPL,RKCS1(R5) ;START SPINDLE CMD
4720 025646 013737 001432 005450          : *          MOV      T10,TEMP1          ;SETUP TIMEOUT
4721 025654 004737 032102          : *          JSR      PC,FRDY          ;FIND RDY
4722 025660 104121          : *          ERROR   121          ;RDY NOT SET AFTER START SPIN CMD
4723          : *
4724 025662 004737 034240          : *          JSR      PC,CALCLK          ;CALIB TIME TO GO THRU 'LOOP'
4725          : *
4726 025666 012737 026252 001176          : *          MOV      #95,$ESCAPE
4727          : *
4728 025674 013737 001410 001412          : *          MOV      HZ,COUNT
4729 025702 012737 000074 001414          : *          MOV      #60.,SEC
4730 025710 012765 000001 000026          : *          MOV      #1,RKMR1(R5)
4731 025716 004737 035202          : *          JSR      PC,CLKON          ;TURN CLOCK ON FOR 60 SEC TIMEOUT
4732 025722 004737 032526          : *          JSR      PC,GSTAT
4733 025726 032737 000040 005434          : *          BIT      #D.HOHM,HMR2
4734 025734 001406          : *          BEQ      25          ;BR IF HEAD HOME = 0
4735 025736 005737 001416          : *          TST      TIMUP
4736 025742 001767          : *          BEQ      15
4737 025744 004737 035310          : *          JSR      PC,CLKOF
4738 025750 104067          : *          ERROR   67          ;HEAD HOME NOT CLEARED DURING LOAD CMD
4739          : *
4740 025752 004737 035310          : *          JSR      PC,CLKOF          ;
4741 025756 005037 001472          : *          CLR      LPCNT          ;SETUP FOR TIME 1
4742 025762 012765 000001 000026          : *          MOV      #1,RKMR1(R5)          ;SELECT WORD 1
4743          : *
4744 025770 004737 032526          : *          JSR      PC,GSTAT
4745 025774 032737 000020 005434          : *          BIT      #D.TFOK,HMR2
4746 026002 001006          : *          BNE      45          ;BR IF GOT TFOK
4747 026004 004737 034672          : *          JSR      PC,LOOP          ;ELSE GO THRU LOOP
4748 026010 005737 001472          : *          TST      LPCNT          ;TEST FOR OVERFLOW
4749 026014 001365          : *          BNE      35          ;BR IF NO
4750 026016 104070          : *          ERROR   70          ;TRACK FOLL OK NOT SET DURING LOAD CMD
4751 026020 013737 001472 001462          : *          MOV      LPCNT,TIM1          ;STORE LOOP COUNT FOR TIME 1
4752          : *
4753          : *
4754 026026 005037 001472          : *          CLR      LPCNT          ;SETUP FOR TIME 2
4755 026032 012765 000001 000026          : *          MOV      #1,RKMR1(R5)
4756          : *

```



```

4757 026040 004737 032526 5$: JSR PC,GSTAT
4758 026044 032737 004000 005434 BIT #D,REV,HMR2
4759 026152 001006 BNE B$ ;BR IF GOT REV
4760 026154 004737 034672 JSR PC,LOOP
4761 026160 005737 001472 TST LPCNT ;TEST FOR OVERFLOW
4762 026154 001365 BNE B$ ;BR IF NO
4763 026066 104071 ERROR 71 ;REV NOT SET DURING LOAD
4764 026070 013737 001472 001464 6$: MOV LPCNT,TIM2 ;STORE LOOP COUNT FOR TIME 2
4765
4766
4767 026076 005037 001472 CLR LPCNT ;SETUP FOR TIME 3
4768 026102 012765 000001 000026 MOV #1,RKMR1(R5)
4769
4770 026110 004737 032526 7$: JSR PC,GSTAT
4771 026114 032737 004000 005434 BIT #D,REV,HMR2
4772 026122 001406 BEQ B$
4773 026124 004737 034672 JSR PC,LOOP
4774 026130 005737 001472 TST LPCNT
4775 026134 001365 BNE B$
4776 026136 104072 ERROR 72 ;REV NOT CLEARED DURING LOAD
4777 026140 013737 001472 001466 8$: MOV LPCNT,TIM3 ;STORE LOOP COUNT FOR TIME 3
4778
4779
4780 026146 104401 044076 TYPE MSG20 ;TIME 1 MEASUREMENT
4781 026152 013737 001462 081160 MOV TIM1,STMP0 ;SETUP FOR MULT
4782 026160 004737 034750 JSR PC,TYPTIM ;TYPE TIME IN USEC
4783
4784 026164 104401 044171 TYPE MSG21 ;TIME 2 MEASUREMENT
4785 026170 013737 001464 001160 MOV TIM2,STMP0
4786 026176 004737 034750 JSR PC,TYPTIM
4787
4788 026202 104401 044263 TYPE MSG22 ;TIME 3 MEASUREMENT
4789 026206 013737 001466 001160 MOV TIM3,STMP0
4790 026214 004737 034750 JSR PC,TYPTIM
4791
4792 026220 104401 001205 TYPE ,SCRLF
4793 026224 104401 001205 TYPE ,SCRLF
4794 026230 005037 001176 CLR $ESCAPE
4795
4796 026234 013737 001442 005450 MOV T5000,TEMP1
4797 026242 004737 032414 JSR PC,FATT2 ;FIND ATTN
4798 026246 104074 ERROR 74 ;NO ATTN AFTER START SPIN CMD
4799 026250 000412 BR TST23 ;GO TO NEXT TEST
4800
4801 026252 005037 001176 9$: CLR $ESCAPE
4802 026256 013737 001440 005452 MOV T100,TEMP2
4803 026264 004737 022320 JSR PC,FATT1 ;FIND ATTN
4804 026270 104074 ERROR 74 ;NO ATTN AFTER START SPIN CMD
4805 026272 000137 031434 JMP $EOP ;ABORT DRIVE

```

```

4806
4807 *****
4808 *TEST 23 MEASURE ROTATIONAL SPEED
4809 *
4810 * THIS TEST MEASURES INDEX TIMING BY:
4811 *
4812 * A. CHANGE FORMAT TO 20 SECTOR & READ HEADER.

```

```

4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825 026276 000004
4826 026300 012737 000001 001174
4827 026306 012706 001100
4828
4829 026312 104401 046050
4830 026316 000137 030520
4831
4832 026322 004737 032570
4833 026326 104024
4834
4835 026330 004737 034240
4836
4837 026334 005000
4838 026336 005037 001476
4839 026342 005037 001500
4840
4841 026346 012765 010025 000000 15:
4842 026354 013737 001444 005450
4843 026362 004737 032102
4844 026366 104171
4845
4846 026370 005700
4847 026372 001007
4848
4849 026374 004737 032650
4850 026400 005737 001426
4851 026404 001402
4852 026406 104064
4853
4854 026410 000464
4855
4856 026412 005037 001472
4857 026416 012765 000025 000000 45:
4858 026424 013737 001444 005450
4859 026432 032765 000200 000000 25:
4860 026440 001007
4861 026442 004737 034672
4862 026446 005337 005450
4863 026452 001367
4864 026454 104171
4865 026456 000441
4866
4867 026460 004737 032650 35:
4868 026464 005737 001426

```

```

:*
:* CONTROLLER RDY STARTS THE TIMER
:* B. CHANGE FORMAT TO 22 SECTOR & READ HEADER.
:* CONTROLLER RDY ENDS THE TIMER.
:*
:* INDEX TIMING IS THE TIME BETWEEN THE 2 CONT. RDY TIMES.
:* THIS IS BECAUSE A CHANGE OF FORMAT INHIBITS SECTOR PULSES
:* UNTIL THE NEXT INDEX APPEARS--THUS KEEPING A GIVEN
:* FORMAT COMPLETE THROUGHOUT AN ENTIRE CYLINDER
:*
:* THE TIME IS THE AVERAGE OF 100 READINGS.

```

```

*****
TST23: SCOPE
MOV #1, $TIMES ;:DO 1 ITERATION
MOV #STACK, SP ;RESTORE STK PTR
TYPE ,MSG42 ;BYPASSING NEXT 4 TESTS
JMP VELTST ;GO TO VELOCITY TEST
JSR PC, SUBCLR
ERROR 24 ;CERR AFTER SCLR
JSR PC, CALCLK ;CALIB TIME TO GO THRU 'LOOP'
CLR RO ;ITERATION COUNTER
CLR SUM ;LO WORD OF REV TIME SUM
CLR SUM+2 ;HI WORD OF REV TIME SUM
15: MOV #<CFMT!RDHEAD>, RKCS1(R5) ;CHANGE TO 20 SECTOR FMT & READ HEADER
MOV T50000, TEMP1 ;SETUP TIMEOUT
JSR PC, FRDY ;FIND RDY
ERROR 171 ;NO RDY AFTER READ HDR CMD
TST RO ;TEST FOR SECTOR 0 AFTER FORMAT CHANGE
BNE 45 ;DO ONLY ONCE
JSR PC, RDSEC
TST SECTOR
BEQ 45
ERROR 64 ;CANT FIND SECTOR 0 FROM INDEX
;AFTER FORMAT CHANGE & RDY REC'D
BR TST24 ;GOTO NEXT TEST
45: CLR LPCNT ;COUNT PASSES THRU LOOP
MOV #RDHEAD, RKCS1(R5) ;CHANGE TO 22 SECTOR FMT & READ HEADER
MOV T50000, TEMP1 ;SETUP TIMEOUT
25: BIT #RDY, RKCS1(R5)
BNE 35 ;EXIT IF GOT RDY
JSR PC, LOOP ;ELSE GO THRU LOOP
DEC TEMP1
BNE 25
ERROR 171 ;NO RDY AFTER READ HDR CMD
BR TST24 ;GO TO NEXT TST
35: JSR PC, RDSEC
TST SECTOR

```

```

4869 026470 001402          BEQ    SS
4870 026472 104064          ERROR  64          ;CANT FIND SECTOR 0 FROM INDEX
4871                                ;AFTER FORMAT CHANGE & RDY REC'D
4872 026474 000432          BR     TST24          ;GOTO NEXT TEST
4873
4874 026476 013702 001472    SS:    MOV    LPCNT,R2          ;FROM _OOP
4875 026502 013703 001474    MOV    LPTIM,R3          ;FROM CALCLK
4876 026506 010246          MOV    R2,-(SP)          ;PUT THE MULTIPLIER ON THE STACK
4877 026510 010346          MOV    R3,-(SP)          ;PUT THE MULTIPLICAND ON THE STACK
4878 026512 004737 041714    JSR    PC,@#MULT          ;CALL THE MULTIPLY ROUTINE
4879 026516 012616          MOV    (SP)+,(SP)        ;DISREGARD THE MSB'S
4880 026520 012603          MOV    (SP)+,R3          ;GET THE LSB'S OF THE PRODUCT
4881
4882 026522 060337 001476    ADD    R3,SUM            ;R3 CONTAINS LOW ORDER PRODUCT
4883 026526 005537 001500    ADC    SUM+2
4884
4885 026532 005200          INC    R0
4886 026534 020027 000144    CMP    R0,#100.
4887 026540 001302          BNE   IS
4888
4889 026542 104401 044442    TYPE   ,MSG24            ;AVG ROTATIONAL TIME
4890
4891 026546 004737 035016    JSR    PC,AVGTIM          ;CALC & TYPEOUT AVG TIME
4892 026552 104401 001205    TYPE   ,$CALF
4893 026556 104401 001205    TYPE   ,$CALF
4894
4895          ;*****
4896          ;*TEST 24          MEASURE MAX SEEK TIME
4897          ;*
4898          ;* THIS TEST MEASURES THE MAX SEEK TIME BETWEEN CYLINDERS 0 & 410
4899          ;* THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
4900          ;* IF NOT WITHIN LIMITS TO BE SUPPLIED.
4901          ;* MAX SEEK TIME SHOULD BE LESS THAN 70MS.
4902          ;*
4903          ;*****
4904 026562 000004          TST24: SCOPE
4905 026564 012737 000001 001174    MOV    #1,$TIMES          ;DO 1 ITERATION
4906 026572 012706 001100    MOV    #STACK,SP
4907
4908
4909 026576 004737 034240    JSR    PC,CALCLK          ;CALIB TIME TO GO THRU 'LOOP'
4910
4911 026602 005000          CLR    R0                ;ITERATION COUNTER
4912 026604 005037 001476    CLR    SUM                ;LO WORD OF FOWARD SEEK TIME
4913 026610 005037 001500    CLR    SUM+2              ;HI WORD OF FOWARD SEEK TIME
4914 026614 005037 001502    CLR    SUM1                ;LO WORD OF REVERSE SEEK TIME
4915 026620 005037 001504    CLR    SUM1+2              ;HI WORD OF REVERSE SEEK TIME
4916
4917 026624 004737 032570    IS:    JSR    PC,SUBCLR
4918 026630 104024          ERROR  24                ;CERR AFTER SCLR
4919
4920 026632 012737 027124 001176    MOV    #7,$$ESCAPE
4921
4922 026640 012765 000632 000020    2S:    MOV    #410.,RKDC(R5)    ;SETUP FOR CYL 410.
4923 026646 012765 000017 000000    MOV    #SEEK,RKCS1(R5)   ;SEEK CMD
4924 026654 013737 001434 005450    MOV    T50,TEMP1

```

4925	026662	004737	032102			JSR	PC,FRDY	;FIND RDY
4926	026666	104131				ERROR	131	;NO RDY AFTER SEEK CMD
4927								
4928	026670	005037	001472			CLR	LPCNT	;COUNT PASSES THRU LOOP
4929	026674	013704	001222			MOV	SUNIT,R4	
4930	026700	004737	034716			JSR	PC,FATT3	;FIND ATTN AND STEP 'LPCNT'
4931	026704	104132				ERROR	132	;NO ATTN AFTER SEEK CMD
4932								
4933	026706	004737	032526			JSR	PC,GSTAT	
4934	026712	032737	100000	005406		BIT	#CFRR,HCS1	
4935	026720	001401				BEQ	55	
4936	026722	104210				ERROR	210	;CERR AFTER SEEK CMD.
4937								
4938	026724	013737	001472	001160	55:	MOV	LPCNT,\$TMP0	;FROM LOOP
4939	026732	013737	001474	001162		MOV	LPTIM,\$TMP1	;FROM CALCLK
4940	026740	013746	001160			MOV	\$TMP0,-(SP)	::PUT THE MULTIPLIER ON THE STACK
4941	026744	013746	001162			MOV	\$TMP1,-(SP)	::PUT THE MULTIPLICAND ON THE STACK
4942	026750	004737	041714			JSR	PC,\$MULT	::CALL THE MULTIPLY ROUTINE
4943	026754	012637	001162			MOV	(SP)+,\$TMP1	::GET THE LSB'S OF THE PRODUCT
4944	026760	012637	001164			MOV	(SP)+,\$TMP1+2	::GET THE MSB'S OF THE PRODUCT
4945								
4946	026764	005765	000020			TST	RKDC(R5)	
4947	026770	001414				BEQ	65	;BR IF THIS SEEK WAS REVERSE TO CYL 0
4948								
4949	026772	063737	001162	001476		ADD	\$TMP1,SUM	;SUM UP FOWARD SEEK TIMES (LSB)
4950	027000	005537	001500			ADC	SUM+2	
4951	027004	063737	001164	001500		ADD	\$TMP2,SUM+2	;MSB
4952								
4953	027012	004737	032570			JSR	PC,SUBCLR	
4954	027016	104024				ERROR	24	;CERR AFTER SCLR
4955	027020	000712				BR	25	;SEEK TO CYL 0
4956								
4957	027022	063737	001162	001502	65:	ADD	\$TMP1,SUM1	;SUM UP REVERSE SEEK TIMES (LSB)
4958	027030	005537	001504			ADC	SUM1+2	
4959	027034	063737	001164	001504		ADD	\$TMP2,SUM1+2	;MSB
4960								
4961	027042	005200				INC	RD	
4962	027044	020027	000144			CMP	RD,#100.	;ALL SEEKS DONE?
4963	027050	001265				BNE	15	;BR & REPEAT IF NO TO CYL 410.
4964								
4965	027052	104401	044622			TYPE	,MSG28	;FOWARD SEEK TIME
4966	027056	004737	035016			JSR	PC,AVGTIM	;CALC & TYPE AVG TIME (FOWARD)
4967								
4968	027062	104401	044701			TYPE	,MSG29	;REVERSE SEEK TIME
4969	027066	013737	001502	001476		MOV	\$SUM1,SUM	;SETUP FOR
4970	027074	013737	001504	001500		MOV	\$SUM1+2,SUM+2	;AVGTIM ROUTINE
4971	027102	004737	035016			JSR	PC,AVGTIM	;CALC & TYPE AVG TIME (REVERSE)
4972								
4973	027106	104401	001205			TYPE	,\$SCLF	
4974	027112	104401	001205			TYPE	,\$SCLF	
4975	027116	005037	001176			CLR	\$ESCAPE	
4976	027122	000464				BR	TST25	::GO TO NEXT TEST
4977								
4978	027124	005037	001176		75:	CLR	\$ESCAPE	
4979								
4980	027130	012765	100000	000000		MOV	#CCLR,RKCS1(R5)	

```

4981 027136 013765 001222 000010      MOV      $UNIT,RKCS2(R5)
4982 027144 012765 000013 000000      MOV      #RECAL,RKCS1(R5)          ;RECAL CMD
4983                                     ;RESET CYL DIFF/OFFSET & CYL ADDR REG
4984                                     ;IN RKMR2 & RKMR3 RESP.
4985 027152 013737 001432 005450      MOV      T10,TEMP1              ;SETUP TIMEOUT
4986 027160 004737 032102      JSR      PC,FRDY                ;FIND RDY
4987 027164 104124      ERROR   124                    ;RDY NOT SET AFTER RECAL CMD
4988
4989 027166 012765 000001 000026      MOV      #1,RKMR1(R5)          ;SELECT WORD 1
4990 027174 004737 032526      JSR      PC,GSTAT
4991 027200 032737 020000 005434      BIT      #D.RTZ,HMR2
4992 027206 001001      BNE     64$
4993 027210 104244      ERROR   244                    ;RTZ NOT SET DURING RECAL CMD
4994 027212 013737 001432 005452 64$:      MOV      T10,TEMP2              ;SETUP TIMEOUT
4995 027220 004737 032320      JSR      PC,FATT1              ;FIND ATTN
4996 027224 104055      ERROR   55                    ;NO ATTN AFTER RECAL CMD
4997
4998 027226 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
4999 027234 013765 001222 000010      MOV      $UNIT,RKCS2(R5)      ;DRIVE#
5000 027242 012765 000005 000000      MOV      #CLEAR,RKCS1(R5)    ;DRIVE CLEAR CMD
5001 027250 013737 001432 005450      MOV      T10,TEMP1              ;SETUP TIMEOUT
5002 027256 004737 032102      JSR      PC,FRDY                ;FIND RDY
5003 027262 104151      ERROR   151                    ;NO RDY AFTER DRIVE CLEAR CMD
5004 027264 004737 032266      JSR      PC,TSTATN            ;TEST FOR ATTN
5005 027270 000401      BR      65$
5006 027272 104154      ERROR   154                    ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021 027274 000004      ST25: SCOPE
5022 027276 012737 000001 001174      MOV      #1,$TIMES            ;;DO 1 ITERATION
5023 027304 012706 001100      MOV      #STACK,SP
5024
5025
5026 027310 004737 034240      JSR      PC,CALCLK            ;CALIB TIME TO GO THRU 'LOOP'
5027
5028 027314 005000      CLR     RO                    ;ITERATION COUNTER
5029 027316 005037 001476      CLR     SUM                   ;LO WORD OF FOWARD SEEK TIME
5030 027322 005037 001500      CLR     SUM+2                 ;HI WORD OF FOWARD SEEK TIME
5031 027326 005037 001502      CLR     SUM1                  ;LO WORD OF REVERSE SEEK TIME
5032 027332 005037 001504      CLR     SUM1+2                ;HI WORD OF REVERSE SEEK TIME
5033
5034 027336 004737 032570      JSR      PC,SUBCLR
5035 027342 104024      ERROR   24                    ;CERR AFTER SCLR
5036

```

```

*****
;TEST 25      MEASURE MIN SEEK TIME
;
; THIS TEST MEASURES THE MIN SEEK TIME BETWEEN CYLINDER 0 & 1
; THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
; IF NOT WITHIN LIMITS TO BE SUPPLIED.
; MIN SEEK TIME SHOULD BE LESS THAN 10MS.
*****

```

```

*****
;ST25: SCOPE
MOV      #1,$TIMES            ;;DO 1 ITERATION
MOV      #STACK,SP
;
JSR      PC,CALCLK            ;CALIB TIME TO GO THRU 'LOOP'
;
CLR     RO                    ;ITERATION COUNTER
CLR     SUM                   ;LO WORD OF FOWARD SEEK TIME
CLR     SUM+2                 ;HI WORD OF FOWARD SEEK TIME
CLR     SUM1                  ;LO WORD OF REVERSE SEEK TIME
CLR     SUM1+2                ;HI WORD OF REVERSE SEEK TIME
;
JSR      PC,SUBCLR
ERROR   24                    ;CERR AFTER SCLR

```

M09

UNIBUS RKO6 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 95
 DZR6IB.P11 T25 MEASURE MIN SEEK TIME

SEQ 0096

5037	027344	012737	027636	001176		MOV	#75, \$ESCAPE	
5038								
5039	027352	012765	000301	000020		MOV	#1, RKDC(R5)	; SETUP FOR CYL 1
5040	027360	012765	000317	000000	25:	MOV	#SEEK, RKCS1(R5)	; SEEK CMD
5041	027366	013737	001434	005450		MOV	T50, TMP1	
5042	027374	004737	032102			JSR	PC, FROD	; FIND RDY
5043	027400	104131				ERROR	131	; NO RDY AFTER SEEK CMD
5044								
5045	027402	005037	001472			CLR	LPCNT	; COUNT PASSES THRU LOOP
5046	027406	013704	001222			MOV	\$UNIT, R4	
5047	027412	004737	034716			JSR	PC, FATT3	; FIND ATTN AND STEP 'LPCNT'
5048	027416	104132				ERROR	132	; NO ATTN AFTER SEEK CMD
5049								
5050	027420	004737	032526			JSR	PC, GSTAT	
5051	027424	032737	100000	005406		BIT	#CERR, HCS1	
5052	027432	001401				BEQ	\$S	
5053	027434	104210				ERROR	210	; CERR AFTER SEEK CMD.
5054								
5055	027436	013737	001472	001160	55:	MOV	LPCNT, \$TMP0	; FROM LOOP
5056	027444	013737	001474	001162		MOV	LPTIM, \$TMP1	; FROM CALCLK
5057	027452	013746	001160			MOV	\$TMP0, -(SP)	;; PUT THE MULTIPLIER ON THE STACK
5058	027456	013746	001162			MOV	\$TMP1, -(SP)	;; PUT THE MULTIPLICAND ON THE STACK
5059	027462	004737	041714			JSR	PC, \$MULT	;; CALL THE MULTIPLY ROUTINE
5060	027466	012637	001162			MOV	(SP)+, \$TMP1	;; GET THE LSB'S OF THE PRODUCT
5061	027472	012637	001164			MOV	(SP)+, \$TMP1+2	;; GET THE MSB'S OF THE PRODUCT
5062								
5063	027476	005765	000020			TST	RKDC(R5)	
5064	027502	001414				BEQ	\$S	; BR IF THIS SEEK WAS REVERSE TO CYL 0
5065								
5066	027504	063737	001162	001476		ADD	\$TMP1, SUM	; SUM UP FOWARD SEEK TIMES (LSB)
5067	027512	005537	001500			ADC	SUM+2	
5068	027516	063737	001164	001500		ADD	\$TMP2, SUM+2	; MSB
5069								
5070	027524	004737	032570			JSR	PC, SUBCLR	
5071	027530	104024				ERROR	24	; CERR AFTER SCLR
5072	027532	000712				BR	\$S	; SEEK TO CYL 0
5073								
5074	027534	063737	001162	001502	65:	ADD	\$TMP1, SUM1	; SUM UP REVERSE SEEK TIMES (LSB)
5075	027542	005537	001504			ADC	SUM1+2	
5076	027546	063737	001164	001504		ADD	\$TMP2, SUM1+2	; MSB
5077								
5078	027554	005200				INC	R0	
5079	027556	020027	000144			CMP	R0, #100.	; ALL SEEKS DONE?
5080	027562	001265				BNE	\$S	; BR & REPEAT IF NO TO CYL 1
5081								
5082	027564	104401	044761			TYPE	MSG30	; FOWARD SEEK TIME
5083	027570	004737	035016			JSR	PC, AVGTIM	; CALC & TYPE AVG TIME (FOWARD)
5084								
5085	027574	104401	045047			TYPE	MSG31	; REVERSE SEEK TIME
5086	027600	013737	001502	001476		MOV	SUM1, SUM	; SETUP FOR
5087	027606	013737	001504	001500		MOV	SUM1+2, SUM+2	; AVGTIM ROUTINE
5088	027614	004737	035016			JSR	PC, AVGTIM	; CALC & TYPE AVG TIME (REVERSE)
5089								
5090	027620	104401	001205			TYPE	\$SCLF	
5091	027624	104401	001205			TYPE	\$SCLF	
5092	027630	005037	001176			CLR	\$ESCAPE	

```

5093 027634 000464          BR      TST26          ;;GO TO NEXT TEST
5094
5095 027636 005037 001176      7$:   CLR      $ESCAPE
5096
5097 027642 012765 100000 000000    MOV      #CCLR,RKCS1(R5)
5098 027650 013765 001222 000010    MOV      $UNIT,RKCS2(R5)
5099 027656 012765 000013 000000    MOV      #RECAL,RKCS1(R5)          ;RECAL CMD
5100                                     ;RESET CYL DIFF/OFFSET & CYL ADDR PEG
5101                                     ;IN RKMR2 & RKMR3 RESP.
5102 027664 013737 001432 005450    MOV      T10,TEMP1
5103 027672 004737 032102          JSR      PC,FRDY
5104 027676 104124          ERROR    124          ;FIND RDY
5105                                     ;RDY NOT SET AFTER RECAL CMD
5106 027700 012765 000001 000026    MOV      #1,RKMR1(R5)          ;SELECT WORD 1
5107 027706 004737 032526          JSR      PC,GSTAT
5108 027712 032737 020000 005434    BIT      #D.RTZ,HMR2
5109 027720 001001          BNE      64$
5110          ERROR    244
5111 027724 013737 001432 005452    64$:  MOV      T10,TEMP2          ;RTZ NOT SET DURING RECAL CMD
5112 027732 004737 032320          JSR      PC,FATT1          ;SETUP TIMEOUT
5113 027736 104055          ERROR    55          ;FIND ATTN
5114                                     ;NO ATTN AFTER RECAL CMD
5115 027740 012765 100000 000000    MOV      #CCLR,RKCS1(R5)
5116 027746 013765 001222 000010    MOV      $UNIT,RKCS2(R5)          ;DRIVE#
5117 027754 012765 000005 000000    MOV      #CLEAR,RKCS1(R5)          ;DRIVE CLEAR CMD
5118 027762 013737 001432 005450    MOV      T10,TEMP1
5119 027770 004737 032102          JSR      PC,FRDY          ;SETUP TIMEOUT
5120 027774 104151          ERROR    151          ;FIND RDY
5121 027776 004737 032266          JSR      PC,TSTATN          ;NO RDY AFTER DRIVE CLEAR CMD
5122 030002 000401          BR      65$          ;TEST FOR ATTN
5123 030004 104154          ERROR    154          ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5124 030006          65$:
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138 030006 000004          TST26: SCOPE
5139 030010 012737 000001 001174    MOV      #1,$TIMES          ;;DO 1 ITERATION
5140 030016 012706 001100          MOV      #STACK,SP
5141
5142
5143 030022 004737 034240          JSR      PC,CALCLK          ;CALIB TIME TO GO THRU 'LOOP'
5144
5145 030026 005000          CLR      RO          ;ITERATION COUNTER
5146 030030 005037 001476          CLR      SUM          ;LO WORD OF FOWARD SEEK TIME
5147 030034 005037 001500          CLR      SUM+2          ;HI WORD OF FOWARD SEEK TIME
5148 030040 005037 001502          CLR      SUM1          ;LO WORD OF REVERSE SEEK TIME

```

```

*****
:TEST 26          MEASURE 137 CYLINDER SEEK TIME
:

```

```

* THIS TEST MEASURES THE AVERAGE SEEK TIME BETWEEN CYLINDERS 0 & 137
* THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
* IF NOT WITHIN LIMITS TO BE SUPPLIED.
* AVERAGE SEEK TIME SHOULD BE LESS THAN 40MS
*

```

```

*****

```

5149	030044	005037	001504		CLR	SUM1+2		;HI WORD OF REVERSE SEEK TIME
5150	030050	004737	032570	15:	JSR	PC,SUBCLR		
5151	030054	104024			ERROR	24		;CERR AFTER SCLR
5152								
5153								
5154	030056	012737	030350	001176	MOV	#75,SESCAPE		
5155								
5156	030064	012765	000211	000020	MOV	#137,RKDC(R5)		;SETUP FOR CYL 137.
5157	030072	012765	000017	000000	MOV	#SEEK,RKCS1(R5)		;SEEK CMD
5158	030100	013737	001434	005450	MOV	T50,TEMP1		
5159	030106	004737	032102		JSR	PC,FRDY		;FIND RDY
5160	030112	104131			ERROR	131		;NO RDY AFTER SEEK CMD
5161								
5162	030114	005037	001472		CLR	LPCNT		;COUNT PASSES THRU LOOP
5163	030120	013704	001222		MOV	SUNIT,R4		
5164	030124	004737	034716		JSR	PC,FAIT3		;FIND ATTN AND STEP 'LPCNT'
5165	030130	104132			ERROR	132		;NO ATTN AFTER SEEK CMD
5166								
5167	030132	004737	032526		JSR	PC,GSTAT		
5168	030136	032737	100000	005406	BIT	#CERR,HCS1		
5169	030144	001401			BEG	55		
5170	030146	104210			ERROR	210		;CERR AFTER SEEK CMD.
5171								
5172	030150	013737	001472	001160	MOV	LPCNT,STMP0		;FROM LOOP
5173	030156	013737	001474	001162	MOV	LPTIM,STMP1		;FROM CALCLK
5174	030164	013746	001160		MOV	STMP0,-(SP)		::PUT THE MULTIPLIER ON THE STACK
5175	030170	013746	001162		MOV	STMP1,-(SP)		::PUT THE MULTIPLICAND ON THE STACK
5176	030174	004737	041714		JSR	PC,#MULT		::CALL THE MULTIPLY ROUTINE
5177	030200	012637	001162		MOV	(SP)+,STMP1		::GET THE LSB'S OF THE PRODUCT
5178	030204	012637	001164		MOV	(SP)+,STMP1+2		::GET THE MSB'S OF THE PRODUCT
5179								
5180	030210	005765	000020		TST	RKDC(R5)		
5181	030214	001414			BEG	65		;BR IF THIS SEEK WAS REVERSE TO CYL 0
5182								
5183	030216	063737	001162	001476	ADD	STMP1,SUM		;SUM UP FOWARD SEEK TIMES (LSB)
5184	030224	005537	001500		ADC	SUM+2		
5185	030230	063737	001164	001500	ADD	STMP2,SUM+2		;MSB
5186								
5187	030236	004737	032570		JSR	PC,SUBCLR		
5188	030242	104024			ERROR	24		;CERR AFTER SCLR
5189	030244	000712			BR	25		;SEEK TO CYL 0
5190								
5191	030246	063737	001162	001502	ADD	STMP1,SUM1		;SUM UP REVERSE SEEK TIMES (LSB)
5192	030254	005537	001504		ADC	SUM1+2		
5193	030260	063737	001164	001504	ADD	STMP2,SUM1+2		;MSB
5194								
5195	030266	005200			INC	R0		
5196	030270	020027	000144		CMP	R0,#100		;ALL SEEKS DONE?
5197	030274	001265			BNE	15		;BR & REPEAT IF NO TO CYL 137.
5198								
5199	030276	104401	045136		TYPE	MSG32		;FOWARD SEEK TIME
5200	030302	004737	035016		JSR	PC,AVGTIM		;CALC & TYPE AVG TIME (FOWARD)
5201								
5202	030306	104401	045215		TYPE	MSG33		;REVERSE SEEK TIME
5203	030312	013737	001502	001476	MOV	SUM1,SUM		;SETUP FOR
5204	030320	013737	001504	001500	MOV	SUM1+2,SUM+2		;AVGTIM ROUTINE


```

5205 030326 004737 035016 JSR PC,AVGTIM ;CALC & TYPE AVG TIME (REVERSE)
5206
5207 030332 104401 001205 TYPE .SCRLF
5208 030336 104401 001205 TYPE .SCRLF
5209 030342 005037 001176 CLR $ESCAPE
5210 030346 000464 BR TST27 ;;GO TO NEXT TEST
5211
5212 030350 005037 001176 7$: CLR $ESCAPE
5213
5214 030354 012765 100000 000000 MOV #CLR,RKCS1(R5)
5215 030362 013765 001222 000010 MOV $UNIT,RKCS2(R5)
5216 030370 012765 000013 000000 MOV #RECAL,RKCS1(R5) ;RECAL CMD
5217 ;RESET CYL DIFF/OFFSET & CYL ADDR REG
5218 ;IN RKMR2 & RKMR3 RESP.
5219 030376 013737 001432 005450 MOV T10,TEMP1 ;SETUP TIMEOUT
5220 030404 004737 032102 JSR PC,FRDY ;FIND RDY
5221 030410 104124 ERROR 124 ;RDY NOT SET AFTER RECAL CMD
5222
5223 030412 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
5224 030420 004737 032526 JSR PC,GSTAT
5225 030424 032737 020000 005434 BIT #D.RTZ,HMR2
5226 030432 001001 BNE 64$
5227 030434 104244 ERROR 244 ;RTZ NOT SET DURING RECAL CMD
5228 030436 013737 001432 005452 64$: MOV T10,TEMP2 ;SETUP TIMEOUT
5229 030444 004737 032320 JSR PC,FATT1 ;FIND ATTN
5230 030450 104055 ERROR 55 ;NO ATTN AFTER RECAL CMD
5231
5232 030452 012765 100000 000000 MOV #CLR,RKCS1(R5)
5233 030460 013765 001222 000010 MOV $UNIT,RKCS2(R5) ;DRIVE#
5234 030466 012765 000005 000000 MOV #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
5235 030474 013737 001432 005450 MOV T10,TEMP1 ;SETUP TIMEOUT
5236 030502 004737 032102 JSR PC,FRDY ;FIND RDY
5237 030506 104151 ERROR 151 ;NO RDY AFTER DRIVE CLEAR CMD
5238 030510 004737 032266 JSR PC,TSTATN ;TEST FOR ATTN
5239 030514 000401 BR 65$
5240 030516 104154 ERROR 154 ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5241
5242 65$:
5243
5244
5245 030520
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256 030520 000004
5257 030522 012737 000001 001174
5258 030530 012706 001100
5259
5260 030534 005000

```

VELTST:
:*****
:TEST 27 MEASURE MAX VELOCITY OF HEADS
:
: THIS TESTS MAX VELOCITY BY DOING SEEKS BETWEEN
: CYL 0 & 383 AND MEASURING THE TIME BETWEEN CYLINDERS
: 128 & 256. SINCE THE DISTANCE BETWEEN CYL 128 & 256 IS KNOWN,
: THE AVERAGE VELOCITY OF 100 SEEKS IS CALCULATED & TYPED
: IF NOT WITHIN THE SPECIFIED LIMITS TO BE SUPPLIED.
:*****
↑TST27: SCOPE
MOV #1,\$TIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR
CLR RO ;ITERATION COUNTER

5261	030536	005037	001476		CLR	SUM		;LO WORD OF FOWARD SEEK TIME
5262	030542	005037	001500		CLR	SUM+2		;HI WORD OF FOWARD SEEK TIME
5263	030546	005037	001502		CLR	SUM1		;LO WORD OF REVERSE SEEK TIME
5264	030552	005037	001504		CLR	SUM1+2		;HI WORD OF REVERSE SEEK TIME
5265								
5266	030556	004737	034456		JSR	PC,VELCAL		;CLOCK CALIB SIMILAR TO CALCLK
5267								
5268	030562	004737	032570	15:	JSR	PC,SUBCLR		
5269	030566	104024			ERROR	24		;CERR AFTER SCLR
5270								
5271	030570	012737	031264	001176	MOV	#14\$, \$ESCAPE		
5272								
5273	030576	012765	000577	000020	MOV	#383.,RKDC(R5)		
5274	030604	012765	030017	000000	25:	MOV	#SEEK,RKCS1(R5)	;SEEK CMD TO CYL 400
5275	030612	013737	001434	005450	MOV	T50,TEMP1		
5276	030620	004737	032102		JSR	PC,FRDY		;FIND RDY
5277	030624	104131			ERROR	131		;NO RDY AFTER SEEK CMD
5278								
5279	030626	013737	001436	005450	MOV	T500,TEMP1		
5280	030634	004737	033004		645:	JSR	PC,QKCYLD	;READ CYL DIFF (NO SHIFTING)
5281	030640	032737	004000	001402	BIT	#BIT11,CYLDIF		;CHECK BIT 7 (NO SHIFTING)
5282	030646	001004			BNE	655		;BR IF SET
5283	030650	005337	005450		DEC	TEMP1		
5284	030654	001367			BNE	645		
5285	030656	104110			ERROR	110		;CANNOT FIND CYL 128
5286								
5287	030660	005037	001472		655:	CLR	LPCNT	
5288	030664	004737	033004		665:	JSR	PC,QKCYLD	
5289	030670	032737	004000	001402	BIT	#BIT11,CYLDIF		;CHECK BIT 7 (NO SHIFTING)
5290	030676	001406			BEQ	675		
5291								
5292	030700	004737	034672		JSR	PC,LOOP		
5293	030704	005737	001472		TST	LPCNT		
5294	030710	001365			BNE	665		
5295	030712	104111			ERROR	111		;CANNOT FIND CYL 256
5296	030714				675:			
5297	030714	013737	001444	005450	MOV	T50000,TEMP1		
5298	030722	004737	032414		JSR	PC,FATT2		;FIND ATTN
5299	030726	104132			ERROR	132		;NO ATTN AFTER SEEK CMD
5300								
5301	030730	032737	100000	005406	BIT	#CERR,HCS1		
5302	030736	001401			BEQ	75		
5303	030740	104210			ERROR	210		;CERR AFTER SEEK CMD
5304								
5305	030742	013702	001472		75:	MOV	LPCNT,R2	;FROM LOOP
5306	030746	013703	001474		MOV	LPTIM,R3		;FROM VELCAL
5307	030752	010246			MOV	R2,-(SP)		:::PUT THE MULTIPLIER ON THE STACK
5308	030754	010346			MOV	R3,-(SP)		:::PUT THE MULTIPLICAND ON THE STACK
5309	030756	004737	041714		JSR	PC,@\$MULT		:::CALL THE MULTIPLY ROUTINE
5310	030762	012616			MOV	(SP)+,(SP)		:::DISREGARD THE MSB'S
5311	030764	012603			MOV	(SP)+,R3		:::GET THE LSB'S OF THE PRODUCT
5312	030766	060337	001476		ADD	R3,SUM		;SUM UP FOWARD SEEK TIMES
5313	030772	005537	001500		ADC	SUM+2		
5314								
5315	030776	004737	032570		JSR	PC,SUBCLR		
5316	031002	104024			ERROR	24		;CERR AFTER SCLR

E10

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 100
 DZR6IB.P11 T27 MEASURE MAX VELOCITY OF HEADS

SEQ 0101

```

5317
5318 031004 012765 000017 000000      MOV      #SEEK,RKCS1(R5) ;SEEK TO CYL 0
5319 031012 013737 001434 005450      MOV      T50,TEMP1
5320 031020 004737 032102      JSR      PC,FROY        ;FIND RDY
5321 031024 104131      ERROR    131          ;NO RDY AFTER SEEK CMD
5322
5323 031026 013737 001436 005450      MOV      T500,TEMP1
5324 031034 004737 033004      JSR      PC,GKCYLD      ;READ CYL DIFF (NO SHIFTING)
5325 031040 032737 004000 001402 68$:      BIT      #BIT11,CYLDIF  ;CHECK BIT 7 (NO SHIFTING)
5326 031046 001004      BNE      69$          ;BR IF SET
5327 031050 005337 005450      DEC      TEMP1
5328 031054 001367      BNE      68$
5329 031056 104111      ERROR    111          ;CANNOT FIND CYL 256
5330
5331 031060 005037 001472      CLR      LPCNT        69$:
5332 031064 004737 033004      JSR      PC,GKCYLD      70$:
5333 031070 032737 004000 001402      BIT      #BIT11,CYLDIF  ;CHECK BIT 7 (NO SHIFTING)
5334 031076 001406      BEQ      71$
5335
5336 031100 004737 034672      JSR      PC,LOOP
5337 031104 005737 001472      TST      LPCNT
5338 031110 001365      BNE      70$
5339 031112 104110      ERROR    110          ;CANNOT FIND CYL 128
5340 031114
5341 031114 013737 001444 005450 71$:      MOV      T50000,TEMP1
5342 031122 004737 032414      JSR      PC,FATT2
5343 031126 104132      ERROR    132          ;FIND ATTN
5344
5345 031130 032737 100000 0 5406      BIT      #CERR,HCS1    ;NO ATTN AFTER SEEK CMD
5346 031136 001401      BEQ      12$
5347 031140 104210      ERROR    210          ;CERR AFTER SEEK CMD
5348
5349 031142 013702 001472      MOV      LPCNT,R2      12$:
5350 031146 013703 001474      MOV      LPTIM,R3
5351 031152 010246      MOV      R2,-(SP)      ;;PUT THE MULTIPLIER ON THE STACK
5352 031154 010346      MOV      R3,-(SP)      ;;PUT THE MULTIPLICAND ON THE STACK
5353 031156 004737 041714      JSR      PC,#SMULT     ;;CALL THE MULTIPLY ROUTINE
5354 031162 012616      MOV      (SP)+,(SP)    ;;DISREGARD THE MSB'S
5355 031164 012603      MOV      (SP)+,R3      ;;GET THE LSB'S OF THE PRODUCT
5356 031166 060337 001502      ADD      R3,SUM1
5357 031172 005537 001504      ADC      SUM1+2
5358
5359 031176 005200      INC      R0
5360 031200 020027 000144      CMP      R0,#100.     ;DONE 100 SEEKS?
5361 031204 001402      BEQ      13$          ;BR IF YES
5362 031206 000137 030562      JMP      1$           ;ELSE DO AGAIN
5363
5364 031212 104401 045275      TYPE     MSG34        13$:
5365 031216 004737 035056      JSR      PC,AVGSP     ;AVG MAX FWD SPEED
5366
5367 031222 104401 045363      TYPE     MSG35        ;AVG MAX REV SPEED
5368 031226 013737 001502 001476      MOV      SUM1,SUM
5369 031234 013737 001504 001500      MOV      SUM1+2,SUM+2 ;SETUP FOR
5370 031242 004737 035056      JSR      PC,AVGSP     ;AVGSP ROUTINE
5371
5372 031246 104401 001205      TYPE     ,$CRLF

```

F10

5373	031252	104401	001205		TYPE	\$CRLF	
5374	031256	005037	001176		CLR	\$ESCAPE	
5375	031262	000464			BR	\$EOP	
5376							
5377	031264	005037	001176	14\$:	CLR	\$ESCAPE	
5378							
5379	031270	012765	100000	000000	MOV	\$CCLR,RKCS1(R5)	
5380	031276	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	
5381	031304	012765	000013	000000	MOV	\$RECAL,RKCS1(R5)	
5382							;RECAL CMD
5383							;RESET CYL DIFF/OFFSET & CYL ADDR REG
5384	031312	013737	001432	005450	MOV	T10,TEMP1	;IN RKMR2 & RKMR3 RESP.
5385	031320	004737	032102		JSR	PC,FRDY	;SETUP TIMEOUT
5386	031324	104124			ERROR	124	;FIND RDY
5387							;RDY NOT SET AFTER RECAL CMD
5388	031326	012765	000001	000026	MOV	\$1,RKMR1(R5)	;SELECT WORD 1
5389	031334	004737	032526		JSR	PC,GSTAT	
5390	031340	032737	020000	005434	BIT	\$D,RTZ,HMR2	
5391	031346	001001			BNE	72\$	
5392	031350	104244			ERROR	244	;RTZ NOT SET DURING RECAL CMD
5393	031352	013737	001432	005452	72\$:	MOV	T10,TEMP2
5394	031360	004737	032320		JSR	PC,FATT1	;SETUP TIMEOUT
5395	031364	104055			ERROR	55	;FIND ATTN
5396							;NO ATTN AFTER RECAL CMD
5397	031366	012765	100000	000000	MOV	\$CCLR,RKCS1(R5)	
5398	031374	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	;DRIVE#
5399	031402	012765	000005	000000	MOV	\$CLEAR,RKCS1(R5)	;DRIVE CLEAR CMD
5400	031410	013737	001432	005450	MOV	T10,TEMP1	;SETUP TIMEOUT
5401	031416	004737	032102		JSR	PC,FRDY	;FIND RDY
5402	031422	104151			ERROR	151	;NO RDY AFTER DRIVE CLEAR CMD
5403	031424	004737	032266		JSR	PC,TSTATN	;TEST FOR ATTN
5404	031430	000401			BR	73\$	
5405	031432	104154			ERROR	154	;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5406	031434			73\$:			
5407							
5408							
5409							
5410							

```

5411
5412
5413
5414
5415
5416
5417
5418
5419 031434
5420
5421 031434 000004
5422 031436 012706 001100
5423 031442 005237 001220
5424 031446 023737 005472 001220
5425 031454 001403
5426 031456 000137 012754
5427 031462 000004
5428 031464 005037 001102
5429 031470 005037 001174
5430 031474 005237 001216
5431 031500 042737 100000 001216
5432 031506 005327
5433 031510 000001
5434 031512 003022
5435 031514 012737
5436 031516 000001
5437 031520 031510
5438 031522 104401 031567
5439 031526 013746 001216
5440 031532 104405
5441 031534 104401 031564
5442 031540 013700 000042
5443 031544 001405
5444 031546 000005
5445 031550 004710
5446 031552 000240
5447 031554 000240
5448 031556 000240
5449 031560
5450 031560 000137
5451 031562 011474
5452 031564 377 377 000
5453 031567 015 042412 042116
5454 031574 050040 051501 020123
5455 031602 000043

```

```

.SBTTL END OF PASS ROUTINE
;*****
; *INCREMENT THE PASS NUMBER ($PASS)
; *TYPE "END PASS #XXXXX" WHERE XXXXX IS A DECIMAL NUMBER)
; *IF THERES A MONITOR GO TO IT
; *IF THERE ISN'T JUMP TO ST5
$EOP:
SCOPE
MOV #STACK, SP
INC $DEVCT ; INCR COUNT FOR # DRIVES CHECKED
CMP DRIVS, $DEVCT ; ARE ALL DRIVES PRESENT TESTED?
BEQ $EOP1+2 ; BR IF YES
JMP NUDRV ; ELSE TEST NEXT DRIVE PRESENT
$EOP1:
CLR $STNM ; ZERO THE TEST NUMBER
CLR $TIMES ; ZERO THE NUMBER OF ITERATIONS
INC $PASS ; INCREMENT THE PASS NUMBER
BIC #100000, $PASS ; DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ; LOOP?
$EOPCT: .WORD 1
BGT $DOAGN ; YES
MOV (PC)+, 2(PC)+ ; RESTORE COUNTER
$ENDCT: .WORD 1
TYPE $ENDMG ; TYPE "END PASS #"
MOV $PASS, -(SP) ; SAVE $PASS FOR TYPEOUT
TYPDS ; GO TYPE--DECIMAL ASCII WITH SIGN
TYPE $ENULL ; TYPE A NULL CHARACTER
$GET42: MOV #42, R0 ; GET MONITOR ADDRESS
BEQ $DOAGN ; BRANCH IF NO MONITOR
RESET ; CLEAR THE WORLD
$ENDAD: JSR PC, (R0) ; GO TO MONITOR
NOP ; SAVE ROOM
NOP ; FOR
NOP ; ACT11
$DOAGN: JMP 2(PC)+ ; RETURN
$RTNAD: .WORD ST5
$ENULL: .BYTE -1, -1, 0 ; NULL CHARACTER STRING
$ENDMG: .ASCIIZ <15><12>/END PASS #/

```



```

5512 031756 000761          BR      75          ;FOUND TERMINATOR, EXIT
5513
5514 031760 104401 046221    4$:   TVOC      EMI          ; ONLY 0-7 ALLOWED.
5515 031764 000137 010724    JMP      PRGSRT      ; START ALL OVER
5516
5517 031770 005201          5$:   INC      R1          ; S/B NO MORE THAN 8 DIFF
5518 031772 001330          BNE     1$          ; DRIVES TYPED IN.
5519 031774 000771          BR      4$          ; IF NORE, HAVE ERROR.
5520
5521 031776 005237 005522    6$:   INC      SIZFLG     ; DO TEST 1 (SIZING)
5522 032002 000207          RTS     PC          ; EXIT.
5523
5524
5525 ;ROUTINE TO INPUT RKBAS OR DEFALLT.
5526 ;
5527
5528 032004 104412          GBA:   RDOCT
5529 032006 012600          MOV     (SP)+,R0     ; GET LOW ORDER FROM STACK
5530 032010 005700          TST    R0
5531 032012 001403          BEQ    1$          ; BRANCH IF DEFAULT.
5532 032014 010037 001264    MOV     R0,$BASE
5533 032020 000207          RTS     PC
5534 032022 012737 177440 001264 1$:   MOV     #177440,$BASE ; DEFAULT VALUE
5535 032030 000207          RTS     PC
5536
5537
5538 ;ROUTINE TO INPUT RKVEC OR DEFAULT
5539 ;
5540
5541 032032 104412          GINT:  RDOCT
5542 032034 012600          MOV     (SP)+,R0     ; GET LOW ORDER FROM STACK
5543 032036 005700          TST    R0
5544 032040 001405          BEQ    1$          ; BRANCH IF DEFAULT
5545 032042 010037 001334    MOV     R0,RKVEC
5546 032046 004737 032064    JSR    PC,SETINT
5547 032052 000207          RTS     PC
5548 032054 012737 000210 001334 1$:   MOV     #210,RKVEC   ; DEFAULT VALUE
5549 032062 000771          BR      2$
5550
5551
5552 ;ROUTINE TO SETUP INTERRUPT VECTOR & PRIORITY
5553 ;
5554
5555 032064 013700 001334    SETINT: MOV     RKVEC,R0
5556 032070 012720 036046    MOV     #INTER,(R0)+ ; INTER ADDR TO RKVEC
5557 032074 013710 001336    MOV     RKPRI,(R0)  ; PRS TO RKVEC+2
5558 032100 000207          RTS     PC
5559
5560
5561
5562 ;ROUTINE TO FIND CONTROLLER READY (RDY) DURING A DELAY
5563 ;ENTER WITH A COUNT IN TEMP1
5564 ;RETURN IF RDY NOT PRESENT (ERROR CONDITION)
5565 ;RETURN +2 IF RDY PRESENT (SKIP OVER ERROR)
5566 ;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
5567 ;

```

```

5568 032102 032765 000200 000000 FRDY: BIT #RDY,RKCS1(R5)
5569 032110 001006 BNE IS
5570 032112 005337 005450 DEC TEMP1
5571 032116 001371 BNE FRDY
5572 032120 004737 032140 JSR PC,HOLD ;STORE ALL RK611 REGS IN HOLDING REGS.
5573 032124 000207 RTS PC ;NO RDY, EXIT
5574 032126 062716 000002 IS: ADD #2,(SP) ;SKIP OVER ERROR
5575 032132 004737 032140 JSR PC,HOLD
5576 032136 000207 RTS PC
5577
5578
5579 ;STORE ALL RK611 REGISTERS IN HOLDING REGS
5580
5581
5582 032140 016537 000000 005406 HOLD: MOV RKCS1(R5),HCS1
5583 032146 016537 000010 005410 MOV RKCS2(R5),HCS2
5584 032154 016537 000002 005412 MOV RKWC(R5),HWC
5585 032162 016537 000004 005414 MOV RKBA(R5),HBA
5586 032170 016537 000006 005416 MOV RKDA(R5),HDA
5587 032176 016537 000012 005420 MOV RKDS(R5),HDS
5588 032204 016537 000014 005422 MOV RKER(R5),HER
5589 032212 016537 000016 005424 MOV RKASOF(R5),HASOF
5590 032220 016537 000020 005426 MOV RKDC(R5),HDC
5591 032226 016537 000026 005432 MOV RKMR1(R5),HMR1
5592 032234 016537 000034 005434 MOV RKMR2(R5),HMR2
5593 032242 016537 000036 005436 MOV RKMR3(R5),HMR3
5594 032250 016537 000030 005440 MOV RKECPS(R5),HPOS
5595 032256 016537 000032 005442 MOV RKECPT(R5),HPAT
5596 032264 000207 RTS PC
5597
5598
5599 ;ROUTINE TO CHECK FOR CORRECT ATTN
5600 ;RETURN IF ATTN NOT PRESENT (ERROR CONDITION)
5601 ;RETURN +2 IF ATTN PRESENT (SKIP OVER ERROR)
5602
5603 032266 010446 STATN: MOV R4,-(SP) ;SAV R4
5604 032270 013704 MOV $UNIT,R4
5605 032274 136437 005376 005425 BITB ATTN(R4),HASOF+1
5606 032302 001404 BEQ IS ;BRANCH IF ATTN NOT PRESENT
5607 032304 012604 MOV (SP)+,R4 ;RESTOR R4
5608 032306 062716 000002 ADD #2,(SP) ;INCR RET ADDR TO JUMP OVER ERROR.
5609 032312 000207 RTS PC
5610 032314 012604 IS: MOV (SP)+,R4 ;RESTOR R4
5611 032316 000207 RTS PC
5612
5613
5614 ;ROUTINE TO FIND ATTN WITHIN TIMES GREATER THAN 1 SEC
5615 ;ENTER WITH TIME IN SECONDS IN TEMP2
5616 ;RETURN IF NO ATTN (ERROR CONDITION)
5617 ;RETURN +2 IF ATTN FOUND
5618 ;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
5619
5620
5621 032320 010446 FATT1: MOV R4,-(SP) ;SAV R4
5622 032322 012737 177777 005450 IS: MOV #1,TEMP1
5623 032330 013704 001222 MOV $UNIT,R4
    
```


K10

```

5624 032334 136465 005376 000017 1$: BITB ATTN(R4),RKASOF+1(R5) ;FIND CORRECT ATTN
5625 032342 001014 BNE 2$
5626 032344 005337 005450 DEC TEMP1
5627 032350 001371 BNE 1$
5628 032352 005337 005452 DEC TEMP2
5629 032356 001361 BNE 3$
5630 032360 005065 000026 CLR RKMR1(R5) ;SELECT WORD 0
5631 032364 004737 032526 JSR PC,GSTAT ;GET LATEST STATUS
5632 032370 012604 MOV (SP)+,R4 ;RESTOR R4
5633 032372 000207 RTS PC
5634 032374 005065 000026 2$: CLR RKMR1(R5)
5635 032400 004737 032526 JSR PC,GSTAT ;GET STATUS AFTER ATTN SEEN
5636 032404 012604 MOV (SP)+,R4 ;RESTOR R4
5637 032406 062716 000002 ADD #2,(SP) ;SKIP OVER ERROR
5638 032412 000207 RTS PC
  
```

```

;ROUTINE TO FIND ATTN WITHIN 1 SEC
;ENTER WITH COUNT IN TEMP1
;RETURN IF NO ATTN (ERROR)
;RETURN +2 IF ATTN FOUND
;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
  
```

```

5648 032414 010446 FATT2: MOV R4,-(SP) ;SAV R4
5649 032416 013704 001222 2$: MOV $UNIT,R4
5650 032422 136465 005376 000017 BITB ATTN(R4),RKASOF+1(R5) ;FIND CORRECT ATTN
5651 032430 001011 BNE 1$
5652 032432 005337 005450 DEC TEMP1
5653 032436 001367 BNE 2$
5654 032440 005065 000026 CLR RKMR1(R5) ;SELECT WORD 0
5655 032444 004737 032526 JSR PC,GSTAT ;GET LATEST STATUS.
5656 032450 012604 MOV (SP)+,R4 ;RESTOR R4
5657 032452 000207 RTS PC
5658 032454 005065 000026 1$: CLR RKMR1(R5)
5659 032460 004737 032526 JSR PC,GSTAT
5660 032464 012604 MOV (SP)+,R4 ;RESTOR R4
5661 032466 062716 000002 ADD #2,(SP) ;SKIP OVER ERROR
5662 032472 000207 RTS PC
  
```

```

;ENTER WITH A COUNT IN TEMP1
;THE DELAY IS APPROX 17 US/ITERATION + 12 US TO EXIT
;WHEN COUNT IS 0. BASED ON AN 11/05
  
```

```

5668 032474 005737 005450 DLY: TST TEMP1 ;5.6 US
5669 032500 001403 BEQ 1$ ;2.5 US
5670 032502 005337 005450 DEC TEMP1 ;6.8 US
5671 032506 000772 BR DLY ;2.5 US
5672 032510 000207 1$: RTS PC ;3.8 US
  
```

```

;THIS ROUTINE TYPES BYPASSED DRIVE#. ENTER WITH DRIVE# IN R0
  
```

```

5677 032512 104401 043634 BYP: TYPE MSG14 ;BYPASS DRIVE
5678 032516 010046 MOV R0,-(SP) ;SAVE R0 FOR TYPEOUT
5679 ;TYPE DR#
  
```

5680 032520 104403
5681 032522 001
5682 032523 000
5683 032524 00C207
5684
5685
5686
5687
5688
5689
5690

TYPOS
.BYTE 1
.BYTE 0
RTS PC

::GO TYPE--OCTAL ASCII
::TYPE 1 DIGIT(S)
::SUPPRESS LEADING ZEROS

:: THIS ROUTINE DOES THE SELECT DRIVE COMMAND TO GET STATUS
:: IT THEN WAITS FOR CONTROLLER READY
:: IF RDY NOT RECEIVED BY THE TIMEOUT, AN ERROR IS FLAGGED
::

5691 032526 013746 005450
5692 032532 013765 001222 000010
5693 032540 012765 000001 000000
5694 032546 013737 001432 005450
5695 032554 004737 032102
5696 032560 104117
5697 032562 012637 005450
5698 032566 000207
5699
5700

GSTAT: MOV TEMP1, -(SP) ;SAVE TEMP1
MOV #UNIT, RKCS2(R5) ;CURRENT DRIVE #
MOV #SELDRV, RKCS1(R5) ;GET STATUS WITH SELECT DRIVE CMD
MOV T10, TEMP1
JSR PC, FRDY ;FIND RDY
ERROR 117 ;RDY NOT SET BY END OF SELECT DRIVE CMD
MOV (SP)+, TEMP1 ;RESTOR TEMP1.
RTS PC

5701
5702
5703
5704
5705
5706

:: THIS ROUTINE DOES A SUBSYSTEM CLEAR & WAITS FOR CONTROLLER READY
:: IF RDY IS NOT RECEIVED BY THE END OF THE TIMEOUT, AN ERROR IS FLAGGED.
:: THE ROUTINE THEN GETS CURRENT STATUS & CHECKS FOR CONTROLLER ERROR (CERR)
:: RETURN IF CERR SET
:: RETURN +2 IF CERR CLEAR
::

5707 032570 012765 000040 000010
5708 032576 013737 001432 005450
5709 032604 004737 032102
5710 032610 104120
5711 032612 013765 001222 000010
5712 032620 005065 000026
5713 032624 004737 032526
5714 032630 032737 100000 005406
5715 032636 001401
5716 032640 000207
5717 032642 062716 000002
5718 032646 000207
5719
5720

SUBCLR: MOV #SCLR, RKCS2(R5) ;SUBSYS CLEAR
MOV T10, TEMP1
JSR PC, FRDY ;FIND RDY
ERROR 120 ;RDY NOT SET BY END OF SCLR
MOV #UNIT, RKCS2(R5) ;CURRENT DRIVE #
CLR RKMR1(R5) ;SELECT WORD 0
JSR PC, GSTAT ;GET STATUS
BIT #CERR, HCS1 ;CHECK FOR CONT ERROR
BEQ 15
RTS PC
15: ADD #2, (SP) ;SKIP OVER ERROR
RTS PC

5721
5722
5723 032650 012765 000003 000026
5724 032656 004737 032526
5725 032662 013737 005436 001426
5726 032670 042737 177017 001426
5727 032676 006237 001426
5728 032702 006237 001426
5729 032706 006237 001426
5730 032712 006237 001426
5731 032716 000207
5732

:: READ THE SECTOR COUNT IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'SECTOR'
::
RDSEC: MOV #3, RKMR1(R5) ;WORD 3
JSR PC, GSTAT
MOV HMR3, SECTOR
BIC #C<M. SECT>, SECTOR
ASR SECTOR ;RIGHT JUSTIFY
ASR SECTOR ;SECTOR
ASR SECTOR ;INFO
ASR SECTOR
RTS PC

5733
5734
5735 032720 012765 000002 000026

:: READ THE CYL DIFF/OFFSET IN RKMR2, RIGHT JUSTIFY IT & STORE IT IN 'CYLDIF'
::
RD CYLD: MOV #2, RKMR1(R5) ;WORD 2

M10

```

5736 032726 004737 032526            JSR    PC,GSTAT
5737 032732 013737 005434 001402        MOV    HMR2,CYLDIF
5738 032740 042737 160017 001402        BIC    #1C<M.CDIF>,CYLDIF
5739 032746 006237 001402            ASR    CYLDIF            ;RIGHT JUSTIFY
5740 032752 006237 001402            ASR    CYLDIF            ;CYL DIFF/OFFSET
5741 032756 006237 001402            ASR    CYLDIF            ;INFO
5742 032762 006237 001402            ASR    CYLDIF
5743 032766 023727 001402 000777        CMP    CYLDIF,#777        ;CHK TO SEE IF RET IN COMPL. FORM
5744 032774 001002                    BNE    1$                ;BR IF NOT
5745 032776 005037 001402            CLR    CYLDIF            ;CLR IF YES
5746 033002 000207                    1$:    RTS                PC
5747
5748                                    ;
5749                                    ;QUICK SELECT DRIVE COMMAND TO OBTAIN CYL DIFF
5750
5751 033004 013746 005450            QKCLD: MOV    TEMP1,-(SP)        ;SAVE TEMP1
5752 033010 012765 000002 000026        MOV    #2,RKMR1(R5)       ;SELECT WORD 2
5753 033016 012765 000001 000000        MOV    #SELDRV,RKCS1(R5)   ;SELECT DRIVE CMD
5754 033024 013737 001432 005450        MOV    T10,TEMP1
5755 033032 032765 000200 000000        1$:    BIT    #RDY,RKCS1(R5)     ;TEST FOR CONT RDY
5756 033040 001004                    BNE    2$                ;BR IF THERE
5757 033042 005337 005450            DEC    TEMP1
5758 033046 001371                    BNE    1$
5759 033050 104117                    ERROR  117               ;NO RDY AFTER SEL DRV CMD
5760
5761 033052 016537 000034 001402        2$:    MOV    RKMR2(R5),CYLDIF
5762 033060 042737 160017 001402        BIC    #1C<M.CDIF>,CYLDIF   ;GET CYL DIFF ONLY (NO SHIFTING)
5763 033066 012637 005450            MOV    (SP)+,TEMP1       ;RESTORE TEMP1
5764 033072 000207                    RTS    PC
5765
5766                                    ;
5767                                    ;READ THE CYL ADDR IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'CYLADD'
5768
5768 033074 012765 000002 000026        RDCYLA: MOV    #2,RKMR1(R5)     ;WORD 2
5769 033102 004737 032526            JSR    PC,GSTAT
5770 033106 013737 005436 001404        MOV    HMR3,CYLADD
5771 033114 042737 160017 001404        BIC    #1C<M.CADD>,CYLADD
5772 033122 006237 001404            ASR    CYLADD            ;RIGHT JUSTIFY
5773 033126 006237 001404            ASR    CYLADD            ;CYL ADDR
5774 033132 006237 001404            ASR    CYLADD            ;INFO
5775 033136 006237 001404            ASR    CYLADD
5776 033142 000207                    RTS    PC
5777
5778                                    ;
5779                                    ;FIND SECTOR 17
5780                                    ;RETURN IF NOT FOUND
5781                                    ;RETURN +4 IF FOUND
5782
5782 033144 013737 001442 005450        FSEC17: MOV    T5000,TEMP1     ;SETUP TIMEOUT
5783 033152 004737 032650            1$:    JSR    PC,RDSEC        ;READ SECTOR
5784 033156 023727 001426 000021        CMP    SECTOR,#17.       ;TEST FOR SECTOR 17
5785 033164 001014                    BNE    2$                ;BR IF NOT 17
5786
5787 033166 004737 032650            JSR    PC,RDSEC
5788 033172 023727 001426 000021        CMP    SECTOR,#17.
5789 033200 001412                    BEQ    3$                ;BR IF READ SAME TWICE
5790 033202 004737 032650            JSR    PC,RDSEC        ;ELSE TRY 1 MORE TIME
5791 033206 023727 001426 000021        CMP    SECTOR,#17.
  
```

```

5792 033214 001404          B&Q      3$          ;BR IF 17
5793
5794 033216 005337 005450  2$:      DEC      TEMP1
5795 033222 001353          BNE      1$          ;TRY AGAIN
5796 033224 000207          RTS      PC
5797
5798 033226 062716 000004  3$:      ADD      #4,(SP) ;SKIP OVER ERROR
5799 033232 000207          RTS      PC
5800
5801 ;ROUTINE TO FIND HEADS HOME IN RKMP2 WORD 1 BEFORE SPECIFIED DELAY
5802 ;ENTER WITH TIME IN SECONDS IN TEMP2
5803 ;RETURN IF NOT FOUND
5804 ;RETURN+2 IF FOUND - SKIP OVER ERROR
5805
5806 033234 012737 177777 005450 FHDHM:  MOV      #-1,TEMP1 ;ALL 1'S
5807 033242 012765 000001 000026  MOV      #1,RKMR1(R5) ;WORD 1
5808 033250 004737 032526  1$:      JSR      PC,GSTAT
5809 033254 032737 000040 005434  BIT      #D.HDHM,HMR2
5810 033262 001007          BNE      2$
5811 033264 005337 005450  DEC      TEMP1
5812 033270 001367          BNE      1$
5813 033272 005337 005452  DEC      TEMP2
5814 033276 001356          BNE      FHDHM
5815 033300 000207          RTS      PC
5816 033302 062716 000002  2$:      ADD      #2,(SP) ;SKIP OVER ERROR
5817 033306 000207          RTS      PC
5818
5819 ;ROUTINE TO FIND LOAD HEADS IN RKMR2 WORD 1 BEFORE THE TIMEOUT
5820 ;RETURN IF NOT FOUND
5821 ;RETURN+2 IF FOUND: SKIP OVER ERROR
5822
5823 033310 012737 000372 005450 FLOAD:  MOV      #250,TEMP1
5824 033316 012765 000001 000026  MOV      #1,RKMR1(R5) ;WORD 1
5825 033324 004737 032526  1$:      JSR      PC,GSTAT
5826 033330 032737 010000 005434  BIT      #D.LOAD,HMR2
5827 033336 001004          BNE      2$
5828 033340 005337 005450  DEC      TEMP1
5829 033344 001367          BNE      1$
5830 033346 000207          RTS      PC
5831 033350 062716 000002  2$:      ADD      #2,(SP) ;SKIP OVER ERROR
5832 033354 000207          RTS      PC
5833
5834 ;FILL HEADER TABLE WITH 66 WORDS OF VALID HEADERS
5835 ;ENTER WITH CYL # IN 'CALADD'
5836 ;ENTER WITH HEAD # IN 'HEAD'
5837 ;ENTER WITH FORMAT IN 'FORMAT'
5838
5839 033356 010046          FHDTAB: MOV      R0,-(SP) ;SAV R0
5840 033360 010146          MOV      R1,-(SP) ;SAV R1
5841 033362 012700 001550  MOV      #HDTAB,R0 ;HEADER WORD TABLE ADDR
5842 033366 005001          CLR      R1 ;SECTOR COUNTER
5843 033370 013737 001514 001516  MOV      HEAD,HD1
5844 033376 006337 001516  ASL      HD1
5845 033402 005337 001516  ASL      HD1
5846 033406 006337 001516  ASL      HD1
5847 033412 006337 001516  ASL      HD1

```

```

5848 033416 006337 001516      ASL    HD1          ;SETUP HEAD # FOR WORD 2 OF HEADER
5849 033422 013737 001520 001522  MOV    FORMAT,FMT1
5850 033430 000337 001522      SWAB   FMT1
5851 033434 006337 001522      ASL    FMT1          ;SETUP FORMAT FOR WORD 2 OF HEADER
5852
5853 033440 013720 001406      IS:   MOV    CALADD,(R0)+ ;HEADER WORD 1-CYL ADDR
5854 033444 010110          MOV    R1,(R0)        ;HEADER WORD 2-SECTOR NO
5855 033446 053710 001516      BIS    HD1,(R0)       ;
5856 033452 053710 001522      BIS    FMT1,(R0)      ;
5857 033458 052710 140000      BIS    #'BIT14'3IT15),(R0) ; -GOOD SECTOR FLAGS
5858
5859 033462 013737 001406 005450  MOV    CALADD,TEMP1
5860 033470 011037 005452      MOV    (R0),TEMP2
5861 033474 043737 001406 005452  BIC    CALADD,TEMP2
5862 033502 042037 005450      BIC    (R0)+,TEMP1
5863 033506 053737 005450 005452  BIS    TEMP1,TEMP2
5864 033514 013720 005452      MOV    TEMP2,(R0)+ ;HEADER WORD 3-HEADER CHECK
5865
5866 033520 005201          INC    R1              ;SECTOR CTR
5867 033522 020127 000026      CMP    R1,#22         ;ALL 22 SECTORS DONE? (66 WORDS)
5868 033526 001344          BNE   IS              ;BR IF NO
5869
5870 033530 012601          MOV    (SP)+,R1       ;RESTOR R1
5871 033532 012600          MOV    (SP)+,R0       ;RESTOR R0
5872 033534 000207          RTS                   PC
5873
5874
5875 ; THIS ROUTINE SORTS THE RHTAB TABLE FROM WHATEVER SECTOR IT BEGINS
5876 ; WITH AND RE-WITES THE INFO IN SRTTAB TABLE TO BEGIN WITH SECTOR 0
5877
5878 033536 010046      SORT: MOV    R0,-(SP)      ;SAVE R0
5879 033540 010146      MOV    R1,-(SP)      ;SAVE R1
5880 033542 004737 032650  JSR    PC,R0SEC
5881 033546 062737 000001 001426  ADD    #1,SECTOR
5882 033554 004737 033644  JSR    PC,MULT6      ;MULT SECTOR BY 6
5883
5884 033560 012700 000204      MOV    #132,R0
5885 033564 163700 001426      SUB    SECTOR,R0     ;R0-SECTOR TO R0 = INDEX
5886 033570 010037 001426      MOV    R0,SECTOR
5887 033574 062737 001754 001426  ADD    #RHTAB,SECTOR ;SAVE INDEX
5888
5889 033602 062700 001754      ADD    #RHTAB,R0    ;INDEX TO BOT HALF OF RHTAB
5890 033606 012701 002160      MOV    #SRTTAB,R1   ;INDEX TO TOP HALF OF SRTTAB
5891
5892 033612 012021          IS:   MOV    (R0)+,(R1)+ ;PUT BOTTOM OF RHTAB TO TOP OF SRTTAB
5893 033614 020027 002160      CMP    R0,#RHTAB+132
5894 033620 001374          BNE   IS
5895
5896 033622 012700 001754          2S:   MOV    #RHTAB,R0
5897 033626 012021          MOV    (R0)+,(R1)+ ;PUT TOP OF RHTAB TO BOT OF SRTTAB
5898 033630 020037 001426      CMP    R0,SECTOR
5899 033634 001374          BNE   2S
5900
5901 033636 012601          MOV    (SP)+,R1       ;RESTOR R1
5902 033640 012600          MOV    (SP)+,R0       ;RESTOR R0
5903 033642 000207          RTS                   PC

```

```

5904
5905
5906
5907
5908 033644 006337 001426
5909 033650 013746 001426
5910 033654 006337 001426
5911 033660 062637 001426
5912 033664 000207
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922 033666 010446
5923 033670 013746 001160
5924 033674 013746 001162
5925
5926 033700 013737 005416 001160
5927 033706 013737 005416 001162
5928
5929 033714 042737 177740 001160
5930 033722 042737 174377 001162
5931 033730 000337 001162
5932
5933 033734 005737 001160
5934 033740 001414
5935 033742 005337 001160
5936 033746 013737 001160 001426
5937 033754 013737 001162 001514
5938 033762 013737 005426 001374
5939 033770 000440
5940
5941 033772 005737 001162
5942 033776 001414
5943 034000 005337 001162
5944 034004 013737 001162 001514
5945 034012 012737 000025 001426
5946 034020 013737 005426 001374
5947 034026 000421
5948
5949 034030 005737 005426
5950 034034 001414
5951 034036 005337 005426
5952 034042 013737 005426 001374
5953 034050 012737 000002 001514
5954 034056 012737 000025 001426
5955 034064 000402
5956
5957 034066 104044
5958 034070 000453
5959

```

;MULT BY 6. ENTER WITH DESIRED # IN 'SECTOR'
 MULT6: ASL SECTOR ;2 X SECTOR
 MOV SECTOR,-(SP)
 ASL SECTOR ;4 X SECTOR
 ADD (SP)+,SECTOR ;(4 X 5)+(2 X 5) = 6 X SECTOR
 RTS PC

;THIS ROUTINE IS ENTERED IF THERE IS A READ HEADER MISMATCH.
 ;IT COMPARES THE CYLINDER, SECTOR AND HEAD NUMBERS OF WHERE THE
 ;HEADS WERE AT THE TIME OF READING AGAINST THE BSE INFO
 ;OBTAINED IN A PREVIOUS TEST.
 ;RETURN IF NO COMPARE: ERROR CONDITION
 ;RETURN+2 IF COMPARE FOUND: SKIP OVER ERROR

```

TRUERR: MOV R4,-(SP) ;SAVE R4
MOV STMP0,-(SP) ;SAVE
MOV STMP1,-(SP) ;SAVE
MOV HDA,STMP0 ;READ TRK SECTOR INFO
MOV HDA,STMP1
BIC #1C<37>,STMP0 ;TMP0 HAS SECTOR INFO
BIC #1C<3400>,STMP1 ;TMP1 HAS HEAD INFO
SWAB STMP1
TST STMP0 ;SEE IF SECTOR 0
BEQ 6$
DEC STMP0 ;GET ACTUAL WLE SECTOR
MOV STMP0,SECTOR ;STORE SECTOR
MOV STMP1,HEAD ;STORE HEAD
MOV HDC,CCYL ;STORE CYLINDER
BR 9$
6$: TST STMP1 ;SEE IF HEAD 0
BEQ 7$ ;BR IF YES
DEC STMP1 ;GET ACTUAL WLE HEAD
MOV STMP1,HEAD
MOV #21,SECTOR
MOV HDC,CCYL
BR 9$
7$: TST HDC ;SEE IF CYL 0
BEQ 8$
DEC HDC ;GET ACTUAL WLE CYL
MOV HDC,CCYL
MOV #21,HEAD
MOV #21,SECTOR
BR 9$
8$: ERROR 44 ;RKDC & RKDA INDICATES WCE
BR 5$ ;OCCURRED AT CYL 411,HD 2 SECTOR 21

```

```

5960 034072 012704 003374 95:   MOV   #BSE22+8, R4
5961 034076 012437 001540 15:   MOV   (R4)+, WORD      ;GET CYL# OFF BSE TABLE
5962 034102 023727 001540 177777  CMP   WORD, #1         ;SEE IF ALL 1'S
5963 034110 001406  BEQ   25              ;EXIT IF YES
5964 034112 023737 001540 001374  CMP   WORD, CCYL      ;COMPARE CYL #
5965 034120 001410  BEQ   35              ;BR IF CYL MATCH
5966 034122 005724  TST   (R4)+          ;ADV TO NEXT CYL WORD
5967 034124 000764  BR    15
5968
5969 034126 012637 001162 25:   MOV   (SP)+, STMP1    ;RESTORE
5970 034132 012637 001160  MOV   (SP)+, STMP0    ;RESTORE
5971 034136 012604  MOV   (SP)+, R4       ;RESTOR R4
5972 034140 000207  RTS   PC
5973
5974 034142 011437 001540 35:   MOV   (R4), WORD      ;GET HEAD & SECTOR FROM BSE TABLE
5975 034146 042737 177400 001540  BIC   #177400, WORD    ;KEEP SECTOR# ONLY
5976 034154 023737 001540 001426  CMP   WORD, SECTOR    ;SECTOR COMPARE?
5977 034162 001402  BEQ   45              ;BR IF YES
5978 034164 005724  TST   (R4)+          ;ELSE GET NEXT CYL # OFF TABLE
5979 034166 000743  BR    15
5980
5981 034170 012437 001540 45:   MOV   (R4)+, WORD      ;GET HEAD & SECTOR FROM BSE TABLE
5982 034174 000337 001540  SWAB  WORD
5983 034200 042737 177400 001540  BIC   #177400, WORD    ;KEEP HEAD# ONLY
5984 034206 023737 001540 001514  CMP   WORD, HEAD      ;HEAD COMPARE?
5985 034214 001401  BEQ   55              ;BR IF YES
5986 034216 000727  BR    15
5987
5988 034220 012637 001162 55:   MOV   (SP)+, STMP1    ;RESTORE
5989 034224 012637 001160  MOV   (SP)+, STMP0    ;RESTORE
5990 034230 012604  MOV   (SP)+, R4       ;RESTOR R4
5991 034232 062716 000002  ADD   #2, (SP)        ;SKIP OVER ERR ON RETURN
5992 034236 000207  RTS   PC
5993

```

```

; THIS ROUTINE CALIBRATES ANY PDP-11 AGAINST 2 CONSECUTIVE 16MS TICKS FROM AN L OR P CLOC
; THE 1ST TICK CLEARS A COUNTER 'LPCNT'. THE PROGRAM THEN GOES THRU A LOOP.
; EACH TIME IT GOES THRU THE LOOP, LPCNT IS INCREMENTED BY 1.
; THE 2ND TICK STOPS INCREMENTING LPCNT AND THE CLOCK INTERRUPT IS TURNED OFF.
; NOW, FOR THIS PARTICULAR PDP-11, THE TIME TO GO THRU THE ABOVE LOOP ONE IS
; 16.6MS DIVIDED BY THE CONTENTS OF LPCNT.
; AN APPROX VALUE FOR A PDP11-05 MAY BE 0.4MS TO GO THRU THE LOOP ONCE
; FROM THIS POINT ON, WHENEVER ACCURACIES GREATER THAN WHAT THE L CLOCK
; CAN PROVIDE ARE NECESSARY, THE EVENT TO BE TIMED IS COMPARED AGAINST
; THE NUMBER OF TIMES THE LOOP IS PASSED THRU AND MULT. BY THE ABOVE
; CALCULATED FIGURE

```

```

6009
6010 034240 104413  CALCLK: SAVREG
6011 034242 005000  CLR   R0              ; ITERATION CTR
6012 034244 005037 001476  CLR   SUM             ; SUM OF 256 ITERATIONS
6013
6014 034250 012737 000001 001412 55:   MOV   #1, COUNT      ; GO THRU CLOCK HANDLER
6015 034256 012737 000001 001414  MOV   #1, SEC        ; ONLY ONCE

```

```

DZR618.P11      SUBROUTINES
6016 034264 012737 177777 005450      MUV      #1,TEMP1      ;ALL 1'S FOR TIMEOUT
6017 034272 004737 035202              JSR      PC,CLKON
6018
6019
6020 034276 005737 001416      1$:     TST      TIMUP
6021 034302 001007              BNE      2$           ;BR IF GOT 1'ST TICK
6022 034304 005337 005450      DEC      TEMP1
6023 034310 001372              BNE      1$           ;BR IF TIMEOUT NOT DONE
6024 034312 104401 044355      TYPE    MSG23        ;NO CLOCK INTR PRESENT, ABORT TIMING TEST
6025 034316 000137 031434      JMP      $EOP        ;ABORT DRIVE
6026
6027 034322 012737 000001 001412 2$:     MOV      #1,COUNT    ;GOT 1'ST TICK
6028 034330 012737 000001 001414      MOV      #1,SEC
6029 034336 005037 001472      CLR      LPCNT       ;LOOP COUNTER
6030 034342 005037 001416      CLR      TIMUP       ;CLEAR BEFORE 2'ND TICK
6031
6032 034346 005737 001416      3$:     TST      TIMUP
6033 034352 001006              BNE      4$           ;BR IF GOT 2'ND TICK
6034 034354 004737 034672      JSR      PC,LOOP
6035 034360 000240              NOP
6036 034362 000240              NOP      ;THESE NOP'S
6037 034364 000240              NOP      ;PROVIDE A
6038 034366 000767              BR       3$          ;DELAY
6039
6040 034370 004737 035310 001476 4$:     JSR      PC,CLKOF    ;GOT 2'ND TICK, TURN OFF CLOCK
6041 034374 063737 001472      ADD     LPCNT,SUM
6042
6043 034402 005200              INC      R0
6044 034404 020027 000400      CMP     R0,#256.    ;ALL ITERATIONS DONE?
6045 034410 001317              BNE      5$           ;BR IF NO
6046
6047 034412 000337 001476      SWAB   SUM          ;DIVIDE BY 256
6048 034416 105037 001477      CLRB   SUM+1       ;TO GET AVERAGE
6049
6050 034422 005000              CLR     R0          ;CLEAR FOR DIV
6051 034424 005001              CLR     R1
6052 034426 005002              CLR     R2
6053 034430 005004              CLR     R4
6054 034432 012703 040432      MOV     #16666.,R3  ;LSB DIVIDEND (16666 US)
6055 034436 013705 001476      MOV     SUM,R5      ;DIVISOR (AVERAGE OF 256)
6056 034442 004737 036170      JSR     PC,M.DPID   ;DO DIVIDE
6057 034446 010337 001474      MOV     R3,LPTIM    ;STORE QUOTIENT
6058
6059
6060 034452 104414              RESREG
6061 034454 000207              RTS     PC
6062
6063
6064
6065
6066
6067
6068 034456 104413      ;THIS ROUTINE IS EXACTLY THE SAME AS THE ABOVE EXCEPT THAT
6069 034460 005000      ;A JSR PC QKCYLD HAS BEEN INSERTED TO TAKE UP SOME TIME
6070 034462 005037 001476      ;TO MATCH THE TIME TAKEN BY THE ROUTINE THAT USES THIS ROUTINE
6071
VELCAL: SAVREG
        CLR     R0          ; ITERATION CTR
        CLR     SUM        ; SUM OF 256 ITERATIONS

```



```

6072 034466 012737 000001 001412 5$: MOV #1,COUNT ;SEE ABOVE CALCLK FOR COMMENTS
6073 034474 012737 000001 001414 MOV #1,SEC
6074 034502 012737 177777 005450 MOV #-1,TEMP1
6075 034510 004737 035202 JSR PC,CLKON
6076
6077 034514 005737 001416 1$: TST TIMUP
6078 034520 001007 SNE 2$
6079 034522 005337 005450 DEC TEMP1
6080 034526 001372 BNE 1$
6081 034530 104401 044355 TYPE ,MSG23 ;NO CLOCK
6082 034534 000137 031434 JMP $EOP
6083
6084 034540 012737 000001 001412 2$: MOV #1,COUNT
6085 034546 012737 000001 001414 MOV #1,SEC
6086 034554 005037 001472 CLR LPCNT
6087 034560 005037 001416 CLR TIMUP
6088
6089 034564 005737 001416 3$: TST TIMUP
6090 034570 001005 BNE 4$
6091 034572 004737 033004 JSR PC,QKCYLD ;THIS IS ONLY DIFFERENCE BETWEEN
6092 ;PRECEDING SUBROUTINE
6093 034576 004737 034672 JSR PC,LOOP
6094 034602 000770 BR 3$
6095
6096 034604 004737 035310 4$: JSR PC,CLKOF
6097 034610 063737 001472 001476 ADD LPCNT,SUM
6098
6099 034616 005200 INC R0
6100 034620 020027 000400 CMP R0,#256.
6101 034624 001320 BNE 5$
6102
6103 034626 000337 001476 SWAB SUM
6104 034632 105037 001477 CLRB SUM+1
6105
6106 034636 005000 CLR R0
6107 034640 005001 CLR R1
6108 034642 005002 CLR R2
6109 034644 005004 CLR R4
6110 034646 012703 040432 MOV #16666.,R3
6111 034652 013705 001476 MOV SUM,R5
6112 034656 004737 036170 JSR PC,H.DPID
6113 034662 010337 001474 MOV R3,LPTIM
6114 034666 104414 RESREG
6115 034670 000207 RTS PC
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126 034672 010046 LOOP: MOV R0,-(SP) ;SAVE R0
6127 034674 012700 MOV #50.,R0
    
```

```

;WITH 50 IN R0, IT TAKES APPROX 400 US FOR AN 11/05 TO GO THRU THIS
;LOOP AND INCREMENT 'LPCNT' ONCE.
;THIS 400 US IS APPROX 2.5% OF 16MS GIVING A RESOLUTION OF 0.4MS
;PER COUNT IN 'LPCNT'.
;WHEN USED BY THE 'CALCLK' ROUTINE, LPCLK SHOULD BE APPROX 40(10)=50(8)
;WITH AN 11/05 TO 200(10)=264(8) FOR AN 11/70
    
```

```

6128 034700 005300
6129 034702 001401
6130 034704 000775
6131 034706 005237 001472
6132 034712 012600
6133 034714 000207
6134
6135
6136
6137 034716 136465 005376 000017
6138 034724 001006
6139 034726 004737 034672
6140 034732 005737 001472
6141 034736 001367
6142 034740 000207
6143 034742 062716 000002
6144 034746 000207
6145
6146
6147
6148
6149
6150 034750
6151 034750 013746 001474
6152 034754 013746 001160
6153 034760 004737 041714
6154 034764 012637 001160
6155 034770 012637 001162
6156 034774 012746 001160
6157
6158 035000 004737 041424
6159 035004 004737 041654
6160 035010 104401 044504
6161 035014 000207
6162
6163
6164
6165
6166 035016 104413
6167 035020 005000
6168 035022 005001
6169 035024 005004
6170 035026 013703 001476
6171 035032 013702 001500
6172 035036 012705 000144
6173 035042 004737 035150
6174 035046 104401 044504
6175 035052 104414
6176 035054 000207
6177
6178
6179
6180
6181 035056 104413
6182 035060 005000
6183 035062 005001

```

```

1S:   DEC   R0
      BEQ   2S
      BR    1S
2S:   INC   LPCNT
      MOV   (SP)+,R0      ;RESTORE R0
      RTS   PC

; THIS ROUTINE FINDS ATTN FROM A SEEK AND RETURNS
FATT3: BITB  ATTN(R4),RKASOF+1(R5) ;TEST FOR ATTN
      BNE  1S ;EXIT IF THERE
      JSR  PC,LOOP ;ELSE GO THRU LOOP
      TST  LPCNT ;TEST FOR OVERFLOW
      BNE  FATT3 ;BR IF NO AND TRY AGAIN
      RTS  PC
1S:   ADD   #2,(SP) ;JUMP OVER ERROR
      RTS  PC

; ROUTINE TO MULT 8 TYPE A 2 WORD PRODUCT
; ENTER WITH LPCNT IN $TMP0
; USED FOR LOW VEL TIMES IN UNLD & LOADING
TYPTIM:
      MOV   LPTIM,-(SP) ;;PUT THE MULTIPLIER ON THE STACK
      MOV   $TMP0,-(SP) ;;PUT THE MULTIPLICAND ON THE STACK
      JSR  PC,$MULT ;;CALL THE MULTIPLY ROUTINE
      MOV   (SP)+,$TMP0 ;;GET THE LSB'S OF THE PRODUCT
      MOV   (SP)+,$TMP0+2 ;;GET THE MSB'S OF THE PRODUCT
      MOV   #TMP0,-(SP) ;;PUSH LSB ONTO STACK
      JSR  PC,$DB2D ;;CONVERT TO ASCII STRING
      JSR  PC,$SUPRS ;;TYPE TIMES
      TYPE MSG25 ;;MICRO SEC
      RTS  PC

; ROUTINE USED BY TIMING TO OBTAIN THE AVERAGE OF 100 TIMES IN MICRO SECONDS
AVGTIM: SAVREG
      CLR   R0 ;CLEAR FOR DIV
      CLR   R1
      CLR   R4
      MOV   SUM,R3 ;LSB DIVIDEND
      MOV   SUM+2,R2 ;MSB DIVIDEND
      MOV   #100,R5 ;DIVISOR
      JSR  PC,DIVTYP
      TYPE ,MSG25 ;USEC
      RESREG
      RTS  PC

; ROUTINE TO CALC & TYPE AVERAGE SPEED
AVGSP: SAVREG
      CLR   R0 ;CLEAR FOR DIV
      CLR   R1

```

```

6184 035064 013703 001476      MOV      SUM,R3          ;LSB DIVIDEND
6185 035070 013702 001500      MOV      SUM+2,R2       ;MSB DIVIDEND
6186 035074 012705 103240      MOV      #103240,R5    ;LSB DIVISOR
6187 035100 012704 000001      MOV      #1,R4         ;MSB DIVISOR
6188                                ;TO DIVIDE BY 100000(10)
6189                                ;100000(10)=303240(8)
6190 035104 004737 036170      JSR      PC,M.DPID     ;GO DIVIDE
6191 035110 010337 001162      MOV      R3,$TMP1     ;SAVE QUOTIENT
6192
6193 035114 005000                CLR      R0            ;CLEAR FOR DIV
6194 035116 005001                CLR      R1
6195 035120 005002                CLR      R2
6196 035122 005004                CLR      R4
6197 035124 012703 001231      MOV      #665.,R3     ;LSB DIVIDEND
6198 035130 013705 001162      MOV      $TMP1,R5     ;DIVISOR
6199 035134 004737 035150      JSR      PC,DIVTYP
6200 035140 104401 045452      TYPE    ,MSG36        ;INCHES/SEC
6201 035144 104414
6202 035146 000207
6203
6204                                ;ROUTINE TO DIVIDE & TYPE WITH SUPPRESSED 0'S
6205
6206 035150 004737 036170      DIVTYP: JSR      PC,M.DPID ;GO DIVIDE
6207 035154 010337 001160      MOV      R3,$TMP0     ;STORE LSB QUOTIENT
6208 035160 010237 001162      MOV      R2,$TMP1     ;STORE MSB QUOTIENT
6209 035164 012746 001160      MOV      #,$TMP0,-(SP) ;PUSH LSB ON STACK
6210 035170 004737 041424      JSR      PC,$DB2D     ;CONVERT TO ASCII
6211 035174 004737 041654      JSR      PC,$SUPRS    ;TYPE IT
6212 035200 000207
6213
6214                                ;ROUTINE TO TURN L OR P CLOCK INTERRUPT ON
6215
6216 035202 012746 000000      CLKON:  MOV      #PRO,-(SP) ;PSW LOADED TO BE
6217 035206 012746 035214      MOV      #25,-(SP)    ;LSI-11 COMPATABLE
6218 035212 000002                RTI                    ;ENABLE CLOCK INTERRUPTS
6219 035214 005037 001416      25:    CLR      TIMUP
6220 035220 005737 005516      TST     PCLKF
6221 035224 001004                BNE     1$            ;BRANCH IF P-CLOCK PRESENT
6222 035226 012777 000100 144112      MOV      #100,$LKS   ;L-CLOCK, ENABLE INT
6223 035234 000207                RTS     PC
6224 035236 012777 177777 144076      1$:    MOV      #-1,$PKSB ;P-CLOCK, ALL 1'S
6225 035244 012777 000135 144066      MOV      #135,$PKSB  ;ENABLE INT, CT UP, REP INT
6226 035252 000207                RTS     PC            ;LINE FREQ & RUN
6227
6228                                ;KW11-L & KW11-P INTERRUPT HANDLER
6229
6230 035254 005037 001416      CLOCK:  CLR      TIMUP
6231 035260 005337 001412      DEC     COUNT
6232 035264 001010                BNE     1$
6233 035266 013737 001410 001412      MOV      HZ,COUNT
6234 035274 005337 001414      DEC     SEC
6235 035300 001002                BNE     1$
6236 035302 005237 001416      INC     TIMUP        ;SORRY, TIME IS UP
6237 035306 000C02      1$:    RTI
6238
6239                                ;ROUTINE TO TURN L OR P CLOCK INTERRUPT OFF

```

6240
6241 035310 012746 000340
6242 035314 012746 035322
6243 035320 000002
6244 035322 005737 005516
6245 035326 001003
6246 035330 005077 144012
6247 035334 000207
6248 035336 005077 143776
6249 035342 000207

```

;CLKOF:  MOV    #PR7,-(SP)      ;PSW LOADED TO BE
          MOV    #2$,-(SP)      ;LSI-11 COMPATABLE
          RTI                    ;LOCK OUT ALL INTERRUPTS
2$:      TST    PCLKF
          BNE    1$             ;BRACH IF P-CLOCK PRESENT
          CLR    @LKS           ;L-CLOCK, CLEAR INTERRUPT
          RTS    PC
1$:      CLR    @PKS           ;P-CLOCK, CLEAR INTERRUPT
          RTS    PC
    
```

6250
6251
6252
6253
6254

```

;THIS ROUTINE CHECKS RKMR2 (MSGA)-WORDS 0 & 1
;ENTER WITH SHOULD BE VALUE IN SBMR2.
;RETURN IF NO COMPARE, RETURN +2 IF COMPARE
    
```

6255 035344 013746 005450
6256 035350 053737 001222 005444
6257 035356 013737 005444 005450
6258 035364 004737 035524
6259 035370 013737 005450 005444
6260 035376 023737 005444 005434
6261 035404 001005
6262 035406 012637 005450
6263 035412 062716 000002
6264 035416 000207
6265 035420 012637 005450
6266 035424 000207

```

;CKMR2:  MOV    TEMP1,-(SP)     ;SAV TEMP1
          BIS    $UNIT,SBMR2    ;INSERT DPIPE #
          MOV    SBMR2,TEMP1
          JSR    PC,SBPAR       ;GET PARITY FOR SBMR2
          MOV    TEMP1,SBMR2    ;NOW HAS PARITY
          CMP    SBMR2,HMR2     ;SHOULD BE SAME
          BNE    1$
          MOV    (SP)+,TEMP1    ;RESTOR TEMP1
          ADD    #2,(SP)        ;COMPARE OK, SKIP OVER ERROR.
          RTS    PC
1$:      MOV    (SP)+,TEMP1    ;RESTOR TEMP1
          RTS    PC
    
```

6267
6268
6269
6270
6271
6272
6273

```

;THIS ROUTINE CHECKS RKMR3 (MSGB)-WORDS 0 & 1
;ENTER WITH SHOULD BE VALUE IN SBMR3
;RETURN IF NO COMPARE, RETURN +2 IF COMPARE
    
```

6274 035426 013746 005450
6275 035432 013737 005432 005450
6276 035440 042737 177774 005450
6277 035446 053737 005450 005446
6278 035454 013737 005446 005450
6279 035462 004737 035524
6280 035466 013737 005450 005446
6281 035474 023737 005446 005436
6282 035502 001005
6283 035504 012637 005450
6284 035510 062716 000002
6285 035514 000207
6286 035516 012637 005450
6287 035522 000207

```

;CKMR3:  MOV    TEMP1,-(SP)     ;SAV TEMP1
          MOV    HMR1,TEMP1
          BIC    #1C(A.ID),TEMP1
          BIS    TEMP1,SBMR3    ;INSERT WORD #
          MOV    SBMR3,TEMP1
          JSR    PC,SBPAR       ;GET PARITY FOR SBMR3
          MOV    TEMP1,SBMR3    ;NOW HAS PARITY
          CMP    SBMR3,HMR3     ;SHOULD BE SAME
          BNE    1$
          MOV    (SP)+,TEMP1    ;RESTOR TEMP1
          ADD    #2,(SP)        ;COMPARE OK, SKIP OVER ERROR.
          RTS    PC
1$:      MOV    (SP)+,TEMP1    ;RESTOR TEMP1
          RTS    PC
    
```

6288
6289
6290
6291
6292
6293
6294
6295

```

;THIS ROUTINE GENERATES PARITY FOR SBMR2 AND SBMR3
;ENTER WITH SBMR2 / SBMR3 IN TEMP1
;TEMP1 IS ROTATED LEFT 17 TIMES. EACH TIME THE CARRY BIT IS SET,
;R1 IS INCREMENTED. AT THE END OF 17 ROTATES (TEMP1 BACK TO ORIG),
;R1 BIT 0 IS EXAMINED. IF IT IS SET, INDICATING AN ODD # OF 1'S,
;THE PARITY BIT IS NOT SET IN B.
    
```



```

6352 035654 006 .BYTE 6 ;;TYPE 6 DIGIT(S)
6353 035655 000 .BYTE 0 ;;SUPPRESS LEADING ZEROS
6354 035656 032777 001000 143254 1$: BIT #SW9,2SWR ;;LOOP ON ERROR?
6355 035664 001403 BEQ 2$ ;;NO BRANCH
6356 035666 022626 CMP (SP)+,(SP)+ ;;YES, RESTORE STACK
6357 035670 000177 143212 JMP 2$LPADR ;;GO TO STARTING ADDR OF TEST
6358 ;; THAT GAVE BAD TIMEOUT
6359 035674 032777 040000 143236 2$: BIT #SW14,2SWR ;;LOOP ON TEST?
6360 035702 001401 BEQ 3$ ;;NO BRANCH
6361 035704 000002 RTI ;;YES
6362
6363 035706 000000 3$: HALT ;;UNEXPECTED TIME OUT OCCURRED
6364 ;;AS INDICATED. YOU CAN LOOP ON
6365 ;;ERROR, LOOP ON TEST OR INHIBIT
6366 ;;ERROR TYPEOUT BY SETTING THOSE
6367 ;;SWITCHES.
6368
6369 035710 022626 CMP (SP)+,(SP)+ ;;RESTORE STACK
6370 035712 000137 031462 JMP $EOP1 ;;ABORT TESTS
6371
6372 .SBTTL UNEXPECTED INTERRUPT HANDLER
6373
6374 ;;
6375 ;;THIS ROUTINE CHECKS SW13 (INH ERR TYPEOUT), SW9 (LOOP ON ERR)
6376 ;;& SW14 (LOOP ON TEST).
6377 ;;
6378
6379 035716 011600 BADINT: MOV (SP),RO ;;SAVE PC WHERE INT OCCURRED
6380 035720 005740 TST -(RO) ;;GET PC BEFORE UPDATE
6381 035722 032777 020000 143210 BIT #SW13,2SWR ;;INHIBIT ERR TYPEOUT?
6382 035730 001005 BNE 1$ ;;YES, DONT TYPE
6383 035732 104401 046440 TYPE EM4 ;;ABORT TESTS, UNEXP INT @ PC=
6384 035736 010046 MOV RO,-(SP) ;;SAVE RO FOR TYPEOUT
6385 ;;TYPE PC
6386 035740 104403 TYPOS ;;GO TYPE--OCTAL ASCII
6387 035742 006 .BYTE 6 ;;TYPE 6 DIGIT(S)
6388 035743 000 .BYTE 0 ;;SUPPRESS LEADING ZEROS
6389
6390 035744 032777 001000 143166 1$: BIT #SW9,2SWR ;;LOOP ON ERROR?
6391 035752 001403 BEQ 2$ ;;NO BRANCH
6392 035754 022626 CMP (SP)+,(SP)+ ;;YES, RESTORE STACK
6393 035756 000177 143124 JMP 2$LPADR ;;GO TO THE STARTING ADDR OF
6394 ;;TEST THAT GAVE UNEXP. INT.
6395 035762 032777 040000 143150 2$: BIT #SW14,2SWR ;;LOOP ON TEST?
6396 035770 001401 BEQ 3$ ;;NO BRANCH
6397 035772 000002 RTI ;;YES.
6398
6399 035774 000000 3$: HALT ;;UNEXPECTED INTERRUPT OCCURRED AS
6400 ;;INDICATED. YOU CAN LOOP ON ERROR,
6401 ;;LOOP ON TEST OR INHIBIT
6402 ;;ERROR TYPEOUT BY SETTING THOSE
6403 ;;SWITCHES
6404
6405 035776 022626 CMP (SP)+,(SP)+ ;;RESTORE STACK
6406 036000 000137 031462 JMP $EOP1 ;;ABORT TESTS
6407

```

```

6408 .SBTTL MEMORY CHECK ENABLE TRAP
6409
6410 036004 012737 036020 001176 MEMERR: MOV #15,SESCAPE
6411 036012 011637 001354 MOV (SP),TRAPPC ;STORE PC
6412 036016 104041 ERROR 41 ;UNEXP MEM PARITY TRAP
6413 036020 005037 001176 15: CLR SESCOAPE
6414 036024 032777 001000 143106 BIT #SW9,SWR ;CHECK IF LOOP ON ERROR
6415 036032 001001 BNE 25 ;YES, FORCE STACK AND TRY AGAIN
6416 036034 000002 RTI ;ELSE RETURN
6417
6418 036036 012706 001100 25: MOV #STACK,SP ;INIT STACK
6419 036042 000177 143042 JMP #SLPERA ;LOOP ON ERROR
6420

```

```

6421 .SBTTL RK06 INTERRUPT HANDLER
6422
6423 INTER: NOP
6424 036046 000240 NOP
6425 036050 000240 NOP
6426 036052 000240 NOP
6427 036054 011600 MOV (SP),RO ;SAVE PC WHERE INT OCCURRED.
6428 036056 005740 TST -(RO) ;GET PC BEFORE UPDATE.
6429 036060 104401 043307 TYPE MSG6 ;INT AT PC=
6430 036064 010046 MOV RO,-(SP) ;SAVE RO FOR TYPEOUT
6431 ;TYPE PC
6432 036066 104403 TYPOS ;GO TYPE--OCTAL ASCII
6433 036070 006 .BYTE 6 ;TYPE 6 DIGIT(S)
6434 036071 000 .BYTE 0 ;SUPPRESS LEADING ZEROS
6435 036072 000000 HALT
6436 036074 000240 NOP
6437 036076 000240 NOP
6438 036100 000002 RTI

```

```

6439 .SBTTL POWER DOWN AND UP ROUTINES
6440
6441 ;POWER DOWN ROUTINE
6442
6443 SPWRDN: MOV #SPWRUP,PWRVEC ;SET UP VECTOR
6444 036102 012737 036114 000024 HALT
6445 036110 000000 BR -2 ;HANG UP.
6446 036112 000776
6447
6448 ;POWER UP ROUTINE
6449
6450 SPWRUP: CLR SPWRCT ;WAIT LOOP FOR TTY
6451 036120 005237 036166 15: INC SPWRCT ;WAIT FOR THE INCR
6452 036124 001375 BNE 15 ;OF WORD
6453 036126 012737 036102 000024 MOV #SPWRDN,PWRVEC ;SET POWER DOWN VECTOR
6454 036134 012737 000340 000026 MOV #PR7,PWRVEC+2 ;PRIORITY 7
6455 036142 012737 000340 000036 MOV #PR7,TRAPVEC+2 ;LOCKOUT ALL INTERRUPTS FOR TRAPS
6456 036150 012706 001100 MOV #STACK,SP ;INITIALIZE STACK
6457 036154 104401 043477 TYPE ,MSG11 ;REPORT POWER FAIL
6458 036160 000005 RESET
6459 036162 000137 013106 JMP PFSRT
6460
6461 036166 000000 SPWRCT: 0 ;WAIT COUNT FOR TTY
6462
6463 ;

```

```

6464      :DIVISION UTILITY ROUTINE
6465      :
6466      :R0-R1-R2-R3=DIVIDEND
6467      :R4-R5=DIVISOR
6468      :R0-R1=REMAINDER AFTER DIVISION
6469      :R2-R3=QUOTIENT AFTER DIVISION
6470      :ENTER WITH JSR PC,M.DPID
6471      :
6472      036170 012746 000040      M.DPID: MOV      #40,-(SP)      ;COUNTER FOR DIVISION CYCLES
6473      036174 010446              MOV      R4,-(SP)      ;HI ORDER
6474      036176 010546              MOV      R5,-(SP)      ;LO ORDER TO THE STACK
6475      036200 005466 000002      NEG      2(SP)         ;FORM NEGATIVE
6476      036204 005416              NEG      @SP            ;VERSION OF DIVISOR
6477      036206 005666 000002      SBC      2(SP)
6478      036212 061601              ADD      @SP,R1
6479      036214 005500              ADC      R0            ;PERFORM INIT SUBT.
6480      036216 066600 000002      ADD      2(SP),R0
6481      036222 103445              BCS     M.DP50         ;IF CARRY THEN OVERFLOW HAS OCCURRED
6482      036224 005046              CLR      -(SP)         ;THIS IS A LONGER LASTING CARRY BIT
6483      036226 006.03              M.DP40: ROL      R3
6484      036230 006102              ROL      R2
6485      036232 006101              ROL      R1
6486      036234 006100              ROL      R0
6487      036236 005716              TST     @SP            ;TEST CARRY INDICATOR
6488      036240 001410              BEQ     M.DP41         ;IF TO CARRY THEN ADD, ELSE SUBT.
6489      036242 005016              CLR      @SP            ;CLEAR UP FOR NEXT TIME
6490      036244 066601 000002      ADD      2(SP),R1
6491      036250 005500              ADC      R0            ;ADD -(DIVISOR)
6492      036252 005516              ADC      @SP            ;SET CARRY
6493      036254 066600 000004      ADD      4(SP),R0
6494      036260 000404              BR      M.DP42
6495      :
6496      036262 067501              M.DP41: ADD      R5,R1
6497      036264 005500              ADC      R0            ;ADD +(DIVISOR)
6498      036266 005516              ADC      @SP            ;SET CARRY
6499      036270 060400              ADD      R4,R0
6500      036272 005516              M.DP42: ADC      @SP
6501      036274 005716              TST     @SP            ;SET CARRY
6502      036276 001401              BEQ     .+4            ;TEST THE UPDATE INDICATOR
6503      036300 005203              INC     R3            ;IF 0, FORGET IT
6504      036302 005366 000006      DEC     6(SP)         ;NO CARRY POSSIBLE HERE
6505      036306 003347              BGT     M.DP40         ;DECREMENT CTR
6506      036310 006003              ROR     R3            ;BR IF MORE TO DO
6507      036312 103404              BCS     M.DP44
6508      036314 060501              ADD     R5,R1
6509      036316 005500              ADC     R0
6510      036320 060400              ADC     R4,R0
6511      036322 000241              CLC
6512      :
6513      036324 006103              M.DP44: ROL     R3
6514      036326 062706 000010      ADD     #10,SP        ;ADJUST STACK BY 4 WORDS
6515      036332 000242              CLV
6516      036334 000207              RTS     PC
6517      :
6518      036336 062706 000006      M.DP50: ADD     #6,SP
6519      036342 000262              SEV
    
```


N11

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 122
DZR61B.P11 POWER DOWN AND UP ROUTINES

SEQ 0123

6520 036344 000207
6521

RIS PC

.SBTTL SCOPE HANDLER ROUTINE

```

6522
6523
6524
6525
6526
6527
6528
6529
6530
6531
6532
6533
6534
6535
6536 036346
6537 036346 104407
6538 036350 032777 040000 142562
6539 036356 001114
6540
6541 036360 000416
6542
6543 036362 013746 000004
6544 036366 012737 036406 000004
6545 036374 005737 177060
6546 036400 012637 000004
6547 036404 000463
6548 036406 022626
6549 036410 012637 000004
6550 036414 000423
6551 036416
6552 036416 032777 000400 142514
6553 036424 001404
6554 036426 127737 142506 001102
6555 036434 001465
6556 036436 105737 001103
6557 036442 001421
6558 036444 123737 001115 001103
6559 036452 101015
6560 036454 032777 001000 142456
6561 036462 001404
6562 036464 013737 001110 001106
6563 036472 000446
6564 036474 105037 001103
6565 036500 005037 001174
6566 036504 000415
6567 036506 032777 004000 142424
6568 036514 001011
6569 036516 005737 001211
6570 036522 001406
6571 036524 005237 001104
6572 036530 023737 001174 001104
6573 036536 002024
6574 036540 012737 000001 001104
6575 036546 013737 036624 001174
6576 036554 105237 001102
6577 036560 113737 001102 001214

```

```

*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>
*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1 LOOP ON TEST
*SW11=1 INHIBIT ITERATIONS
*SW09=1 LOOP ON ERROR
*SW08=1 LOOP ON TEST IN SWR<7:0>
*CALL
* SCOPE ;;SCOPE=IOT

```

```

$SCOPE:
1$: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
BIT #BIT14,$SWR ;;LOOP ON PRESENT TEST?
BNE $OVER ;;YES IF SW14=1
*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR 6$ ;;IF RUNNING ON THE "XOR" TESTER CHANGE
;;THIS INSTRUCTION TO A "NOP" (NOP=240)
MOV #ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
MOV #5,$ERRVEC ;;SET FOR TIMEOUT
TST #177060 ;;TIME OUT ON XOR?
MOV (SP)+,$ERRVEC ;;RESTORE THE ERROR VECTOR
BR $SVLAD ;;GO TO THE NEXT TEST
5$: CMP (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
MOV (SP)+,$ERRVEC ;;RESTORE THE ERROR VECTOR
BR 7$ ;;LOOP ON THE PRESENT TEST
6$: *****END OF CODE FOR THE XOR TESTER*****
BIT #BIT08,$SWR ;;LOOP ON SPEC. TEST?
BEQ 2$ BR IF NO
CMPB $SWR,$STNM ;;ON THE RIGHT TEST? SWR<7:0>
BEQ $OVER BR IF YES
2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
BEQ 3$ BR IF NO
CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
BHI 3$ BR IF NO
BIT #BIT09,$SWR ;;LOOP ON ERROR?
BEQ 4$ BR IF NO
7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
BR $OVER
4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
BR 1$ ;;ESCAPE TO THE NEXT TEST
3$: BIT #BIT11,$SWR ;;INHIBIT ITERATIONS?
BNE 1$ BR IF YES
TST $PASS ;;IF FIRST PASS OF PROGRAM
BEQ 1$ INHIBIT ITERATIONS
INC $ICNT ;;INCREMENT ITERATION COUNT
CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
BGE $OVER BR IF MORE ITERATION REQUIRED
1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
$SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
MOVB $STNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX

```

```

6578 036566 011637 001106      MOV      (SP), $LPADR      ;; SAVE SCOPE LOOP ADDRESS
6579 036572 011637 001110      MOV      (SP), $LPERR     ;; SAVE ERROR LOOP ADDRESS
6580 036576 005037 001176      CLR      $ESCAPE          ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
6591 036602 112737 000001 001115  MOVB     #1, $ERMAX        ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
6582 036610 013777 001102 142324 $OVER:  MOV      $STNM, $DISPLAY  ;; DISPLAY TEST NUMBER
6583 036616 013716 001106      MOV      $LPADR, (SP)    ;; FUDGE RETURN ADDRESS
6584 036622 000002      RTI                          ;; FIXES PS
6585 036624 003720  $MYCNT: 2000.             ;; MAX. NUMBER OF ITERATIONS
6586                                     .SBTTL  ERROR HANDLER ROUTINE
6587
6588                                     ;; *****
6589                                     ;; THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
6590                                     ;; *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
6591                                     ;; *AND GO TO TYPERR ON ERROR
6592                                     ;; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6593                                     ;; *SW15=1      HALT ON ERROR
6594                                     ;; *SW13=1      INHIBIT ERROR TYPEOUTS
6595                                     ;; *SW10=1      BELL ON ERROR
6596                                     ;; *SW09=1      LOOP ON ERROR
6597                                     ;; *CALL
6598                                     ;; *      ERROR      N      ;; ERROR=EMT AND N=ERROR ITEM NUMBER
6599
6600 036626                                     $ERROR:
6601 036626 104407      CKSWR                                     ;; TEST FOR CHANGE IN SOFT-SWR
6602 036630 105237 001103 7$:      INCB      $ERFLG          ;; SET THE ERROR FLAG
6603 036634 001775      BEQ      7$              ;; DON'T LET THE FLAG GO TO ZERO
6604 036636 013777 001102 142276  MOV      $STNM, $DISPLAY  ;; DISPLAY TEST NUMBER AND ERROR FLAG
6605 036644 032777 002000 142266  BIT      #BIT10, $SWR     ;; BELL ON ERROR?
6606 036652 001402      BEQ      1$              ;; NO - SKIP
6607 036654 104401 001200      TYPE     $BELL          ;; RING BELL
6608 036660 005237 001112 1$:      INC      $ERTL          ;; COUNT THE NUMBER OF ERRORS
6609 036664 011637 001116      MOV      (SP), $ERRPC    ;; GET ADDRESS OF ERROR INSTRUCTION
6610 036670 162737 000002 001116  SUB      #2, $ERRPC
6611 036676 117737 142214 001114  MOVB     $ERRPC, $ITEMB   ;; STRIP AND SAVE THE ERROR ITEM CODE
6612 036704 032777 020000 142226  BIT      #BIT13, $SWR     ;; SKIP TYPEOUT IF SET
6613 036712 001004      BNE     20$             ;; SKIP TYPEOUTS
6614 036714 004737 054364      JSR     PC, TYPERR       ;; GO TO USER ERROR ROUTINE
6615 036720 104401 001205      TYPE     , $CALF
6616 036724
6617 036724 122737 000001 001230 20$:  CMPB     #APTENV, $ENV    ;; RUNNING IN APT MODE
6618 036732 001007      BNE     2$              ;; NO SKIP APT ERROR REPORT
6619 036734 113737 001114 036746  MOVB     $ITEMB, 21$     ;; SET ITEM NUMBER AS ERROR NUMBER
6620 036742 004737 037552      JSR     PC, $ATY4       ;; REPORT FATAL ERROR TO APT
6621 036746      000
6622 036747      000 21$:  .BYTE   0
6623 036750 000777      .BYTE   0
6624 036752 005777 142162 22$:  BR      22$             ;; APT ERROR LOOP
6625 036756 100002 2$:      TST     $SWR           ;; HALT ON ERROR
6626 036760 000000      BPL     3$             ;; SKIP IF CONTINUE
6627 036762 104407      HALT                    ;; HALT ON ERROR!
6628 036764 032777 001000 142146 3$:  CKSWR                                     ;; TEST FOR CHANGE IN SOFT-SWR
6629 036772 001402      BIT     #BIT09, $SWR    ;; LOOP ON ERROR SWITCH SET?
6630 036774 013716 001110      BEQ     4$              ;; BR IF NO
6631 037000 005737 001176      MOV     $LPERR, (SP)    ;; FUDGE RETURN FOR LOOPING
6632 037004 001402      TST     $ESCAPE        ;; CHECK FOR AN ESCAPE ADDRESS
6633 037006 013716 001176      BEQ     5$              ;; BR IF NONE
6633 037006 013716 001176      MOV     $ESCAPE, (SP)  ;; FUDGE RETURN ADDRESS FOR ESCAPE

```

```

6634 037012
6635 037012 022737 031550 000042
6636 037020 001001
6637 037022 000000
6638 037024
6639 037024 000002
6640
6641
6642
6643
6644
6645
6646
6647
6648
6649
6650
6651
6652
6653
6654
6655
6656
6657 037026 105737 001157
6658 037032 100002
6659 037034 000000
6660 037036 000430
6661 037040 010046
6662 037042 017600 000002
6663 037046 122737 000001 001230
6664 037054 001011
6665 037056 132737 000100 001231
6666 037064 001405
6667 037066 010037 037076
6668 037072 004737 037542
6669 037076 000000
6670 037100 132737 000040 001231
6671 037106 001003
6672 037110 112046
6673 037112 011005
6674 037114 005726
6675 037116 012600
6676 037120 062716 000002
6677 037124 000002
6678 037126 122716 000011
6679 037132 001430
6680 037134 122716 000200
6681 037140 001006
6682 037142 005726
6683 037144 104401
6684 037146 001205
6685 037150 105037 037304
6686 037154 000755
6687 037156 004737 037240
6688 037162 123726 001156
6689 037166 001350

```

```

5$: CMP #SENDAD,2#42 ;;ACT-11 AUTO-ACCEPT?
   BNE 6$ ;;BRANCH IF NO
   HALT ;;YES
6$: RTI ;;RETURN
.SBTTL TYPE ROUTINE

*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
* TYPE
* MESADR
*

$TYPE: TSTB $TPFLG ;; IS THERE A TERMINAL?
        BPL 1$ BR IF YES
        HALT HERE IF NO TERMINAL
        BR 3$ LEAVE
1$: MOV RO,-(SP) SAVE RO
   MOV 2(SP),RO GET ADDRESS OF ASCIZ STRING
   CMPB #APTENV,$ENV RUNNING IN APT MODE
   BNE 62$ NO GO CHECK FOR APT CONSOLE
   BITB #APTPOOL,$ENVM SPOOL MESSAGE TO APT
   BEQ 62$ NO GO CHECK FOR CONSOLE
   MOV RO,61$ SETUP MESSAGE ADDRESS FOR APT
   JSR PC,$ATY3 SPOOL MESSAGE TO APT
61$: .WORD 0 MESSAGE ADDRESS
62$: BITB #APTCSUP,$ENVM APT CONSOLE SUPPRESSED
   BNE 60$ YES, SKIP TYPE OUT
2$: MOVB (RO)+,-(SP) PUSH CHARACTER TO BE TYPED ONTO STACK
   BNE 4$ BR IF IT ISN'T THE TERMINATOR
   TST (SP)+ IF TERMINATOR POP IT OFF THE STACK
60$: MOV (SP)+,RO RESTORE RO
3$: ADD #2,(SP) ADJUST RETURN PC
   RTI RETURN
4$: CMPB #HT,(SP) BRANCH IF <HT>
   BEQ 8$
   CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
   BNE 5$
   TST (SP)+ ;;POP <CR><LF> EQUIV
   TYPE TYPE A CR AND LF
   $CRLF
   CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
   BR 2$ GET NEXT CHARACTER
5$: JSR PC,$TYPEC GO TYPE THIS CHARACTER
6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
   BNE 2$ IF NO GO GET NEXT CHAR.

```

```

6690 037170 013746 001154      MOV      $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED
6691                                ;; AND THE NULL CHAR.
6692 037174 105366 000001      7$:      DECB      1(SP)      ;; DOES A NULL NEED TO BE TYPED?
6693 037200 002770                BLT      6$                ;; BR IF NO--GO POP THE NULL OFF OF STACK
6694 037202 004737 037240      JSR      PC,$TYPEC      ;; GO TYPE A NULL
6695 037206 105337 037304      DECB      $CHARCNT      ;; DO NOT COUNT AS A COUNT
6696 037212 000770                BR       7$                ;; LOOP

```

;HORIZONTAL TAB PROCESSOR

```

6700 037214 112716 000040      8$:      MOV      #' (SP)      ;; REPLACE TAB WITH SPACE
6701 037220 004737 037240      9$:      JSR      PC,$TYPEC      ;; TYPE A SPACE
6702 037224 132737 000007 037304      BIT      #7,$CHARCNT      ;; BRANCH IF NOT AT
6703 037232 001372                BNE      9$                ;; TAB STOP
6704 037234 005726                TST      (SP)+            ;; POP SPACE OFF STACK
6705 037236 000724                BR       2$                ;; GET NEXT CHARACTER
6706 037240 105777 141704      $TYPEC: TST      2$TPS          ;; WAIT UNTIL PRINTER IS READY
6707 037244 100375                BPL      $TYPEC
6708 037246 116677 000002 141676      MOV      2(SP),2$TPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
6709 037254 122766 000015 000002      CMP      #CR,2(SP)        ;; IS CHARACTER A CARRIAGE RETURN?
6710 037262 001003                BNE      1$                ;; BRANCH IF NO
6711 037264 105037 037304      CLRB      $CHARCNT      ;; YES--CLEAR CHARACTER COUNT
6712 037270 000406                BR       $TYPEX           ;; EXIT
6713 037272 122766 000012 000002      1$:      CMP      #LF,2(SP)        ;; IS CHARACTER A LINE FEED?
6714 037300 001402                BEQ      $TYPEX           ;; BRANCH IF YES
6715 037302 105227                INCB      (PC)+            ;; COUNT THE CHARACTER
6716 037304 000000                $CHARCNT: .WORD 0        ;; CHARACTER COUNT STORAGE
6717 037306 000207                $TYPEX:  RTS      PC

```

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

6718
6719
6720
6721
6722
6723
6724
6725
6726
6727
6728
6729
6730
6731
6732
6733
6734
6735
6736
6737
6738
6739
6740
6741
6742
6743
6744
6745

```

```

;*****
;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
;REPLACED WITH SPACES.
;CALL:
;*      MOV      NUM,-(SP)      ;; PUT THE BINARY NUMBER ON THE STACK
;*      TYPDS                ;; GO TO THE ROUTINE

```

```

$TYPDS:
MOV      R0,-(SP)      ;; PUSH R0 ON STACK
MOV      R1,-(SP)      ;; PUSH R1 ON STACK
MOV      R2,-(SP)      ;; PUSH R2 ON STACK
MOV      R3,-(SP)      ;; PUSH R3 ON STACK
MOV      R5,-(SP)      ;; PUSH R5 ON STACK
MOV      -20200,-(SP)  ;; SET BLANK SWITCH AND SIGN
MOV      20(SP),R5     ;; GET THE INPUT NUMBER
BPL      1$            ;; BR IF INPUT IS POS.
NEG      R5            ;; MAKE THE BINARY NUMBER POS.
MOV      #'-,1(SP)     ;; MAKE THE ASCII NUMBER NEG.
1$:      CLR      R0            ;; ZERO THE CONSTANTS INDEX
MOV      #5,R3         ;; SETUP THE OUTPUT POINTER
MOV      #' ,(R3)+     ;; SET THE FIRST CHARACTER TO A BLANK
2$:      CLR      R2            ;; CLEAR THE BCD NUMBER

```

```

6746 037360 016001 037514      MOV      $DTBL(R0),R1      ;; GET THE CONSTANT
6747 037364 160105      3$: SUB      R1,R5          ;; FORM THIS BCD DIGIT
6748 037366 002402      BLT      4$              ;; BR IF DONE
6749 037370 005202      INC      R2              ;; INCREASE THE BCD DIGIT BY 1
6750 037372 000774      BR      3$
6751 037374 060105      4$: ADD      R1,R5          ;; ADD BACK THE CONSTANT
6752 037376 005702      TST      R2              ;; CHECK IF BCD DIGIT=0
6753 037400 001002      BNE      5$              ;; FALL THROUGH IF 0
6754 037402 105716      TSTB     (SP)            ;; STILL DOING LEADING 0'S?
6755 037404 100407      BMI      7$              ;; BR IF YES
6756 037406 106316      5$: ASLB     (SP)            ;; MSD?
6757 037410 103003      BCC      6$              ;; BR IF NO
6758 037412 116663 000001 177777  MOVB     1(SP),-1(R3)      ;; YES--SET THE SIGN
6759 037420 052702 000060 6$: BIS      #'0,R2          ;; MAKE THE BCD DIGIT ASCII
6760 037424 052702 000040 7$: BIS      #' ,R2          ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
6761 037430 110223      MOVB     R2,(R3)+         ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
6762 037432 005720      TST      (R0)+           ;; JUST INCREMENTING
6763 037434 020027 000010  CMP      R0,#10          ;; CHECK THE TABLE INDEX
6764 037440 002746      BLT      2$              ;; GO DO THE NEXT DIGIT
6765 037442 003002      BGT      8$              ;; GO TO EXIT
6766 037444 010502      MOV      R5,R2           ;; GET THE LSD
6767 037446 000764      BR      6$              ;; GO CHANGE TO ASCII
6768 037450 105726      8$: TSTB     (SP)+         ;; WAS THE LSD THE FIRST NON-ZERO?
6769 037452 100003      BPL      9$              ;; BR IF NO
6770 037454 116663 177777 177776 9$: MOVB     -1(SP),-2(R3)   ;; YES--SET THE SIGN FOR TYPING
6771 037462 105013      CLRB     (R3)            ;; SET THE TERMINATOR
6772 037464 012605      MOV      (SP)+,R5        ;; POP STACK INTO R5
6773 037466 012603      MOV      (SP)+,R3        ;; POP STACK INTO R3
6774 037470 012602      MOV      (SP)+,R2        ;; POP STACK INTO R2
6775 037472 012601      MOV      (SP)+,R1        ;; POP STACK INTO R1
6776 037474 012600      MOV      (SP)+,R0        ;; POP STACK INTO R0
6777 037476 104401 037524  TYPE     $DBLK           ;; NOW TYPE THE NUMBER
6778 037502 016666 000002 000004  MOV      2(SP),4(SP)      ;; ADJUST THE STACK
6779 037510 012616      MOV      (SP)+,(SP)
6780 037512 000002      RTI                          ;; RETURN TO USER
6781 037514 023420      $DTBL: 10000.
6782 037516 001750      1000.
6783 037520 000144      100.
6784 037522 000012      10.
6785 037524 000004      $DBLK: .BLKW 4
6786      .SBTTL APT COMMUNICATIONS ROUTINE
6787
6788 *****
6789 037534 112737 000001 040000 $ATY1: MOVB     #1,$FFLG      ;; TO REPORT FATAL ERROR
6790 037542 112737 000001 037776 $ATY3: MOVB     #1,$MFLG      ;; TO TYPE A MESSAGE
6791 037550 000403      BR      $ATYC
6792 037552 112737 000001 040000 $ATY4: MOVB     #1,$FFLG      ;; TO ONLY REPORT FATAL ERROR
6793 037560      $ATYC:
6794 037560 010046      MOV      R0,-(SP)        ;; PUSH R0 ON STACK
6795 037562 010146      MOV      R1,-(SP)        ;; PUSH R1 ON STACK
6796 037564 105737 037776      TSTB     $MFLG           ;; SHOULD TYPE A MESSAGE?
6797 037570 001450      BEQ      5$              ;; IF NOT: BR
6798 037572 122737 000001 001230  CMPB     #APTENV,$ENV     ;; OPERATING UNDER APT?
6799 037600 001031      BNE      3$              ;; IF NOT: BR
6800 037602 132737 000100 001231  BITB     #APTPOOL,$ENVM   ;; SHOULD SPOOL MESSAGES?
6801 037610 001425      BEQ      3$              ;; IF NOT: BR

```

```

6802 037612 017600 000004      MOV      24(SP),R0      ;;GET MESSAGE ADDR.
6803 037616 062766 000002 000004  ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
6804 037624 005737 001210      15:     TST      $MSGTYPE     ;;SEE IF DONE W/ LAST XMISSION?
6805 037630 001375          BNE      15           ;;IF NOT: WAIT
6806 037632 010037 001224      MOV      R0,$MSGAD     ;;PUT ADDR IN MAILBOX
6807 037636 105720          25:     TSTB     (R0)+      ;;FIND END OF MESSAGE
6808 037640 001376          BNE      25           ;;
6809 037642 163700 001224      SUB      $MSGAD,R0     ;;SUB START OF MESSAGE
6810 037646 006200          ASR      RC           ;;GET MESSAGE LNTH IN WORDS
6811 037650 010037 001226      MOV      R0,$MSGLGT    ;;PUT LENGTH IN MAILBOX
6812 037654 012737 000004 001210      MOV      #4,$MSGTYPE   ;;TELL APT TO TAKE MSG.
6813 037662 000413          BR       55           ;;
6814 037664 017637 000004 037710  35:     MOV      24(SP),4$     ;;PUT MSG ADDR IN JSR LINKAGE
6815 037672 062766 000002 000004  ADD      #2,4(SP)      ;;BUMP RETURN ADDRESS
6816 037700 013746 177776      MOV      177776,-(SP)  ;;PUSH 177776 ON STACK
6817 037704 004737 037026      JSR      PC,$TYPE     ;;CALL TYPE MACRO
6818 037710 000000          45:     .WORD    0
6819 037712          55:
6820 037712 105737 040000      105:    TSTB     $FFLG       ;;SHOULD REPORT FATAL ERROR?
6821 037716 001416          BEQ      12$         ;;IF NOT: BR
6822 037720 005737 001230      TST      $ENV         ;;RUNNING UNDER APT?
6823 037724 001413          BEQ      12$         ;;IF NOT: BR
6824 037726 005737 001210      115:    TST      $MSGTYPE     ;;FINISHED LAST MESSAGE?
6825 037732 001375          BNE      11$         ;;IF NOT: WAIT
6826 037734 017637 000004 001212  MOV      24(SP),$FATAL  ;;GET ERROR #
6827 037742 062766 000002 000004  ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
6828 037750 005237 001210      INC      $MSGTYPE     ;;TELL APT TO TAKE ERROR
6829 037754 105037 040000      125:    CLRB     $FFLG       ;;CLEAR FATAL FLAG
6830 037760 105037 037777      CLRB     $LFLG        ;;CLEAR LOG FLAG
6831 037764 105037 037776      CLRB     $MFLG        ;;CLEAR MESSAGE FLAG
6832 037770 012601      MOV      (SP)+,R1     ;;POP STACK INTO R1
6833 037772 012600      MOV      (SP)+,R0     ;;POP STACK INTO R0
6834 037774 000207      RTS      PC          ;;RETURN
6835 037776      000      $MFLG: .BYTE    0      ;;MESSG. FLAG
6836 037777      000      $LFLG: .BYTE    0      ;;LOG FLAG
6837 040000      000      $FFLG: .BYTE    0      ;;FATAL FLAG

```

```

6838      040002      .EVEN
6839      000200      APTSIZE=200
6840      000001      APTENV=001
6841      000100      APTSPOOL=100
6842      000040      APTCSUP=040
6843      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
6844
6845      ;*****
6846      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
6847      ;*OCTAL (ASCII) NUMBER AND TYPE IT.
6848      ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
6849      ;*CALL:
6850      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6851      ;*      TYPOS      ;;CALL FOR TYPEOUT
6852      ;*      .BYTE    N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
6853      ;*      .BYTE    M              ;;M=1 OR 0
6854      ;*
6855      ;*      ;;1=TYPE LEADING ZEROS
6856      ;*      ;;0=SUPPRESS LEADING ZEROS
6857      ;*$STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST

```

```

6858 ;*STYPOS OR STYPOC
6859 ;*CALL:
6860 ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6861 ;*      TYPON                      ;;CALL FOR TYPEOUT
6862 ;*
6863 ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6864 ;*CALL:
6865 ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6866 ;*      TYPOC                      ;;CALL FOR TYPEOUT
6867 ;*
6868 040002 017646 000000          STYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
6869 040006 116637 000001 040225  MOVB     1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
6870 040014 112637 040227          MOVB     (SP)+,SOMODE+1    ;;NUMBER OF DIGITS TO TYPE
6871 040020 062716 000002          ADD      #2,(SP)          ;;ADJUST RETURN ADDRESS
6872 040024 000406          BR      $TYPON
6873 040026 112737 000001 040225  STYPOC: MOVB     #1,SOFILL      ;;SET THE ZERO FILL SWITCH
6874 040034 112737 000006 040227  MOVB     #6,SOMODE+1      ;;SET FOR SIX(6) DIGITS
6875 040042 112737 000005 040224  STYPON: MOVB     #5,SOCNT      ;;SET THE ITERATION COUNT
6876 040050 010346          MOV      R3,-(SP)        ;;SAVE R3
6877 040052 010446          MOV      R4,-(SP)        ;;SAVE R4
6878 040054 010546          MOV      R5,-(SP)        ;;SAVE R5
6879 040056 113704 040227          MOVB     $SOMODE+1,R4    ;;GET THE NUMBER OF DIGITS TO TYPE
6880 040062 005404          NEG      R4
6881 040064 062704 000006          ADD      #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
6882 040070 110437 040226          MOVB     R4,SOMODE      ;;SAVE IT FOR USE
6883 040074 113704 040225          MOVB     $SOFILL,R4     ;;GET THE ZERO FILL SWITCH
6884 040100 016605 000012          MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
6885 040104 005003          CLR      R3             ;;CLEAR THE OUTPUT WORD
6886 040106 006105          1$: ROL      R5           ;;ROTATE MSB INTO "C"
6887 040110 000404          BR      3$
6888 040112 006105          2$: ROL      R5           ;;FORM THIS DIGIT
6889 040114 006105          ROL      R5
6890 040116 006105          ROL      R5
6891 040120 010503          MOV      R5,R3
6892 040122 006103          3$: ROL      R3           ;;GET LSB OF THIS DIGIT
6893 040124 105337 040226          DECB     $SOMODE        ;;TYPE THIS DIGIT?
6894 040130 100016          BPL      7$             ;;BR IF NO
6895 040132 042703 177770          BIC      #177770,R3     ;;GET RID OF JUNK
6896 040136 001002          BNE      4$             ;;TEST FOR 0
6897 040140 005704          TST      R4             ;;SUPPRESS THIS 0?
6898 040142 001403          BEQ      5$             ;;BR IF YES
6899 040144 005204          4$: INC      R4           ;;DON'T SUPPRESS ANYMORE 0'S
6900 040146 052703 000060          BIS      #'0,R3        ;;MAKE THIS DIGIT ASCII
6901 040152 052703 000040          5$: BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
6902 040156 110337 040222          MOVB     R3,#$          ;;SAVE FOR TYPING
6903 040162 104401 040222          TYPE     #$            ;;GO TYPE THIS DIGIT
6904 040166 105337 040224          7$: DECB     $SOCNT      ;;COUNT BY 1
6905 040172 003347          BGT      2$             ;;BR IF MORE TO DO
6906 040174 002402          BLT      6$             ;;BR IF DONE
6907 040176 005204          INC      R4             ;;INSURE LAST DIGIT ISN'T A BLANK
6908 040200 000744          BR      2$             ;;GO DO THE LAST DIGIT
6909 040202 012605          6$: MOV      (SP)+,R5     ;;RESTORE R5
6910 040204 012604          MOV      (SP)+,R4     ;;RESTORE R4
6911 040206 012603          MOV      (SP)+,R3     ;;RESTORE R3
6912 040210 016666 000002 000004  MOV      2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
6913 040216 012616          MOV      (SP)+,(SP)
    
```



```

6914 040220 000002
6915 040222 000
6916 040223 000
6917 040224 000
6918 040225 000
6919 040226 000000
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930 040230 022737 000176 001140
6931 040236 001074
6932 040240 105777 140700
6933 040244 100071
6934 040246 117746 140674
6935 040252 042716 177600
6936 040256 022726 000007
6937 040262 001062
6938 040264 123727 001134 000001
6939 040272 001456
6940
6941 040274 104401 041115
6942 040300 104401 041122
6943 040304 013746 000176
6944 040310 104402
6945 040312 104401 041133
6946 040316 005046
6947 040320 005046
6948 040322 105777 140616
6949 040326 100375
6950
6951 040330 117746 140612
6952 040334 042716 177600
6953
6954
6955
6956 040340 021627 000025
6957 040344 001005
6958 040346 104401 041110
6959 040352 062706 000006
6960 040356 000757
6961
6962
6963 040360 021627 000015
6964 040364 001022
6965 040366 005766 000004
6966 040372 001403
6967 040374 016677 000002 140536
6968 040402 062706 000006
6969 040406 104401 001205

R/I
BS: .BYTE 0
.SOCNT: .BYTE 0
.SOFILL: .BYTE 0
.SOMODE: .WORD 0
.SBTTL TTY INPUT ROUTINE

::RETURN
::STORAGE FOR ASCII DIGIT
::TERMINATOR FOR TYPE ROUTINE
::OCTAL DIGIT COUNTER
::ZERO FILL SWITCH
::NUMBER OF DIGITS TO TYPE

:*****
:ENABL LSB

:*****
:SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
:ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
:SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
:WHEN OPERATING IN TTY FLAG MODE.
$CKSWR: CMP #SWREG,SWR :: IS THE SOFT-SWR SELECTED?
BNE 15$ :: BRANCH IF NO
TSTB 2$TKS :: CHAR THERE?
BPL 15$ :: IF NO, DON'T WAIT AROUND
MOV#B 2$TKB,-(SP) :: SAVE THE CHAR
BIC #177,(SP) :: STRIP-OFF THE ASCII
CMP #7,(SP)+ :: IS IT A CONTROL G?
BNE 15$ :: NO, RETURN TO USER
CMPB $AUTOB,#1 :: ARE WE RUNNING IN AUTO-MODE?
BEQ 15$ :: BRANCH IF YES

$GTSWR: TYPE , $CNTLG :: ECHO THE CONTROL-G (1G)
TYPE , $MSWR :: TYPE CURRENT CONTENTS
MOV $SWREG,-(SP) :: SAVE SWREG FOR TYPEOUT
TYPOC :: GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE , $MNEW :: PROMPT FOR NEW SWR
19$: CLR -(SP) :: CLEAR COUNTER
CLR -(SP) :: THE NEW SWR
7$: TSTB 2$TKS :: CHAR THERE?
BPL 7$ :: IF NOT TRY AGAIN

MOV#B 2$TKB,-(SP) :: PICK UP CHAR
BIC #177,(SP) :: MAKE IT 7-BIT ASCII

9$: CMP (SP),#25 :: IS IT A CONTROL-U?
BNE 10$ :: BRANCH IF NOT
TYPE , $CNTLU :: YES, ECHO CONTROL-U (1U)
20$: ADD #6,SP :: IGNORE PREVIOUS INPUT
BR 19$ :: LET'S TRY IT AGAIN

10$: CMP (SP),#15 :: IS IT A <CR>?
BNE 16$ :: BRANCH IF NO
TST 4(SP) :: YES, IS IT THE FIRST CHAR?
BEQ 11$ :: BRANCH IF YES
MOV 2(SP),2$SWR :: SAVE NEW SWR
11$: ADD #6,SP :: CLEAR UP STACK
14$: TYPE , $CRLF :: ECHO <CR> AND <LF>

```

```

6970 040412 123727 001135 000001      CMPB  $INTAG,#1      ;;RE-ENABLE TTY KBD INTERRUPTS?
6971 040420 001003                      BNE   15$           ;;BRANCH IF NOT
6972 040422 012777 000100 140514      MOV   #100,2$TKS   ;;RE-ENABLE TTY KBD INTERRUPTS
6973 040430 000002                      RTI                      ;;RETURN
6974 040432 004737 037240      15$: JSR   PC,$TYPEC  ;;ECHO CHAR
6975 040436 021627 000060      16$: CMP   (SP),#60  ;;CHAR < 0?
6976 040442 002420                      BLT   18$           ;;BRANCH IF YES
6977 040444 021627 000067      CMP   (SP),#67    ;;CHAR > 7?
6978 040450 003015                      BGT   18$           ;;BRANCH IF YES
6979 040452 042726 000060      BIC   #60,(SP)+   ;;STRIP-OFF ASCII
6980 040456 005766 000002      TST   2(SP)       ;;IS THIS THE FIRST CHAR
6981 040462 001403                      BEQ   17$           ;;BRANCH IF YES
6982 040464 006316                      ASL   (SP)        ;;NO, SHIFT PRESENT
6983 040466 006316                      ASL   (SP)        ;;CHAR OVER TO MAKE
6984 040470 006316                      ASL   (SP)        ;;ROOM FOR NEW ONE.
6985 040472 005266 000002      17$: INC  2(SP)     ;;KEEP COUNT OF CHAR
6986 040476 056616 177776      BIS   -2(SP),(SP) ;;SET IN NEW CHAR
6987 040502 000707                      BR    7$           ;;GET THE NEXT ONE
6988 040504 104401 001204      18$: TYPE $QUES    ;;TYPE ?(CR)(LF)
6989 040510 000720                      BR    20$         ;;SIMULATE CONTROL-U
6990 .DSABL LSB

```

```

6991
6992
6993 *****
6994 *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
6995 *CALL:
6996 *      RDCHR      ;; INPUT A SINGLE CHARACTER FROM THE TTY
6997 *      RETURN HERE ;; CHARACTER IS ON THE STACK
6998 *                ;; WITH PARITY BIT STRIPPED OFF
6999
7000

```

```

7001 040512 011646      $RDCHR: MOV   (SP),-(SP)  ;;PUSH DOWN THE PC
7002 040514 016666 000004 000002  MOV   4(SP),2(SP) ;;SAVE THE PS
7003 040522 105777 140416  1$: TSTB 2$TKS    ;;WAIT FOR
7004 040526 100375                      BPL   1$           ;;A CHARACTER
7005 040530 117766 140412 000004  MOVB  2$TKB,4(SP) ;;READ THE TTY
7006 040536 042766 177600 000004  BIC   #1C(177),4(SP) ;;GET RID OF JUNK IF ANY
7007 040544 026627 000004 000023  CMP   4(SP),#23  ;;IS IT A CONTROL-S?
7008 040552 001013                      BNE   3$           ;;BRANCH IF NO
7009 040554 105777 140364      2$: TSTB 2$TKS    ;;WAIT FOR A CHARACTER
7010 040560 100375                      BPL   2$           ;;LOOP UNTIL ITS THERE
7011 040562 117746 140360  MOVB  2$TKB,-(SP) ;;GET CHARACTER
7012 040566 042716 177600  BIC   #1C177,(SP) ;;MAKE IT 7-BIT ASCII
7013 040572 022627 000021  CMP   (SP)+,#21  ;;IS IT A CONTROL-Q?
7014 040576 001366                      BNE   2$           ;;IF NOT DISCARD IT
7015 040600 000750                      BR    1$           ;;YES, RESUME
7016 040602 026627 000004 000140  3$: CMP   4(SP),#140 ;;IS IT UPPER CASE?
7017 040610 002407                      BLT   4$           ;;BRANCH IF YES
7018 040612 026627 000004 000175  CMP   4(SP),#175 ;;IS IT A SPECIAL CHAR?
7019 040620 003003                      BGT   4$           ;;BRANCH IF YES
7020 040622 042766 000040 000004  BIC   #40,4(SP)  ;;MAKE IT UPPER CASE
7021 040630 000002      4$: RTI          ;;GO BACK TO USER

```

```

7022 *****
7023 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
7024 *CALL:
7025 *      RDLIN     ;; INPUT A STRING FROM THE TTY

```

```

7026      :*      RETURN HERE      :: ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
7027      :*      :: TERMINATOR WILL BE A BYTE OF ALL 0'S
7028
7029 040632 010346 SRDLIN: MOV R3, -(SP)      :: SAVE R3
7030 040634 005046 CLR -(SP)      :: CLEAR THE RUBOUT KEY
7031 040636 012703 041066 1S: MOV #STTYIN, R3      :: GET ADDRESS
7032 040642 022703 041110 2S: CMP #STTYIN+22, R3      :: BUFFER FULL?
7033 040646 101456 BLOS 4S      :: BR IF YES
7034 040650 104410 RDCR      :: GO READ ONE CHARACTER FROM THE TTY
7035 040652 112613 MOVB (SP)+, (R3)      :: GET CHARACTER
7036 040654 122713 000177 10S: CMPB #177, (R3)      :: IS IT A RUBOUT
7037 040660 001022 BNE 5S      :: BR IF NO
7038 040662 005716 TST (SP)      :: IS THIS THE FIRST RUBOUT?
7039 040664 001007 BNE 6S      :: BR IF NO
7040 040666 112737 000134 041064 MOVB #' \, 9S      :: TYPE A BACK SLASH
7041 040674 104401 041064 TYPE 9S
7042 040700 012716 177777 MOV 6-1, (SP)      :: SET THE RUBOUT KEY
7043 040704 005303 6S: DEC R3      :: BACKUP BY ONE
7044 040706 020327 041066 CMP R3, #STTYIN      :: STACK EMPTY?
7045 040712 103434 BLO 4S      :: BR IF YES
7046 040714 111337 041064 MOVB (R3), 9S      :: SETUP TO TYPEOUT THE DELETED CHAR.
7047 040720 104401 041064 TYPE 9S      :: GO TYPE
7048 040724 000746 BR 2S      :: GO READ ANOTHER CHAR.
7049 040726 005716 5S: TST 5P      :: RUBOUT KEY SET?
7050 040730 001406 BEQ 7S      :: BR IF NO
7051 040732 112737 000134 041064 MOVB #' \, 9S      :: TYPE A BACK SLASH
7052 040740 104401 041064 TYPE 9S
7053 040744 005016 CLR (SP)      :: CLEAR THE RUBOUT KEY
7054 040746 122713 000025 7S: CMPB #25, (R3)      :: IS CHARACTER A CTRL U?
7055 040752 001003 BNE 8S      :: BR IF NO
7056 040754 104401 041110 TYPE #SCNTLU      :: TYPE A CONTROL "U"
7057 040760 000726 BR 1S      :: GO START OVER
7058 040762 122713 000022 8S: CMPB #22, (R3)      :: IS CHARACTER A "r"?
7059 040766 001011 BNE 3S      :: BRANCH IF NO
7060 040770 105013 CLRB (R3)      :: CLEAR THE CHARACTER
7061 040772 104401 001205 TYPE #SCRLF      :: TYPE A "CR" & "LF"
7062 040776 104401 041066 TYPE #STTYIN      :: TYPE THE INPUT STRING
7063 041002 000717 BR 2S      :: GO PICKUP ANOTHER CHARACTER
7064 041004 104401 001204 4S: TYPE #QUES      :: TYPE A '?'
7065 041010 000712 BR 1S      :: CLEAR THE BUFFER AND LOOP
7066 041012 111337 041064 3S: MOVB (R3), 9S      :: ECHO THE CHARACTER
7067 041016 104401 041064 TYPE 9S
7068 041022 122723 000015 CMPB #15, (R3)+      :: CHECK FOR RETURN
7069 041026 001305 BNE 2S      :: LOOP IF NOT RETURN
7070 041030 105063 177777 CLRB -1(R3)      :: CLEAR RETURN (THE 15)
7071 041034 104401 001206 TYPE #LF      :: TYPE A LINE FEED
7072 041040 005726 TST (SP)+      :: CLEAN RUBOUT KEY FROM THE STACK
7073 041042 012603 MOV (SP)+, R3      :: RESTORE R3
7074 041044 011646 MOV (SP), -(SP)      :: ADJUST THE STACK AND PUT ADDRESS OF THE
7075 041046 016666 000004 000002 MOV 4(SP), 2(SP)      :: FIRST ASCII CHARACTER ON IT
7076 041054 012766 041066 000004 MOV #STTYIN, 4(SP)
7077 041062 000002 RTI      :: RETURN
7078 041064 000 9S: .BYTE 0      :: STORAGE FOR ASCII CHAR. TO TYPE
7079 041065 000 .BYTE 0      :: TERMINATOR
7080 041066 000022 000004 STTYIN: .BLKB 22      :: RESERVE 22 BYTES FOR TTY INPUT
7081 041110 052536 005015 000 SCNTLU: .ASCIZ /U/<15><12>      :: CONTROL "U"

```

```

7082 041115 136 006507 000012 $CNTLG: .ASCIZ /IG<15><12> /;CONTROL "G"
7083 041122 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
7084 041130 020075 0C
7085 041133 040 047040 053505 $MNEW: .ASCIZ / NEW = /
7086 041140 036440 000040
7087 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
7088
7089 ;:*****
7090 ;:THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
7091 ;:CHANGE IT TO BINARY.
7092 ;:THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
7093 ;:OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
7094 ;:FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
7095 ;:THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
7096 ;:CALL:
7097 ;: * RDOCT ;: READ AN OCTAL NUMBER
7098 ;: * RETURN HERE ;: LOW ORDER BITS ARE ON TOP OF THE STACK
7099 ;: * ;: HIGH ORDER BITS ARE IN $HIOCT
7100
7101 041144 011646 000004 000002 $RDOCT: MOV (SP),-(SP) ;: PROVIDE SPACE FOR THE
7102 041146 016666 000004 000002 MOV 4(SP),2(SP) ;: INPUT NUMBER
7103 041154 010046 MOV RO,-(SP) ;: PUSH RO ON STACK
7104 041156 010146 MOV R1,-(SP) ;: PUSH R1 ON STACK
7105 041160 010246 MOV R2,-(SP) ;: PUSH R2 ON STACK
7106 041162 104411 1$: RDLIN ;: READ AN ASCIZ LINE
7107 041164 012600 MOV (SP)+,RO ;: GET ADDRESS OF 1ST CHARACTER
7108 041166 010037 041272 MOV RO,$$ ;: AND SAVE IT
7109 041172 005001 CLR R1 ;: CLEAR DATA WORD
7110 041174 005002 CLR R2
7111 041176 112046 2$: MOVB (RO)+,-(SP) ;: PICKUP THIS CHARACTER
7112 041200 001420 BEQ $$ ;: IF ZERO GET OUT
7113 041202 122716 000060 CMPB #'0,(SP) ;: MAKE SURE THIS CHARACTER
7114 041206 003026 BGT 4$ ;: IS AN OCTAL DIGIT
7115 041210 122716 000067 CMPB #'7,(SP)
7116 041214 002423 BLT 4$
7117 041216 006301 ASL R1 ;: *2
7118 041220 006102 ROL R2
7119 041222 006301 ASL R1 ;: *4
7120 041224 006102 ROL R2
7121 041226 006301 ASL R1 ;: *8
7122 041230 006102 ROL R2
7123 041232 042716 177770 BIC #'C7,(SP) ;: STRIP THE ASCII JUNK
7124 041236 062601 ADD (SP)+,R1 ;: ADD IN THIS DIGIT
7125 041240 000756 BR 2$ ;: LOOP
7126 041242 005726 3$: TST (SP)+ ;: CLEAN TERMINATOR FROM STACK
7127 041244 010166 000012 MOV R1,12(SP) ;: SAVE THE RESULT
7128 041250 010237 041302 MOV R2,$HIOCT
7129 041254 012602 MOV (SP)+,R2 ;: POP STACK INTO R2
7130 041256 012601 MOV (SP)+,R1 ;: POP STACK INTO R1
7131 041260 012600 MOV (SP)+,RO ;: POP STACK INTO RO
7132 041262 000002 RTI ;: RETURN
7133 041264 005726 4$: TST (SP)+ ;: CLEAN PARTIAL FROM STACK
7134 041266 105010 CLRB (RO) ;: SET A TERMINATOR
7135 041270 104401 TYPE ;: TYPE UP THRU THE BAD CHAR.
7136 041272 000000 5$: .WORD 0
7137 041274 :04401 001204 TYPE ,SQUES ;: "?" "CR" & "LF"

```

M12

```

7138 041300 000730          BR      15          ;; TRY AGAIN
7139 041302 000000          $HI OCT: WORD 0          ;; HIGH ORDER BITS GO HERE
7140                                     .SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
7141
7142                                     ;;*****
7143                                     ;;THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
7144                                     ;;UNSIGNED OCTAL ASCII NUMBER.
7145                                     ;;CALL
7146                                     ;;
7147                                     ;;      MOV      #PNTR, -(SP)          ;; POINTER TO LOW WORD OF BINARY NUMBER
7148                                     ;;      JSR      PC, 2#$DB20          ;; CALL THE ROUTINE
7149                                     ;;      RETURN          ;; THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK
7150
7151 041304 104413          $DB20: SAVREG          ;; SAVE ALL REGISTERS
7152 041306 016601 000002          MOV      2(SP), R1          ;; PICKUP THE POINTER TO LOW WORD
7153 041312 012705 041423          MOV      #SOCTVL+13., R5          ;; POINTER TO DATA TABLE
7154 041316 012704 000014          MOV      #12., R4          ;; DO ELEVEN CHARACTERS
7155 041322 012703 177770          MOV      #1C7, R3          ;; MASK
7156 041326 012100          MOV      (R1)+, R0          ;; LOWER WORD
7157 041330 012101          MOV      (R1)+, R1          ;; HIGH WORD
7158 041332 005002          CLR      R2          ;; TERMINATOR
7159 041334 110245          15:  MOVB   R2, -(R5)          ;; PUT CHARACTER IN DATA TABLE
7160 041336 010002          MOV      R0, R2          ;; GET THIS DIGIT
7161 041340 0C5304          DEC      R4          ;; COUNT THIS CHARACTER
7162 041342 003007          BGT     35          ;; BR IF NOT THE LAST DIGIT
7163 041344 001405          BEQ     25          ;; BR IF IT IS THE LAST DIGIT
7164 041346 005205          INC     R5          ;; ALL DIGITS DONE-ADJUST POINTER FOR FIRST
7165 041350 010566 000002          MOV     R5, 2(SP)          ;; ASCII CHAR. & PUT IT ON THE STACK
7166 041354 104414          RESREG          ;; RESTORE ALL REGISTERS
7167 041356 000207          RTS     PC          ;; RETURN TO USER
7168 041360 006203          25:  ASR     R3          ;; POSITION THE MASK FOR THE LAST DIGIT
7169 041362 006001          35:  ROR     R1          ;; POSITION THE BINARY NUMBER FOR
7170 041364 076000          ROR     R0          ;; THE NEXT OCTAL DIGIT
7171 041366 006001          ROR     R1
7172 041370 006000          ROR     R0
7173 041372 006001          ROR     R1
7174 041374 006000          ROR     R0
7175 041376 040302          BIC     R3, R2          ;; MASK OUT ALL JUNK
7176 041400 062702 000060          ADD     #'0, R2          ;; MAKE THIS CHAR. ASCII
7177 041404 000753          BR      15          ;; GO PUT IT IN THE DATA TABLE
7178 041406 000016          $OCTVL: .BLKB 14          ;; RESERVE DATA TABLE
7179                                     .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
7180
7181                                     ;;*****
7182                                     ;;THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
7183                                     ;;DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
7184                                     ;;POSITIVE.
7185                                     ;;CALL
7186                                     ;;
7187                                     ;;      MOV      #PNTR, -(SP)          ;; POINTER TO LOW WORD OF BINARY NUMBER
7188                                     ;;      JSR      PC, 2#$DB20          ;; CALL THE ROUTINE
7189                                     ;;      RETURN          ;; THE FIRST ADDRESS OF ASCII
7190                                     ;; IS ON THE STACK
7191
7192 041424 104413          $DB20: SAVREG          ;; SAVE REGISTERS
7193 041426 016602 000002          MOV     2(SP), R2          ;; PICKUP THE DATA POINTER
  
```

```

7194 041432 012700 041604      MOV      #SDECVL,R0      ;;GET ADDRESS OF "SDECVL" STRING
7195 041436 010066 000002      MOV      R0,2(SP)      ;;PUT ADDRESS OF ASCII STRING ON STACK
7196 041442 012201      MOV      (R2)+,R1      ;;PICKUP THE BINARY NUMBER
7197 041444 012202      MOV      (R2)+,R2
7198 041446 012737 0C0012 041522      MOV      #10,R4      ;;SET UP TO DO 10 CONVERSIONS
7199 041454 012704 041534      MOV      #STNPNR,R4      ;;ADDRESS OF TEN POWER
7200 041460 012705 041536      MOV      #STNPNR+2,R5
7201 041464 005003      15:     CLR      R3      ;;CLEAR PARTIAL
7202 041466 161401      23:     SUB      (R4),R1      ;;SUBTRACT TEN POWER
7203 041470 005602      SBC      R2
7204 041472 161502      SUB      (R5),R2
7205 041474 002402      BLT      35:      ;;BR IF TEN POWER TOO LARGE
7206 041476 005203      INC      R3      ;;ADD 1 TO PARTIAL
7207 041500 000772      BR      25:      ;;LOOP
7208 041502 062401      35:     ADD      (R4)+,R1      ;;RESTORE SUBTRACTED VALUE
7209 041504 005502      ADC      R2
7210 041506 062402      ADD      (R4)+,R2
7211 041510 022525      CMP      (R5)+,(R5)+  ;;MOVE TO NEXT TEN POWER
7212 041512 052703 000060      BIS      #0,R3      ;;CHANGE PARTIAL TO ASCII
7213 041516 110320      MOVSB   R3,(R0)+      ;;SAVE IT
7214 041520 005327      DEC      (PC)+      ;;DONE?
7215 041522 000000      45:     .WORD   0
7216 041524 001357      BNE     15:      ;;BR IF NO
7217 041526 105020      CLRB   (R0)+      ;;TERMINATE
7218 041530 104414      RESREG      ;;RESTORE REGISTERS
7219 041532 000207      RTS      PC      ;;RETURN
7220 041534 145000      STNPNR: 145000      ;;1.0E09
7221 041536 035632      35632
7222 041540 160400      160400      ;;1.0E08
7223 041542 002765      2765
7224 041544 113200      113200      ;;1.0E07
7225 041546 000230      230
7226 041550 041100      041100      ;;1.0E06
7227 041552 000017      17
7228 041554 103240      103240      ;;1.0E05
7229 041556 000001      1
7230 041560 023420      23420      ;;1.0E04
7231 041562 000000      0
7232 041564 001750      1750      ;;1.0E03
7233 041566 000000      0
7234 041570 000144      144      ;;1.0E02
7235 041572 000000      0
7236 041574 000012      12      ;;1.0E01
7237 041576 000000      0
7238 041600 000001      1      ;;1.0E00
7239 041602 000000      0
7240 041604 000014      SDECVL: .BLKB 12.      ;;RESERVE STORAGE FOR ASCII STRING
7241      .SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCII ROUTINE
7242
7243      ;*****
7244      ;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
7245      ;UNSIGNED DECIMAL ASCII NUMBER.
7246      ;CALL
7247      ;      MOV      NUMBER,-(SP)      ;;PUT BINARY NUMBER ON THE STACK
7248      ;      JSR      PC,@#SB20      ;;CALL
7249      ;      RETURN      ;;ADDRESS OF THE 1ST ASCII CHAR.IS ON THE STACK
    
```

```

7250
7251
7252 041620 016637 000002 041650 $$B2D: MOV 2(SP),1$ ;;SAVE BINARY NUMBER
7253 041626 012746 041650 MOV 1$,-(SP) ;;SET POINTER
7254 041632 004737 041424 JSR PC,$$SDB2D ;;CALL DOUBLE LENGTH CONVERT
7255 041636 062716 000005 ADD 1$(SP) ;;ONLY ALLOW FIVE CHARACTERS
7256 041642 012666 000002 MOV (SP)+,2(SP) ;;PICKUP POINTER
7257 041646 000207 RTS PC ;;RETURN
7258 041650 000000 000000 1$: .WORD 0,0
7259 .SBTTL TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
7260
7261 *****
7262 *THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
7263 *LEADING NUMBERS.
7264 *CALL
7265 * MOV $NUMADR,-(SP) ;;FIRST ADDRESS OF ASCIZ STRING
7266 * JSR PC,$$SUPRS
7267
7268
7269 041654 010046 $$SUPRS: MOV R0,-(SP) ;;SAVE R0
7270 041656 016600 000004 MOV 4(SP),R0 ;;PICKUP THE POINTER
7271 041662 105710 1$: TSTB (R0) ;;TERMINATOR?
7272 041664 001403 BEQ 2$ ;;BR IF YES
7273 041666 122720 000060 CMPB 8'0,(R0)+ ;;IS THIS AN ASCII "0" ?
7274 041672 001773 BEQ 1$ ;;BR IF YES
7275 041674 005300 2$: DEC R0 ;;BACKUP BY "1"
7276 041676 010037 041704 MOV R0,3$ ;;SAVE FOR TYPING
7277 041702 004401 TYPE ;;GO TYPE
7278 041704 000000 3$: .WORD 0 ;;ASCIZ POINTER GOES HERE
7279 041706 012600 MOV (SP)+,R0 ;;RESTORE R0
7280 041710 012616 MOV (SP)+,(SP) ;;RESTORE THE STACK
7281 041712 000207 RTS PC ;;RETURN
7282 .SBTTL INTEGER MULTIPLY ROUTINE
7283
7284 *****
7285 *CALL
7286 * MOV MULTIPLER,-(SP)
7287 * MOV MULTIPLICAND,-(SP)
7288 * JSR PC,$$MULT
7289 * RETURN ;;PRODUCT IS ON THE STACK
7290
7291 *
7292 * STACK PRODUCT
7293 * -----
7294 * TOP LSB'S
7295 * +2 MSB'S
7296
7297 041714 010046 $MULT: MOV R0,-(SP) ;;PUSH R0 ON STACK
7298 041716 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
7299 041720 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
7300 041722 005046 CLR -(SP) ;;CLEAR THE SIGN KEY
7301 041724 016601 000012 MOV 12(SP),R1 ;;GET THE MULTIPLICAND
7302 041730 100002 1$: BPL 1$ ;;BR IF PLUS
7303 041732 005216 INC (SP) ;;SET THE SIGN KEY
7304 041734 005401 NEG R1 ;;MAKE THE MULTIPLICAND POSTIVE
7305 041736 016602 000014 1$: MOV 14(SP),R2 ;;GET THE MULTIPLIER

```

```

7306 041742 100002          BPL      2$          ;; BR IF PLUS
7307 041744 005316          DEC      (SP)       ;; UPDATE THE SIGN KEY
7308 041746 005402          NEG      R2         ;; MAKE THE MULTIPLIER POSTIVE
7309 041750 012746 000021 2$: MOV      #17.,-(SP)  ;; SET THE LOOP COUNT
7310 041754 005000          CLR      R0        ;; SETUP FOR THE MULTIPLY LOOP
7311 041756 103001          3$: BCC     4$          ;; DON'T ADD IF MULTIPLICAND = 0
7312 041760 060200          ADD     R2,R0
7313 041762 006000          4$: ROR     R0
7314 041764 006001          ROR     R1
7315 041766 005316          DEC     (SP)
7316 041770 001372          BNE     3$
7317 041772 022616          CMP     (SP)+,(SP)  ;; SHOULD PRODUCT BE NEGATIVE?
7318 041774 001403          SEQ     5$          ;; GO TO EXIT IF NO
7319 041776 005400          NEG     R0
7320 042000 005401          NEG     R1
7321 042002 005600          SBC     R0
7322 042004 005726          5$: TST     (SP)+    ;; CLEAR SIGN INFO. OFF OF STACK
7323 042006 010066 000012  MOV     R0,12(SP)  ;; PUT THE PRODUCT ON THE STACK (MSB'S)
7324 042012 010166 000010  MOV     R1,10(SP)  ;; LSB'S
7325 042016 012602          MOV     (SP)+,R2   ;; POP STACK INTO R2
7326 042020 012601          MOV     (SP)+,R1   ;; POP STACK INTO R1
7327 042022 012600          MOV     (SP)+,R0   ;; POP STACK INTO R0
7328 042024 000207          RTS     PC

```

.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

```

7329
7330
7331 *****
7332 *SAVE R0-R5
7333 *CALL:
7334 * SAVREG
7335 *UPON RETURN, FROM $SAVREG THE STACK WILL LOOK LIKE:
7336 *
7337 *TOP---(+16)
7338 * +2---(+18)
7339 * +4---R5
7340 * +6---R4
7341 * +8---R3
7342 * +10---R2
7343 * +12---R1
7344 * +14---R0

```

```

7345
7346 $SAVREG:
7347 042026 010046          MOV     R0,-(SP)   ;; PUSH R0 ON STACK
7348 042030 010146          MOV     R1,-(SP)   ;; PUSH R1 ON STACK
7349 042032 010246          MOV     R2,-(SP)   ;; PUSH R2 ON STACK
7350 042034 010346          MOV     R3,-(SP)   ;; PUSH R3 ON STACK
7351 042036 010446          MOV     R4,-(SP)   ;; PUSH R4 ON STACK
7352 042040 010546          MOV     R5,-(SP)   ;; PUSH R5 ON STACK
7353 042042 016646 000022  MOV     22(SP),-(SP)  ;; SAVE PS OF MAIN FLOW
7354 042046 016646 000022  MOV     22(SP),-(SP)  ;; SAVE PC OF MAIN FLOW
7355 042052 016646 000022  MOV     22(SP),-(SP)  ;; SAVE PS OF CALL
7356 042056 016646 000022  MOV     22(SP),-(SP)  ;; SAVE PC OF CALL
7357 042062 000002          RTI

```

```

7358
7359 *RESTORE R0-R5
7360 *CALL:
7361 * RESREG

```


7362 042064
 7363 042064 012666 000022
 7364 042070 012666 000022
 7365 042074 012666 000022
 7366 042100 012666 000022
 7367 042104 012605
 7368 042106 012604
 7369 042110 012603
 7370 042112 012602
 7371 042114 012601
 7372 042116 012600
 7373 042120 000002

\$RESREG:
 MOV (SP)+,R22(SP) ;; RESTORE PC OF CALL
 MOV (SP)+,R22(SP) ;; RESTORE PS OF CALL
 MOV (SP)+,R22(SP) ;; RESTORE PC OF MAIN FLOW
 MOV (SP)+,R22(SP) ;; RESTORE PS OF MAIN FLOW
 MOV (SP)+,R5 ;; POP STACK INTO R5
 MOV (SP)+,R4 ;; POP STACK INTO R4
 MOV (SP)+,R3 ;; POP STACK INTO R3
 MOV (SP)+,R2 ;; POP STACK INTO R2
 MOV (SP)+,R1 ;; POP STACK INTO R1
 MOV (SP)+,R0 ;; POP STACK INTO R0
 RTI

.SBTTL TRAP DECODER

 *THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 *AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 *OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 *GO TO THAT ROUTINE.

7382 042122 010046
 7383 042124 016600 000002
 7384 042130 005740
 7385 042132 111000
 7386 042134 006300
 7387 042136 016000 042156
 7388 042142 000200

\$TRAP: MOV R0, -(SP) ;; SAVE R0
 MOV 2(SP), R0 ;; GET TRAP ADDRESS
 TST -(R0) ;; BACKUP BY 2
 MOVB (R0), R0 ;; GET RIGHT BYTE OF TRAP
 ASL R0 ;; POSITION FOR INDEXING
 MOV \$TRPAD(R0), R0 ;; INDEX TO TABLE
 RTS R0 ;; GO TO ROUTINE

:: THIS IS USE TO HANDLE THE "GETPRI" MACRO

7393 042144 011646
 7394 042146 016666 000004 000002
 7395 042154 000002

\$TRAP2: MOV (SP), -(SP) ;; MOVE THE PC DOWN
 MOV 4(SP), 2(SP) ;; MOVE THE PSW DOWN
 RTI ;; RESTORE THE PSW

.SBTTL TRAP TABLE

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 *BY THE "TRAP" INSTRUCTION.

7403
 7404 042156 042144
 7405 042160 037026
 7406 042162 040026
 7407 042164 040002
 7408 042166 040042
 7409 042170 037310
 7410
 7411 042172 040300
 7412
 7413 042174 040230
 7414 042176 040512
 7415 042200 040632
 7416 042202 041144
 7417 042204 042026

ROUTINE

 \$TRPAD: .WORD \$TRAP2
 \$TYPE ;; CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
 \$TYPOC ;; CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 \$TYFOS ;; CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
 \$TYPON ;; CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
 \$TYPDS ;; CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
 \$GTSWR ;; CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
 \$CKSWR ;; CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
 \$RDCHR ;; CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
 \$RDLIN ;; CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
 \$RDOCT ;; CALL=RDOCT TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
 \$SAVREG ;; CALL=SAVREG TRAP+13(104413) SAVE R0-R5 ROUTINE

E13

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2 MACY11 27(732) 01-OCT-76 10:38 PAGE 139
DZR6IB.P11 TRAP TABLE

SEC 0140

7418 042206 042064
7419 042210 035576
7420

\$RESREG ::CALL=RESREG TRAP+14(104414) RESTORE R0-R5 ROUTINE
SCOPI\$::CALL=SCOPI TRAP+15(104415) INTERNAL LOOP ON ERROR

7421				
7422				
7423				
7424				
7425	042212	005015	047125	041111
7426	042220	051525	051040	030113
7427	042226	020066	051104	053111
7428	042234	020105	044504	043501
7429	042242	047516	052123	041511
7430	042250	005015	050011	051101
7431	042256	020124	004462	030460
7432	042264	006462	012	
7433	042267	015	004412	025052
7434	042274	025052	020052	040503
7435	042302	052125	047511	020116
7436	042310	025052	025052	006452
7437	042316	012		
7438	042317	015	052012	044510
7439	042324	020123	051120	043517
7440	042332	040522	020115	044123
7441	042340	052517	042114	020040
7442	042346	042502	044040	046101
7443	042354	042524	020104	047117
7444	042362	054514	040440	020124
7445	042370	044124	020105	047105
7446	042376	104		
7447	042377	015	047412	020106
7448	042404	020101	040520	051523
7449	042412	020054	052117	042510
7450	042420	053522	051511	020105
7451	042426	042510	042101	051105
7452	042434	020123	051127	052111
7453	042442	042524	020116	047117
7454	042450	052040	042510	
7455	042454	005015	044504	045523
7456	042462	041440	051101	051124
7457	042470	042111	042507	046440
7458	042476	054501	041040	020105
7459	042504	042514	052106	044440
7460	042512	020116	047101	052440
7461	042520	043116	051117	040515
7462	042526	052124	042105	051440
7463	042534	040524	042524	005015
7464	042542	005015	046101	047523
7465	042550	020054	051104	053111
7466	042556	051505	052040	020117
7467	042564	042502	052040	051505
7468	042572	042524	020104	044123
7469	042600	052517	042114	044040
7470	042606	053101	035105	005015
7471	042614	005015	027101	044040
7472	042622	040505	051504	046440
7473	042630	047101	040525	046114
7474	042636	020131	047514	042101
7475	042644	042105		
7476	042646	005015	027102	041440

.SBTTL SERVICE MESSAGES

MSG1: .ASCII <CR><LF>/UNIBUS RK06 DRIVE DIAGNOSTIC/

.ASCII <CR><LF>/ PART 2 012/<CR><LF>

.ASCII <CR><LF>/ ***** CAUTION *****/<CR><LF>

.ASCII <CR><LF>/THIS PROGRAM SHOULD BE HALTED ONLY AT THE END/

.ASCII <CR><LF>/OF A PASS, OTHERWISE HEADERS WRITTEN ON THE/

.ASCII <CR><LF>/DISK CARTRIDGE MAY BE LEFT IN AN UNFORMATTED STATE/<CR><LF>

.ASCII <CR><LF>/ALSO, DRIVES TO BE TESTED SHOULD HAVE:/<CR><LF>

.ASCII <CR><LF>/A. HEADS MANUALLY LOADED/

.ASCII <CR><LF>/B. CORRECT PORT SELECTED/

7477	042654	051117	042522	052103	
7478	042662	050040	051117	020124	
7479	042670	042523	042514	052103	
7480	042676	042105			
7481	042700	005015	027103	053440	.ASCII <CR><LF>/C. WRITE LOCK DISABLED/
7482	042706	044522	042524	046040	
7483	042714	041517	020113	044504	
7484	042722	040523	046102	042105	
7485	042730	005015	027104	042040	.ASCII <CR><LF>/D. DRIVE READY INDICATOR ON/<CR><LF>
7486	042736	044522	042526	051040	
7487	042744	040505	054504	044440	
7488	042752	042116	041511	052101	
7489	042760	051117	047440	006516	
7490	042766	012			
7491	042767	015	042012	044522	.ASCII <CR><LF>/DRIVES NOT TO BE TESTED MUST HAVE/
7492	042774	042526	020123	047516	
7493	043002	020124	047524	041040	
7494	043010	020105	042524	052123	
7495	043016	042105	046440	051525	
7496	043024	020124	040510	042526	
7497	043032	005015	047502	044124	.ASCIZ <CR><LF>/BOTH PORTS DESELECTED/
7498	043040	050040	051117	051524	
7499	043046	042040	051505	046105	
7500	043054	041505	042524	000104	
7501					
7502					
7503	043062	005015	042502	051440	MSG2: .ASCIZ <CR><LF>/BE SURE TO PUT SCRATCH PACK IN DRIVE D/
7504	043070	051125	020105	047524	
7505	043076	050040	052125	051440	
7506	043104	051103	052101	044103	
7507	043112	050040	041501	020113	
7508	043120	047111	042040	044522	
7509	043126	042526	030040	000	
7510	043133	015	042012	044522	MSG3: .ASCIZ <CR><LF>/DRIVE(S) TO BE TESTED: /
7511	043140	042526	051450	020051	
7512	043146	047524	041040	020105	
7513	043154	042524	052123	042105	
7514	043162	020072	000		
7515	043165	015	052012	050131	MSG4: .ASCIZ <CR><LF>/TYPE BUSS ADDRESS IF NOT 177440 /
7516	043172	020105	052502	051523	
7517	043200	040440	042104	042522	
7518	043206	051523	044440	020106	
7519	043214	047516	020124	033461	
7520	043222	032067	030064	000040	
7521	043230	005015	054524	042520	MSG5: .ASCIZ <CR><LF>/TYPE CONTROLLER INTERRUPT VECTOR IF NOT 210 /
7522	043236	041440	047117	051124	
7523	043244	046117	042514	020122	
7524	043252	047111	042524	051122	
7525	043260	050125	020124	042526	
7526	043266	052103	051117	044440	
7527	043274	020106	047516	020124	
7528	043302	030462	020060	000	
7529	043307	015	044412	052116	MSG6: .ASCIZ <CR><LF>/INTERRUPT OCCURRED AT PC=/
7530	043314	051105	052522	052120	
7531	043322	047440	041503	051125	
7532	043330	042522	020104	052101	

H13

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZR6IB.P11 SERVICE MESSAGES

MACY11 27(732) 01-OCT-76 10:38 PAGE 142

SEQ 0143

7533	043336	050040	036503	000	
7534	043343	015	042012	044522	MSG7: .ASCIZ <CR><LF>/DRIVE 0 WILL NOT BE TESTED/
7535	043350	042526	030040	053440	
7536	043356	046111	020114	047516	
7537	043364	020124	042502	052040	
7538	043372	051505	042524	000104	
7539	043400	005015	042522	042101	MSG8: .ASCIZ <CR><LF>/READ DATA WITH OFFSET TEST/<CR><LF>
7540	043406	042040	052101	020101	
7541	043414	044527	044124	047440	
7542	043422	043106	042523	020124	
7543	043430	042524	052123	005015	
7544	043436	000			
7545	043437	015	044012	040505	MSG9: .ASCIZ <CR><LF>/HEAD NO. /
7546	043444	020104	047516	000056	
7547	043452	005015	053412	046111	MSG10: .ASCIZ <CR><LF><LF>/WILL TEST DRIVES: /
7548	043460	020114	042524	052123	
7549	043466	042040	044522	042526	
7550	043474	035123	000		
7551	043477	015	005012	047520	MSG11: .ASCIZ <CR><LF><LF>/POWER UP RESTART TO TEST 1/<CR><LF>
7552	043504	042527	020122	050125	
7553	043512	051040	051505	040524	
7554	043520	052122	052040	020117	
7555	043526	042524	052123	030440	
7556	043534	005015	000		
7557	043537	116	047117	006505	MSG12: .ASCIZ /NONE/<CR><LF>
7558	043544	000012			
7559	043546	005015	047516	046040	MSG13: .ASCII <CR><LF>/NO L OR P CLOCKS PRESENT/
7560	043554	047440	020122	020120	
7561	043562	046103	041517	051513	
7562	043570	050040	042522	042523	
7563	043576	052116			
7564	043600	005015	046101	020114	.ASCIZ <CR><LF>/ALL TIMING TESTS BYPASSED/
7565	043606	044524	044515	043516	
7566	043614	052040	051505	051524	
7567	043622	041040	050131	051501	
7568	043630	042523	000104		
7569	043634	005015	054502	040520	MSG14: .ASCIZ <CR><LF>/BYPASSING DRIVE /
7570	043642	051523	047111	020107	
7571	043650	051104	053111	020105	
7572	043656	000			
7573	043657	015	005012	051104	MSG15: .ASCIZ <CR><LF><LF>/DRIVE /
7574	043664	053111	020105	000	
7575	043671	015	042012	044522	MSG16: .ASCIZ <CR><LF>/DRIVE SERIAL NO. /
7576	043676	042526	051440	051105	
7577	043704	040511	020114	047516	
7578	043712	020056	000		
7579	043715	015	041412	051101	MSG17: .ASCIZ <CR><LF>/CARTRIDGE SERIAL NO. /
7580	043722	051124	042111	042507	
7581	043730	051440	051105	040511	
7582	043736	020114	047516	020056	
7583	043744	000			
7584	043745	015	052012	046511	MSG18: .ASCIZ <CR><LF>/TIME BETWEEN OUTER LIMIT & HEADS HOME DURING UNLOADING: /
7585	043752	020105	042502	053524	
7586	043760	042505	020116	052517	
7587	043766	042524	020122	044514	
7588	043774	044515	020124	020046	

7589	044002	042510	042101	020123
7590	044010	047510	042515	042040
7591	044016	051125	047111	020107
7592	044024	047125	047514	042101
7593	044032	047111	035107	000040
7594	044040	005015	054502	040520
7595	044046	051523	047111	020107
7596	044054	046101	020114	051127
7597	044062	052111	020105	042524
7598	044070	052123	006523	000012
7599	044076	005015	044524	042515
7600	044104	041040	052105	042527
7601	044112	047105	044040	040505
7602	044120	051504	044040	046517
7603	044126	020105	020046	051124
7604	044134	041501	020113	047506
7605	044142	046114	047440	020113
7606	044150	052504	044522	043516
7607	044156	046040	040517	044504
7608	044164	043516	020072	000
7609	044171	015	052012	046511
7610	044176	020105	042502	053524
7611	044204	042505	020116	052517
7612	044212	042524	020122	044514
7613	044220	044515	020124	020046
7614	044226	047111	042516	020122
7615	044234	044514	044515	020124
7616	044242	052504	044522	043516
7617	044250	046040	040517	044504
7618	044256	043516	020072	000
7619	044263	015	052012	046511
7620	044270	020105	042502	053524
7621	044276	042505	020116	047111
7622	044304	042516	020122	044514
7623	044312	044515	020124	020046
7624	044320	052517	042524	020122
7625	044326	044514	044515	020124
7626	044334	052504	044522	043516
7627	044342	046040	040517	044504
7628	044350	043516	020072	000
7629	044355	015	047012	020117
7630	044362	046103	041517	020113
7631	044370	047111	042524	051122
7632	044376	050125	051524	050040
7633	044404	042522	042523	052116
7634	044412	020054	041101	051117
7635	044420	044524	043516	052040
7636	044426	046511	047111	020107
7637	044434	042524	052123	000123
7638	044442	005015	053101	051105
7639	044450	043501	020105	044524
7640	044456	042515	043040	051117
7641	044464	030440	051040	053105
7642	044472	046117	052125	047511
7643	044500	035116	000040	
7644	044504	052440	000123	

MSG19: .ASCIZ <CR><LF>/BYPASSING ALL WRITE TESTS/<CR><LF>

MSG20: .ASCIZ <CR><LF>/TIME BETWEEN HEADS HOME & TRACK FOLL OK DURING LOADING: /

MSG21: .ASCIZ <CR><LF>/TIME BETWEEN OUTER LIMIT & INNER LIMIT DURING LOADING: /

MSG22: .ASCIZ <CR><LF>/TIME BETWEEN INNER LIMIT & OUTER LIMIT DURING LOADING: /

MSG23: .ASCIZ <CR><LF>/NO CLOCK INTERRUPTS PRESENT, ABORTING TIMING TESTS/

MSG24: .ASCIZ <CR><LF>/AVERAGE TIME FOR 1 REVOLUTION: /

MSG25: .ASCIZ / US/

7645	044510	005015	040412	047502
7646	044516	052122	047111	020107
7647	044524	040504	040524	052040
7648	044532	051505	051524	023040
7649	044540	043440	044517	043516
7650	044546	052040	020117	044524
7651	044554	044515	043516	052040
7652	044562	051505	051524	005015
7653	044570	000012		
7654	044572	005015	040412	046114
7655	044600	042040	044522	042526
7656	044606	020123	042524	052123
7657	044614	042105	005015	000012
7658	044622	005015	053101	051105
7659	044630	043501	020105	044524
7660	044636	042515	043040	051117
7661	044644	032040	030061	041440
7662	044652	046131	047111	042504
7663	044660	020122	047506	040527
7664	044666	042122	051440	042505
7665	044674	035113	020040	000
7666	044701	015	040412	042526
7667	044706	040522	042507	052040
7668	044714	046511	020105	047506
7669	044722	020122	030464	020060
7670	044730	054503	044514	042116
7671	044736	051105	051040	053105
7672	044744	051105	042523	051440
7673	044752	042505	035113	020040
7674	044760	000		
7675	044761	015	040412	042526
7676	044766	040522	042507	052040
7677	044774	046511	020105	047506
7678	045002	020122	020061	054503
7679	045010	044514	042116	051105
7680	045016	024040	020060	047524
7681	045024	030440	020051	047506
7682	045032	040527	042122	051440
7683	045040	042505	035113	020040
7684	045046	000		
7685	045047	015	040412	042526
7686	045054	040522	042507	052040
7687	045062	046511	020105	047506
7688	045070	020122	020061	054503
7689	045076	044514	042116	051105
7690	045104	024040	020060	047524
7691	045112	030440	020051	042522
7692	045120	042526	051522	020105
7693	045126	042523	045505	020072
7694	045134	000040		
7695	045136	005015	053101	051105
7696	045144	043501	020105	044524
7697	045152	042515	043040	051117
7698	045160	030440	033463	041440
7699	045166	046131	047111	042504
7700	045174	020122	047506	040527

MSG26: .ASCIZ <CR><LF><LF>/ABORTING DATA TESTS & GOING TO TIMING TESTS/<CR><LF><LF>

MSG27: .ASCIZ <CR><LF><LF>/ALL DRIVES TESTED/<CR><LF><LF>

MSG28: .ASCIZ <CR><LF>/AVERAGE TIME FOR 410 CYLINDER FOWARD SEEK: /

MSG29: .ASCIZ <CR><LF>/AVERAGE TIME FOR 410 CYLINDER REVERSE SEEK: /

MSG30: .ASCIZ <CR><LF>/AVERAGE TIME FOR 1 CYLINDER (0 TO 1) FOWARD SEEK: /

MSG31: .ASCIZ <CR><LF>/AVERAGE TIME FOR 1 CYLINDER (0 TO 1) REVERSE SEEK: /

MSG32: .ASCIZ <CR><LF>/AVERAGE TIME FOR 137 CYLINDER FOWARD SEEK: /

7701	045202	042122	051440	042505
7702	045210	035113	020340	000
7703	045215	015	040412	042526
7704	045222	040522	042507	052040
7705	045230	046511	020105	047506
7706	045236	020122	031461	020067
7707	045244	054503	044514	042116
7708	045252	051105	051040	053105
7709	045260	051105	042523	051440
7710	045266	042505	035113	020040
7711	045274	000		
7712	045275	015	040412	042526
7713	045302	040522	042507	043040
7714	045310	053517	051101	020104
7715	045316	050123	042505	020104
7716	045324	042502	053524	042505
7717	045332	020116	054503	044514
7718	045340	042116	051105	020123
7719	045346	031061	020070	020046
7720	045354	032462	035066	020040
7721	045362	000		
7722	045363	015	040412	042526
7723	045370	040522	042507	051040
7724	045376	053105	051105	042523
7725	045404	051440	042520	042105
7726	045412	041040	052105	042527
7727	045420	047105	041440	046131
7728	045426	047111	042504	051522
7729	045434	030440	034062	023040
7730	045442	031040	033065	020072
7731	045450	000040		
7732	045452	020040	047111	044103
7733	045460	051505	050040	051105
7734	045466	051440	041505	047117
7735	045474	000104		
7736	045476	005015	047516	053440
7737	045504	044522	042524	041440
7738	045512	042510	045503	042440
7739	045520	051122	051117	040440
7740	045526	020124	040515	020130
7741	045534	047520	044523	044524
7742	045542	042526	047440	043106
7743	045550	042523	000124	
7744	045554	005015	047516	053440
7745	045552	044522	042524	041440
7746	045570	042510	045503	042440
7747	045576	051122	051117	040440
7748	045604	020124	040515	020130
7749	045612	042516	040507	044524
7750	045620	042526	047440	043106
7751	045626	042523	006524	000012
7752	045634	005015	051127	052111
7753	045642	020105	044103	041505
7754	045650	020113	040506	046111
7755	045656	051125	020105	052101
7756	045664	047440	043106	042523

MSG33: .ASCIZ <CR><LF>/AVERAGE TIME FOR 137 CYLINDER REVERSE SEEK: /

MSG34: .ASCIZ <CR><LF>/AVERAGE FOWARD SPEED BETWEEN CYLINDERS 128 & 256: /

MSG35: .ASCIZ <CR><LF>/AVERAGE REVERSE SPEED BETWEEN CYLINDERS 128 & 256: /

MSG36: .ASCIZ / INCHES PER SECOND/

MSG37: .ASCIZ <CR><LF>/NO WRITE CHECK ERROR AT MAX POSITIVE OFFSET/

MSG38: .ASCIZ <CR><LF>/NO WRITE CHECK ERROR AT MAX NEGATIVE OFFSET/<CR><LF>

MSG39: .ASCIZ <CR><LF>/WRITE CHECK FAILURE AT OFFSET =/

7757	045672	020124	000075
7758	045676	005015	047503
7759	045704	020104	047516
7760	045712	042522	042101
7761	045720	042101	051440
7762	045726	047524	020122
7763	045734	047506	047440
7764	045742	054503	020114
7765	045750	060	030464
7766	045751	015	047412
7767	045756	046101	043511
7768	045764	047105	020124
7769	045772	052122	044522
7770	046000	020105	051525
7771	046006	005015	000
7772	046011	015	041012
7773	046016	051501	044523
7774	046024	040440	046114
7775	046032	046511	047111
7776	046040	02524	052123
7777	046046	000012	006523
7778	046050	005015	047522
7779	046056	044524	047117
7780	046064	020054	040515
7781	046072	044515	026116
7782	046100	042116	030440
7783	046106	041440	046131
7784	046114	042505	020113
7785	046122	042515	123
7786	046125	015	040412
7787	046132	052040	046505
7788	046140	040522	044522
7789	046146	041040	050131
7790	046154	042523	020104
7791	046162	044524	020114
7792	046170	05203	040522
7793	046176	051505	041440
7794	046204	041040	020105
7795	046212	051120	053117
7796	046220	000	042105
7797			
7798			
7799			
7800			
7801	046221	015	042412
7802	046226	051117	020054
7803	046234	054514	030040
7804	046242	051110	020125
7805	046250	046101	047514
7806	046256	026104	052040
7807	046264	040440	040507
7808	046272	005015	000
7809	046275	104	044522
7810	046302	021440	044440
7811	046310	045522	051503
7812	046316	040503	047116

MSG40: .ASCII <CR><LF>/COULD NOT READ BAD SECTOR INFO ON CYL 410/

.ASCIZ <CR><LF>/OR ALIGNMENT CARTRIDGE USED/<CR><LF>

MSG41: .ASCIZ <CR><LF>/BYPASSING ALL TIMING TESTS/<CR><LF>

MSG42: .ASCII <CR><LF>/ROTATIONAL, MAX-MIN, AND 137 CYL SEEK TIMES/

.ASCIZ <CR><LF>/ARE TEMPORARILY BYPASSED UNTIL ACCURACIES CAN BE IMPROVED/

.SBTTL ERROR MESSAGES

EM1: .ASCIZ <CR><LF>/ERROR, ONLY 0 THRU 7 ALLOWED, TRY AGAIN.<CR><LF>

EM2: .ASCIZ /DRIVE # IN RKCS2 CANNOT BE READ BACK CORRECTLY IN RKMR2/

7813	046324	041040	020105	042522	
7814	046332	042101	041040	041501	
7815	046340	020113	047503	051122	
7816	046346	041505	046124	020131	
7817	046354	047111	051040	046513	
7818	046362	031122	000		
7819	046355	015	040412	047502	EM3: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED TIME OUT AT PC=/'
7820	046372	052122	052040	051505	
7821	046400	051524	027056	052456	
7822	046406	042516	050130	041505	
7823	046414	042524	020104	044524	
7824	046422	042515	047440	052125	
7825	046430	040440	020124	041520	
7826	046436	000075			
7827	046440	005015	041101	051117	EM4: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED INTERRUPT AT PC=/'
7828	046446	020124	042524	052123	
7829	046454	027123	027056	047125	
7830	046462	054105	042520	052103	
7831	046470	042105	044440	052116	
7832	046476	051105	052522	052120	
7833	046504	040440	020124	041520	
7834	046512	000075			
7835	046514	042115	020123	042523	EM5: .ASCIZ /MDS SET IN RKCS2/'
7836	046522	020124	047111	051040	
7837	046530	041513	031123	000	
7838	046535	125	042506	051440	EM6: .ASCIZ /UFE SET IN RKCS2/'
7839	046542	052105	044440	020116	
7840	046550	045522	051503	000062	
7841	046556	051104	020101	047111	EM7: .ASCIZ /DPA IN RKDS & NED IN RKCS2 RESET; WRONG PORT SELECTED?/'
7842	046564	051040	042113	020123	
7843	046572	020046	042516	020104	
7844	046600	047111	051040	041513	
7845	046606	031123	051040	051505	
7846	046614	052105	020073	051127	
7847	046622	047117	020107	047520	
7848	046630	052122	051440	046105	
7849	046636	041505	042524	037504	
7850	046644	023			
7851	046645	104	044522	042526	EM8: .ASCIZ /DRIVE PRESENT BUT NOT SPECIFIED BY OPERATOR/'
7852	046652	050040	042522	042523	
7853	046660	052116	041040	052125	
7854	046666	047040	052117	051440	
7855	046674	042520	044503	044506	
7856	046702	042105	041040	020131	
7857	046710	050117	051105	052101	
7858	046716	051117	000		
7859	046721	104	044522	042526	EM9: .ASCIZ /DRIVE NOT PRESENT BUT SPECIFIED BY OPERATOR/'
7860	046726	047040	052117	050040	
7861	046734	042522	042523	052116	
7862	046742	041040	052125	051440	
7863	046750	042520	044503	044506	
7864	046756	042105	041040	020131	
7865	046764	050117	051105	052101	
7866	046772	051117	000		
7867	046775	101	047502	052122	EM10: .ASCIZ /ABORT TESTS...CANNOT REFERENCE CONTROLLER REGISTER/'
7868	047002	052040	051505	051524	

7869	047010	027056	041456	047101	
7870	047016	047516	020124	042522	
7871	047024	042506	042522	041516	
7872	047032	020105	047503	052116	
7873	047040	047522	046114	051105	
7874	047046	051040	043505	051511	
7875	047054	042524	000122		
7876	047060	051104	020101	047111	EM11: .ASCIZ /DRA IN RKDS & NED IN RKCS2 BOTH SET/
7877	047066	051040	042113	020123	
7878	047074	020046	042516	020104	
7879	047102	047111	051040	041513	
7880	047110	031123	041040	052117	
7881	047116	020110	042523	000124	
7882	047124	047503	052116	047522	EM12: .ASCIZ /CONTROLLER NOT READY IN RKCS1/
7883	047132	046114	051105	047040	
7884	047140	052117	051040	040505	
7885	047146	054504	044440	020116	
7886	047154	045522	051503	000061	
7887	047162	047516	040440	052124	EM13: .ASCIZ /NO ATTN IN RKASOF/
7888	047170	020116	047111	051040	
7889	047176	040513	047523	000106	
7890	047204	047125	054105	042520	EM14: .ASCIZ /UNEXPECTED MEMORY PARITY TRAP/
7891	047212	052103	042105	046440	
7892	047220	046505	051117	020131	
7893	047226	040520	044522	054524	
7894	047234	052040	040522	000120	
7895	047242	045522	041504	023040	EM15: .ASCII /RKDC & RKDA INDICATE THAT WCE OCCURRED AT/
7896	047250	051040	042113	020101	
7897	047256	047111	044504	040503	
7898	047264	042524	052040	040510	
7899	047272	020124	041527	020105	
7900	047300	041517	052503	051122	
7901	047306	042105	040440	124	
7902	047313	015	041412	046131	.ASCIZ <CR><LF>/CYL 411. TRACK 2, SECTOR 21/
7903	047320	032040	030461	020054	
7904	047326	051124	041501	020113	
7905	047334	026062	051440	041505	
7906	047342	047524	020122	030462	
7907	047350	000			
7908	047351	103	047101	047516	EM16: .ASCIZ /CANNOT READ BAD SECTOR INFORMATION/
7909	047356	020124	042522	042101	
7910	047364	041040	042101	051440	
7911	047372	041505	047524	020122	
7912	047400	047111	047506	046522	
7913	047406	052101	047511	000116	
7914	047414	042515	051523	043501	EM17: .ASCIZ /MESSAGE A0 ERROR/
7915	047422	020105	030101	042440	
7916	047430	051122	051117	000	
7917	047435	115	051505	040523	EM18: .ASCIZ /MESSAGE B0 ERROR/
7918	047442	042507	041040	020060	
7919	047450	051105	047522	000122	
7920	047456	042515	051523	043501	EM19: .ASCIZ /MESSAGE A1 ERROR/
7921	047464	020105	030501	042440	
7922	047472	051122	051117	000	
7923	047477	115	051505	040523	EM20: .ASCIZ /MESSAGE B1 ERROR/
7924	047504	042507	041040	020061	

7925	047512	051105	047522	000122	
7926	047520	042503	051122	051440	EM21: .ASCIZ /CERR SET IN RKCS1/
7927	047526	052105	044440	020116	
7928	047534	045522	051503	000061	
7929	047542	047526	020114	040526	EM24: .ASCIZ /VOL VALID NOT SET IN RKMR2/
7930	047550	044514	020104	047516	
7931	047556	020124	042523	020124	
7932	047564	047111	051040	046513	
7933	047572	031122	000		
7934	047575	103	046131	040440	EM36: .ASCIZ /CYL ADDR IN RKMR3 NOT SAME AS RKDC/
7935	047602	042104	020122	047111	
7936	047610	051040	046513	031522	
7937	047616	047040	052117	051440	
7938	047624	046501	020105	051501	
7939	047632	051040	042113	000103	
7940	047640	054503	020114	044504	EM39: .ASCIZ /CYL DIFF & OFFSET IN RKMR2 NOT CLEARED/
7941	047646	043106	023040	047440	
7942	047654	043106	042523	020124	
7943	047662	047111	051040	046513	
7944	047670	031122	047040	052117	
7945	047676	041440	042514	051101	
7946	047704	042105	000		
7947	047707	103	046131	040440	EM40: .ASCIZ /CYL ADDR IN RKMR3 NOT CLEARED/
7948	047714	042104	020122	047111	
7949	047722	051040	046513	031522	
7950	047730	047040	052117	041440	
7951	047736	042514	051101	042105	
7952	047744	000			
7953	047745	103	046131	040440	EM41: .ASCIZ /CYL ADDR IN RKMR3 DID NOT REMAIN CLEARED/
7954	047752	042104	020122	047111	
7955	047760	051040	046513	031522	
7956	047766	042040	042111	047040	
7957	047774	052117	051040	046505	
7958	050002	044501	020116	046103	
7959	050010	040505	042522	000104	
7960	050016	052101	047124	047040	EM55: .ASCIZ /ATTN NOT CLEARED IN RKASOF/
7961	050024	052117	041440	042514	
7962	050032	051101	042105	044440	
7963	050040	020116	045522	051501	
7964	050046	043117	000		
7965	050051	110	040505	051504	EM60: .ASCIZ /HEADS HOME NOT FOUND IN RKMR2/
7966	050055	044040	046517	020105	
7967	050064	047516	020124	047506	
7968	050072	047125	020104	047111	
7969	050100	051040	046513	031122	
7970	050106	000			
7971	050107	104	042524	051440	EM62: .ASCIZ /DTE SET IN RKER/
7972	050114	052105	044440	020116	
7973	050122	045522	051105	000	
7974	050127	104	052114	051440	EM63: .ASCIZ /DLT SET IN RKCS2/
7975	050134	052105	044440	020116	
7976	050142	045522	051503	000062	
7977	050150	042522	042101	044040	EM65: .ASCIZ /READ HEADER ERROR/
7978	050156	040505	042504	020122	
7979	050164	051105	047522	000122	
7980	050172	054503	020114	042101	EM66: .ASCIZ /CYL ADDR IN RKMR3 INCORRECT/

7981	050200	051104	044440	020116		
7982	050206	045522	051115	020063		
7983	050214	047111	047503	051122		
7984	050222	041505	000124			
7985	050226	046101	043511	046516	EM69:	.ASCIZ .ALIGNMENT CARTRIDGE USED
7986	050234	047105	020124	040503		
7987	050242	052122	044522	043504		
7988	050250	020105	051525	042105		
7989	050256	000				
7990	050257	122	055124	047040	EM74:	.ASCIZ /RTZ NOT SET IN RKMR2
7991	050264	052117	051440	052105		
7992	050272	044440	020116	045522		
7993	050300	051115	000062			
7994	050304	051127	052111	020105	EM80:	.ASCIZ /WRITE CHECK ERROR SET IN RKCS2/
7995	050312	044103	041505	020113		
7996	050320	051105	047522	020122		
7997	050326	042523	020124	047111		
7998	050334	051040	041513	031123		
7999	050342	000				
8000	050343	127	044522	042524	EM81:	.ASCIZ /WRITE CHECK COMMAND NOT FUNCTIONING/
8001	050350	041440	042510	045503		
8002	050356	041440	046517	040515		
8003	050364	042116	047040	052117		
8004	050372	043040	047125	052103		
8005	050400	047511	044516	043516		
8006	050406	000				
8007	050407	122	040505	020104	EM82:	.ASCIZ /READ DATA DID NOT COMPARE WITH WRITE DATA/
8008	050414	040504	040524	042040		
8009	050422	042111	047040	052117		
8010	050430	041440	046517	040520		
8011	050436	042522	053440	052111		
8012	050444	020110	051127	052111		
8013	050452	020105	040504	040524		
8014	050460	000				
8015	050461	104	052101	020101	EM83:	.ASCIZ /DATA CHECK ERROR SET IN RKER/
8016	050466	041103	041505	020113		
8017	050474	051105	047522	020122		
8018	050502	042523	020124	047111		
8019	050510	051040	042513	000122		
8020	050516	043117	051506	052105	EM85:	.ASCIZ /OFFSET STATUS BIT IN RKMR2 CLEARED/
8021	050524	051440	040524	052524		
8022	050532	020123	044502	020124		
8023	050540	047111	051040	046513		
8024	050546	031122	041440	042514		
8025	050554	051101	047105	000		
8026	050561	117	047106	042523	EM86:	.ASCIZ /OFFSET REG IN RKMR2 NOT = RKMSOF/
8027	050566	020124	042522	020107		
8028	050574	047111	051040	046513		
8029	050602	031122	047040	052117		
8030	050610	036440	051040	040513		
8031	050616	047523	000106			
8032	050622	044504	020104	047516	EM88:	.ASCIZ /DID NOT FIND SECTOR 0 FROM INDEX/
8033	050630	020124	044506	042116		
8034	050636	051440	041505	047524		
8035	050644	020122	020060	051106		
8036	050652	046517	044440	042116		

8037	050660	054105	000		
8038	050663	110	040505	051504	EM89: .ASCIZ /HEADS HOME NOT CLEARED IN RKMR2/
8039	050670	044040	046517	020105	
8040	050676	047516	020124	046103	
8041	050704	040505	042522	020104	
8042	050712	047111	051040	046513	
8043	050720	031122	000		
8044	050723	124	040522	045503	EM90: .ASCIZ /TRACK FOLL CK NOT SET IN RKMR2/
8045	050730	043040	046117	020114	
8046	050736	045517	047040	052117	
8047	050744	051440	052105	044440	
8048	050752	020116	045522	051115	
8049	050760	000062			
8050	050762	042522	020126	047516	EM91: .ASCIZ /REV NOT SET IN RKMR2/
8051	050770	020124	042523	020124	
8052	050776	047111	051040	046513	
8053	051004	031122	000		
8054	051007	122	053105	047040	EM92: .ASCIZ /REV NOT CLEARED IN RKMR2/
8055	051014	052117	041440	042514	
8056	051022	051101	042105	044440	
8057	051030	020116	045522	051115	
8058	051036	000062			
8059	051040	042522	042101	047111	EM93: .ASCIZ /READING WRONG CYLINDER # IN HEADER...MISPOSITION/
8060	051046	020107	051127	047117	
8061	051054	020107	054503	044514	
8062	051062	042116	051105	021440	
8063	051070	044440	020116	042510	
8064	051076	042101	051105	027056	
8065	051104	046456	051511	047520	
8066	051112	044523	044524	047117	
8067	051120	000			
8068	051121	117	043106	042523	EM94: .ASCIZ /OFFSET STATUS BIT IN RKMR2 NOT CLEARED/
8069	051126	020124	052123	052101	
8070	051134	051525	041040	052111	
8071	051142	044440	020116	045522	
8072	051150	051115	020062	047516	
8073	051156	020124	046103	040505	
8074	051164	042522	000104		
8075	051170	047506	046522	052101	EM95: .ASCIZ /FORMAT BIT NOT SET IN RKMR2/
8076	051176	041040	052111	047040	
8077	051204	052117	051440	052105	
8078	051212	044440	020116	045522	
8079	051220	051115	000062		
8080	051224	040503	047116	052117	EM96: .ASCIZ /CANNOT FIND SECTOR 17/
8081	051232	043040	047111	020104	
8082	051240	042523	052103	051117	
8083	051246	030440	000067		
8084	051252	042510	042101	051440	EM97: .ASCIZ /HEAD SWITCHING LONGER THAN 16 MS/
8085	051260	044527	041524	044510	
8086	051266	043516	046040	047117	
8087	051274	042507	020122	044124	
8088	051302	047101	030440	020066	
8089	051310	051515	000		
8090	051313	103	047101	047516	EM98: .ASCIZ /CANNOT FIND CYLINDER 128/
8091	051320	020124	044506	042116	
8092	051326	041440	046131	047111	

8093	051334	042504	020122	031061
8094	051342	000070		
8095	051344	040503	047116	052117
8096	051352	043040	047111	020104
8097	051360	054503	044514	042116
8098	051366	051105	031040	033065
8099	051374	000		
8100	051375	104	044522	042526
8101	051402	047440	043106	052040
8102	051410	040522	045503	051440
8103	051416	052105	044440	020116
8104	051424	045522	051115	000063
8105	051432	044504	020104	047516
8106	051440	020124	047507	052040
8107	051446	020117	054503	044514
8108	051454	042116	051105	030440
8109	051462	000060		
8110				
8111				
8112				
8113	051464	042524	052123	047040
8114	051472	027117	020040	041520
8115	051500	000		
8116	051501	122	046513	031122
8117	051506	051011	046513	031522
8118	051514	051011	042513	004522
8119	051522	045522	051504	051011
8120	051530	041513	030523	051011
8121	051536	041513	031123	000
8122	051543	122	040513	047523
8123	051550	000106		
8124	051552	044123	052517	042114
8125	051560	041040	000105	
8126	051564	045522	051111	004462
8127	051572	045522	051115	004463
8128	051600	045522	051105	051011
8129	051606	042113	004523	045522
8130	051614	041504	051011	041513
8131	051622	030523	051011	041513
8132	051630	031123	000	
8133	051633	122	046513	031122
8134	051640	020040	051040	046513
8135	051646	031522	020040	051040
8136	051654	042113	020103	020040
8137	051662	051106	046517	041440
8138	051670	046131	020040	047524
8139	051676	041440	046131	020040
8140	051704	054503	020114	044504
8141	051712	043106	000	
8142	051715	122	046513	031122
8143	051722	051011	046513	031522
8144	051730	051011	042513	004522
8145	051736	045522	051504	051011
8146	051744	042113	004501	045522
8147	051752	051503	004461	045522
8148	051760	051503	000062	

EM99: .ASCIZ /CANNOT FIND CYLINDER 256/

EM100: .ASCIZ /DRIVE OFF TRACK SET IN RKMR3/

EM101: .ASCIZ /DID NOT GO TO CYLINDER 10/

.SBTTL DATA HEADERS

DH1: .ASCIZ /TEST NO. PC/

DH2: .ASCIZ /RKMR2 RKMR3 RKER RKDS RKCS1 RKCS2/

DH3: .ASCIZ /RKASOF/

DH4: .ASCIZ /SHOULD BE/

DH5: .ASCIZ /RKMR2 RKMR3 RKER RKDS RKDC RKCS1 RKCS2/

DH6: .ASCIZ /RKMR2 RKMR3 RKDC FROM CYL TO CYL CYL DIFF/

DH7: .ASCIZ /RKMR2 RKMR3 RKER RKDS RKDA RKCS1 RKCS2/

8261	053105	117	020.16	042523	DH42:	.ASCIZ /ON SECTORS 0,2,4,6 OR 8 CYL 410 TRACK 2/
8262	053112	052103	051117	020123		
8263	053120	026060	026062	026064		
8264	053126	020066	051117	034040		
8265	053134	020040	054503	020114		
8266	053142	030464	020060	051124		
8267	053150	041501	020113	000062		
8268	053156	047117	051440	041505	DH43:	.ASCIZ /ON SECTORS 1,3,5,7 OR 9 CYL 410 TRACK2/
8269	053164	047524	051522	030440		
8270	053172	031454	032454	033454		
8271	053200	047440	020122	020071		
8272	053206	041440	046131	032040		
8273	053214	030061	052040	040522		
8274	053222	045503	000062			
8275	053226	047506	046522	052101	DH44:	.ASCIZ /FORMAT & ALL READ-WRITE TESTS WILL BE BYPASSED/
8276	053234	023040	040440	046114		
8277	053242	051040	040505	026504		
8278	053250	051127	052111	020105		
8279	053256	042524	052123	020123		
8280	053264	044527	046114	041040		
8281	053272	020105	054502	040520		
8282	053300	051523	042105	000		
8283	053305	101	052106	051105	DH47:	.ASCIZ /AFTER READ HEADER COMMAND WITH MOVEMENT/
8284	053312	05.740	040505	020104		
8285	053320	042510	042101	051105		
8286	053326	041440	046517	040515		
8287	053334	042116	053440	052111		
8288	053342	020110	047515	042526		
8289	053350	042515	052116	000		
8290	053355	101	052106	051105	DH51:	.ASCIZ /AFTER SEEK TO SELF COMMAND/
8291	053362	051440	042505	020113		
8292	053370	047524	051440	046105		
8293	053376	020106	047503	046515		
8294	053404	047101	000104			
8295	053410	044527	044124	044440	DH52:	.ASCIZ /WITH INTENTIONAL MISCOMPARE/
8296	053416	052116	047105	044524		
8297	053424	047117	046101	046440		
8298	053432	051511	047503	050115		
8299	053440	051101	000105			
8300	053444	052504	044522	043516	DH53:	.ASCIZ /DURING OFFSET COMMAND/
8301	053452	047440	043106	042523		
8302	053460	020124	0475.3	046515		
8303	053466	047101	000104			
8304	053472	043101	042524	020122	DH54:	.ASCIZ /AFTER FORMAT CHANGE AND CONTR READY REC'D/
8305	053500	047506	046522	052101		
8306	053506	041440	040510	043516		
8307	053514	020105	047101	020104		
8308	053522	047503	052116	020122		
8309	053530	042522	042101	020131		
8310	053536	042522	023503	000104		
8311	053544	045522	051115	004462	DH56:	.ASCIZ /RKMR2 RKMR3 RKDC CYL # HEADER WORD 0/
8312	053552	045522	051115	004463		
8313	053560	045522	041504	041411		
8314	053566	046131	021440	044011		
8315	053574	040505	042504	020122		
8316	053602	047527	042122	030040		

8317	053610	000			
8318	053611	101	052106	051105	DH57: .ASCIZ /AFTER WRITE COMMAND WITH OFFSET/
8319	053616	053440	044522	042524	
8320	053624	041440	046517	040515	
8321	053632	042116	053440	052111	
8322	053640	020110	043117	051506	
8323	053646	052105	000		
8324	053651	104	052101	020101	DH58: .ASCIZ /DATA WAS SHOULD BE/
8325	053656	040527	020123	051411	
8326	053664	047510	046125	020104	
8327	053672	042502	000		
8328					
8329					.SBTTL ERROR OUTPUT DATA
8330					
8331		053676			.EVEN
8332	053676	001214	001116	005434	DT1: \$TESTN,\$ERRPC,HMR2,HMR3,HER,HDS,HCS1,HCS2,SBMR2,SBMR3
8333	053704	005436	005422	005420	
8334	053712	005406	005410	005444	
8335	053720	005446			
8336	053722	001214	001116	005434	DT2: \$TESTN,\$ERRPC,HMR2,HMR3,HER,HDS,HCS1,HCS2,HASOF
8337	053730	005436	005422	005420	
8338	053736	005406	005410	005424	
8339	053744	001214	001354		DT3: \$TESTN,TRAPP
8340	053750	001214	001116	005434	DT4: \$TESTN,\$ERRPC,HMR2,HMR3,HDC,FRCYL,TOCYL,CALDIF
8341	053756	005436	005426	001370	
8342	053764	001372	001400		
8343	053770	001214	001116	005434	DT5: \$TESTN,\$ERRPC,HMR2,HMR3,HER,HDS,HDA,HCS1,HCS2,SBMR2,SBMR3
8344	053776	005436	005422	005420	
8345	054004	005416	005406	005410	
8346	054012	005444	005446		
8347	054016	001214	001116	005434	DT6: \$TESTN,\$ERRPC,HMR2,HMR3,HCS1,HCS2,WD2,WD1
8348	054024	005436	005406	005410	
8349	054032	001510	001506		
8350	054036	001214	001116	005434	DT7: \$TESTN,\$ERRPC,HMR2,HMR3,HDA,WDCNT,HOWD,TEMP1
8351	054044	005436	005416	001524	
8352	054052	001542	005450		
8353	054056	001214	001116	005434	DT8: \$TESTN,\$ERRPC,HMR2,HMR3,HDC,TOCYL,FRCYL,CALDIF
8354	054064	005436	005426	001372	
8355	054072	001370	001400		
8356	054076	001214	001116	005434	DT9: \$TESTN,\$ERRPC,HMR2,HMR3,HDC,TOCYL,RHTAB
8357	054104	005436	005426	001372	
8358	054112	001754			
8359	054114	001214	001116	005434	DT10: \$TESTN,\$ERRPC,HMR2,HMR3,HCS1,HCS2,HDC,HDA
8360	054122	005436	005406	005410	
8361	054130	005426	005416		
8362					
8363					.SBTTL ERROR DATA FORMATS
8364					
8365	054134	000002			DF1: 2
8366	054136	002	000		.BYTE 2,0
8367	054140	051501			DH2
8368	054142	006	000		.BYTE 6,0
8369					
8370	054144	000001			DF2: 1
8371	054146	002	000		.BYTE 2,0
8372					

8373	054150	000003		DF3:	3	
8374	054152	000	000		.BYTE	0,0
8375	054154	051464			DH1	
8376	054156	002	000		.BYTE	2,0
8377	054160	052107			DH11	
8378	054162	006	000		.BYTE	6,0
8379						
8380	054164	000002		DF4:	2	
8381	054166	002	000		.BYTE	2,0
8382	054170	052154			DH12	
8383	054172	006	000		.BYTE	6,0
8384						
8385	054174	000003		DF6:	3	
8386	054176	000	000		.BYTE	0,0
8387	054200	051464			DH1	
8388	054202	002	000		.BYTE	2,0
8389	054204	051633			DH6	
8390	054206	006	000		.BYTE	6,0
8391						
8392	054210	000004		DF8:	4	
8393	054212	000	000		.BYTE	0,0
8394	054214	051464			DH1	
8395	054216	002	000		.BYTE	2,0
8396	054220	051501			DH2	
8397	054222	006	000		.BYTE	6,0
8398	054224	051552			DH4	
8399	054226	001	000		.BYTE	1,0
8400						
8401	054230	000004		DF9:	4	
8402	054232	000	000		.BYTE	0,0
8403	054234	051464			DH1	
8404	054236	002	000		.BYTE	2,0
8405	054240	051501			DH2	
8406	054242	006	000		.BYTE	6,0
8407	054244	051552			DH4	
8408	054246	002	000		.BYTE	2,0
8409						
8410	054250	000003		DF10:	3	
8411	054252	000	000		.BYTE	0,0
8412	054254	051464			DH1	
8413	054256	002	000		.BYTE	2,0
8414	054260	051501			DH2	
8415	054262	006	000		.BYTE	6,0
8416						
8417	054264	000004		DF11:	4	
8418	054266	000	000		.BYTE	0,0
8419	054270	051464			DH1	
8420	054272	002	000		.BYTE	2,0
8421	054274	051501			DH2	
8422	054276	006	000		.BYTE	6,0
8423	054300	051543			DH3	
8424	054302	001	000		.BYTE	1,0
8425						
8426	054304	000002		DF14:	2	
8427	054306	002	000		.BYTE	2,0
8428	054310	053003			DH40	

8429	054312	006	000
8430			
8431			
8432	054314	000003	
8433	054316	000	000
8434	054320	051464	
8435	054322	002	000
8436	054324	051715	
8437	054326	007	000
8438			
8439	054330	000004	
8440	054332	000	000
8441	054334	053226	
8442	054336	000	000
8443	054340	051464	
8444	054342	002	000
8445	054344	051501	
8446	054346	006	000
8447	054350	000003	
8448	054352	000	000
8449	054354	051464	
8450	054356	002	000
8451	054360	053544	
8452	054362	005	000
8453			
8454			
8455			
8456			
8457			
8458			
8459			
8460			
8461			
8462			
8463			
8464	054364	104413	
8465	054366	113700	001114
8466	054372	042700	177400
8467	054376	005300	
8468	054400	006300	
8469	054402	006300	
8470	054404	006300	
8471	054406	062700	005524
8472	054412	012037	054426
8473	054416	001404	
8474	054420	104401	001205
8475	054424	104401	
8476	054426	000000	
8477	054430	012037	054444
8478	054434	001404	
8479	054436	104401	001205
8480	054442	104401	
8481	054444	000000	
8482	054446	012001	
8483	054450	001455	
8484	054452	005004	

```

.BYTE 6,0
DF15: 3
       .BYTE 0,0
       DH1
       .BYTE 2,0
       DH7
       .BYTE 7,0
DF17: 4
       .BYTE 0,0
       DH44
       .BYTE 0,0
       DH1
       .BYTE 2,0
       DH2
       .BYTE 6,0
DF20: 3
       .BYTE 0,0
       DH1
       .BYTE 2,0
       DH56
       .BYTE 5,0

```

```

.EVEN
:*****
:SBTTL TYPE ERROR ROUTINE
:*ENTRY JSR PC,TYP ERR
:*RETURN RTS PC
:*
:*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
:*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
:*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
:*THE ERROR.
:*****
†TYPERR: SAVREG
        MOV      $ITEMB,RO      ;ENTER ERROR NUMBER
        BIC      #177400,RO     ;CLEAR SIGN EXTENSION
        DEC      RO             ;FORM INDEX FOR ERROR TABLE
        ASL      RO
        ASL      RO
        ASL      RO
1$:     ADD      #SERATB,RO      ;FORM ADDRESS OF ERROR ENTRY
        MOV      (RO)+,2$      ;GET EM POINTER
        BEQ     3$             ;BRANCH IF THERE ISN'T ONE
        TYPE    ,SCLF         ;TYPE CARRIAGE RETURN LINE FEED
        TYPE    ,SCLF         ;TYPE ERROR MESSAGE (EM)
2$:     .WORD   0              ;EM POINTER GOES HERE
3$:     MOV      (RO)+,4$      ;GET DH POINTER
        BEQ     5$             ;BRANCH IF THERE ISN'T ONE
        TYPE    ,SCLF         ;TYPE CR-LF
        TYPE    ,SCLF         ;TYPE DATA HEADER
4$:     .WORD   0              ;DH POINTER GOES HERE
5$:     MOV      (RO)+,R1      ;GET DT POINTER
        BEQ     20$           ;BRANCH IF THERE ARE NONE
        CLR     R4            ;SET INDENT SWITCH

```

```

8485 054454 012000      MOV      (R0)+,R0      ;GET DF POINTER
8486 054456 012002      MOV      (R0)+,R2      ;STORE NUMBER OF DH'S
8487 054460 001446      BEQ      17$           ;DH NUM IS 0-BRANCH
8488 054462 005104      COM      R4           ;NO INDENT
8489 054464 104401 0C1205      TYPE     $CRLF
8490 054470 112003 10$:      MOVW    (R0)+,R3      ;GET & STORE NUMBER OF DATA WORDS
8491 054472 105720      TSTB    (R0)+        ;BUMP PAST FORMAT WORD
8492 054474 005703      TST     R3           ;TEST IF ANY DATA FOR THIS HEADER
8493 054476 001407      BEQ     14$          ;NO - SKIP DATA PRINT
8494 054500 013146 11$:      MOV     2(R1)+,-(SP)  ;PUT FIRST DATA WORD ON STACK
8495 054502 104402      TYPOC
8496 054504 005303      DEC     R3           ;TYPE IT
8497 054506 001403      BEQ     14$          ;MORE DATA WORDS
8498 054510 104401 054640      TYPE     SPACE2      ;NO-BRANCH
8499 054514 000771      BR      11$         ;TYPE SEPARATORS
8500 054516 005302 14$:      DEC     R2           ;LOOP
8501 054520 002431      BLE     20$         ;MORE DH'S?
8502 054522 104401 001205      TYPE     $CRLF      ;NO-BRANCH
8503 054526 005760 000002      TST     2(R0)       ;ONLY A DH IN THIS REQUEST?
8504 054528 001404      BEQ     15$         ;YES-BRANCH BYPASS INDENT
8505 054530 005104      COM     R4           ;INDENT?
8506 054536 001002      BNE     15$         ;NO-BRANCH
8507 054540 104401 054640      TYPE     SPACE2      ;YES-TYPE SPACES
8508 054544 012037 054552 15$:      MOV     (R0)+,16$   ;GET NEXT DH POINTER
8509 054550 104401      TYPE     DH         ;TYPE DH
8510 054552 000000 16$:      .WORD   0           ;DH POINTER GOES HERE
8511 054554 105710      TSTB    (R0)        ;TYPE A DT?
8512 054556 001003      BNE     21$         ;YES-BRANCH
8513 054560 062700 000002      ADD     #2,R0       ;INCREMENT OF POINTER
8514 054564 000754      BR      14$        ;SEE IF END OF DF BLOCK
8515 054566 104401 001205 21$:      TYPE     $CRLF
8516 054572 005704      TST     R4         ;INDENT?
8517 054574 001335      BNE     10$        ;NO-BRANCH
8518 054576 104401 054640 17$:      TYPE     SPACE2      ;YES-TYPE SPACES
8519 054580 000732      BR      10$        ;LOOP
8520 054604 104414 20$:      RESREG
8521 054606 032777 010000 124324      BIT     #SW12,#SWR   ;SEE IF ABORT DRV AFTER 20 ERRORS
8522 054614 001410      BEQ     25$         ;BR IF NO
8523 054616 023727 001103 000024      CMP     $ERFLG,#20. ;ELSE SEE IF HAVE 20 ERRORS
8524 054624 001004      BNE     25$         ;BR IF NO
8525 054626 012706 001100      MOV     #STACK,SP   ;ELSE RESTORE STACK PTR
8526 054632 000137 031434      JMP     $EOP        ;AND GO TO NEXT DRV
8527
8528
8529 054636 000207 25$:      RTS     PC
8530 054640 020040 000      SPACE2: .ASCIZ/ /   ;2 SPACES
8531
8532 ; ODT-11 -- V005A
8533 ; DEC-11-UODPA-A-LA
8534
8535 ; COPYRIGHT 1969,1970,1972
8536 ; DIGITAL EQUIPMENT CORPORATION
8537 ; MAYNARD, MASSACHUSETTS 01754
8538 ; .ENABL ABS,AMA
8539 ; .EVEN
8540 ; .+=60

```

```

8541 000000 R0 = %0 ; REGISTER
8542 000001 R1 = %1 ; NAMING
8543 000002 R2 = %2 ; CONVENTIONS
8544 000003 R3 = %3
8545 000004 R4 = %4
8546 000005 R5 = %5
8547 000006 SP = %6
8548 000007 PC = %7
8549 177776 ST = 177776 ;STATUS REGISTER
8550
8551 000014 O.TVEC = 14 ;TRT VECTOR LOCATION
8552 000340 O.STM = 340 ;PRIORITY MASK - STATUS REGISTER
8553 000020 O.TBT = 20 ;T-BIT MASK - STATUS REGISTER
8554 000003 TRT = 000003 ;TRT INSTRUCTION
8555 000006 RTT = 000006 ;RTT INSTRUCTION
8556
8557
8558
8559
8560
8561
8562 177562 O.RDB = 177562 ;R DATA BUFFER
8563 177560 O.RCSR = 177560 ;R C/SR
8564 177566 O.TDB = 177566 ;T DATA BUFFER
8565 177564 O.TCSR = 177564 ;T C/SR
8566
8567
8568
8569
8570
8571
8572
8573 054724 000413 O.ODT: BR O.STRT ;NORMAL ENTRY
8574 054726 000417 BR O.RST ;RESTART
8575 054730 013737 177776 054704 O.ENTR: MOV ST,O.UST ;RE-ENTER -- SAVE STATUS
8576 054736 013737 000016 177776 MOV O.TVEC+2,ST ;SET UP LOCAL STATUS
8577 054744 010737 054702 MOV PC,O.UPC ;FAKE THE PC
8578 054750 000137 056102 JMP O.BK1
8579
8580
8581
8582
8583
8584
8585
8586
8587
8588
8589
8590
8591
8592
8593
8594
8595
8596
054754 012706 054664 O.STRT: MOV #O.URO,SP ;SET UP STACK
054760 010637 054700 MOV SP,O.USB ;FAKE THE SAVED STACK
054764 000414 BR O.RST1 ;CLEAR BREAKPOINT TABLES
054766 004037 056310 O.RST: JSR O,O.SVR ;SAVE REGISTERS
054772 013777 054722 177716 MOV O.UIN,JO.ADR1 ;REMOVE THE BREAKPOINT
055000 113704 054706 MOVB O.PRI,R4 ;GET ODT PRIORITY
055004 106004 RORB R4 ;SHIFT
055006 106004 RORB R4 ; INTO
055010 106004 RORB R4 ; POSITION
055012 110437 177776 MOVB R4,ST ;STORE IN STATUS
055016 000127 O.RST1: JMP (PC)+
055020 000403 BR O.45
055022 012737 000002 056012 MOV #RTI,O.RTIT ;SET TO RTI IF 11/20 OR /05
055030 105037 056731 O.45: CLRB O.P ;DISALLOW PROCEED
055034 012737 000340 C00016 MOV #O.STM,O.TVEC+2 ;STATUS WORD TO TRT VECTOR + 2
055042 012737 056072 000014 MOV #O.BRK,O.TVEC ;PC TO TRT VECTOR
055050 000447 BR O.RALL ;CLEAR BREAKPOINT TABLES

```

Z

Z

```

8597
8598
8599
8600
8601 055052 004537 056532
8602 055056 012704 056755
8603 055062 120024
8604 055064 001413
8605 055066 022704 056763
8606 055072 101373
8607 055074 042700 177770
8608 055100 010004
8609 055102 006304
8610 055104 062704 054664
8611 055110 005202
8612 055112 000444
8613 055114 162704 056746
8614 055120 000770
8615
8616
8617
8618 055122 004737 056656
8619 055126 010502
8620 055130 061202
8621 055132 006202
8622 055134 103421
8623 055136 006302
8624 055140 005722
8625 055142 010205
8626 055144 000137 055416
8627
8628
8629
8630 055150 005702
8631 055152 001406
8632 055154 006204
8633 055156 103410
8634 055160 006304
8635 055162 010437 054716
8636 055166 000412
8637 055170 012737 056772 054716
8638 055176 000406
8639
8640
8641
8642
8643
8644
8645 055200 052705 000001
8646 055204 012700 000077
8647 055210 004537 056610
8648 055214 004537 056710
8649 055220 005004
8650 055222 005002
8651 055224 004537 056532
8652 055230 022700 000060

```

```

: SPECIAL NAME HANDLER
: DEPENDS UPON THE EXPLICIT ORDER OF THE TWO TABLES O.TL AND O.URD
O.REGT: JSR 5,O.GET ;SPECIAL NAME, GET ONE MORE CHARACTER
MOV #O.TL,R4 ;TABLE START ADDRESS
O.RSP: CMPB RD,(R4)+ ;IS THIS THE CORRECT CHARACTER?
BEQ O.SP ;JUMP IF YES
CMP #O.TL+O.LG,R4 ;IS THE SEARCH DONE?
BHI O.RSP ;BRANCH IF NOT
BIC #177770,R0 ;MASK OFF OCTAL
MOV RD,R4
O.SP1: ASL R4
ADD #O.URD,R4 ;GENERATE ADDRESS
INC R2 ;SET FOUND FLAG
BR O.SCAN ;GO FIND NEXT CHARACTER
O.SP: SUB #O.TL-7,R4 ;CORRECT CONSTANT
BR O.SP1

:
: ← HANDLER - OPEN INDEXED ON THE PC
O.ORPC: JSR PC,O.TCLS
MOV R5,R2 ;CURRENT ADDRESS IN R2
ADD #R2,R2 ;COMPUTE
ASR R2 ;MOVE ONE BIT TO CARRY
BCS O.ERR ;ERROR IF ODD NUMBER
ASL R2 ;RESTORE WORD
TST (R2)+ ;AND INCREMENT BY TWO
MOV R2,R5 ;UPDATE CAD
JMP O.OP2 ;GO FINISH UP

:
: B HANDLER - SET AND REMOVE BREAKPOINTS
O.BKPT: TST R2 ;IF NO NUMBER TYPED
BEQ O.RALL ;REMOVE BREAKPOINT
ASR R4 ;CHECK IF ODD
BCS O.ERR ;JUMP IF ODD
ASL R4 ;RESTORE ONE BIT
MOV R4,O.ADR1 ;SET A BREAKPOINT
BR O.DCD
O.RALL: MOV #O.TRTC,O.ADR1 ;CLEAR BREAKPOINT
BR O.DCD

:
: COMMAND DECODER - ODT11
:
: REGISTERS R0-R4 MAY BE USED,
: REGISTER R5 WILL BE CONSIDERED SAFE
O.ERR: BIS #1,R5 ;CLOSE EVERYTHING
MOV #1,R0 ;? TO BE TYPED
JSR 5,O.ETYP ;OUTPUT ?
O.DCD: JSR 5,O.CJLS ;TYPE <CR><LF>
O.DCD1: CLR R4 ;R4 CONTAINS THE CONVERTED OCTAL
CLR R2 ;R2 IS THE NUMBER FOUND FLAG
O.SCAN: JSR 5,O.GET ;GET A CHAR, RETURN IN R0
CMP #0,R0 ;COMPARE WITH ASCII 0

```


8653	055234	101013		BHI	0.CLGL	:CHECK LEGALITY IF NON-NUMERIC
8654	055236	022700	000267	CMP	#7,R0	:COMPARE WITH ASCII 7
8655	055242	103410		BLJ	0.CLGL	:CHECK LEGALITY IF NOT OCTAL
8656	055244	042700	177770	BIC	#177770,R0	:CONVERT TO BCD
8657	055250	006304		ASL	R4	:MAKE ROOM
8658	055252	006304		ASL	R4	:IN
8659	055254	006304		ASL	R4	:R4
8660	055256	060004		ADD	R0,R4	:PACK THREE BITS IN R4
8661	055260	005202		INC	R2	:R2 HAS NUMERIC FLAG
8662	055262	000760		BR	0.SCAN	:AND TRY AGAIN
8663	055264	005001		O.CLGL: CLR	R1	:CLEAR INDEX
8664	055266	120061	055741	O.LGL1: CMPB	R0,0.LGCH(R1)	:DO THE CODES MATCH?
8665	055272	00:405		BEQ	0.LGL2	:JUMP IF YES
8666	055274	005201		INC	R1	:SET INDEX FOR NEXT SEARCH
8667	055276	020127	000014	CMP	R1,#0.CLGT	:IS THE SEARCH DONE?
8668	055302	103336		BHIS	0.ERR	:OOPS!
8669	055304	000770		BR	0.LGL1	:RE-LOOP
8670	055306	006301		O.LGL2: ASL	R1	:MULTIPLY BY TWO
8671	055310	000171	055314	JMP	#0.LGDR(R1)	:GO TO PROPER ROUTINE
8672						
8673	055314	055344		O.LGDR: O.WRD	:	/ OPEN WORD
8674	055316	055376		O.CRET	:	CARRIAGE RETURN CLOSE
8675	055320	055052		O.REGT	:	\$ REGISTER OPS
8676	055322	055706		O.GO	:	G GO TO ADDRESS K
8677	055324	055410		O.OP1	:	<LF> MODIFY, CLOSE, OPEN NEXT
8678	055326	055122		O.ORPC	:	+ OPEN RELATED, INDEX - PC
8679	055330	055442		O.BACK	:	+ OPEN PREVIOUS

```

8680 055332 055452
8681 055334 055530
8682 055336 055524
8683 055340 055150
8684 055342 056014
8685 000030
8686
8687
8688
8689
8690 055344 005702
8691 055346 001410
8692 055350 010405
8693 055352 006205
8694 055354 103711
8695 055356 006305
8696 055360 011500
8697 055362 004537 056446
8698 055366 000714
8699 055370 042705 000001
8700 055374 000766
8701
8702
8703 055376 004737 056656
8704 055402 052705 000001
8705 055406 000702
8706
8707
8708
8709 055410 004737 056656
8710 055414 005725
8711 055416 004537 056702
8712 055422 010500
8713 055424 004537 056446
8714 055430 012700 000057
8715 055434 004537 056610
8716 055440 000744
8717
8718
8719
8720 055442 004737 056656
8721 055446 005745
8722 055450 000762
8723
8724
8725
8726 055452 006205
8727 055454 103737
8728 055456 006305
8729 055460 012700 000040
8730 055464 004537 056610
8731 055470 160504
8732 055472 005304
8733 055474 005304
8734 055476 010400
8735 055500 010402

```

```

0.OFST : 0 OFFSET
0.WSCH : W SEARCH WORD
0.EFF : E SEARCH EFFECTIVE ADDRESS
0.BKPT : B BREAKPOINTS
0.PROC : P PROCEED

0.LGL = -0.LGDR ;LGL MUST EQUAL 2X CHLGT ALWAYS
:
: PROCESS / - OPEN WORD
0.WRD: TST R2 ;GET VALUE IF R2 IS NON-ZERO
: BEQ 0.WRDA ;SKIP OTHERWISE
: MOV R4,R5 ;PUT VALUE IN CAD
0.WRD1: ASR R5 ;MOVE ONE BIT TO CARRY
0.ERR2: BCS 0.ERR ;JUMP IF ODD ADDRESS
: ASL R5 ;RESTORE THE CARRY BIT
: MOV 2R5,R0 ;GET CONTENTS OF WORD
: JSR 5,0.CADV ;GO GET AND TYPE OUT 2CAD
: BR 0.DCD1 ;GO BACK TO DECODER
0.WRDA: BIC #1,R5 ;CLEAR CLOSED BIT
: BR 0.WRD1 ;GO BACK TO MAIN-LINE
:
: PROCESS CARRIAGE RETURN
0.CRET: JSR PC,0.TCLS ;CLOSE LOCATION
: BIS #1,R5 ;CLOSE EVERYTHING
: BR 0.DCD ;RETURN TO DECODER
:
: PROCESS <LF>, OPEN NEXT WORD
0.OP1: JSR PC,0.TCLS ;CLOSE PRESENT CELL
: TST (R5)+ ;GENERATE NEW ADDRESS
0.OP2: JSR 5,0.CRLF ;<CR><LF>
: MOV R5,R0 ;NUMBER TO TYPE
: JSR 5,0.CADV ;TYPE OUT ADDRESS
: MOV #1,R0 ;TYPE A /
: JSR 5,0.FTYP
: BR 0.WRD1 ;GO PROCESS IT
:
: PROCESS ↑, OPEN PREVIOUS WORD
0.BACK: JSR PC,0.TCLS ;GENERATE NEW ADDRESS
: TST -(R5) ;GO DO THE REST
: BR 0.OP2
:
: PROCESS 0, COMPUTE OFFSET
0.OFST: ASR R5 ;GET LOW ORDER BIT
: BCS 0.ERR2 ;ERROR IF CLOSED
: ASL R5 ;RESTORE WORD
: MOV #1,R0 ;TYPE ONE BLANK
: JSR 5,0.FTYP ;AS A SEPARATOR
: SUB R5,R4 ;COMPUTE
: DEC R4 ;
: DEC R4 ; 16 BIT OFFSET
: MOV R4,R0 ;TYPE A
: MOV R4,R2 ;SAVE R4

```

8736	055502	004537	056446
8737	055506	010200	
8738	055510	006200	
8739	055512	103402	
8740	055514	004537	056446
8741	055520	000137	055220
8742			
8743			
8744			
8745			

```

JSR    S.O.CADV      ;NUMBER IN RO - WORD MODE
MOV    R2,RO
ASR    RO             ;DIVIDE BY TWO
BCS    0.OF1         ;BRANCH IF ODD
JSR    S.O.CADV      ;NUMBER IN RO - BYTE MODE
0.OF1: JMP    0.DCD1 ;ALL DONE

: SEARCHES - $MSK HAS THE MASK
:           $MSK+2 HAS THE FWA
:           $MSK+4 HAS THE LWA

```

8746					
8747	055524	005201		0.EFF: INC R1	;SET EFFECTIVE SEARCH
8748	055526	000401		BR 0.WDS	
8749	055530	005001		0.WSCH: CLR R1	;SET WORD SEARCH
8750	055532	005702		0.WDS: TST R2	;CHECK FOR OBJECT FOUND
8751	055534	001621		0.ERR1: BEQ 0.ERR	;ERROR IF NO OBJECT
8752	055536	013702	054712	MOV 0.MSK+2,R2	;SET ORIGIN
8753	055542	013705	054710	MOV 0.MSK,R5	;SET MASK
8754	055546	005105		COM R5	;AND COMPLEMENT IT
8755	055550	020237	054714	0.WDS2: CMP R2,0.MSK+4	; IS THE SEARCH ALL DONE?
8756	055554	101217		BHI 0.DCD	; YES
8757	055556	011200		MOV 2R2,R0	; GET OBJECT
8758	055560	005701		TST R1	;NO
8759	055562	001027		BNE 0.EFF1	;BRANCH IF EFFECTIVE SEARCH
8760	055564	010046		MOV R0,-(SP)	
8761	055566	010403		MOV R4,R3	;EXCLUSIVE OR
8762	055570	040400		BIC R4,R0	; IS DONE
8763	055572	042603		BIC (SP)+,R3	
8764	055574	050003		BIS R0,R3	; IN A VERY
8765	055576	040503		BIC R5,R3	; FANCY MANNER HERE
8766	055600	001016		0.WDS3: BNE 0.WDS4	;AND RESULT WITH MASK
8767	055602	010446		MOV R4,-(SP)	;RE-LOOP IF NO MATCH
8768	055604	004537	056702	JSR 5,0.CRLF	;REGISTERS R2,R4, AND R5 ARE SAFE
8769	055610	010200		MOV R2,R0	;TYPE (CR,LF)
8770	055612	004537	056446	JSR 5,0.CADV	;GET READY TO TYPE
8771	055616	012700	000057	MOV #1/,R0	;TYPE ADDRESS
8772	055622	004537	056610	JSR 5,0.FTYP	;SLASH TO R0
8773	055626	011200		MOV 2R2,R0	;TYPE IT
8774	055630	004537	056446	JSR 5,0.CADV	;GET CONTENTS
8775	055634	012604		MOV (SP)+,R4	;TYPE CONTENTS
8776	055636	005722		0.WDS4: TST (R2)+	;RESTORE R4
8777	055640	000743		BR 0.WDS2	;INCREMENT TO NEXT CELL AND
8778	055642	020004		0.EFF1: CMP R0,R4	;RETURN
8779	055644	001755		BEQ 0.WDS3	; IS (X)=K?
8780	055646	010003		MOV R0,R3	;TYPE IF EQUAL
8781	055650	060203		ADD R2,R3	; (X) TO R3
8782	055652	005203		INC R3	; (X)+X
8783	055654	005203		INC R3	; (X)+X+2
8784	055656	020304		CMP R3,R4	; IS (X)+X+2=K?
8785	055660	001747		BEQ 0.WDS3	;BRANCH IF EQUAL
8786	055662	042700	177400	BIC #177400,R0	;WIPE OUT EXTRANEIOUS BITS
8787	055666	110000		MOVB R0,R0	;EXTEND SIGN
8788	055670	000257		CCC	
8789	055672	006300		ASL R0	;MULTIPLY BY TWO
8790	055674	005200		INC R0	;ADD TWO
8791	055676	005200		INC R0	
8792	055700	060200		ADD R2,R0	;ADD PC
8793	055702	020004		CMP R0,R4	;IS THE RESULT A PROPER REL. BRANCH?
8794	055704	000735		BR 0.WDS3	
8795					
8796					
8797					
8798	055706	105037	056731	0.GO: CLRB 0.P	;DISALLOW PROCEED
8799	055712	006204		ASR R4	;CHECK LOW ORDER BIT
8800	055714	103617		BCS 0.ERR2	;ERROR IF ODD NUMBER
8801	055716	006304		ASL R4	;RESTORE WORD

PROCESS G - GO

```

8802 055720 010437 054702      MOV      R4,0.UPC      ;SET UP NEW PC
8803 055724 112737 000340 177776  MOVB     #0,STM,ST     ;SET HIGH PRIORITY
8804 055732 004537 056400      JSR      5,0.RSTT     ;RESTORE TELETYPE
8805 055736 105037 056730 0.TBIT:  CLRB     0,T        ;CLEAR BOTH
8806 055742 042737 000020 054704  BIC     #0,TBT,0.UST  ;T-BIT FLAGS
8807 055750 017737 176742 054722  MOV     @0,ADR1,0.UIN ;SAVE INSTRUCTION
8808 055756 013777 056772 176732  MOV     0,TRC,@0,ADR1 ;REPLACE WITH TRAP
8809 055764 012600 0.G02:  MOV     (SP)+,R0      ;RESTORE
8810 055766 012601      MOV     (SP)+,R1      ;R0
8811 055770 012602      MOV     (SP)+,R2      ;THRU
8812 055772 012603      MOV     (SP)+,R3
8813 055774 012604      MOV     (SP)+,R4
8814 055776 012605      MOV     (SP)+,R5
8815 056000 012606      MOV     (SP)+,SP      ;R5
8816 056002 013746 054704  MOV     0,UST,-(SP)   ;AND SP
8817 056006 013746 054702  MOV     0,UPC,-(SP)  ;AND STATUS
8818 056012 000006 0.RTIT: RTT          ;AND PC
8819                                     ;CHANGED TO RTI FOR 11/20 AND /05
8820                                     ;
8821                                     ; PROCESS P - PROCEED
8822                                     ; ONLY ALLOWED AFTER A BREAKPOINT
8823 056014 105737 056731 0.FROC: TSTB     0,P      ;CHECK LEGALITY OF PROCEED
8824 056020 001645      BEQ     0,ERR1        ;NOT LEGAL
8825 056022 105037 056731      CLRB     0,P          ;CLEAR PROCEED FLAG
8826 056026 005702      TST     R2            ;WAS COUNT SPECIFIED?
8827 056030 001402      BEQ     0,PRI        ;NO
8828 056032 010437 054720      MOV     R4,0.CT      ;YES, PUT AWAY COUNT
8829 056036 112737 000340 177776 0.PRI:  MOVB     #0,STM,ST  ;FORCE HIGH PRIORITY
8830 056044 004537 056400      JSR      5,0.RSTT     ;RESTORE TTY
8831 056050 112737 000340 177776 0.C1:  MOVB     #0,STM,ST  ;SET HIGH PRIORITY
8832 056056 105237 056730      INCB     0,T          ;SET T-BIT FLAG
8833 056062 052737 000020 054704  BIS     #0,TBT,0.UST  ;SET T-BIT
8834 056070 000735      BR      0.G02
8835                                     ;
8836                                     ; BREAKPOINT HANDLER
8837                                     ; A TRT BREAKPOINT CAUSES 0.BRK TO BE ENTERED, WHICH SAVES
8838                                     ; VARIOUS ODDS AND ENDS, FINDS OUT IF THE BREAKPOINT WAS LEGAL,
8839                                     ; AND GIVES CONTROL TO THE COMMAND DECODER
8840                                     ;
8841 056072 012637 054702 0.BRK:  MOV     (SP)+,0.UPC  ;PRIORITY IS 7 UPON ENTRY
8842 056076 012637 054704      MOV     (SP)+,0.UST  ;SAVE STATUS AND PC
8843 056102 004037 056310 0.BK1:  JSR      0,0.SVR     ;SAVE VARIOUS REGISTERS
8844 056106 105737 056730      TSTB     0,T          ;CHECK FOR T-BIT SET
8845 056112 001311      BNE     0,TBIT       ;JUMP IF SET
8846 056114 013777 054722 176574  MOV     0,UIN,@0,ADR1 ;REMOVE BREAKPOINTS
8847 056122 105737 054706      TSTB     0,PRI       ;CHECK IF PRIORITY
8848 056126 100003      BPL     0,BK2        ;IS AS SAME AS USER PGM
8849 056130 113705 054704      MOVB     0,UST,R5    ;PICK UP USER UST IF SO
8850 056134 000407      BR      0,BK3        ;AND DON'T COMPUTE THE PRIORITY
8851 056136 113705 054706 0.BK2:  MOVB     0,PRI,R5    ;OTHERWISE PICK UP ACTUAL PRIORITY
8852 056142 000257      CCC          ;CLEAR CARRY
8853 056144 106005      RORB     R5          ;SHIFT LOW ORDER BITS
8854 056146 106005      RORB     R5          ;INTO
8855 056150 106005      RORB     R5          ;HIGH ORDER
8856 056152 106005      RORB     R5          ;POSITION
8857 056154 110537 177776 0.BK3:  MOVB     R5,ST       ;PUT THE STATUS AWAY WHERE IT BELONGS

```

```

8858 056160 013705 054702      MOV      0,UPC,R5      ;GET PC. IT POINTS TO THE TRT
8859 056164 005745          TST      -(R5)        ;SUBTRACT TWO
8860 056166 010537 054702      MOV      R5,0,UPC     ;FROM THE USER'S PC
8861 056172 020537 054716      CMP      R5,0,ADR1    ;COMPARE WITH LIST
8862 056176 001417          BEQ      0,B2         ;JUMP IF FOUND
8863 056200 004537 056346      JSR      5,0,SVTT     ;SAVE TELETYPE STATUS
8864 056204 004537 056702      JSR      5,0,CRLF     ;
8865 056210 012704 056734      MOV      #0,BD,R4     ;ERROR, NOTHING FOUND
8866 056214 012703 056735      MOV      #0,BD+1,R3
8867 056220 004537 056574      JSR      5,0,TYPE     ;OUTPUT "BE" FOR BAD ENTRY
8868 056224 010500          MOV      R5,R0
8869 056226 042737 000020 054704 BIC      #0,TBT,0,UST ;CLEAR OUT ANY POSSIBLE FAKE T-BIT
8870 056234 000420          BR       0,B3         ;AND CONTINUE
8871 056236 005237 054720      0.B2:  DEC      0,CT
8872 056242 003302          BGT      0,C1         ;JUMP IF REPEAT
8873 056244 012737 000001 054720      MOV      #1,0,CT     ;RESET COUNT TO 1
8874 056252 105237 056731      INCB    0,P          ;ALLOW PROCEED
8875 056256 004537 056346      JSR      5,0,SVTT     ;SAVE TELETYPE STATUS, R4 IS SAFE
8876 056262 012700 000102      MOV      #1,B,R0
8877 056266 004537 056610      JSR      5,0,FTYP     ;TYPE "B"
8878 056272 013700 054716      MOV      0,ADR1,R0   ;GET ADDRESS OF BREAK
8879 056276 004537 056446      0.B3:  JSR      5,0,CADV     ;TYPE ADDRESS
8880 056302 005005          CLR      R5          ;CLEAR CAD
8881 056304 000137 055214      JMP      0,DCD        ;GO TO DECODER
8882
8883      ; SAVE REGISTERS R0-R6 IN INTERNAL STACK
8884
8885      0.SVR: MOV      (SP)+,0,XXX   ;PICK REGISTER FROM STACK AND SAVE
8886 056314 010637 054700      MOV      SP,0,USP    ;SAVE USER STACK ADDRESS
8887 056320 012706 054700      MOV      #0,USP,SP   ;SET TO INTERNAL STACK
8888 056324 010546          MOV      R5,-(SP)    ;SAVE
8889 056326 010446          MOV      R4,-(SP)    ;REGISTERS
8890 056330 010346          MOV      R3,-(SP)    ;1
8891 056332 010246          MOV      R2,-(SP)    ;THRU
8892 056334 010146          MOV      R1,-(SP)    ;5
8893 056336 013746 056726      MOV      0,XXX,-(SP) ;PUT SAVED REGISTER ON STACK
8894 056342 005746          TST      -(SP)
8895 056344 000200          RTS      R0
8896
8897      ; SAVE TELETYPE STATUS
8898
8899 056346 113737 177560 056732      0.SVTT: MOVB    0,RCSR,0,CSR1 ;SAVE R C/SR
8900 056354 113737 177564 056733      MOVB    0,TCSR,0,CSR2 ;SAVE T C/SR
8901 056362 105037 177560          CLRB    0,RCSR      ;CLEAR ENABLE AND MAINTENANCE
8902 056366 105037 177564          CLRB    0,TCSR      ;BITS IN BOTH C/SR
8903 056372 004537 056702      JSR      5,0,CRLF    ;TYPE <CR,LF>
8904 056376 000205          RTS      R5
8905
8906      ; RESTORE TELETYPE STATUS
8907
8908 056400 004537 056702      0.RSTT: JSR      5,0,CRLF    ;<CR,LF> BEFORE RESTORING
8909 056404 105737 177564          TSTB    0,TCSR      ;WAIT READY ON PRINTER
8910 056410 100375          BPL     -4
8911 056412 032737 004000 177560      BIT     #4000,0,RCSR ;CHECK BUSY FLAG ON READER
8912 056420 001403          BEQ     0,RSE1      ;SKIP READY LOOP IF NOT BUSY
8913 056422 105737 177560          TSTB    0,RCSR      ;WAIT READY

```

```

8914 056426 110375          BPL          -4          ; ON READER
8915 056430 113737 056732 177560 0.RSE1: MOVB    0.CSR1,0.RCSR ; RESTORE
8916 056436 113737 056733 177564      MOVB    0.CSR2,0.TCSR ; THE STATUS REGISTERS
8917 056444 000205          RTS      RS
8918
8919          ; TYPE OUT CONTENTS OF WORD OR BYTE WITH ONE TRAILING SPACE
8920          ; WORD IS IN R0
8921
8922 056446 010246          0.CADV: MOV     R2,-(SP)      ; SAVE R2
8923 056450 012704 056771      MOV     #0.BUF+6,R4      ; BUFFER START ADDRESS
8924 056454 012746 000060      MOV     #'0,-(SP)      ; CONSTANT ASCII 0
8925 056460 010002          0.SPC: MOV     R0,R2      ; GET
8926 056462 042702 177770      BIC     #177770,R2      ; OCTAL CHARACTER
8927 056466 061602          ADD     2SP,R2          ; CONVERT TO ASCII
8928 056470 110244          MOVB   R2,-(R4)        ; STORE IN BUFFER
8929 056472 006200          ASR    R0              ; SHIFT THIS MESS
8930 056474 006200          ASR    R0              ; RIGHT
8931 056476 006200          ASR    R0              ; THREE WHOLE PLACES
8932 056500 020427 056764      CMP     R4,#0.BUF+1    ; DONE?
8933 056504 101365          BHI    0.SPC          ; NO
8934 056506 042700 177776      BIC     #177776,R0      ; GET LAST BIT
8935 056512 062500          ADD     (SP)+,R0      ; CONVERT TO ASCII
8936 056514 110044          MOVB   R0,-(R4)        ; AND PUT IT AWAY
8937 056516 012703 056771      MOV     #0.BUF+6,R3    ; LWA
8938 056522 004537 056574      JSR    5,0.FTYP        ; TYPE WHOLE STRING OF CHARACTERS
8939 056526 012602          MOV     (SP)+,R2      ; RESTORE R2
8940 056530 000205          RTS      RS
8941
8942          ; GENERAL CHARACTER INPUT ROUTINE
8943          ; CHARACTER INPUT GOES TO R0
8944
8945 056532 105737 177560          0.GET: TSTB   0.RCSR      ; WAIT FOR
8946 056536 100375          BPL    -4              ; INPUT FROM KEYBOARD
8947 056540 113700 177562      MOVB   0.RDB,R0        ; GET A CHARACTER
8948 056544 004537 056610      JSR    5,0.FTYP        ; ECHO CHARACTER
8949 056550 042700 177600      BIC     #177600,R0      ; STRIP OFF PARITY FROM CHARACTER
8950 056554 001766          BEQ    0.GET          ; IGNORE NULLS
8951 056556 122700 000040      CMPB   #40,R0         ; CHECK FOR SPACES
8952 056562 001763          BEQ    0.GET          ; IGNORE NULLS
8953 056564 122700 000073      CMPB   #'',R0         ; CHECK FOR SEMI-COLON
8954 056570 001760          BEQ    0.GET          ; IGNORE THEM IF FOUND
8955 056572 000205          RTS      RS
8956
8957          ; GENERAL CHARACTER OUTPUT ROUTINE
8958          ; ADDRESS OF FIRST BYTE IN R4,
8959          ; ADDRESS OF LAST BYTE IN R3, (R3)>(R4)
8960
8961 056574 020304          0.TYPE: CMP     R3,R4      ; CHECK FOR COMPLETION
8962 056576 103426          BLO    0.FTYP1        ; EXIT WHEN DONE
8963 056600 112400          MOVB   (R4)+,R0        ; GET A CHARACTER
8964 056602 004537 056610      JSR    5,0.FTYP        ; TYPE ONE CHARACTER
8965 056606 000772          BR     0.TYPE          ; LOOP UNTIL DONE
8966
8967          ; TYPE ONLY ONE CHARACTER (CONTAINED IN R0)
8968
8969 056610 105737 177564          0.FTYP: TSTB   0.TCSR      ; CHECK STATUS

```

```

8970 056614 100375          BPL          -4          ;WAIT UNTIL READY
8971 056616 110037 177566    MOVB        R0,0.TDB    ;TYPE ONE CHARACTER
8972 056622 120037 000045    CMPB        R0,0#45    ;IS CHAR TO BE FILLED?
8973 056626 001012          BNE          0.TYP1     ;NO
8974 056630 113746 000044    MOVB        0#44,-(SP) ;YES, INIT THE COUNT
8975 056634 105737 177564    O.TYP2: TSTB        0.TCSR
8976 056640 100375          BPL          0.TYP2
8977 056642 105037 177566    CLRB        0.TDB      ;GENERATE NULL FILLER
8978 056646 105316          DECB        0.SP
8979 056650 003371          BGT          0.TYP2
8980 056652 005726          TST        (SP)+      ;POP STACK
8981 056654 000205    O.TYP1: RTS          R5
8982
8983          ;
8984          ; CLOSE WORD OR BYTE AND EXIT
8985          ; UPON ENTERING, R2 HAS NUMERIC FLAG, R4 HAS CONTENTS
8986 056656 006205    O.TCLS: ASR          R5          ;GET LOW ORDER BIT
8987 056660 103405          BCS          0.TC        ;JUMP IF ALREADY CLOSED
8988 056662 006305          ASL          R5
8989 056664 005702          TST        R2          ;IF NO NUMBER WAS TYPED THERE IS
8990 056666 001401          BEQ          0.CLS1     ;NO CHANGE TO THE OPEN CELL
8991 056670 010415          MOV         R4,R5      ;STORE WORD
8992 056672 000207    O.CLS1: RTN          PC
8993 056674 005746    O.TC:   TST        -(SP)      ;POP EXTRA CELL FROM STACK
8994 056676 000137 055200    JMP         0.ERR      ;AND SCREAM BLOODY MURDER
8995
8996          ;
8997          ; O.CRLF - TYPE <CR,LF>
8998          ; O.CRLS - TYPE <CR,LF>*
8999 056702 012703 056737    O.CRLF: MOV         0#0.CR+1,R3 ;LWA <CR,LF>
9000 056706 000402          BR          0.CRS
9001 056710 012703 056740    O.CRLS: MOV         0#0.CR+2,R3 ;LWA <CR,LF>*
9002 056714 012704 056736    C.CRS:  MOV         0#0.CR,R4  ;FWA
9003 056720 00-537 056574    JSR         5,0.TYPE    ;TYPE SOMETHING
9004 056724 000205          RTS          R5
9005
9006          ;
9007          ; O.XXX: .WORD 0          ;TEMPORARY STORAGE
9008          ; O.T: .BYTE 0          ; T-BIT FLAG
9009          ; O.P: .BYTE 0          ; PROCEED FLAG = 0 IF PROCEED NOT ALLOWED
9010          ;                      ; = 1 IF PROCEED ALLOWED
9011 055732 000          O.CSR1: .BYTE 0
9012 056733 000          O.CSR2: .BYTE 0
9013          ;
9014 056734 042502    O.BD: .WORD "BE
9015          ;
9016 056736 015          O.CR: .BYTE 015 ; <CR>
9017 056737 012          .BYTE 012 ; <LF>
9018 056740 052          .BYTE '*' ; *
9019          ;
9020 056741 057          O.LGCH: .BYTE '/' ; /
9021 056742 015          .BYTE 015 ; CARRIAGE RETURN
9022 056743 044          .BYTE 'S' ; S
9023 056744 107          .BYTE 'G' ; G
9024 056745 012          .BYTE 012 ; <LF>
9025 056746 137          .BYTE '*' ; *

```


ABASE = 177440	466	507	531#		
ACDW1 = 000000	466	509			
ACDW2 = 000000	466	510			
ACLO = 000010	257#				
ACPUOP = 000000	466	481			
ACT11 = 005466	1173#	2363#			
ADDW0 = 000000	466	511			
ADDW1 = 000000	466	512			
ADDW10 = 000000	466	521			
ADDW11 = 000000	466	522			
ADDW12 = 000000	466	523			
ADDW13 = 000000	466	524			
ADDW14 = 000000	466	525			
ADDW15 = 000000	466	526			
ADDW2 = 000000	466	513			
ADDW3 = 000000	466	514			
ADDW4 = 000000	466	515			
ADDW5 = 000000	466	516			
ADDW6 = 000000	466	517			
ADDW7 = 000000	466	518			
ADDW8 = 000000	466	519			
ADDW9 = 000000	466	520			
ADEVCT = 000000	466	472			
ADEVN = 000000	466	508			
RENV = 000000	466	477			
RENVN = 000000	466	478			
AFATAL = 000000	466	469			
AMADR1 = 000000	466	494			
AMADR2 = 000000	466	498			
AMADR3 = 000000	466	501			
AMADR4 = 000000	466	504			
AMAMS1 = 000000	466	488			
AMAMS2 = 000000	466	496			
AMAMS3 = 000000	466	499			
AMAMS4 = 000000	466	502			
AMSGAD = 000000	466	474			
AMSGLG = 000000	466	475			
AMSGTY = 000000	466	468			
AMTYP1 = 000000	466	489			
AMTYP2 = 000000	466	497			
AMTYP3 = 000000	466	500			
AMTYP4 = 000000	466	503			
APASS = 000000	466	471			
APRIOR = 000000	466				
APTCSU = 000040	6670	6842#			
APTENV = 000001	6617	6663	6798	6840#	
APTSIZ = 000200	2296	6839#			
APTSPO = 000100	6665	6800	6841#		
ASWREG = 000000	466	479			
ATESTN = 000000	466	470			
ATTN = 005376	640#	5605	5624	5650	6137
AUNIT = 000000	466	473			
AUSWR = 000000	466	480			
AVECT1 = 000000	466	505			
AVECT2 = 000000	466	506			
A.GSP = 035056	5365	5370	6181#		

E16

GDRVS	031644	2343	5483#															
GINT	032032	2347	5541#															
GMS	= ***** U	360	7405	7406	7407	7408	7409	7411	7413	7414	7415	7416	7417	7418				
		7419																
GO	= 000001	203#																
GSTAT	032526	2717	2808	2909	2932	2942	2952	2991	3015	3025	3035	3057	3074	3084				
		3131	3148	3158	3181	3189	3199	3235	3258	3277	3287	3304	3331	3341				
		3370	3411	3422	3438	3448	3493	3501	3511	3603	3703	3713	3723	3740				
		3757	3767	3806	3817	3833	3843	3871	3933	3985	3996	4012	4022	4052				
		4062	4072	4109	4126	4136	4163	4219	4229	4239	4259	4276	4286	4311				
		4434	4487	4518	4561	4672	4732	4744	4757	4770	4933	4990	5050	5107				
		5167	5224	5389	5531	5635	5655	5659	5691#	5713	5724	5736	5769	5808				
		5825																
GTSWR	= 104406	7411#																
HASC	005424	1146#	5589#	5605	8336													
HBA	005414	1142#	5585#															
HCS1	005406	1139#	2490	2589	2809	2910	2933	2992	3016	3065	3132	3236	3259	3305				
		3371	3546	3590	3641	3704	3741	4053	4110	4220	4260	4435	4458	4492				
		4523	4562	4587	4638	4934	5051	5168	5301	5345	5592*	5714	8332	8336				
		8343	8347	8359														
HCS2	005410	1140#	2522	2524	2528	2540	2615	2617	2621	2624	2816	3135	3182	3308				
		3744	3872	4113	4263	5583*	8332	8336	8343	8347	8359							
HDA	005416	1143#	5586*	5926	5927	8343	8350	8359										
HDB	005430	1148#																
HDC	005426	1147#	5590*	5938	5946	5949	5951*	5952	8340	8353	8356	8359						
HDS	005420	1144#	2526	2619	5587*	8332	8336	8343										
HDTAB	001550	625#	2893	2899	2902	2977	3220	3355	4416	4423	4426	4547	5841					
HMD	001542	620#	8350															
HD1	001516	608#	5843*	5844*	5845*	5846*	5847*	5848*	5855									
HEAD	001514	607#	2982*	3226*	3361*	4552*	5843	5937*	5944*	5953*	5984							
HER	005422	1145#	2813	3058	5588*	8332	8336	8343										
HMR1	005432	1149#	5591*	6275														
HMR2	005434	1150#	2492	2591	2718	2753	2767	3270	3324	3494	3553	3615	3934	4164				
		4312	4673	4733	4745	4758	4771	4991	5108	5225	5390	5592*	5737	5809				
		5826	6260	8332	8336	8340	8343	8347	8350	8353	8356	8359	8343	8347	8350			
HMR3	005436	1151#	4488	4519	5593*	5725	5770	6281	8332	8336	8340	8343	8347	8350				
		8353	8356	8359														
HOLD	032140	5572	5575	5582#														
HPAT	005442	1153#	5595*															
HPOS	005440	1152#	5594*															
HT	= 003011	62#	6678	6719														
HVRC	= 000400	243#																
HMC	005412	1141#	5584*															
HZ	001410	562#	4728	6233														
IDAE	= 002000	245#																
IE	= 000100	204#																
ILF	= 000001	235#																
INTER	036046	5556	6424#															
INVCYL	005364	633#																
IOTVEC	= 000020	157#	2265*	2266*														
IR	= 000100	222#																
LCLKF	005514	1189#	2379*															
LCVEC	001350	539#	2380															
LF	= 000012	63#	6713	6719	7425	7430	7433	7438	7447	7455	7464	7471	7476	7481				
		7485	7491	7497	7503	7510	7515	7521	7529	7534	7539	7545	7547	7551				
		7557	7559	7564	7569	7573	7575	7579	7584	7594	7599	7609	7619	7629				

MSG42	046050	4829	7778#							
MSG5	043230	2346	7521#							
MSG6	043307	6429	7529#							
MSG7	043343	2361	7534#							
MSG8	043400	3777	7539#							
MSG9	043437	3781	7545#							
MSP =	000100	273#								
MULT6	033644	5882	5908#							
M.ALGN=	040000	349#								
M.CA00=	017760	348#	5771							
M.COIF=	017760	341#	5739	5762						
M.DPID	036170	6056	6112	6190	6206	6472#				
M.DP40	036226	6483#	6505							
M.DP41	036262	6488	6496#							
M.DP42	036272	6494	6500#							
M.DP44	036324	6507	6513#							
M.DP50	036336	6481	6518#							
M.DRV =	000007	340#								
M.HEAD=	007000	351#								
M.ID =	000003	347#	6276							
M.OFST=	017760	342#								
M.PAR =	100000	352#	6313							
M.SECT=	000760	350#	5726							
M.SER =	077770	343#								
NE0 =	010000	228#	2528	2540	2621	2624				
NE1 =	004000	227#								
NUORV	012754	2554	2655#	5426						
NXF =	000004	237#								
OFFERR	001:12	604#	3789#	3877#	3879	3895	3903	3912	3916	
OFFSET=	000:15	195#	3405	3800	3979					
OFST =	000004	256#								
OPI =	020000	248#								
OR =	000200	223#								
O.ADR1	054716	8584#	8635#	8637#	8807	8808*	8846*	8861	8878	9069#
O.BACK	055442	8679	8720#							
O.30	056734	8865	8866	9014#						
O.2KPT	055150	8630#	8683							
O.BK1	056102	8578	8843#							
O.BK2	056136	8848	8851#							
O.BK3	056154	8850	8857#							
O.BRK	056072	8595	9841#							
O.BUF	056763	8923	8932	8937	9042#					
O.B2	056236	8862	8871#							
O.B3	056276	8870	8879#							
O.CADV	056446	8696	8713	8736	8740	8770	8774	8879	8922#	
O.CLGL	055264	8653	8655	8663#						
O.CLGT=	000014	8667	9032#							
O.CLS1	056672	8990	8992#							
O.CR	056736	8999	9001	9002	9016#					
O.CRET	055376	8674	8703#							
O.CRLF	056702	8711	8768	8864	8903	8908	8999#			
O.CRLS	056710	8648	9001#							
O.CRS	056714	9003	9002#							
O.CSR1	056732	8899#	8915	9010#						
O.CSR2	056733	8900#	8916	9011#						
O.CT	054720	8828#	8871#	8873#	9070#					

O.C1	056050	8831#	8872							
O.DCD	055214	8636	8638	8648#	8705	8756	8891			
O.DCD1	055220	8647#	8697	8741						
O.EFF	055524	868c	8747#							
O.EFF1	055E42	8759	8778#							
O.ENTR	054730	8575#								
O.ERR	055200	8622	8633	8645#	8668	8693	8751	8994		
O.ERR1	055534	8751#	8824							
O.ERR2	055354	8693#	8727	8800						
O.FTYP	056610	8647	8715	8730	8772	8877	8948	8964	8969#	
O.GET	056532	8601	8651	8945#	8950	8952	8954			
O.GO	055706	8676	8798#							
O.G02	055764	8809#	8834							
O.LG	000006	8605	9040#							
O.LGCH	056741	8664	9020#	9032						
O.LGOR	055314	8671	8673#	8685						
O.LGL	000030	8685#								
O.LGL1	055266	8664#	8669							
O.LGL2	055306	8665	8670#							
O.MSK	054710	8752	8753	8755	9062#					
O.OOT	054724	377	8573#	9051						
O.OFST	055452	8680	8726#							
O.OF1	055520	8739	8741#							
O.OP1	055410	8677	8709#							
O.OP2	055416	8626	8711#	8722						
O.OPPC	055122	8618#	8678							
O.P	056731	8593#	8798#	8823	8825*	8874*	9008#			
O.PRI	054706	8585	8847	8851	9061#					
O.PROC	056014	8684	8823#							
O.PRI	056036	8827	8829#							
O.PALL	055170	8596	8631	8637#						
O.RCSR =	177560	8563#	8899	8901*	8911	8913	8915*	8945		
O.ROB =	177562	8562#	8947							
O.REGT	055052	8601#	8675							
O.RSE1	056430	8912	8915#							
O.RSP	055062	8603#	8606							
O.RST	054766	8574	8793#							
O.RSTT	056400	8804	8830	8908#						
O.RST1	055016	8582	8590#							
O.RTIT	056012	8592#	8818#							
O.SCAN	055224	8612	8651#	8662						
O.SP	055114	8604	8613#							
O.SPC	056460	8925#	8933							
O.SP1	055102	8609#	8614							
O.STM =	000340	8552#	8594	8803	8829	8831				
O.STRT	054754	8573	8580#							
O.SVR	056310	8583	8843	8885#						
O.SVTT	056346	8863	8875	8899#						
O.T	056730	8805#	8832*	8844	9007#					
O.TBIT	055736	8805#	8845							
O.TBT =	000020	8553#	8806	8833	8869					
O.TC	056674	8987	8993#							
O.TCLS	056656	8618	8703	8709	8720	8986#				
O.TCSR =	177564	8565#	8900	8902*	8909	8916*	8969	8975		
O.TDB =	177566	8564#	8971*	8977*						
O.TL	056755	8602	8605	8613	9034#	9040				

	5659*	5662*	5672*	5683*	5695*	5698*	5709*	5713*	5716*	5718*	5724*	5731*	5736*
	5745*	5764*	5769*	5776*	5783*	5787*	5790*	5796*	5799*	5808*	5815*	5817*	5825*
	5830*	5832*	5872*	5880*	5882*	5903*	5912*	5972*	5992*	6017*	6034*	6040*	6056*
	6061*	6075*	6091*	6093*	6096*	6112*	6115*	6133*	6139*	6142*	6144*	6153*	6158*
	6159*	6161*	6173*	6176*	6190*	6199*	6202*	6206*	6210*	6211*	6212*	6223*	6225*
	6247*	6249*	6258*	6264*	6266*	6279*	6285*	6287*	6316*	6316*	6520*	6614*	6620*
	6668*	6687*	6694*	6701*	6715*	6717*	6817*	6834*	6974*	7157*	7214*	7219*	7254*
	7257*	7281*	7328*	8529*	8548*	8577	8590	8618*	8703*	8703*	8720*	8992*	
PCA = 004000	278#												
PCD = 010000	279#												
PCLKF 005516	1190#	2383*	2390*	6220	6244								
PCVEC 001352	540#	2384	2391										
PCYL 001376	556#												
PFSRT 013106	2698#	6459											
PGE = 002000	226#												
PIP = 020000	264#												
PIPO = 177772	69#												
FIRQVE= 000240	163#												
PKPB 001344	536#												
PKS 001340	534#	2382	2389	6225*	6248*								
PKSB 001342	535#	6224*											
PPTP 005470	1174#	2330*											
PRGSRT 010724	2225	2230	2235	2240	2245	2250#	5515						
PR0 = 000000	86#	6216											
PR1 = 000040	87#												
PR2 = 000100	88#												
PR3 = 000140	89#												
PR4 = 000200	90#												
PR5 = 000240	91#	533											
PR6 = 000300	92#												
PR7 = 000340	93#	2252	2301	2306	2371	2394	6241	6454	6455				
PS = 177775	66#	67											
PSEC 001422	559#												
FSH = 177776	67#												
PWRVEC= 000024	158#	2271*	2272*	6444*	6453*	6454*							
OKCYLD 033004	5280	5288	5324	5332	5751#	6091							
ROCHR = 104410	7034	7414#											
ROCYLA 033074	3457	3619	4086	4496	4527	5768#							
ROCYLD 032720	3468	3474	3524	3530	3563	3569	3624	4082	5735#				
RODATA= 000021	197#	2804	3053										
ROGATE= 100000	272#												
ROHEAD= 000025	139#	3586	4841	4857									
ROLIN = 104411	5483	7106	7415#										
ROOCT = 104412	5528	5541	7416#										
ROSEC 032650	4849	4867	5723#	5783	5787	5790	5880						
ROTAB 004364	631#	3049	3094	3103									
ROY = 000200	205#	4375	4859	5568	5755								
RECAL = 000013	194#	3925	4155	4303	4982	5099	5216	5381					
RESREG= 104414	6060	6114	6175	6201	7166	7218	7418#	8520					
RESVEC= 000010	153#												
RHTAB 001754	626#	3585	3609	5887	5889	5893	5896	8356					
RKASOF= 000016	176#	2432	3402*	3463*	3519*	3558*	3794*	3976*	5589	5624	5650	6137	
RKBA = 000004	171#	2428	2799*	2847*	2902*	2924*	2977*	3006*	3049*	3110*	3122*	3174*	3220*
	3249*	3296*	3355*	3693*	3731*	3864*	4044*	4100*	4211*	4250*	4366*	4426*	4478*
	4509*	4547*	5585										
RKCSI = 000000	169#	2425	2486*	2585*	2762*	2804*	2905*	2928*	2941*	2950*	2987*	3011*	3024*

K16

	3033*	3053*	3073*	3082*	3126*	3147*	3156*	3177*	3188*	3197*	3231*	3254*	3266*
	3276*	3295*	3299*	3320*	3330*	3339*	3366*	3405*	3420*	3437*	3446*	3491*	3483*
	3492*	3500*	3509*	3537*	3536*	3632*	3699*	3712*	3721*	3735*	3756*	3765*	3800*
	3815*	3832*	3841*	3653*	3655*	3867*	3923*	3925*	3941*	3943*	3979*	3994*	4011*
	4020*	4032*	4034*	4048*	4061*	4070*	4081*	4104*	4125*	4134*	4153*	4155*	4171*
	4173*	4215*	4228*	4237*	4254*	4275*	4284*	4301*	4303*	4319*	4321*	4370*	4375*
	4430*	4449*	4482*	4513*	4557*	4578*	4629*	4651*	4657*	4659*	4719*	4841*	4857*
	4859	4923*	4980*	4982*	4998*	5000*	5040*	5097*	5099*	5115*	5117*	5157*	5214*
	5216*	5232*	5234*	5274*	5318*	5379*	5381*	5397*	5399*	5568	5582	5693*	5753*
	5755												
RKCS2 = 000010	173*	2426	2467*	2481*	2485*	2580*	2584*	3008*	3121*	3250*	3297*	3482*	3601
	3694*	3732*	3954*	3865*	3924*	3942*	4033*	4045*	4101*	4154*	4172*	4212*	4251*
	4302*	4320*	4367*	4479*	4510*	4658*	4981*	4999*	5098*	5116*	5215*	5233*	5380*
	5398*	5583	5692*	5707*	5711*								
RKDA = 000006	172*	2429	2800*	2824*	2848*	3796*	4210*	4249*	4369*	5586			
RKDB = 000024	178*	2434	3137	3310	3596	3597	3598	3746	3875	4115	4265		
RKDC = 000020	177*	2433	2801*	3580*	4209*	4248*	4428*	4447*	4477*	4549*	4575*	4922*	4946
	5039*	5063	5156*	5180	5273*	5590							
RKDS = 000012	174*	2430	5587										
RKECPS = 000030	182*	2438	5594										
RKECPT = 000032	183*	2439	5595										
RKER = 000014	175*	2431	5588										
RKMR1 = 000026	179*	2435	2716*	2951*	3034*	3083*	3157*	3198*	3286*	3340*	3410*	3421*	3447*
	3510*	3722*	3766*	3805*	3816*	3842*	3932*	3984*	3995*	4021*	4071*	4135*	4162*
	4238*	4285*	4310*	4670*	4730*	4742*	4755*	4768*	4989*	5106*	5223*	5388*	5591
	5530*	5634*	5654*	5658*	5712*	5723*	5735*	5752*	5768*	5807*	5824*		
RKMR2 = 000034	180*	2436	5532	5761									
RKMR3 = 000036	181*	2437	5593										
RKPRI = 001336	533*	5557											
RKVEC = 001334	532*	2366*	5545*	5548*	5555								
RKWC = 000002	170*	2427	2803*	2903*	2925*	2978*	3009*	3050*	3123*	3175*	3221*	3251*	3295*
	3355*	3695*	3733*	3866*	4046*	4102*	4213*	4252*	4368*	4427*	4480*	4511*	4548*
	5584												
RLS = 000010	219*												
RTT = 000006	855*												
RO = %000000	74*	2290*	2384*	2391*	2393*	2394*	2475*	2485	2494	2496	2503	2514*	2515
	2543	2574*	2584	2593	2598*	2599	2689*	2690	2728*	2729*	2730*	2731	2835*
	2336	2893*	2895*	2896*	2897*	2899	3007*	3096	3098	3106	3111*	3585*	3596*
	3097*	3598*	3780*	3794	3882	3890	3893	3897*	3900*	3910	3971*	3976	4146
	4148*	4204*	4209	4248	4295	4416*	4418*	4419*	4420*	4421*	4423	4465*	4532*
	4533	4536*	4837*	4846	4885*	4866	4911*	4961*	4962	5028*	5078*	5079	5145*
	5195*	5196	5260*	5359*	5360	5442*	5445	5461*	5463*	5484*	5496	5506	5529*
	5530	5532	5542*	5543	5545	5555*	5556*	5557*	5678	5839	5841*	5853*	5854*
	5855*	5856*	5857*	5860	5862	5864*	5871*	5878	5884*	5885*	5886	5889*	5892
	5893	5896*	5897	5898	5902*	6011*	6043*	6044	6050*	6069*	6099*	6100	6106*
	6126	6127*	6128*	6132*	6167*	6182*	6193*	6299	6301*	6308*	6315*	6344*	6345
	6349	6379*	6330	6384	6427*	6428	6430	6479*	6480*	6486*	6491*	6493*	6497*
	6499*	6509*	6510*	6661	6662*	6667	6672	6675*	6732	6742*	6746	6762	6763
	6776*	6794	6802*	6806	6807	6809*	6810*	6811	6833*	7103	7107*	7108	7111
	7131*	7134*	7156*	7160	7170*	7172*	7174*	7194*	7195	7213*	7217*	7269	7270*
	7271	7273	7275*	7276	7279*	7297	7310*	7312*	7313*	7319*	7321*	7323	7327*
	7347	7372*	7382	7383*	7384	7385*	7386*	7387*	7368*	8465*	8466*	8467*	8468*
	8469*	8470*	8471*	8472	8477	8482	8485*	8486	8490	8491	8503	8508	8511
	8513*	8541*	8603	8607*	8608	8646*	8652	8654	8656*	8660	8664	8695*	8712*
	8714*	8729*	8734*	8737*	8738*	8757*	8760	8762*	8764	8769*	8771*	8773*	8778
	8790	8785*	8787*	8789*	8790*	8791*	8792*	8793	8809*	8868*	8876*	8878*	8895*

L16

		8925	8929*	8930*	8931*	8934*	8935*	8936	8947*	8949*	8951	8953	8963*	8971		
R1	=0.000001	8972														
		75#	2476*	2501*	2513	2575*	2535	2597	2605	2605*	2626	2677*	2681	2687		
		2718*	2722*	2723*	2724*	2725*	2726*	2727*	2728	3034*	3035	3098	3103	3475		
		3531	3570	3657*	3660*	3778*	3782	3795*	3796	5197*	5193*	3954	5462*	5464*		
		5435*	5499	5517*	5840	5842*	5854	5866*	5867	5970*	5579	5830*	5842*	5837*		
		5901*	6051*	6107*	6166*	6183*	6194*	6300	6302*	6307*	6311	6314*	6473*	6485*		
		6490*	6496*	6509*	6733	6746*	6747	6751	6753*	6795	6832*	7104	7109*	7117*		
		7119*	7121*	7124*	7127	7130*	7152*	7156	7157*	7169*	7171*	7173*	7196*	7202*		
		7208*	7298	7301*	7304*	7314*	7323*	7324	7325*	7343	7371*	8432*	8494	8542*		
		8663*	8664	8666*	8667	8670*	8671	8747*	8749*	8758	8810*	8832				
		76#	3397*	3402	3463	3465	3469	3519	3521	3525	3558	3560	3564	3648		
		3651*	3652	3656*	3651*	3662	4874*	4876	5305*	5307	5349*	5351	5486*	5467*		
		5491	5497	5506*	5507*	5509	5510	6052*	6108*	6171*	6185*	6195*	6208	6484*		
		6734	6745*	6749*	6752	6759*	6760*	6761	6756*	6774*	7105	7110*	7118*	7120*		
7122*	7128	7129*	7153*	7159	7160*	7175*	7176*	7193*	7196	7197*	7203*	7204*				
7209*	7210*	7299	7305*	7308*	7312	7325*	7349	7370*	8486*	8500*	8543*	8611*				
8619*	8520*	8621*	8623*	8624	6625	8630	6550*	8661*	8629	8735*	8737	8750				
8752*	8755	8757	8769	8773	8776	8781	6732	8811*	8626	8891	8922	8925*				
8926*	8927*	8928	8939*	8989												
R3	=0.000003	7#	2721*	2732*	4875*	4877	4879	4890*	4882	5306*	5308	5310	5311*	5312		
		5350*	5352	5354	5355*	5356	5468*	5493	5504*	6054*	6057	6110*	6113	6170*		
		6184*	6191	6197*	6207	6483*	6503*	6506*	6513*	6735	6743*	6744*	6758*	6761*		
		6770*	6771*	6773*	6876	6885*	6891*	6892*	6895*	6900*	6901*	6902	6911*	7029		
		7031*	7032	7035*	7036	7043*	7044	7046	7054	7058	7060*	7066	7068	7070*		
		7073*	7155*	7168*	7175	7201*	7206*	7212*	7213	7350	7369*	8490*	8492	8496*		
		8544*	8761*	8763*	8764*	8765*	8780*	8781*	8782*	8783*	8784	8812*	8866*	8890		
		8937*	8961	8999*	9001*											
		78#	2719*	2720	2731*	2734*	4929*	5046*	5163*	5489*	5491	5494*	5495	5603		
		5604*	5605	5607*	5610*	5621	5623*	5624	5632*	5636*	5648	5649*	5650	5656*		
		5660*	5922	5960*	5961	5966	5971*	5974	5978	5981	5990*	6053*	6109*	6137		
		6169*	6187*	6196*	6473	6499	6510	6877	6879*	6880*	6881*	6882	6883*	6897		
		6899*	6907*	6910*	7154*	7161*	7199*	7202	7208	7210	7351	7368*	8484*	8488*		
		8505*	8516	8545*	8585*	8596*	8587*	8588*	8589	8602*	8603	8605	8608*	8609*		
8610*	8613*	8632*	8634*	8635	8649*	8657*	8658*	8659*	8660*	8691	8731*	8732*				
8733*	8734	8735	8761	8762	8767	8775*	8778	8784	8793	8799*	8801*	8802				
8813*	8828	8865*	8889	8923*	8928*	8932	8936*	8961	8963	8991	9002*					
R5	=0.000005	79#	2424*	2425	2426	2427	2428	2429	2430	2431	2432	2433	2434	2435		
		2436	2437	2438	2439	2467*	2481*	2485*	2486*	2580*	2584*	2585*	2716*	2762*		
		2799*	2800*	2801*	2803*	2804*	2824*	2847*	2648*	2902*	2903*	2905*	2924*	2925*		
		2928*	2941*	2950*	2951*	2977*	2979*	2987*	3006*	3007*	3009*	3011*	3024*	3033*		
		3034*	3049*	3050*	3053*	3073*	3082*	3083*	3110*	3121*	3122*	3123*	3126*	3137		
		3147*	3156*	3157*	3174*	3175*	3177*	3188*	3147*	3198*	3220*	3221*	3231*	3249*		
		3250*	3251*	3254*	3256*	3276*	3285*	3286*	3295*	3296*	3297*	3299*	3310	3320*		
		3330*	3339*	3340*	3355*	3356*	3366*	3402*	3405*	3410*	3420*	3421*	3437*	3446*		
		3447*	3463*	3481*	3482*	3483*	3492*	3500*	3509*	3510*	3519*	3537*	3558*	3580*		
		3586*	3596	3597	3598	3601	3632*	3693*	3694*	3695*	3699*	3712*	3721*	3722*		
		3731*	3732*	3733*	3735*	3746	3756*	3765*	3766*	3794*	3796*	3800*	3805*	3815*		
		3816*	3832*	3841*	3842*	3853*	3854*	3855*	3864*	3865*	3866*	3867*	3875	3923*		
		3924*	3925*	3932*	3941*	3942*	3943*	3976*	3979*	3984*	3994*	3995*	4011*	4020*		
		4021*	4032*	4033*	4034*	4044*	4045*	4046*	4048*	4061*	4070*	4071*	4081*	4100*		
		4101*	4102*	4104*	4115	4125*	4134*	4135*	4153*	4154*	4155*	4162*	4171*	4172*		
		4173*	4209*	4210*	4211*	4212*	4213*	4215*	4228*	4237*	4238*	4248*	4249*	4250*		
		4251*	4252*	4254*	4265	4275*	4284*	4285*	4301*	4302*	4303*	4310*	4319*	4320*		
		4321*	4366*	4367*	4368*	4369*	4370*	4375	4426*	4427*	4428*	4430*	4447*	4449*		
		4477*	4478*	4479*	4480*	4482*	4509*	4510*	4511*	4513*	4547*	4548*	4549*	4557*		

M16

R6 =%000006
 R7 =%000007
 SAVREG= 104413
 SBMR2 005444

 SBMR3 005446

 SBPAR 035524
 SCLR = 000040
 SCOP1 = 104415

 SCOP18 035576
 SDC = ***** U
 SEC 001414
 SECNT 001420
 SECTOR 001426

 SEEK = 000017
 SELDRV= 000001
 SETINT 032064
 SIZFLG 005522
 SKI = 000002
 SORT 033536
 SP =%000006

4575*	4578*	4629*	4551*	4657*	4658*	4659*	4670*	4719*	4730*	4742*	4755*	4758*
4841*	4857*	4859	4922*	4923*	4946	4980*	4981*	4982*	4983*	4933*	4933*	5000*
5039*	5040*	5063	5097*	5098*	5099*	5106*	5115*	5116*	5117*	5156*	5157*	5190
5114*	5215*	5216*	5223*	5232*	5233*	5234*	5273*	5274*	5218*	5279*	5210*	5331*
5299*	5397*	5398*	5399*	5568	5582*	5583	5584	5585	5536	5587	5513	5589
5590	5591	5592	5593	5594	5595	5624	5630*	5634*	5650	5654*	5653*	5692*
5693*	5707*	5711*	5712*	5723*	5735*	5752*	5753*	5755	5761	5768*	5807*	5824*
6055*	6111*	6137	6172*	6186*	6198*	6174	6496	6508	6736	6728*	6740*	6747*
6751*	6766	6772*	6878	6884*	6886*	6888*	6899*	6890*	6891	6909*	7153*	7159*
7164*	7165	7200*	7204	7211	7352	7307*	8546*	8619	8625*	8645*	8691*	8692*
8694*	8695	8698*	8704*	8710	8712	8721	8726*	8728*	8731	8753*	8754*	8765
8914*	8849*	8851*	8853*	8854*	8955*	8256*	8257	8953*	8859	8850	8861	8868
8990*	8688	8904*	8917*	8940*	8955*	8981*	8986*	8988*	8991*	9004*		
80*	82	2259*	2260*	2261								
81*	83											
6010	6068	6166	6181	7151	7192	7417*	8464					
1158*	2943*	2953*	3026*	3036*	3075*	3085*	3149*	3159*	3190*	3200*	3278*	3288*
3332*	3342*	3412*	3423*	3434*	3449*	3502*	3512*	3714*	3724*	3758*	3768*	3807*
3818*	3834*	3944*	3996*	3997*	4013*	4023*	4063*	4073*	4127*	4137*	4230*	4240*
4277*	4287*	6256*	6257	6259*	6260	8373	8343					
SBMR3 005446	1159*	2946*	2956*	3029*	3039*	3078*	3088*	3152*	3162*	3193*	3203*	3281*
	3335*	3345*	3416*	3427*	3442*	3452*	3505*	3515*	3717*	3727*	3761*	3771*
	3822*	3837*	3847*	3930*	4001*	4016*	4026*	4066*	4076*	4130*	4140*	4233*
	4280*	4290*	6277*	6278	6280*	6281	8332	8343				4243*
	6258	6279	6299*									
	221*	2467	2481	2580	5707							
	2478	2577	2756	2918	3000	3043	3115	3168	3243	3349	4093	4469
	7419*											4502
	6323*	7419										
	7424											
	566*	4729*	6015*	6028*	6073*	6085*	6234*					
	568*											
	571*	4850	4868	5725*	5726*	5727*	5728*	5729*	5730*	5794	5788	5791
	5885	5836*	5887*	5898	5908*	5909	5910*	5911*	5936*	5945*	5954*	5976
	196*	3537	3632	4449	4578	4629	4923	5040	5157	5274	5318	
	189*	2486	2585	3266	3320	5693	5753					
	2367	5546	5555*									
	1192*	2368*	2471	5501*	5521*							
	236*											
	5878*											
	82*	2251*	2252*	2253*	2263*	2281*	2289*	2293	2309	2370*	2371*	2372*
	2397	2400	2419*	2444	2465*	2479*	2503*	2543*	2573*	2579*	2668*	2690*
	2720*	2748*	2757*	2791*	2854*	2876*	2919*	2973*	3001*	3011*	3116*	3169*
	3244*	3350*	3395*	3688*	3782*	3882*	3969*	4094*	4202*	4352*	4409*	4470*
	4613*	4714*	4827*	4876*	4877*	4879*	4880	4906*	4940*	4941*	4943	4944
	5057*	5058*	5060	5061	5140*	5174*	5175*	5177	5178	5258*	5307*	5308*
	5311	5351*	5352*	5354*	5355	5422*	5439*	5484	5529	5542	5574*	5603*
	5608*	5610	5621*	5632	5636	5637*	5648*	5656	5660	5661*	5678*	5691*
	5717*	5751*	5763	5798*	5816*	5831*	5839*	5840*	5870	5871	5878*	5879*
	5902	5909*	5911	5922*	5923*	5924*	5969	5970	5971	5988	5989	5990
	6126*	6132	6143*	6151*	6152*	6154	6155	6156*	6209*	6216*	6217*	6241*
	6255*	6262	6263*	6265	6274*	6283	6284*	6286	6299*	6300*	6314	6315
	6330	6344	6349*	6356	6369	6379	6384*	6392	6405	6411	6418*	6427
	6456*	6472*	6473*	6474*	6475*	6476*	6477*	6478	6460	6482*	6487	6489*
	6492*	6493	6498*	6500*	6501	6504*	6514*	6518*	6543*	6546	6548	6549
	6579	6583*	6609	6630*	6633*	6661*	6662	6672*	6674	6675	6676*	6680

.SRDDE	1#		
.SRDOC	1#	11#	7087
.SREAO	1#	11#	6920
.SR2AZ	1#		
.SSAVE	1#	10#	7329
.SSB2D	1#	12#	7241
.SSB2O	1#		
.SSCOP	1#	10#	6522
.SSIZE	1#		
.SSUPR	1#	12#	7259
.STRAP	1#	11#	7374
.STYPB	1#		
.STYPD	1#	11#	6719
.STYPE	1#	10#	6640
.STYPO	1#	10#	6843
.S40CA	1#		
.1170	1#		

ADC	4883	4950	4958	5067	5075	5184	5192	5313	5357	6479	6491	6492	6497	6498	6500
	6509	7209													
ADD	2824	4882	4949	4951	4957	4959	5066	5068	5074	5076	5183	5185	5191	5193	5312
	5356	5574	5608	5637	5661	5717	5798	5816	5831	5881	5887	5889	5911	5991	6041
	6097	6143	6263	6284	6478	6480	6490	6493	6496	6499	6508	6510	6514	6518	6676
	6751	6803	6815	6827	6871	6881	6959	6968	7124	7176	7208	7210	7255	7312	8471
	8513	8610	8620	8660	8781	8792	8927	8935							
ASL	4537	5844	5845	5846	5947	5848	5851	5908	5910	6982	6983	6994	7117	7119	7121
	7386	8468	8469	8470	8609	8623	8634	8657	8658	8659	8670	8694	8728	8789	8801
	8988														
ASLB	6756														
ASR	5727	5728	5729	5730	5739	5740	5741	5742	5772	5773	5774	5775	6810	7168	8621
	8632	8692	8726	8738	8799	8929	8930	8931	8986						
BCC	6306	6757	7311												
BDS	6481	6507	8622	8633	8693	8727	8739	8800	8987						
BEQ	2297	2472	2529	2541	2596	2606	2622	2625	2627	2679	2682	2684	2686	2810	2814
	2817	2823	2837	2844	2879	2885	2911	2934	2993	3017	3059	3066	3099	3107	3133
	3136	3237	3260	3306	3309	3372	3459	3470	3476	3526	3532	3547	3565	3571	3591
	3602	3610	3616	3621	3626	3642	3653	3663	3705	3742	3745	3873	3880	3894	3896
	3911	3955	4054	4084	4088	4111	4114	4221	4261	4264	4296	4384	4436	4459	4489
	4493	4498	4520	4524	4529	4563	4588	4616	4639	4734	4736	4772	4851	4869	4935
	4947	5052	5064	5169	5181	5290	5302	5334	5346	5361	5425	5443	5492	5500	5509
	5531	5544	5606	5669	5715	5789	5792	5934	5942	5950	5963	5965	5977	5985	6129
	6324	6326	6355	6360	6391	6396	6488	6502	6553	6555	6557	6561	6570	6603	6606
	6629	6632	6666	6679	6714	6797	6801	6821	6823	6898	6939	6966	6981	7050	7112
	7163	7272	7274	7318	8473	8478	8483	8487	8493	8497	8504	8523	8604	8631	8665
	8690	8751	8779	8785	8824	8827	8862	8912	8950	8952	8954	8990			
BGE	6573														
BGT	5434	6505	6765	6905	6978	7019	7114	7162	8872	8979					
BHI	6559	8605	8653	8756	8933										
BHIS	9668														
BIC	2493	2592	2729	5431	5487	5507	5726	5738	5762	5771	5861	5862	5929	5930	5975
	5983	6276	6895	6935	6952	6979	7006	7012	7020	7123	7175	8466	8607	8656	8698
	8762	8763	8765	8786	8806	8869	8926	8934	8949						
BIS	2730	3008	3121	3250	3297	3694	3732	3865	4045	4101	4212	4251	4367	4421	4479
	4510	5855	5856	5857	5863	6256	6277	6313	6759	6760	6900	6901	6986	7212	8645
	8704	8764	8833												
BIT	2490	2522	2524	2526	2528	2540	2589	2615	2617	2619	2621	2624	2753	2767	2809
	2813	2816	2910	2933	2992	3016	3058	3065	3132	3135	3182	3236	3259	3270	3305
	3308	3324	3371	3465	3494	3521	3546	3553	3560	3590	3601	3615	3641	3648	3704
	3741	3744	3872	3890	3934	4053	4110	4113	4164	4220	4260	4263	4312	4375	4435
	4458	4488	4492	4519	4523	4562	4587	4638	4673	4733	4745	4758	4771	4859	4934
	4991	5051	5108	5168	5225	5281	5289	5301	5325	5333	5345	5390	5568	5714	5755
	5809	5826	6311	6323	6346	6354	6359	6381	6390	6395	6414	6538	6552	6560	6567
	6605	6612	6628	8522	8911										
BITB	2296	5605	5624	5650	6137	6665	6670	6702	6800						
BLE	8501														
BLO	7045	8655	8962												
BLOS	7033														
BLT	6693	6748	6764	6906	6976	7017	7116	7205							
BMI	6755														
BNE	2262	2286	2322	2325	2340	2358	2491	2495	2497	2499	2516	2518	2523	2525	2527
	2590	2594	2600	2616	2618	2620	2645	2673	2711	2733	2754	2768	2827	2829	2852
	2900	3104	3183	3271	3325	3466	3495	3522	3554	3561	3649	3891	3904	3913	3917
	3935	4147	4165	4313	4355	4376	4424	4446	4534	4539	4573	4621	4674	4677	4746
	4749	4759	4762	4775	4847	4860	4863	4887	4963	4992	5080	5109	5197	5226	5282

	5284	5294	5326	5328	5338	5391	5465	5473	5496	5498	5511	5518	5569	5571	5625
	5627	5629	5651	5653	5744	5756	5758	5785	5795	5810	5812	5814	5827	5829	5868
	5894	5899	6021	6023	6033	6045	6078	6080	6090	6101	6138	6141	6221	6232	6235
	6245	6261	6282	6309	6312	6347	6382	6415	6452	6539	6568	6613	6618	6636	6664
	6671	6673	6681	6689	6703	6710	6753	6799	6805	6808	6825	6896	6931	6937	6957
	6964	6971	7008	7014	7037	7039	7055	7059	7069	7216	7316	8506	8512	8517	8525
BPL	8759	8766	8845	8973											
	6625	6658	6707	6739	6769	6894	6933	6949	7004	7010	7302	7306	8848	8910	8914
BR	8946	8970	8976												
	2225	2230	2235	2240	2245	2288	2307	2348	2362	2385	2395	2338	2442	2508	2530
	2534	2538	2549	2553	2601	2609	2613	2623	2631	2635	2639	2643	2819	2831	2833
	2840	2849	3108	3141	3314	3472	3488	3528	3567	3750	3860	3908	3914	3948	4039
	4119	4178	4269	4326	4356	4364	4664	4799	4954	4865	4872	4955	4976	5005	5072
	5093	5122	5189	5210	5239	5375	5404	5512	5519	5549	5671	5939	5947	5955	5958
	5967	5979	5986	6038	6094	6130	6446	6494	6541	6547	6550	6563	6566	6623	6660
	6686	6696	6705	6712	6750	6767	6791	6813	6872	6887	6908	6960	6987	6989	7015
	7048	7057	7063	7065	7125	7138	7177	7207	8499	8514	8519	8573	8574	8582	8591
	8596	8612	8614	8636	8638	8662	8669	8697	8699	8705	8716	8722	8748	8777	8794
CCC	8834	8850	8870	8965	9000										
CLC	8788	8852													
CLR	6303	6511													
	2223	2224	2248	2249	2260	2274	2275	2295	2311	2314	2315	2375	2376	2401	2474
	2475	2574	2608	2795	2796	2797	2821	2846	2895	2946	2956	2979	3029	3039	3078
	3088	3152	3162	3193	3203	3222	3281	3291	3335	3345	3357	3410	3416	3427	3431
	3442	3452	3505	3515	3717	3727	3761	3771	3778	3789	3805	3811	3822	3826	3837
	3847	3984	3990	4001	4005	4016	4026	4066	4076	4130	4140	4210	4233	4243	4249
	4280	4290	4410	4447	4465	4467	4536	4541	4575	4576	4669	4686	4741	4754	4767
	4794	4801	4837	4838	4839	4856	4911	4912	4913	4914	4915	4928	4975	4978	5028
	5029	5030	5031	5032	5045	5092	5095	5145	5146	5147	5148	5149	5162	5209	5212
	5260	5261	5262	5263	5264	5287	5331	5374	5377	5428	5429	5463	5501	5630	5634
	5654	5658	5712	5745	5842	6011	6012	6029	6030	6050	6051	6052	6053	6069	6070
	6086	6087	6106	6107	6108	6109	6167	6168	6169	6182	6183	6193	6194	6195	6196
	6219	6230	6246	6248	6302	6413	6450	6482	6489	6565	6580	6742	6745	6885	6946
CLRB	6947	7030	7053	7109	7110	7158	7201	7300	7310	8484	8649	8650	8663	8749	8800
	2227	2229	2232	2233	2239	2243	2247	2734	6048	6104	6564	6685	6711	6771	6829
	6830	6831	7060	7070	7134	7217	8593	8798	8805	8825	8901	8902	8977		
CLV	6515														
CMP	2261	2285	2309	2387	2397	2400	2444	2494	2515	2593	2599	2826	2843	2899	3098
	3103	3106	3469	3475	3525	3531	3564	3570	3609	3620	3652	3662	3893	3910	3954
	4146	4295	4383	4423	4445	4497	4533	4538	4572	4886	4962	5079	5196	5360	5424
	5491	5495	5499	5508	5743	5784	5788	5791	5867	5893	5898	5962	5964	5976	5984
	6044	6100	6260	6281	6356	6369	6392	6405	6548	6572	6635	6763	6930	6936	6956
	6963	6975	6977	7007	7013	7016	7018	7032	7044	7211	7317	8524	8605	8652	8654
CMPB	8667	8755	8778	8784	8793	8861	8932	8961							
	2324	2357	6554	6558	6617	6663	6678	6680	6688	6709	6713	6798	6938	6970	7036
	7054	7058	7068	7113	7115	7273	8603	8664	8951	8953	8972				
COM	6488	8505	8754												
DEC	2607	2732	4862	5283	5327	5432	5570	5626	5628	5652	5670	5757	5794	5811	5813
	5828	5935	5943	5951	6022	6079	6128	6231	6234	6308	6504	7043	7161	7214	7275
	7307	7315	8467	8496	8500	8732	8733	8871							
DECB	6652	6695	6893	6904	8978										
EMT	58														
HALT	360	6363	6399	6435	6445	6626	6637	6659							
INC	2326	2330	2359	2363	2368	2379	2383	2390	2392	2500	2501	2514	2598	2680	2812
	2825	2839	2842	3651	3660	3661	3877	3900	3953	4532	4885	4961	5078	5195	5359
	5423	5430	5464	5474	5494	5504	5505	5517	5521	5866	6043	6099	6131	6236	6307

	6451	6503	6571	6608	6749	6828	6899	6907	6985	7164	7206	7303	8611	8661	8666
	8747	8782	8783	8790	8791										
INCB	6576	6602	6715	8832	8874										
IOT	59														
JMP	364	366	368	370	372	374	377	2328	2329	2341	2446	2520	2554	2575	2881
	2888	2914	2937	2996	3020	3063	2069	3112	3240	3263	3375	3594	3606	3654	3658
	3664	3708	3898	3901	3956	4057	4149	4224	4297	4386	4439	4566	4618	4623	4805
	4830	5362	5426	5450	5515	6025	6082	6357	6370	6393	6406	6419	6459	8527	8578
	8590	8626	8671	8741	8881	8994									
JSR	2313	2323	2343	2345	2347	2360	2367	2469	2483	2488	2510	2532	2536	2551	2582
	2587	2603	2611	2629	2633	2637	2641	2712	2717	2736	2750	2759	2764	2793	2806
	2808	2855	2856	2890	2907	2909	2921	2930	2932	2942	2944	2947	2952	2954	2957
	2974	2984	2989	2991	3003	3013	3015	3025	3027	3030	3035	3037	3040	3046	3055
	3057	3074	3076	3079	3084	3086	3089	3100	3118	3128	3131	3139	3148	3150	3153
	3157	3160	3163	3171	3179	3181	3189	3191	3194	3199	3201	3204	3218	3228	3233
	3234	3246	3256	3258	3268	3277	3279	3282	3287	3289	3292	3301	3304	3312	3322
	3331	3333	3336	3341	3343	3346	3352	3363	3368	3370	3399	3407	3411	3413	3417
	3422	3424	3428	3433	3438	3440	3443	3448	3450	3453	3457	3468	3474	3485	3487
	3493	3501	3503	3506	3511	3513	3516	3524	3530	3539	3543	3563	3569	3576	3588
	3603	3619	3624	3629	3634	3638	3690	3701	3703	3713	3715	3718	3723	3725	3728
	3737	3740	3748	3757	3759	3762	3767	3769	3772	3791	3802	3806	3808	3812	3817
	3819	3823	3828	3833	3835	3838	3843	3845	3848	3857	3859	3869	3871	3876	3929
	3933	3938	3945	3947	3973	3981	3985	3987	3991	3996	3998	4002	4007	4012	4014
	4017	4022	4024	4027	4036	4038	4050	4052	4062	4064	4067	4072	4074	4077	4082
	4086	4096	4106	4109	4117	4126	4128	4131	4136	4138	4141	4159	4163	4168	4175
	4177	4206	4217	4219	4229	4231	4234	4239	4241	4244	4256	4259	4267	4276	4278
	4281	4286	4288	4291	4307	4311	4316	4323	4325	4359	4362	4373	4377	4381	4413
	4432	4434	4451	4455	4474	4484	4487	4496	4505	4515	4518	4527	4544	4554	4559
	4561	4580	4584	4625	4631	4635	4645	4648	4653	4661	4663	4672	4675	4682	4716
	4721	4724	4731	4732	4737	4740	4744	4747	4757	4760	4770	4773	4782	4786	4790
	4797	4803	4832	4835	4843	4849	4861	4867	4878	4891	4909	4917	4925	4930	4933
	4942	4953	4966	4971	4986	4990	4995	5002	5004	5026	5034	5042	5047	5050	5059
	5070	5083	5088	5103	5107	5112	5119	5121	5143	5151	5159	5164	5167	5176	5187
	5200	5205	5220	5224	5229	5236	5238	5265	5268	5276	5280	5288	5292	5298	5309
	5315	5320	5324	5332	5336	5342	5353	5365	5370	5385	5389	5394	5401	5403	5445
	5546	5572	5575	5631	5635	5655	5659	5695	5709	5713	5724	5736	5769	5783	5787
	5790	5808	5825	5880	5882	6017	6034	6040	6056	6075	6091	6093	6096	6112	6139
	6153	6158	6159	6173	6190	6199	6206	6210	6211	6258	6279	6614	6620	6668	6687
	6694	6701	6817	6974	7254	8583	8601	8618	8647	8648	8651	8696	8703	8709	8711
	8713	8715	8720	8730	8736	8740	8768	8770	8772	8774	8804	8830	8843	8863	8864
	8867	8875	8877	8879	8903	8908	8938	8948	8964	9003					
MOV	2222	2228	2234	2237	2238	2242	2244	2251	2252	2253	2259	2263	2265	2266	2267
	2268	2269	2270	2271	2272	2273	2277	2278	2281	2282	2283	2284	2289	2291	2292
	2293	2298	2300	2301	2303	2305	2306	2310	2365	2366	2370	2371	2372	2374	2377
	2380	2381	2384	2388	2391	2393	2394	2418	2419	2423	2424	2441	2464	2465	2467
	2468	2476	2479	2481	2482	2485	2486	2487	2492	2503	2543	2572	2573	2575	2578
	2580	2581	2584	2585	2586	2591	2667	2668	2669	2670	2677	2687	2689	2690	2707
	2708	2716	2718	2719	2720	2721	2728	2747	2748	2757	2762	2763	2790	2791	2799
	2800	2801	2803	2804	2805	2835	2836	2847	2848	2854	2875	2876	2893	2896	2897
	2902	2903	2905	2906	2919	2924	2925	2928	2929	2941	2943	2950	2951	2953	2972
	2973	2977	2978	2981	2982	2983	2987	2988	3001	3006	3007	3009	3011	3012	3024
	3026	3033	3034	3036	3044	3049	3050	3053	3054	3073	3075	3082	3083	3085	3094
	3095	3096	3110	3111	3116	3122	3123	3126	3127	3137	3138	3147	3149	3156	3157
	3159	3169	3174	3175	3177	3178	3188	3190	3197	3198	3200	3216	3217	3220	3221
	3225	3226	3227	3231	3232	3244	3249	3251	3254	3255	3266	3267	3276	3278	3285
	3286	3288	3295	3296	3299	3300	3310	3311	3320	3321	3330	3332	3339	3340	3342

3350	3355	3356	3360	3361	3362	3366	3367	3394	3395	3397	3402	3404	3405	3406
3412	3420	3421	3423	3432	3437	3439	3446	3447	3449	3463	3481	3482	3483	3484
3492	3500	3502	3509	3510	3512	3519	3537	3538	3542	3558	3579	3580	3585	3586
3587	3596	3597	3598	3632	3633	3637	3656	3657	3687	3688	3693	3695	3699	3700
3712	3714	3721	3722	3724	3731	3733	3735	3736	3746	3747	3756	3758	3765	3766
3768	3780	3782	3794	3796	3799	3800	3801	3807	3815	3816	3818	3827	3832	3834
3841	3842	3844	3853	3854	3855	3856	3864	3866	3867	3868	3875	3882	3897	3923
3924	3925	3928	3932	3937	3941	3942	3943	3944	3968	3969	3971	3976	3978	3979
3980	3986	3994	3995	3997	4006	4011	4013	4020	4021	4023	4032	4033	4034	4035
4044	4046	4048	4049	4061	4063	4070	4071	4073	4081	4094	4100	4102	4104	4105
4115	4116	4125	4127	4134	4135	4137	4148	4153	4154	4155	4158	4162	4167	4171
4172	4173	4174	4201	4202	4204	4209	4211	4213	4215	4216	4228	4230	4237	4238
4240	4248	4250	4252	4254	4255	4265	4266	4275	4277	4284	4285	4287	4301	4302
4303	4306	4310	4315	4319	4320	4321	4322	4351	4352	4358	4366	4368	4369	4370
4372	4385	4408	4409	4411	4416	4418	4419	4420	4426	4427	4428	4430	4431	4449
4450	4454	4464	4466	4470	4472	4477	4478	4480	4482	4493	4503	4509	4511	4513
4514	4542	4547	4548	4549	4551	4552	4553	4557	4558	4578	4579	4583	4612	4613
4629	4630	4634	4650	4651	4652	4657	4658	4659	4660	4670	4681	4713	4714	4719
4720	4726	4728	4729	4730	4742	4751	4755	4764	4768	4777	4781	4785	4789	4796
4802	4826	4827	4841	4842	4857	4858	4874	4875	4876	4877	4879	4880	4905	4906
4920	4922	4923	4924	4925	4938	4939	4940	4941	4943	4944	4969	4970	4980	4981
4982	4985	4989	4994	4998	4999	5000	5001	5022	5023	5037	5039	5040	5041	5046
5055	5056	5057	5058	5060	5061	5086	5087	5097	5098	5099	5102	5106	5111	5115
5116	5117	5118	5139	5140	5154	5156	5157	5158	5163	5172	5173	5174	5175	5177
5178	5203	5204	5214	5215	5216	5219	5223	5228	5232	5233	5234	5235	5257	5258
5271	5273	5274	5275	5279	5297	5305	5306	5307	5308	5310	5311	5318	5319	5323
5341	5349	5350	5351	5352	5354	5355	5368	5369	5379	5380	5381	5384	5388	5393
5397	5398	5399	5400	5422	5435	5439	5442	5461	5462	5484	5485	5488	5489	5529
5532	5534	5542	5545	5548	5555	5556	5557	5582	5583	5584	5585	5586	5587	5588
5589	5590	5591	5592	5593	5594	5595	5603	5604	5607	5610	5621	5622	5623	5632
5636	5648	5649	5656	5660	5678	5691	5692	5693	5694	5697	5707	5708	5711	5723
5725	5735	5737	5751	5752	5753	5754	5761	5763	5768	5770	5782	5806	5807	5823
5824	5839	5840	5841	5843	5849	5853	5854	5859	5860	5864	5870	5871	5878	5879
5884	5886	5890	5892	5896	5897	5901	5902	5909	5922	5923	5924	5926	5927	5936
5937	5938	5944	5945	5946	5952	5953	5954	5960	5961	5969	5970	5971	5974	5981
5988	5989	5990	6014	6015	6016	6027	6028	6054	6055	6057	6072	6073	6074	6084
6085	6110	6111	6113	6126	6127	6132	6151	6152	6154	6155	6156	6170	6171	6172
6184	6185	6186	6187	6191	6197	6198	6207	6208	6209	6216	6217	6222	6224	6225
6233	6241	6242	6255	6257	6259	6262	6265	6274	6275	6278	6280	6283	6286	6299
6300	6301	6314	6315	6327	6330	6344	6349	6379	6384	6410	6411	6418	6427	6430
6444	6453	6454	6455	6456	6472	6473	6474	6540	6544	6546	6549	6562	6574	6575
6578	6579	6582	6583	6604	6609	6630	6633	6661	6662	6667	6675	6690	6732	6733
6734	6735	6736	6737	6738	6743	6746	6766	6772	6773	6774	6775	6776	6778	6779
6794	6795	6802	6806	6811	6812	6814	6816	6826	6832	6833	6868	6876	6877	6878
6884	6891	6909	6910	6911	6912	6913	6943	6967	6972	7001	7002	7029	7031	7042
7073	7074	7075	7076	7101	7102	7103	7104	7105	7107	7108	7127	7128	7129	7130
7131	7152	7153	7154	7155	7156	7157	7160	7165	7193	7194	7195	7196	7197	7198
7199	7200	7252	7253	7256	7269	7270	7276	7279	7280	7297	7298	7299	7301	7305
7309	7323	7324	7325	7326	7327	7347	7348	7349	7350	7351	7352	7353	7354	7355
7356	7363	7364	7365	7366	7367	7368	7369	7370	7371	7372	7382	7383	7387	7393
7394	8472	8477	8482	8485	8486	8494	8508	8526	8575	8576	8577	8580	8581	8584
8592	8594	8595	8602	8608	8619	8625	8635	8637	8646	8691	8695	8712	8714	8729
8734	8735	8737	8752	8753	8757	8760	8761	8767	8769	8771	8773	8775	8780	8802
8807	8808	8809	8810	8811	8812	8813	8814	8815	8816	8817	8828	8841	8842	8846
8858	8860	8865	8866	8868	8873	8876	8878	8885	8886	8887	8888	8889	8890	8891
8892	8893	8922	8923	8924	8925	8937	8939	8991	8999	9001	9002			

MOV8	2276	2731	5486	5506	6577	6581	6611	6619	6672	6700	6708	6741	6744	6758	6761
	6770	6789	6790	6792	6869	6870	6873	6874	6875	6879	6882	6883	6902	6934	6951
	7005	7011	7035	7040	7046	7051	7066	7111	7159	7213	7385	8465	8490	8585	8589
	8787	8803	8829	8831	8849	8851	8857	8899	8900	8915	8916	8928	8936	8947	8963
	8971	8974													
NEG	6475	6476	6740	6880	7304	7308	7319	7320							
NOP	5446	5447	5448	6035	6036	6037	6424	6425	6426	6436	6437				
RESET	2250	5444	6458												
ROL	2722	2723	2724	2725	2726	2727	4443	4444	4570	4571	6305	6483	6484	6485	6486
	6513	6886	6888	6889	6890	6892	7118	7120	7122						
ROR	6506	7169	7170	7171	7172	7173	7174	7313	7314						
RORB	8586	8587	8588	8853	8854	8855	8856								
RTI	2254	2290	2373	6218	6237	6243	6328	6331	6361	6397	6416	6438	6584	6639	6677
	6780	6914	6973	7021	7077	7132	7357	7373	7395	8592					
RTS	5466	5476	5502	5522	5533	5535	5547	5558	5573	5576	5596	5609	5611	5633	5638
	5657	5662	5672	5683	5698	5716	5718	5731	5746	5764	5776	5796	5799	5815	5817
	5830	5832	5872	5903	5912	5972	5992	6061	6115	6133	6142	6144	6161	6176	6202
	6212	6223	6226	6247	6249	6264	6266	6285	6287	6316	6516	6520	6717	6834	7167
	7219	7257	7281	7328	7388	8529	8895	8904	8917	8940	8955	8981	8992	9004	
RTT	8818														
SBC	6477	7203	7321												
SEV	6519														
SUB	5885	6610	6747	6809	7102	7204	8613	8731							
SWAP	3795	3797	5850	5931	5982	6047	6103								
TRAP	7397	7406	7407	7408	7409	7411	7413	7414	7415	7416	7417	7418	7419		
TST	2321	2339	2378	2382	2389	2425	2426	2427	2428	2429	2430	2431	2432	2433	2434
	2435	2436	2437	2438	2439	2471	2496	2498	2513	2517	2595	2597	2605	2626	2672
	2678	2681	2683	2685	2710	2822	2828	2851	2878	2894	3458	3625	3879	3895	3903
	3912	3916	4083	4087	4354	4528	4615	4620	4676	4735	4748	4761	4774	4846	4850
	4868	4946	5063	5180	5293	5337	5472	5493	5497	5510	5530	5543	5668	5933	5941
	5949	5966	5978	6020	6032	6077	6089	6140	6220	6244	6345	6390	6428	6487	6501
	6545	6569	6624	6631	6674	6682	6704	6752	6762	6804	6822	6824	6897	6965	6980
	7038	7049	7072	7126	7133	7322	7384	8492	8503	8516	8524	8630	8689	8710	8721
	8750	8759	8775	8826	8859	8894	8980	8989	8993						
TSTB	6325	6556	6657	6706	6754	6768	6796	6807	6820	6932	6948	7003	7009	7271	8491
	8511	8823	8844	8847	8909	8913	8945	8969	8975						
.ASCII	459	460	7425	7430	7433	7438	7447	7455	7464	7471	7476	7481	7485	7491	7559
	7758	7778	7895												
.ASCIIZ	458	461	5453	7081	7082	7083	7085	7497	7503	7510	7515	7521	7529	7534	7539
	7545	7547	7551	7557	7564	7569	7573	7575	7579	7584	7594	7599	7609	7619	7629
	7638	7644	7645	7654	7658	7666	7675	7685	7695	7703	7712	7722	7732	7736	7744
	7752	7776	7772	7786	7801	7809	7819	7827	7835	7838	7841	7851	7859	7867	7876
	7882	7937	7890	7902	7908	7914	7917	7920	7923	7926	7929	7934	7940	7947	7953
	7960	7965	7971	7974	7977	7980	7985	7990	7994	8000	8007	8015	8020	8026	8032
	8038	8044	8050	8054	8059	8068	8075	8080	8084	8090	8095	8100	8105	8113	8116
	8122	8124	8126	8133	8142	8149	8152	8160	8164	8171	8177	8181	8185	8189	8194
	8198	8203	8207	8211	8216	8221	8226	8231	8235	8244	8249	8257	8261	8268	8275
	8283	8290	8295	8300	8304	8311	8318	8324	8530						
.BLKB	7680	7178	7240												
.BLKM	625	626	627	629	630	631	6785								
.BYTE	422	423	428	429	437	438	446	447	448	449	477	478	488	489	496
	497	499	500	502	503	640	2506	2507	2546	2547	2693	2694	3785	3786	3885
	3886	5452	5681	5682	6352	6353	6387	6388	6433	6434	6621	6622	6835	6835	6837
	6915	6916	6917	6918	7078	7079	8366	8368	8371	8374	8376	8378	8381	8383	8386
	8388	8390	8393	8395	8397	8399	8402	8404	8406	8408	8411	8413	8415	8418	8420
	8422	8424	8427	8429	8433	8435	8437	8440	8442	8444	8446	8448	8450	8452	9007

	9008	9010	9011	9016	9017	9018	9020	9021	9022	9023	9024	9025	9026	9027	9028
.DSABL	9029	9030	9031	9034	9035	9036	9037	9038	9039	9044					
.ENABL	6990		6923	8538											
.END	9072														
.ENOC	23	37	39	40	41	58	150	164	365	382	386	398	393	395	402
	416	420	422	450	456	457	458	459	463	466	488	496	499	502	505
	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520
	521	522	523	524	525	526	527	531	1193	2219	2263	2264	2267	2269	2271
	2273	2274	2275	2277	2279	2300	2396	2408	2409	2416	2417	2418	2419	2443	2450
	2451	2462	2463	2464	2465	2473	2481	2507	2508	2547	2548	2557	2558	2570	2571
	2572	2573	2580	2602	2658	2659	2665	2666	2667	2668	2694	2695	2700	2701	2705
	2706	2707	2708	2712	2741	2742	2745	2746	2747	2748	2755	2762	2769	2774	2775
	2788	2789	2790	2791	2832	2834	2853	2865	2866	2873	2874	2875	2876	2916	2924
	2960	2962	2963	2970	2971	2972	2973	2998	3006	3043	3049	3092	3121	3166	3174
	3209	3210	3214	3215	3216	3217	3242	3249	3266	3319	3355	3377	3381	3382	3392
	3393	3394	3395	3608	3664	3667	3668	3685	3686	3687	3688	3731	3775	3786	3787
	3886	3887	3914	3940	3956	3958	3959	3966	3967	3968	3969	4080	4099	4144	4170
	4185	4186	4199	4200	4201	4202	4247	4294	4318	4333	4334	4349	4350	4351	4352
	4357	4365	4385	4389	4390	4406	4407	4408	4409	4441	4472	4508	4568	4602	4603
	4610	4611	4612	4613	4689	4690	4711	4712	4713	4714	4800	4808	4809	4824	4825
	4826	4827	4855	4866	4873	4880	4881	4896	4897	4903	4904	4905	4906	4943	4945
	4977	4997	5013	5014	5020	5021	5022	5023	5060	5062	5094	5114	5130	5131	5137
	5138	5139	5140	5177	5179	5211	5231	5247	5248	5255	5256	5257	5258	5311	5312
	5355	5356	5396	5414	5415	5416	5418	5428	5434	5437	5438	5442	5444	5450	5452
	5453	5456	5682	5683	6154	6156	6353	6354	6388	6389	6434	6435	6525	6528	6533
	6538	6540	6551	6554	6555	6556	6558	6560	6567	6571	6576	6578	6582	6585	6586
	6589	6592	6602	6609	6614	6615	6616	6624	6635	6639	6640	6643	6672	6722	6789
	6790	6793	6820	6835	6846	6923	6924	6926	6954	6990	6994	7022	7023	7031	7033
	7036	7064	7081	7087	7090	7096	7140	7143	7182	7244	7262	7285	7332	7377	7383
	7386	7405	7406	7407	7408	7409	7410	7411	7412	7413	7414	7415	7416	7417	7418
	7419	7502	8455	8464											
.EQUIV	58	59	67	82	83	112	113	114	115	116	117	118	119	120	121
	140	141	142	143	144	145	146	147	148	149					
.EVEN	466	643	6828	8331	8453	8539	9013	9045							
.IF	19	36	38	39	40	41	56	122	150	363	381	384	386	392	394
	401	415	419	421	450	456	457	458	462	463	465	488	496	499	502
	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519
	520	521	522	523	524	525	526	527	531	2219	2258	2263	2265	2267	2269
	2271	2273	2274	2275	2277	2295	2395	2407	2409	2416	2418	2419	2442	2449	2451
	2462	2464	2465	2472	2481	2506	2507	2546	2547	2556	2558	2570	2572	2573	2580
	2601	2657	2659	2665	2667	2668	2693	2694	2699	2701	2705	2707	2708	2711	2740
	2742	2745	2747	2748	2754	2759	2768	2773	2775	2788	2790	2791	2831	2833	2852
	2864	2866	2873	2875	2876	2916	2921	2940	2961	2963	2970	2972	2973	2998	3003
	3023	3046	3072	3118	3146	3171	3208	3210	3214	3216	3217	3242	3246	3266	3319
	3352	3377	3380	3382	3392	3394	3395	3608	3663	3666	3668	3685	3687	3688	3711
	3755	3785	3786	3885	3886	3913	3940	3955	3957	3959	3966	3968	3969	4060	4096
	4124	4170	4184	4186	4199	4201	4202	4227	4274	4318	4332	4334	4349	4351	4352
	4356	4364	4384	4388	4390	4406	4408	4409	4441	4472	4505	4568	4601	4603	4610
	4612	4613	4688	4690	4711	4713	4714	4799	4807	4809	4824	4826	4827	4854	4865
	4872	4879	4881	4895	4897	4903	4905	4906	4943	4945	4976	4997	5012	5014	5020
	5022	5023	5060	5062	5093	5114	5129	5131	5137	5139	5140	5177	5179	5210	5231
	5246	5248	5255	5257	5258	5310	5312	5354	5356	5396	5413	5414	5415	5416	5417
	5418	5420	5433	5436	5438	5442	5444	5450	5452	5453	5681	5682	6154	6156	6352
	6353	6387	6388	6433	6434	6524	6527	6532	6538	6550	6552	6553	6554	6556	6557
	6558	6567	6569	6577	6579	6584	6585	6586	6588	6591	6602	6605	6612	6614	6615

	6617	6624	6628	6635	6639	6640	6642	6663	6721	6788	6790	6793	6820	6835	6845
	6922	6924	6925	6926	6954	6993	6994	7022	7030	7032	7036	7037	7080	7081	7087
	7089	7092	7108	7142	7181	7243	7261	7284	7331	7376	7382	7386	7397	7406	7407
	7408	7409	7410	7411	7413	7414	7415	7416	7417	7418	7419	7424	8454	8463	
.IFF	36	39	40	41	56	382	386	393	395	402	416	419	422	450	463
	466	2263	2396	2408	2409	2417	2418	2419	2443	2450	2451	2453	2464	2465	2473
	2507	2547	2557	2558	2571	2572	2573	2602	2658	2659	2666	2667	2668	2694	2700
	2701	2706	2707	2708	2712	2741	2742	2746	2747	2748	2755	2769	2774	2775	2789
	2790	2791	2832	2834	2853	2865	2866	2874	2875	2876	2962	2963	2971	2972	2973
	3209	3210	3215	3216	3217	3381	3382	3393	3394	3395	3664	3667	3668	3686	3687
	3688	3786	3885	3886	3914	3956	3958	3959	3967	3968	3969	4185	4186	4200	4201
	4202	4333	4334	4350	4351	4352	4357	4365	4385	4389	4390	4407	4408	4409	4602
	4603	4611	4612	4613	4689	4690	4712	4713	4714	4800	4802	4809	4825	4826	4827
	4855	4866	4873	4881	4896	4897	4904	4905	4906	4944	4977	5013	5014	5021	5022
	5023	5061	5094	5130	5131	5138	5139	5140	5178	5211	5247	5248	5256	5257	5258
	5312	5353	5414	5417	5420	5434	5437	5452	5682	6155	6353	6388	6434	6525	6551
	6554	6555	6558	6585	6586	6589	6591	6605	6635	6640	6643	6722	6789	6846	6923
	6926	6934	6996	7001	7022	7023	7032	7064	7080	7090	7143	7182	7244	7262	7285
	7332	7377	7383	7502	8455	8464									
.IFT	4881	4945	5062	5179	5312	5356	6156	6566	6615	6996	7001	7113	7133	7140	
.IFTF	4880	4943	5060	5177	5311	5355	6154	656	6614	6941	6994	6997	7109	7117	7139
.IIF	18	23	28	33	34	35	37	40	41	360	462	466	2264	2267	2273
	2274	2275	2277	2278	2504	2544	2691	3783	3883	5415	5428	5429	5440	5452	5456
	5679	6350	6385	6431	6528	6529	6530	6531	6532	6533	6537	6565	6566	6582	6585
	6586	6592	6593	6594	6595	6596	6601	6627	6635	6640	6719	6923	6944	7072	7081
.IRP	7087	7140	7405	7406	7407	7408	7409	7411	7413	7414	7415	7416	7417	7418	7419
	531	2219	2407	2449	2556	2657	2699	2740	2773	2864	2961	3208	3380	3666	3957
	4184	4332	4388	4601	4688	4807	4895	5012	5129	5246	5420	6732	6772	6794	6795
	6816	6832	6833	7103	7129	7297	7325	7347	7367						
.LIST	1	4	40	164	360	450	451	452	453	454	455	456	463	466	645
	2219	2279	2407	2418	2449	2464	2556	2572	2657	2667	2699	2707	2740	2747	2773
	2790	2864	2875	2961	2972	3208	3216	3380	3394	3666	3687	3957	3968	4184	4201
	4332	4351	4388	4408	4601	4612	4688	4713	4807	4826	4879	4895	4905	4943	5012
	5022	5060	5129	5139	5177	5246	5257	5310	5354	5428	5444	6154	6532	6635	7022
	7397	7405	7406	7407	7408	7409	7410	7411	7412	7413	7414	7415	7416	7417	7418
	7419	7420													
.MACRO	1	41	413	651	671	699	714	733	765	788	820	833	845	869	893
	926	955	999	1071	1085	1109	2295	2407	2449	2556	2657	2699	2740	2773	2864
	2961	3208	3379	3666	3957	4184	4332	4388	4601	4688	4807	4895	5012	5129	5246
	7397														
.MCALL	9	10	11	12	164	463	2279								
.MLIST	1	3	40	164	360	450	451	452	453	454	455	456	463	466	1131
	2219	2279	2407	2418	2449	2464	2556	2572	2657	2667	2699	2707	2740	2747	2773
	2790	2864	2875	2961	2972	3208	3216	3380	3394	3666	3687	3957	3968	4184	4201
	4332	4351	4388	4408	4601	4612	4688	4713	4807	4826	4879	4895	4905	4943	5012
	5022	5060	5129	5139	5177	5246	5257	5310	5354	5428	5444	6154	6532	6635	7022
	7397	7405	7406	7407	7408	7409	7410	7411	7412	7413	7414	7415	7416	7417	7418
	7419	7420													
.NTYPE	4879	4943	5060	5177	5310	5354	6154								
.PAGE	54	353	413	1193	2219	2405	5411	5456	6522	7421					
.REPT	360	450													
.SBTTL	29	43	54	165	185	216	233	252	268	284	297	311	324	338	345
	354	363	379	390	413	463	1193	2220	2257	2405	2407	2449	2556	2657	2699
	2740	2773	2861	2864	2961	3208	3380	3666	3957	4184	4332	4388	4594	4601	4688
	4807	4895	5012	5129	5246	5411	5456	6335	6372	6408	6422	6440	6522	6586	6640
	6719	6786	6843	6920	7087	7140	7179	7241	7259	7282	7329	7374	7397	7422	7799

	8111	8329	8363	8455											
.TITLE	18														
.WORD	360	361	362	387	406	407	408	409	410	411	421	424	425	426	427
	430	431	432	433	434	435	436	439	440	441	450	451	452	453	454
	455	468	469	470	471	472	473	474	475	479	480	481	494	498	501
	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518
	519	520	521	522	523	524	525	526	5433	5436	5451	6669	6716	6818	6919
	7136	7139	7215	7258	7278	7404	8476	8481	8510	9006	9014				

ERRORS DETECTED: 0 HARD 2 SOFT
 DEFAULT GLOBALS GENERATED: 0

*.DZR6IB/CRF/SOL=SYSMAC.C1,DZR6IB.P11
 RUN-TIME: 59 89 16 SECONDS
 RUN-TIME RATIO: 604/165=3.6
 CORE USED: 41K (81 PAGES)

